Поддержка Linq в 1С

# Операции сравнения

Поддерживаются следующие операции сравнения: равно, неравно, больше, больше или равно, меньше, меньше или равно

# Логические операции

Поддерживаются следующие логические операции: NOT, AND, OR.

!с => NOT c

c1 && c2 => c1 AND c2

c1 & c2 => c1 AND c2

c1 || c2 => c1 OR c2

c1 | c2 => c1 OR c2

# Проверка на NULL

value == null => value IS NULL

value != null => value IS NOT NULL

# Арифметические операции

Бинарные операции: +, -, \*, /

Унарная операция отрицания: -

# Операция проверки на тип

value is Type => value REFS DataSource(Type)

# Условие IN со списком значений

values.Contains(value) => value IN (values[0],…,values[N])

!values.Contains(value) => value NOT IN (values[0],…,values[N])

OneSSqlFunctions.In(value, values) => value IN (values[0],…,values[N])

! OneSSqlFunctions.In(value, values) => value NOT IN (values[0],…,values[N])

OneSSqlFunctions.InHierarchy(value, values) => value IN HIERARCHY (values[0],…,values[N])

! OneSSqlFunctions.InHierarchy(value, values) => value NOT IN HIERARCHY (values[0],…,values[N])

# Приведение типа

(bool)value => CAST (value AS BOOLEAN)

OneSSqlFunctions.ToBoolean(value) => CAST (value AS BOOLEAN)

(numberType)value => CAST (value AS NUMBER) , где numberType : short, int, long, float, double

OneSSqlFunctions.ToInt16 (value, length = null) => CAST (value AS NUMBER[(length)])

OneSSqlFunctions.ToInt32 (value, length = null) => CAST (value AS NUMBER[(length)])

OneSSqlFunctions.ToInt64 (value, length = null) => CAST (value AS NUMBER[(length)])

OneSSqlFunctions.ToSingle (value, length = null, precision = null) => CAST (value AS NUMBER[(length, [precision])])

OneSSqlFunctions.ToDouble (value, length = null, precision = null) => CAST (value AS NUMBER[(length, [precision])])

OneSSqlFunctions.ToDecimal(value, length = null, precision = null) => CAST (value AS NUMBER[(length, [precision])])

(string)value => CAST (value AS STRING)

OneSSqlFunctions.ToString(value, length = null) => CAST (value AS STRING[(length)])

(DateTime)value => CAST(value AS DATE)

OneSSqlFunctions.ToDateTime(value) => CAST (value AS DATE)

(RecordType)value => CAST(value AS tableName), где tableName := DataSourceName(RecordType)

OneSSqlFunctions.ToDataRecord(value, tableName) : OneSDataRecord => CAST(value AS tableName)

# Встроенные функции

value.Substring(position, length) => SUBSTRING(value, position, length)

value.Year => YEAR(value)

OneSSqlFunctions.GetQuarter(value) => QUARTER(value)

value.Month => MONTH(value)

value.DayOfYear => DAYOFYEAR(value)

value.Day => DAY(value)

OneSSqlFunctions.GetWeek(value) => WEEK(value)

value.DayOfWeek => DAYWEEK(value)

OneSSqlFunctions.GetDayWeek(value) => DAYWEEK(value)

value.Hour => HOUR(value)

value.Minute => MINUTE(value)

value.Second => SECOND(value)

OneSSqlFunctions.BeginOfPeriod(value, OneSTimePeriodKind) => BEGINOFPERIOD(value, KIND)

OneSSqlFunctions.EndOfPeriod(value, OneSTimePeriodKind) => ENDOFPERIOD(value, KIND)

# Агрегируемые функции

Select(linqExpression).Sum() => SUM(sqlExpression)

Select(linqExpression).Average() => AVG(sqlExpression)

Select(linqExpression).Max() => MAX(sqlExpression)

Select(linqExpression).Min() => MIN(sqlExpression)

Count() => COUNT(\*)

Select(linqExpression).Count() => COUNT(sqlExpression)

Select(linqExpression).Distinct().Count() => COUNT(DISTINCT sqlExpression)

# Методы последовательности

Distinct() => DISTINCT

Take(n) => TOP n

# Прочие выражения

OneSSqlFunctions.Like(linqExpression, pattern[, escapeSymbol]) => sqlExpression LIKE pattern [ESCAPE escapeSymbol]

OneSSqlFunctions.Between(linqExpression, linqExpressionStart, linqExpressionEnd) =>

sqlExpression BETWEEN sqlExpressionStart AND sqlExpressionEnd

linqExpression1 ?? linqExpression2 =>

CASE WHEN sqlExpression1 IS NULL THEN sqlExpression2 ELSE sqlExpression1

linqCondition ? linqExpression1 : linqExpression2 =>

CASE WHEN sqlCondition THEN sqlExpression1 ELSE sqlExpression2