Expression Function Signatures

1 Introduction

The document contains an overview of the different signatures of the core Expression Engine functions to be supported. If not explicitly stated otherwise, the used terminology for the parameters is as defined next:

- ***expression****:* an expression of any of the supported data types: *Boolean*, *Byte*, *DateTime*, *Decimal*, *Double*, *Int16*, *Int32*, *Int64*, *Single*, *String*, *BLOB* and *CLOB.*
- ***expression_no_lob****:* an expression of any of the supported data types except *BLOB* and *CLOB*: *Boolean*, *Byte*, *DateTime*, *Decimal*, *Double*, *Int16*, *Int32*, *Int64*, *Single* and *String*.
- ***numeric_expression***: an expression of any of the supported numeric data types: *Decimal*, *Double*, *Int16*, *Int32*, *Int64* and *Single*.

If not explicitly stated otherwise, the used terminology for a function's return value is as defined next:

- ***diff_rv****:* the function returns a value of any of the supported data types: *Boolean*, *Byte*, *DateTime*, *Decimal*, *Double*, *Int16*, *Int32*, *Int64*, *Single*, *String*, *BLOB* and *CLOB.*
- ***input_data_type:*** the function returns the same data type as used for the parameter.
- ***numeric_rv***: the function returns a value of any of the supported numeric data types: *Decimal*, *Double*, *Int16*, *Int32*, *Int64* and *Single*.

The list of core Expression Engine functions to be supported can be grouped in different categories. The following sections provide an overview of the supported signatures based on those categories.

2 Aggregate Functions

Table 1 provides an overview of the supported signatures for the aggregate functions.

Function	Return Value Data Type	Signature
Avg	Double	Avg ([string,] numeric_expression)
Count	Int64	Count ([string,] expression)
Max	input_data_type	Max ([string,] expression_no_lob)
Median	input_data_type	Median (numeric_expression)
Min	input_data_type	Min ([string,] expression_no_lob)
Stddev	Double	Stddev (numeric_expression)
Sum	Double	Sum ([string,] numeric_expression)

Table 1: Overview of aggregate function signatures

If either of the function *Avg*, *Count*, *Max*, *Min* or *Sum* is used with the optional first parameter the value of this parameter must be either *All* or *Distinct* with *All* being the default value.

3 Conversion Functions

Table 2 provides an overview of the supported signatures for the conversion functions.

Function	Return Value Data Type	Signature
NullValue	diff_rv	NullValue (expression_no_lob1, expression_no_lob2)
ToDate	DateTime	ToDate (string [, format_description])
ToDouble	Double	ToDouble (numeric_expression) ToDouble (string)
ToFloat	Float	ToFloat (numeric_expression) ToFloat (string)
ToInt32	Int32	ToInt32 (numeric_expression) ToInt32 (string)
ToInt64	Int64	ToInt64 (numeric_expression) ToInt64 (string)
ToString	String	ToString (numeric_expression) ToString (date [, format_description])

Table 2: Overview of conversion function signatures

The function *ToDate* and ToString (to convert a date to a string) use a format specification. For details on this please see the corresponding section later in the document.

4 Date Functions

Table 3 provides an overview of the supported signatures for the date functions.

Function	Return Value Data Type	Signature
AddMonths	DateTime	AddMonths (date, numeric_expression)
CurrentDate	DateTime	CurrentDate ()
Extract	DateTime	Extract (string, date)
MonthsBetween	Double	MonthsBetween (date, date)

Table 3: Overview of date function signatures

5 Mathematical Functions

Table 4 provides an overview of the supported signatures for the mathematical functions.

Function	Return Value Data Type	Signature
Abs	input_data_type	Abs (numeric_expression)
Acos	Double	Acos (numeric_expression)
Asin	Double	Asin (numeric_expression)
Atan	Double	Atan (numeric_expression)
Atan2	Double	Atan2 (numeric_expression, numeric_expression)
Cos	Double	Cos (numeric_expression)
Exp	Double	Exp (numeric_expression)
Ln	Double	Ln (numeric_expression)
Log	Double	Log (numeric_expression, numeric_expression)
Mod	diff_rv	Mod (numeric_expression, numeric_expression)
Power	Double	Power (numeric_expression)
Remainder	diff_rv	Remainder (numeric_expression, numeric_expression)
Sin	Double	Sin (numeric_expression)
Sqrt	Double	Sqrt (numeric_expression)
Tan	Double	Tan (numeric_expression)

Table 4: Overview of mathematical function signatures

The functions *Mod* and *Remainder* both return the remainder of a division of two numbers. The difference between both is the used algorithm: whereas *Mod* uses the function *Floor* in its algorithm, *Remainder* uses the function *Round* instead. As a result, the returned remainder may differ. For example, the call *Mod (34.5, 3)* will return 1.5 whereas the call Remainder (34.5, 3) returns -1.5.

6 Numeric Functions

Table 5 provides an overview of the supported signatures for the numeric functions. The parameter *decimal_places* for the function *Round* can be of any numeric data type.

Function	Return Value Data Type	Signature
Ceil	input_data_type	Ceil (numeric_expression)
Floor	input_data_type	Floor (numeric_expression)
Round	input_data_type	Round (numeric_expression [, decimal_places])
Sign	Int16	Sign (numeric_expression)
Trunc	input_data_type	Trunc (numeric_expression [, numeric_expression])
		Trunc (date, string)

Table 5: Overview of numeric function signatures

The function *Sign* returns *-1* if the provided value is less than 0, *0* if the provided value is 0 and *1* if the provided value is bigger than 0.

