

Domain Aggregate Conversion

The Sum, Average, Comma Separated List, Comma Separated Set, Count, First, Last, Maximum, Minimum, StandardDeviation and Variance functions can be selected from the combo box in the Field Wizard or through scripting.

| Scheme Script | JavaScript |
|-----------------------------|--|
| <i>(davg "fieldname")</i> | ⚡ Data.getAverage("FieldName").getValueOverGroup(); ⚡ Data.getAverage("FieldName").getValueOverAll(); |
| <i>(dcount)</i> | ⚡ Data.getRecordCount().getValueOverGroup(); ⚡ Data.getRecordCount().getValueOverAll(); |
| <i>(dmax "fieldname")</i> | ⚡ Data.getMax("FieldName ").getValueOverGroup(); ⚡ Data.getMax("FieldName ").getValueOverAll(); |
| <i>(dmin "fieldname")</i> | ⚡ Data.getMin("FieldName").getValueOverGroup(); ⚡ Data.getMin("FieldName").getValueOverAll(); |
| <i>(dsum "fieldname")</i> | ⚡ Data.getSum(" FieldName").getValueOverGroup(); ⚡ Data.getSum ("FieldName").getValueOverAll(); |
| <i>(dstdev "fieldname")</i> | ⚡ Data.getStandardDeviation("FieldName").getValueOverGroup(); ⚡ Data.getStandardDeviation("FieldName").getValueOverAll(); |
| <i>(dvar "fieldname")</i> | ⚡ Data.getVariance("FieldName").getValueOverGroup(); ⚡ Data.getVariance("FieldName").getValueOverAll(); |

Conversion Functions

| Scheme Script | JavaScript |
|-------------------------------------|------------------------------|
| <i>(to-boolean object)</i> | Boolean(object) |
| <i>(to-short object)</i> | String + 0; String * 1; |
| <i>(to-int object)</i> | parseInt(String [, base]) |
| <i>(to-long object)</i> | String + 0; String * 1; |
| <i>(to-float object)</i> | parseFloat(object) |
| <i>(to-double object)</i> | String + 0; String * 1; |
| <i>(to-hex-string object)</i> | Not Applicable in version 5. |
| <i>(to-binary-string object)</i> | Not Applicable in version 5. |
| <i>(to-octal-string object)</i> | Not Applicable in version 5. |
| <i>(to-date datestring pattern)</i> | Not Applicable in version 5. |
| <i>(bytes-to-image field)</i> | Not Applicable in version 5. |
| <i>(base64-to-image field)</i> | Not Applicable in version 5. |

Date/Time Functions

| Scheme Script | JavaScript |
|--|---|
| <i>(year date1)</i> | getFullYear(); |
| <i>(quarter-of-year date1)</i> | |
| <i>(month date1)</i> | getMonth(); |
| <i>(week-of-year date1)</i> | |
| <i>(week-of-month date1)</i> | |
| <i>(day-of-year date1)</i> | |
| <i>(day-of-month date1)</i> | |
| <i>(day-of-week date1)</i> | |
| <i>(hour-of-day date1)</i> | |
| <i>(hour date1)</i> | getHours(); |
| <i>(minute date1)</i> | getMinutes(); |
| <i>(second date1)</i> | getSeconds(); |
| <i>(millisecond date1)</i> | |
| <i>(now)</i> | new java.util.Date() |
| <i>(date-add date1 increment interval)</i> | <ul style="list-style-type: none"> ⚡ offsetDays(origDate, numDays); ⚡ offsetMonths(origDate, numMonths); ⚡ offsetYears(origDate, numYears); ⚡ offsetDate(origDate, numYears, numMonths, numDays); |
| <i>(date-diff date1 date2 scope)</i> | dateDiff (DateA, DateB, interval); |

General Functions

| Scheme Script | JavaScript |
|--|--|
| <i>(dlookup "dsname", "fieldname")</i> | Data.getString ("FieldName"); |
| <i>(obj "fieldname")</i> | Data.getObject ("FieldName"); |
| <i>(row-no)</i> | Data.getRecordIndex (); |
| <i>(row-count)</i> | Data.getRecordCount (); |
| <i>(page)</i> | #{#} |
| <i>(pages)</i> | #{##} |
| <i>(parameter-lookup "key")</i> | Properties.getProperty ("ParameterName"); |
| <i>(null-zero expression)</i> | |
| | |

Math Functions

The **Math** object is a top-level, built-in JavaScript object which can be accessed without using a constructor or calling a method. It also has static properties and methods for mathematical constants and functions. This means that you can refer to, say, the constant PI as **Math.PI**, and you can call the Tangent function with **Math.tan(x)**.

