InfoSphere BigInsights user-defined functions for DB2 for z/OS

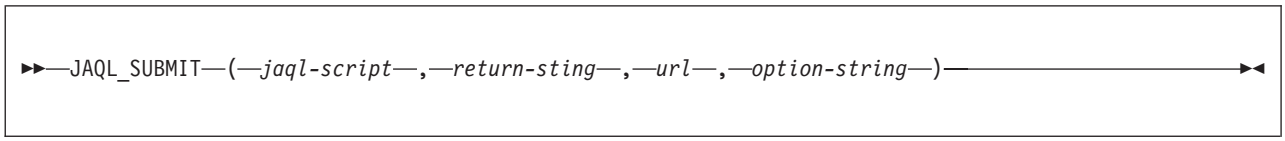
Contents

DB2 for z/OS user-defined functions

You can use the JAQL_SUMBIT and HDFS_READ user-defined functions to format and read data from an HDFS delimiter-separated file to DB2® for z/OS®.

JAQL_SUBMIT

JAQL_SUBMIT is a user-defined function that enables you to invoke an InfoSphere® BigInsights™ ad hoc Jaql query from a DB2 application. This function creates a CSV file from JSON data as input for the HDFS_READ function.



jaql-script
Specifies a script that contains one or more Jaql queries, with or without parameter declarations. *jaql-script* is a VARCHAR value with a maximum length of 8000.

return-string
Specifies a string to return when the Jaql script completes successfully. *return-string* is a VARCHAR value with a maximum length of 512. If *return-string* contains the location for the output of *jaql-script*, the JAQL_SUBMIT function can be specified in the HDFS_READ function as the *file-url* expression.

url
Specifies the URL of a Jaql server that accepts requests from the DB2 client, runs the queries specified in *jaql-script*, and returns results. *url* is a VARCHAR value with a maximum length of 512.

option-string
An expression that specifies a list of name/value pairs that are separated by a space character. *option-string* is a VARCHAR value with a maximum length of 256. *option-string* supports the following name/value pairs:

timeout
timeout identifies the maximum time (in seconds) to wait for results to be returned from the Jaql server specified in the *url* parameter. The timeout value is specified by using the keyword `TIMEOUT` (for example, `TIMEOUT="5000"`).

The default timeout value is 1000 (1000 seconds).

user-name
user-name identifies an InfoSphere BigInsights user name that has access to the Jaql server. The user name is specified in the *url* parameter by using the keyword `USER` (for example, `USER=bi_user01`).

password
password specifies the password of the InfoSphere BigInsights user that is identified in the *user-name* parameter. The password is specified by using the keyword `PASSWORD` (for example, `PASSWORD=s3cuRe`).

Result: The result of the function is the VARCHAR(512) string specified in the *return-string* parameter. If the *jaql-script* fails, SQLCODE -443 will be returned.

Note: Ensure that the InfoSphere BigInsights ad hoc Jaql query application has been deployed before using the JAQL_SUBMIT function.

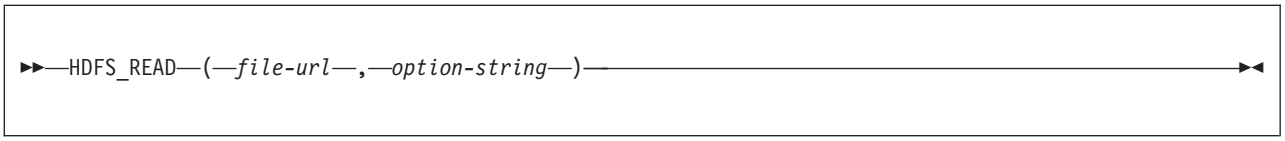
The DSNWLM_WEBSERVICES WLM environment should be modified to run the JAQL_SUBMIT function.

For example, the following SQL statement can be used to create the JAQL_SUBMIT function:

```
CREATE FUNCTION SYSFUN.JAQL_SUBMIT(SCRIPT VARCHAR(8000),
                                   PARMS VARCHAR(512),
                                   URL VARCHAR(512),
                                   OPTIONS VARCHAR(256))
   RETURNS VARCHAR(512)
   LANGUAGE C
   EXTERNAL NAME DSN8JAQL
   PARAMETER STYLE DB2SQL
   PARAMETER CCSID UNICODE
   PARAMETER VARCHAR NULTERM
   FENCED
   NOT DETERMINISTIC
   EXTERNAL ACTION
   DISALLOW PARALLEL
   WLM ENVIRONMENT DSNWLM_WEBSERVICES
   STAY RESIDENT YES
   RUN OPTIONS 'POSIX(ON),XPLINK(ON)';
```

HDFS_READ

HDFS_READ is a user-defined DB2 for z/OS generic table function to read delimiter-separated files from an InfoSphere BigInsights cluster.



