ESQL String Splitter Functions For Splitting Delimited Strings

2

ESQL does not have an inbuilt string splitting function like Java and whilst it's easy enough to build a static function and add the *.jar to the IIB classpath several sites I've worked at have a blanket ban on using Java.

So what does an efficient string splitter in ESQL look like.

ibm-integration-bus extended-sql

asked May 12 at 7:25

TJA

1,381 1 15 23

1 Answer

2

The following four variants on a theme can be used to split an ESQL string.

Rather than add lots of parameters and wind up with fairly convoluted internal logic I chose the option of using the function names instead of flags.

SplitString

Does not add empty strings but will add strings with more than one blank.

```
CREATE PROCEDURE SplitString(
    IN CompositeString CHAR,         -- Composite string that needs to be split
    IN Delimiter CHAR,               -- Delimiter to be used when splitting the
string
    IN ArrayName CHAR,               -- Name of the array for the results of the
function
    IN NewArray BOOLEAN,             -- Use TRUE to clear a pre-existing array,
FALSE appends new element
    IN EnvRef REFERENCE              -- Reference to Environment tree
)
BEGIN
    IF NewArray THEN
        DELETE FIELD EnvRef.SplitterArrays.{ArrayName};
    END IF;
```

```
    IF NOT LASTMOVE(SplitterArrayRef) THEN
        CREATE LASTCHILD OF EnvRef.SplitterArrays AS SplitterArrayRef NAME
ArrayName;
    END IF;

    WHILE LENGTH(Remainder) <> 0 DO
        IF POSITION(Delimiter IN Remainder) > 0 THEN
            DECLARE Element CHAR SUBSTRING(Remainder BEFORE Delimiter);
            IF LENGTH(Element) > 0 THEN
                CREATE LASTCHILD OF SplitterArrayRef NAME 'Element' VALUE
Element;
            END IF;

            SET Remainder = SUBSTRING(Remainder AFTER Delimiter);
        ELSE
            DECLARE Element CHAR Remainder;
            IF LENGTH(Element) > 0 THEN
                CREATE LASTCHILD OF SplitterArrayRef NAME 'Element' VALUE
Element;
            END IF;

            SET Remainder = '';
        END IF;
    END WHILE;
END;
```

SplitStringTrim

Trim leading and trailing blanks from the Element strings.
Does not add empty or blank strings.

```
CREATE PROCEDURE SplitStringTrim(
    IN CompositeString CHAR,         -- Composite string that needs to be split
    IN Delimiter CHAR,               -- Delimiter to be used when splitting the
string
    IN ArrayName CHAR,               -- Name of the array for the results of the
function
    IN NewArray BOOLEAN,             -- Use TRUE to clear a pre-existing array,
FALSE appends new element
    IN EnvRef REFERENCE              -- Reference to Environment tree
)
BEGIN
    IF NewArray THEN
        DELETE FIELD EnvRef.SplitterArrays.{ArrayName};
    END IF;

    DECLARE Element CHAR;
    DECLARE Remainder CHAR TRIM(CompositeString);

    DECLARE SplitterArrayRef REFERENCE TO EnvRef.SplitterArrays.{ArrayName};
    IF NOT LASTMOVE(SplitterArrayRef) THEN
        CREATE LASTCHILD OF EnvRef.SplitterArrays AS SplitterArrayRef NAME
ArrayName;
    END IF;

    WHILE LENGTH(Remainder) <> 0 DO
        IF POSITION(Delimiter IN Remainder) > 0 THEN
            DECLARE Element CHAR TRIM(SUBSTRING(Remainder BEFORE Delimiter));
```

```
            IF LENGTH(Element) > 0 THEN
                CREATE LASTCHILD OF SplitterArrayRef NAME 'Element' VALUE
Element;
            END IF;

            SET Remainder = SUBSTRING(Remainder AFTER Delimiter;
        ELSE
            DECLARE Element CHAR TRIM(Remainder);
            IF LENGTH(Element) > 0 THEN
                CREATE LASTCHILD OF SplitterArrayRef NAME 'Element' VALUE
Element;
            END IF;

            SET Remainder = '';
        END IF;
    END WHILE;
END;
```