The function Trunk is used to process numeric and date expressions. The parameter list and result of the request differs:

- If used to process a numeric expression, the number of digits after a decimal point is truncated to the number specified as the second parameter. For example, a call to *Trunc (123.456, 2)* returns *123.45* where the number of digits after the decimal point is dropped from 3 to 2 as indicated by the second function parameter.
- If used to process a date, the second parameter specified what to return. For example, a call to *Trunc (27-OCT-92, 'YEAR')* will return *01-JAN-92*. Supported values for the second parameter are *YEAR*, *MONTH*, *DAY*, *HOUR* and *MINUTE*.

7 String Functions

Table 6 provides an overview of the supported signatures for the string functions. The table uses the following expressions as defined next:

- ***source_string***: A string used to search in.
- ***search_string***: A string being searched in a source string
- ***pad_to_length***: A numeric expression identifying the length a string should be padded to.
- ***pad_string***: A string used to pad a given string to the requested length. The pad string can contain multiple characters.
- ***start_pos***: A numeric expression to identify a start position within a string to be used for a request.
- ***string_length***: The length of a string to be extracted from a given string starting at a given position.
- ***old_char_set***: A string of characters to be replaced.
- ***new_char_set***: A string of characters representing the replacements for the characters listed in the parameter *old_char_set*.
- ***trim_id***: The kind of operation expected by the function. This is a string and can be any of the following: *LEADING*, *TRAILING*, *BOTH*.

Function	Return Value Data Type	Signature
Concat	String	Concat (string, string)
Instr	Int64	Instr (source_string, search_string)
Length	Int64	Length (string)
Lower	String	Lower (string)
Lpad	String	Lpad (string, pad_to_length [, pad_string])
Ltrim	String	Ltrim (string)
Rpad	String	Rpad (string, pad_to_length [, pad_string])
Rtrim	String	Rtrim (string)
Soundex	String	Soundex (string)
Substr	String	Substr (string, start_pos [, sub_string_length])
Translate	String	Translate (string, old_char_set, new_char_set
Trim	String	Trim (string [, trim_id])
Upper	String	Upper (string)

Table 6: Overview of string function signatures

The functions *Instr* is 1-based and returns *0* if the search-string is not found within the given string.

8 Format Specifications

This section outlines the proposed format language for the functions *ToString* (to convert a date into a string) and *ToDate* (to convert a string to a date).

8.1 Function *ToDate*

This function takes a string value representing date and/or time information and converts it to a date object. The optional format specification parameter defines the format used in the string to represent the date. For example, for a string containing the date *April 2, 1998* the format specification should contain *Month DD, YYYY*. Table 7 provides an overview of the format specification language that is intended to be supported.

Abbreviation	Description
YY	Indicates that a year is defined as a two digit number (example: *07*).
YYYY	Indicates that a year is defined as a four digit number (example: *2007*).
MONTH	Indicates that a month is defined by its name, all in uppercase letters (example: *APRIL*).
month	Indicates that a month is defined by its name, all in lowercase letters (example: *april*).
Month	Indicates that a month is defined by its name with the first letter being an uppercase letter (example: *April*).
MON	Indicates that a month is defined by its abbreviation, all in uppercase letters (example: *APR*).
mon	Indicates that a month is defined by its abbreviation, all in lowercase letters (example: *apr*).
MM	Indicates that a month is defined by its number (example: *04*).
DAY	Indicates that a day is defined by its name, all in uppercase letters (example: *SUNDAY*).
day	Indicates that a day is defined by its name, all in lowercase letters (example: *sunday*).
Day	Indicates that a day is defined by its name with the first letter being an uppercase letter (example: *Sunday*).
DY	Indicates that a day is defined by its abbreviation, all in uppercase letters (example: *SUN*).
dy	Indicates that a day is defined by its abbreviation, all in uppercase letters (example: *SUN*).
DD	Indicates that a day is defined by its number (example: *04*).
hh24	Indicates that a hour is defined by its number in the range [0-24]
hh12	Indicates that a hour is defined by its number in the range [1-12]
hh	Represents a default representation of an hour. Defaults to hh24.
mm	Indicates a minute definition
ss	Indicates a second definition
ms	Indicates a millisecond definition
am | pm	The meridiem. Only considered if used with the time range [1-12] (format hh12).

Table 7: Format specification parameters

The original specification for the function signature has the format specification listed as an optional parameter. It should be considered making it a mandatory parameter as without the format information the converting a string containing date and/or time information into a date may not succeed.

Note that in the listed example, separators in the date string will be represented by itself in the format string. Therefore, a call to the function *ToDate (‘April 2, 1998’, ‘Month DD YYYY’)* may fail because the format specification misses the comma-separator. On the other side, the call *ToDate (‘April 2, 1998’, ‘month DD YYYY’)* where the format specification for the month (according to the format, the value *april* is expected) does not match the actual value (*April*), the conversion will still succeed.

8.2 Function *ToString* – Date Conversion

This function takes a date value and creates a representation of it in a string. The optional format specification parameter allows the user to define the structure of the string to be created. For example, if the date information is *1998-APR-02*, the format specification may request that a string with the value *April 2, 1998* is created.

The format instruction is build using the same date/time abbreviations as defined in table 7. Therefore, the format is open and any kind of combination is supported. Please note that this may mean that a RDBMS provider may not be able to use native (build-in) functions to execute the request and hence has to rely on the Expression Engine to do the job. This will result in a performance penalty.

The function will not support format constructs that – for example – return the number of a day or week within a year for a given date.

For the above example, if the call was *ToString (1998-APR-02, ‘MONTH DD, YY’)*, the returned string would have the value *APRIL 02, 98*.