The HDFS_READ function reads one delimiter-separated file from HDFS and returns a table with one row for each record (or row) in the file. The HDFS_READ function can read only delimiter-separated files that are encoded in ASCII or UTF-8.

If the number of result columns is less than the number of fields in each record, the function returns all fields of a record. For example, if the result table contains four columns, but each record in the file contains six fields, only the first four fields of each record are returned. If a field has no value (an empty field), a null value is returned for the corresponding column of the result table.

The function uses the InfoSphere BigInsights REST API for the data transfer.

file-url
An expression that specifies the HttpFS URL of the file in HDFS. *file-url* is a VARCHAR value with a maximum length of 512. It consists of three parts: the

server address, the web application, and the path of the file in HDFS. The web application must be `webhdsf`. For example, '`http://172.16.134.134:14000/webhdsf/v1/path/to/file.txt`'.

The HDFS_READ function accepts a file that has multiple parts as input. In this case, the path to the file in HDFS specifies a directory. The directory must contain an `_SUCCESS` file. If the directory contains an `_SUCCESS` file, the HDSF_READ function reads the data from the files in the directory. The HDFS_READ function will not read data from files where the file name starts with an underscore character (_).

option-string
An expression that specifies a list of name-value pairs that are separated by a space character. *option-string* is a VARCHAR value with a maximum length of 256. *option-string* supports the following name-value pairs:

delimiter
delimiter identifies the delimiter that is used in the file that is specified in *file-url*. The delimiter is specified by using the keyword `DELIMITER` or `DELIM` (for example, `DELIMITER=","` or `DELIM=" "`). Characters in the 7-bit ASCII range can be specified by using a quoted value. The actual delimiter character is a single-byte string. For the tab character, the delimiter can be specified by using `\t`.The default delimiter is the comma character (`,`).

user-name
user-name identifies an InfoSphere BigInsights user name that has access to the file specified in the *file-url*. The user name is specified by using the keyword `USER` (for example, `USER=bi_user01`).

password
password specifies the password of the InfoSphere BigInsights user that is identified in the *user-name* parameter. The password is specified by using the keyword `PASSWORD` (for example, `PASSWORD=s3cuRe`).

Delimited format: The basic rules for delimiter-separated values for use with the HDFS_READ function are as follows:

- Delimiter-separated files must be encoded in ASCII or UTF-8
- Fields are separated by the delimiter character
- Records/rows are separated by the new line character
- All records have the same number of fields, in the same order
- Fields that contain either the delimiter character or the new line character must be enclosed within double quotation mark characters

The following SQL statement can be used to create the HDFS_READ function:

```
CREATE FUNCTION SYSFUN.HDFS_READ(URL VARCHAR(256),
OPTIONS VARCHAR(256))
RETURNS GENERIC TABLE
LANGUAGE C
EXTERNAL NAME DSN8HDFS
PARAMETER STYLE DB2SQL
PARAMETER CCSID UNICODE
PARAMETER VARCHAR NULTERM
FINAL CALL
FENCED
NOT DETERMINISTIC
EXTERNAL ACTION
DISALLOW PARALLEL
SCRATCHPAD 200
```

```
WLM ENVIRONMENT DSNWLM_WEBSERVICES
STAY RESIDENT YES
RUN OPTIONS 'POSIX(ON),XPLINK(ON)'
CARDINALITY 100000;
```

SQL statement to create HDFS_READ:

The DSNWLM_WEBSERVICES WLM environment needs to be modified to run the HDFS_READ function.

Examples:

Assume that the following comma-separated value file (CSV file) exists in HDFS:

```
1997,Ford, E350,"ac, abs, moon",3000.00
1999,Chevy, "Venture ""Extended Edition""","",4900.00
1996,Jeep,Grand Cherokee,"MUST SELL!
air, moon roof, loaded",4799.00
2000,Toyota,Camry,,6700.0
```

The following SELECT statement returns the contents of the CSV file in HDFS as a DB2 table:

```
SELECT *
   FROM TABLE (HDFS_READ('http://bigins.example.com:14000/webhdfs/v1/path/to/cars.csv', ''))
       AS f(YEAR INT,
            MAKE VARCHAR(10),
            MODEL VARCHAR(30),
            DESCRIPTION VARCHAR(40));
```

The result table would be similar to the following. The fifth field for each record is not included in the result table because the SELECT statement did not define a fifth column in the generated table:

YEAR	MAKE	MODEL	DESCRIPTION
1997	Ford	E350	ac, abs, moon
1999	Chevy	Venture "Extended Edition"	(an empty string)
1996	Jeep	Grand Cherokee	MUST SELL! air, moon roof, loaded
2000	Toyota	Camry	(null value)

Index