EBook Gratis

APRENDIZAJE apache-pig

Free unaffiliated eBook created from
Stack Overflow contributors.

#apache-pig

Tabla de contenido

Acerca de

You can share this PDF with anyone you feel could benefit from it, downloaded the latest version from: apache-pig

It is an unofficial and free apache-pig ebook created for educational purposes. All the content is extracted from Stack Overflow Documentation, which is written by many hardworking individuals at Stack Overflow. It is neither affiliated with Stack Overflow nor official apache-pig.

Use the content presented in this book at your own risk; it is not guaranteed to be correct nor accurate, please send your feedback and corrections to info@zzzprojects.com

Capítulo 1: Empezando con apache-pig

Observaciones

Esta sección proporciona una descripción general de qué es apache-pig y por qué un desarrollador puede querer usarlo.

También debe mencionar cualquier tema grande dentro de apache-pig, y vincular a los temas relacionados. Dado que la Documentación para apache-pig es nueva, es posible que deba crear versiones iniciales de esos temas relacionados.

Examples

Instalación o configuración

Linux

Requisitos (r0.16.0)

Obligatorio

Según la documentación actual de `Apache-Pig` , solo admite `Unix` operativos `Unix` y `Windows` .

- Hadoop 0.23.X, 1.X o 2.X
- Java 1.6 o versiones posteriores instaladas y variable de entorno JAVA_HOME establecida en el directorio de instalación de Java

Opcional

- Python 2.7 o más (UDF de Python)
- Ant 1.8 (para construcciones)

Descarga la última versión de Pig

Descargue la última versión de pig desde http://pig.apache.org/releases.html#Download

Instalación

```
mkdir Pig
cd Downloads/
tar zxvf pig-(latest-version).tar.gz
tar zxvf pig-(latest-version).tar.gz
mv pig-(latest-version).tar.gz/* /home/Pig/
```

Configuración

Después de instalar Apache Pig, tenemos que configurarlo.

Abra el archivo .bashrc

```
vim ~/.bashrc
```

En el archivo .bashrc, establezca las siguientes variables:

```
export PIG_HOME = /home/Pig
export PATH  = PATH:/home/Pig/bin
```

Guarde el archivo y vuelva a cargar Bashrc en el entorno usando

```
. ~/.bashrc
```

Verificando la versión de Pig

```
pig –version
```

Si la instalación se realiza correctamente, el comando anterior muestra el número de versión de Pig instalado.

Prueba de instalación de cerdo

```
pig –h
```

Esto debería mostrar todos los comandos posibles asociados con cerdo.

Tu cerdo ahora está instalado localmente y puedes ejecutarlo usando un parámetro local como

```
pig –x local
```

Conectando a Hadoop

Si Hadoop1.x o 2.x está instalado en el clúster y la variable de entorno HADOOP_HOME está configurada.

puede conectar pig a Hadoop agregando la línea en el .bashrc como antes

```
export PIG_CLASSPATH = $HADOOP_HOME/conf
```

Cerdo corriendo

Modos de Ejecución

Puede ejecutar Pig utilizando el comando `pig` *(bin / pig)* o ejecutando el archivo `jar` *(java -cp pig.jar)*

`PIG` scripts `PIG` se pueden ejecutar en 3 modos diferentes:

- **Modo local**

```
pig -x local ...
```

- **Modo Mapreduce** (modo predeterminado)

```
pig -x mapreduce ...
     (or)
pig ...
```

- **Tez Modo Local**

```
pig -x tez ...
```

Modo interactivo

Pig se puede ejecutar en modo interactivo utilizando el shell `Grunt` . Las instrucciones y comandos de Pig Latin se pueden ingresar interactivamente en este shell.

Ejemplo

```
$ pig -x <mode> <enter>
grunt>
```

`Mode` puede ser uno de los modos de ejecución como se explica en la sección anterior.

Por lotes

Pig también se puede ejecutar en modo batch. Aquí se proporciona un archivo `.pig` que contiene una lista de instrucciones y comandos de cerdo.

Ejemplo

```
$ pig -x <mode> <script.pig>
grunt>
```

Del mismo `Mode` , el `Mode` puede ser uno de los modos de ejecución, como se explicó en la sección anterior.

