Cloudera Streaming Analytics 1.5.0

Using Javascript User Functions

Date published: 2019-12-17
Date modified: 2021-09-29

https://docs.cloudera.com/

Legal Notice

Contents

Creating a JavaScript function

With SQL Stream Builder, you can create user functions to write powerful functions in JavaScript, that you can use to enhance the functionality of SQL.

About this task

User functions can be simple translation functions like Celsius to Fahrenheit, more complex business logic, or even looking up data from external sources. User functions are written in JavaScript. When you write them, you create a library of useful functions.

Procedure

1. Navigate to the Streaming SQL Console.
   a) Go to your cluster in Cloudera Manager.
   b) Select SQL Stream Builder from the list of services.
   c) Click SQLStreamBuilder Console.

   The **Streaming SQL Console** opens in a new window.
2. Select Console from the left-hand menu.
3. Select Functions > Add .
4. Name the function HELLO_WORLD, give it a short description, select JavaScript as the language.
5. Select STRING as output type and add STRING to the input type by selecting it from the list and clicking the plus button.
6. Paste this code into the JavaScript editor:

```
// check to see if the card is VISA
function HELLO_WORLD(card){
    var cardType = "Other";
    if (card.charAt(0) == 4){
      cardType = "Visa";
    }
    return cardType;
}
HELLO_WORLD($p0); // this line must exist
```

7. Click Save.
8. Once created, you can use a User Function in your SQL statement:

```
-- simple usage
SELECT HELLO_WORLD(card) AS IS_VISA
FROM ev_sample_fraud;

-- in the predicate
SELECT amount, card
FROM ev_sample_fraud
```

```
WHERE HELLO_WORLD(card) = "Visa";
```

Note: Valid inputs can be a field in the source virtual table or any other valid input. Functions must be in upper case.

Note: User Functions have access to the Java 8 API, this increases the overall usefulness and power. For example:

```
function GETPLANE(icao) {
  try {
    var c = new java.net.URL('http://yyyyyy.io' + icao).openConnec
tion();
    c.requestMethod='GET';
    var reader = new java.io.BufferedReader(new java.io.InputStreamRe
ader(c.inputStream));
    return reader.readLine();
  } catch(err) {
    return "Unknown: " + err;
  }
}
GETPLANE($p0);
```

Developing JavaScript functions

When developing JavaScript functions that are more complicated than just simple logic, it is recommended to use the jjs command-line utility to create and iterate while writing functions.

About this task

After the function performs the required task, migrate it to the console. Additionally these files/functions can be saved in a source code control system like git/Github.

Procedure

1. Create a file for your function.

   Note: It is recommended to name the file with the same name as that of the function.

2. Create some sample input when calling the function.
3. Call jjs on the command line to test the function.

```
$>cat TO_EPOCH.js
function TO_EPOCH(strDate) {
   var strFmt = "yyyy-MM-dd HH:ss:mm";
   var c = new java.text.SimpleDateFormat(strFmt).parse(strDate).getTime()
/1000;
   return c.toString();
}

print(TO_EPOCH("2019-02-02 22:23:13"));

then
$>jjs TO_EPOCH.js
1549167203
```

What to do next

After you have successfully developed the JavaScript code, copy and paste only the function to your code window when creating the JavaScript function in SQL Stream Builder.

Adding Java to the Functions language option

You can create User Defined Functions (UDF) using Java after manually adding the UDF function JAR file that contains the UDF class to the Flink connectors. After creating the function in the SQL window, you can select Java as a language for the UDFs.

Procedure

1. Create a Java UDF function JAR file based on the following example:

```
package udf;
import org.apache.flink.table.functions.ScalarFunction;

public class UdfTest extends ScalarFunction {
   public String eval(String input){
       return "Hello World " + input;
   }
}
```

   Note: The function needs to be named as ‘eval’ in the JAR file.

2. Copy the JAR file to the flink connectors folder:

```
scp <JAR_LOCATION>/<JAR_FILENAME> <YOUR_HOSTNAME>:/usr/share/flink-connec
tors
```

3. Navigate to the Streaming SQL Console.
   a) Go to your cluster in Cloudera Manager.
   b) Select SQL Stream Builder from the list of services.
   c) Click SQLStreamBuilder Console.

   The **Streaming SQL Console** opens in a new window.
4. Run the following query to create the function:

```
CREATE FUNCTION myFunction AS 'udf.UdfTest’ LANGUAGE java;
```

5. Test the function by running a SELECT query.

```
SELECT myFunction('test');
```