For more information on the Math function, you may refer to the following web link:-
<http://www.devguru.com/Technologies/ecmascript/quickref/math.html>

| Scheme Script | JavaScript |
|-----------------------------|--------------------------------------|
| (e) | Math.E |
| (pi) | Math.PI |
| (random) | Math.random() ; |
| (to-degrees radian) | |
| (to-radians degree) | |
| (log number) | Math.log (number); |
| (exp number) | Math.exp (number); |
| (abs number) | Math.abs (number); |
| (sin number) | Math.sin (number); |
| (cos number) | Math.cos (number); |
| (tan number) | Math.tan (number); |
| (asin number) | Math.asin (number); |
| (acos number) | Math.acos (number); |
| (atan number) | Math.atan (number); |
| (max number1 number2 ...) | Math.max (number1, number2); |
| (min number1 number2 ...) | Math.min (number1, number2); |
| (zero? number) | |
| (positive? number) | |
| (negative? number) | |
| (odd? number) | |
| (even? number) | |
| (quotient number1 number2) | parseInt (number1 / number2) |
| (remainder number1 number2) | var remainder = number1 % number2; |
| (round number) | Math.round (number) |
| (sqrt number) | Math.sqrt (number) |
| (expt number1 number2) | Math.pow (base, exponent) |

Program Flow Functions

| Scheme Script | JavaScript |
|---|---|
| (if <test> <consequent> <alternate>) | if (condition) { action1 } else { action2 } |
| (cond <clause1> <clause2> ...) | switch (expression){ case label1: statement1 break case label2: statement2 break default: statement3; } |

Text Functions

| Scheme Script | JavaScript |
|---|---|
| (char->integer <i>character</i>) | |
| (integer->char <i>number</i>) | |
| (string-length <i>string</i>) | String.length; |
| (format [<i>fieldname</i>] <i>pattern</i> <i>decimalplaces</i>) | <ul style="list-style-type: none"> ↙ Format.formatNumber(Number n, int decimalPlaces) ↙ Format.formatNumber(Number n, int decimalPlaces, String locale) ↙ Format.formatNumber(Number n, String pattern) ↙ Format.formatCurrency(Number n) ↙ Format.formatCurrency(Number n, String locale) ↙ Format.formatPercent(Number n, int decimalPlaces) ↙ Format.ormatPercent(Number n, String locale, int decimalPlaces) ↙ Format.formatDate(Date d, String pattern) ↙ Format.formatDate(Date d, String pattern, String locale) |
| (substring <i>string start end</i>) | String.substring(indexA, indexB) ; |
| (leftstring <i>string length</i>) | leftTrim (String); |
| (rightstring <i>string length</i>) | rightTrim (String); |
| (string-append <i>string1 string2 ...</i>) | StringA + StringB |
| (trim <i>string</i>) | trim (String) |
| (uppercase <i>string</i>) | String.toUpperCase(); |
| (lowercase <i>string</i>) | String.toLowerCase(); |

Data Cache Functions

For more information on the Cache Functions in version 5, please refer to the **Chapter 7: Elixir Report JavaScript Reference (DataCache and DataCacheManager)** of the User Manual shipped together with the Elixir Report Professional Edition.

| Scheme Script | JavaScript |
|---|---|
| (<i>create-data-cache-store</i>) | Not Applicable |
| (<i>destroy-data-cache-store</i>) | Not Applicable |
| (<i>clear-data-cache-store</i>) | Not Applicable |
| (<i>load-data-cache-sep</i> <i>nameparameters separator</i>) | DataCache.loadCache (String dsname, Properties props); |
| (<i>load-data-cache name</i> <i>parameters</i>) | Not Applicable |
| (<i>load-data-cache-default-jdbc</i> <i>nameparameters</i>) | DataCache getCache (Object cacheName); |
| (<i>dlookup-cache-row name</i> <i>fieldname row-number</i>) | DataCache.getObject (int rowIndex, String fieldName); |
| (<i>select-cache name</i>) | DataCache.filter (String fieldName, Object value); |
| (<i>dcursor-move row-number</i>) | DataCache.moveToRow (int rowIndex); |
| (<i>dcursor-look fieldname value</i>) | DataCache.getObject (int rowIndex, String fieldName); |
| (<i>dcursor-lookup fieldname</i>) | DataCache.getString (String fieldName) |

| | |
|--|------------------------------|
| <i>(load-data-cache-ejb-mtd datasourcename keyset keyvalueset)</i> | Not Applicable in version 5. |
| <i>(load-data-cache-ejb-home datasourcename valueset)</i> | Not Applicable in version 5. |

Variable Cache Functions

| Scheme Script | JavaScript |
|------------------------------|--|
| <i>(create-var-cache)</i> | No equivalent JavaScript. These functions will be supported under standard JavaScript. |
| <i>(destroy-var-cache)</i> | |
| <i>(clear-all-vars)</i> | |
| <i>(put-var name object)</i> | |
| <i>(get-var name)</i> | |
| <i>(remove-var name)</i> | |

Resource Bundles Functions

For more information on the usage of Resource Bundles function, please refer to the java.util Class ResourceBundle

| Scheme Script | JavaScript |
|-------------------------------|--|
| <i>(load-bundle)</i> | ResourceBundle myResources = ResourceBundle. getBundle ("MyResources", currentLocale); |
| <i>(load-bundle-template)</i> | ResourceBundle myResources = ResourceBundle. getBundle ("MyResources"); |
| <i>(resource-text)</i> | myResources. getString (String key); |
| <i>(resource-text-locale)</i> | Not Applicable in version 5. |

Note:

For more information on the JavaScript function supported in Elixir Report 5, please refer to the **Chapter 7: Elixir Report JavaScript** of the User Manual shipped together with the Elixir Report Professional Edition.