Ejemplo de conteo de palabras en cerdo

Fichero de entrada

```
Mary had a little lamb
its fleece was white as snow
and everywhere that Mary went
the lamb was sure to go.
```

Código de conteo de palabras de cerdo

```
-- Load input from the file named Mary, and call the single
-- field in the record 'line'.
input = load 'mary' as (line);

-- TOKENIZE splits the line into a field for each word.
-- flatten will take the collection of records returned by
-- TOKENIZE and produce a separate record for each one, calling the single
-- field in the record word.
words = foreach input generate flatten(TOKENIZE(line)) as word;

-- Now group them together by each word.
grpd = group words by word;

-- Count them.
cntd = foreach grpd generate group, COUNT(words);

-- Print out the results.
dump cntd;
```

Salida

```
Mary,2
had,1
a,1
little,1
lamb,2
its,1
fleece,1
was,2
white,1
as,1
snow,1
and,1
everywhere,1
that,1
went,1
the,1
sure,1
to,1
go,1
```

¿Qué es el cerdo?

Pig proporciona un motor para ejecutar flujos de datos en paralelo en Hadoop. Incluye un lenguaje, Pig Latin, para expresar estos flujos de datos. Pig Latin incluye operadores para muchas de las operaciones de datos tradicionales (unirse, ordenar, filtrar, etc.), así como la capacidad para que los usuarios desarrollen sus propias funciones para leer, procesar y escribir datos. Pig es un proyecto de código abierto Apache. Esto significa que los usuarios pueden descargarlo gratuitamente como fuente o binario, usarlo por sí mismos, contribuir a él y, según los términos de la Licencia de Apache, usarlo en sus productos y cambiarlo según lo crea conveniente.

Lea Empezando con apache-pig en línea: https://riptutorial.com/es/apache-pig/topic/3244/empezando-con-apache-pig

Capítulo 2: Operación Cubo

Examples

CUBO basico

Teniendo en cuenta el siguiente caso:

Tenemos datos de eventos de usuario con 4 dimensiones:

1. Cubeta de prueba A / B (prod / prueba)
2. Tipo de cliente (web / móvil)
3. módulo (pedido / informe)
4. evento (clic / ver)

```
test    mobile    order_module    click
prod    web    order_module    view
prod    mobile    order_module    click
```

Un sistema de informes puede querer reportar métricas para diferentes combinaciones, tales como:

1. ¿Cuál es el número total de **clics** del usuario **móvil** ?
2. ¿Cuál es el número total de **clics** de diferentes cubos de prueba?
3. ¿ **Cuál es la vista** total de **la** página **web** para **order_module** en **prod** bucket?

Como puedes ver hay muchas combinaciones. No es bastante eficiente el tiempo si solo almacenamos las métricas de granularidad más pequeñas y luego las enrollamos cuando recibimos una consulta. Así que una solución es **pre-calcular TODAS las combinaciones** .

Así es como podríamos hacer eso con la operación CUBE de PIG.

```
example = LOAD './cube.example' AS (product:chararray, client:chararray, module:chararray,
action:chararray);

cubed_data = CUBE example BY CUBE(product, client, module, action);

final_data = FOREACH cubed_data GENERATE $0, COUNT_STAR($1);

dump final_data;
```

Producirá resultados de todas las combinaciones y cuentas totales. Vea la salida del volcado anterior: con estas estadísticas, podríamos responder preguntas anteriores con una respuesta directa. No se necesita más agregación.

```
((prod,web,order_module,view),1)
((prod,web,order_module,),1)
((prod,web,,view),1)
((prod,web,,),1)
```

```
((prod,mobile,order_module,click),1)
((prod,mobile,order_module,),1)
((prod,mobile,,click),1)
((prod,mobile,,),1)
((prod,,order_module,view),1)
((prod,,order_module,click),1)
((prod,,order_module,),2)
((prod,,,view),1)
((prod,,,click),1)
((prod,,,),2)
((test,mobile,order_module,click),1)
((test,mobile,order_module,),1)
((test,mobile,,click),1)
((test,mobile,,),1)
((test,,order_module,click),1)
((test,,order_module,),1)
((test,,,click),1)
((test,,,),1)
((,web,order_module,view),1)
((,web,order_module,),1)
((,web,,view),1)
((,web,,),1)
((,mobile,order_module,click),2)
((,mobile,order_module,),2)
((,mobile,,click),2)
((,mobile,,),2)
((,,order_module,view),1)
((,,order_module,click),2)
((,,order_module,),3)
((,,,view),1)
((,,,click),2)
((,,,),3)
```

Lea Operación Cubo en línea: https://riptutorial.com/es/apache-pig/topic/6120/operacion-cubo