SplitStringAddEmpty

Add empty elements to the SplitterArray, ensuring there is at least one element. Blanks are retained.

```
CREATE PROCEDURE SplitStringAddEmpty(
    IN CompositeString CHAR,         -- Composite string that needs to be split
    IN Delimiter CHAR,               -- Delimiter to be used when splitting the
string
    IN ArrayName CHAR,               -- Name of the array for the results of the
function
    IN NewArray BOOLEAN,             -- Use TRUE to clear a pre-existing array,
FALSE appends new element
    IN EnvRef REFERENCE              -- Reference to Environment tree
)
BEGIN
    IF NewArray THEN
        DELETE FIELD EnvRef.SplitterArrays.{ArrayName};
    END IF;

    DECLARE Element CHAR;
    DECLARE Remainder CHAR CompositeString;
    DECLARE EndsWithDelimiter BOOLEAN ENDSWITH(Remainder, Delimiter);

    DECLARE SplitterArrayRef REFERENCE TO EnvRef.SplitterArrays.{ArrayName};
    IF NOT LASTMOVE(SplitterArrayRef) THEN
        CREATE LASTCHILD OF EnvRef.SplitterArrays AS SplitterArrayRef NAME
ArrayName;
    END IF;

    IF LENGTH(Remainder) = 0 THEN
        CREATE LASTCHILD OF SplitterArrayRef NAME 'Element' VALUE '';
    ELSE
        WHILE LENGTH(Remainder) <> 0 DO
            IF POSITION(Delimiter IN Remainder) > 0 THEN
                CREATE LASTCHILD OF SplitterArrayRef NAME 'Element' VALUE
SUBSTRING(Remainder BEFORE Delimiter);

                SET Remainder = SUBSTRING(Remainder AFTER Delimiter);
            ELSE
```

```
                CREATE LASTCHILD OF SplitterArrayRef NAME 'Element' VALUE
Remainder;

                SET Remainder = '';
            END IF;
        END WHILE;

        IF EndsWithDelimiter THEN
            CREATE LASTCHILD OF SplitterArrayRef NAME 'Element' VALUE '';
        END IF;
    END IF;
END;
```

SplitStringAddEmptyTrim

Add empty elements to the SplitterArray, ensuring there is at least one element. Trim leading and trailing blanks from the Element strings.

```
CREATE PROCEDURE SplitStringAddEmptyTrim(
    IN CompositeString CHAR,         -- Composite string that needs to be split
    IN Delimiter CHAR,               -- Delimiter to be used when splitting the
string
    IN ArrayName CHAR,               -- Name of the array for the results of the
function
    IN NewArray BOOLEAN,             -- Use TRUE to clear a pre-existing array,
FALSE appends new element
    IN EnvRef REFERENCE              -- Reference to Environment tree
)
BEGIN
    IF NewArray THEN
        DELETE FIELD EnvRef.SplitterArrays.{ArrayName};
    END IF;

    DECLARE Element CHAR;
    DECLARE Remainder CHAR TRIM(CompositeString);
    DECLARE EndsWithDelimiter BOOLEAN ENDSWITH(Remainder, Delimiter);

    DECLARE SplitterArrayRef REFERENCE TO EnvRef.SplitterArrays.{ArrayName};
    IF NOT LASTMOVE(SplitterArrayRef) THEN
        CREATE LASTCHILD OF EnvRef.SplitterArrays AS SplitterArrayRef NAME
ArrayName;
    END IF;

    IF LENGTH(Remainder) = 0 THEN
        CREATE LASTCHILD OF SplitterArrayRef NAME 'Element' VALUE '';
    ELSE
        WHILE LENGTH(Remainder) <> 0 DO
            IF POSITION(Delimiter IN Remainder) > 0 THEN
                CREATE LASTCHILD OF SplitterArrayRef NAME 'Element' VALUE
TRIM(SUBSTRING(Remainder BEFORE Delimiter));

                SET Remainder = SUBSTRING(Remainder AFTER Delimiter);
            ELSE
                CREATE LASTCHILD OF SplitterArrayRef NAME 'Element' VALUE
TRIM(Remainder);

                SET Remainder = '';
            END IF;
```

```
            END WHILE;

            IF EndsWithDelimiter THEN
                CREATE LASTCHILD OF SplitterArrayRef NAME 'Element' VALUE '';
            END IF;
        END IF;
    END;
```

edited May 13 at 0:30

answered May 12 at 7:25

TJA

1,381 1 15 23