Capítulo 3: Operador de carga

Examples

Cargando datos del mercado de valores

Asumamos los siguientes datos del mercado de valores almacenados en `HDFS` . Es un archivo `CSV` con campos: **Símbolo, Fecha, Abrir, Alto, Cerrar y Volumen** .

```
ABT,20160106,42.310001,42.98,42.209999,42.560001,5906000
BAC,20160201,14.05,14.09,13.8,13.96,105739400
CAS,20160129,1.9,1.97,1.83,1.84,34500
DCA,20160129,3.46,3.54,3.46,3.51,84600
ECL,20160114,103.480003,105.400002,102.480003,104.82,1485000
FAF,20160201,34.040001,34.82,33.939999,34.639999,1222600
TYL,20160201,156.070007,159.550003,155.690002,158.259995,177100
UTL,20160201,38.610001,39.889999,38.57,39.27,119500
VTR,20160128,54.09,54.73,53.549999,53.790001,2441300
WWE,20160201,17.629999,18,17.27,17.799999,734100
XRX,20160104,10.41,10.43,10.13,10.3,9122600
YUM,20160104,71.32,72.25,70.639999,72.209999,3466300
ZTR,20160104,12.1,12.14,11.98,12.11,60200
```

Ejemplo 1 Una simple instrucción `LOAD` para los datos anteriores se vería así:

```
stocks = load '/user/pig/stock.txt' using PigStorage(',') as
             (sym:chararray, date:int, open:float, high:float, low:float,
              close:float, vol:int);
```

Cargando datos de ElasticSearch

Antes de verificar la sintaxis específica, veamos cómo configurar su entorno para cargar los complementos necesarios.

Preparar

Para cargar datos directamente desde ElasticSearch, debe descargar el complemento `elasticsearch-hadoop` . Tiene diferentes formas de hacerlo funcionar, para una configuración rápida puede hacer lo siguiente.

Para que funcione, debe colocar el archivo jar `elasticsearch-hadoop-<version>.jar` en una carpeta del nodo donde tenga instalado el servidor porcino. En mi caso, es bastante común buscar en Google, también necesitaba agregar `commons-httpclient-<version>.jar` dentro de esa carpeta.

Luego puede ejecutar `pig` en modo shell (llamado `grunt`) simplemente escribiendo `pig` en la consola. Luego tienes que cargar esos dos tarros de la siguiente manera:

```
REGISTER /path/to/jars/commons-httpclient-<version>.jar;
REGISTER /path/to/jars/elasticsearch-hadoop-<version>.jar;
```

Ahora estás listo para escribir algo de código.

Ejemplo

Revisemos la sintaxis para cargar datos de un caso complejo.

```
DATA = LOAD 'my_index/log' USING org.elasticsearch.hadoop.pig.EsStorage(
'es.nodes=https://server1:port1,https://server2:port2,https://server3:port3',
'es.query=?q=*',
'es.net.ssl=true',
'es.net.http.auth.user=user',
'es.net.http.auth.pass=pass',
'es.net.ssl.keystore.type=JKS',
'es.net.ssl.truststore.location=file:///path/to/truststore.jks',
'es.net.ssl.truststore.pass=pass');
```

Este es el ejemplo completo, ahora vamos a analizarlo paso a paso.

- `es.nodes` contiene la lista de los nodos de su clúster ElasticSearch. Debe especificar sus nodos como una lista separada por comas, con el puerto asociado.
- `es.query` contiene la consulta que se enviará a ElasticSearch para obtener datos. También puede poner una consulta en formato DSL, pero tenga cuidado de que solo se considere la parte coincidente de la consulta. Si intenta limitar el número de campos a través de la consulta DSL, no funcionará: para lograrlo, debe usar el parámetro `es.read.source.filter` . ejemplo de una consulta DSL: `'es.query = { "query":{ "match_all":{} } }'`
- `es.net.ssl=true` se explica por sí mismo, también debe proporcionar las credenciales de inicio de sesión a ElasticSearch con `es.net.http.auth.user` y `es.net.http.auth.pass` .
- `es.net.ssl.keystore.type` si necesita un almacén de confianza, puede seleccionar aquí el tipo. En el parámetro `es.net.ssl.truststore.location` , establece la ubicación del archivo, tenga cuidado de agregar `file://` prefijo `file://` , y en el parámetro `es.net.ssl.truststore.pass` establece la contraseña del almacén de confianza expediente.

Algunas configuraciones útiles

- `es.read.source.filter=field1,field2,field3` permite obtener solo los campos especificados de ElasticSearch (en este ejemplo tres).
- `es.output.json=true` permite recuperar datos en un formato de valor clave (JSON). La configuración en `false` devolverá los datos en formato CSV (predeterminado).

Lea Operador de carga en línea: https://riptutorial.com/es/apache-pig/topic/7257/operador-de-carga

Creditos

S. No	Capítulos	Contributors
1	Empezando con apache-pig	Bhavesh, Brian Armstrong, Community, DhiwaTdG, Mzzzzzz, pratiklodha
2	Operación Cubo	Wenzhong
3	Operador de carga	Andrea Romagnoli, DhiwaTdG