This document contains ODBC scalar functions with the details of each function and its ODBC compatibility version.

ODBC scalar functions are categorized as follows:

String Functions

Function	Description
ASCII(*string_exp*) (ODBC 1.0)	Returns the ASCII code value of the leftmost character of *string_exp* as an integer.
BIT_LENGTH(*string_exp*) (ODBC 3.0)	Returns the length in bits of the string expression. Does not work only for string data types, therefore will not implicitly convert *string_exp* to string but instead will return the (internal) size of whatever datatype it is given.
CHAR(*code*) (ODBC 1.0)	Returns the character that has the ASCII code value specified by *code*. The value of *code* should be between 0 and 255; otherwise, the return value is data source–dependent.
CHAR_LENGTH(*string_exp*) (ODBC 3.0)	Returns the length in characters of the string expression, if the string expression is of a character data type; otherwise, returns the length in bytes of the string expression (the smallest integer not less than the number of bits divided by 8). (This function is the same as the CHARACTER_LENGTH function.)
CHARACTER_LENGTH(*string_exp*) (ODBC 3.0)	Returns the length in characters of the string expression, if the string expression is of a character data type; otherwise, returns the length in bytes of the string expression (the smallest integer not less than the number of bits divided by 8). (This function is the same as the CHAR_LENGTH function.)
CONCAT(*string_exp1*,*string_exp2*) (ODBC 1.0)	Returns a character string that is the result of concatenating *string_exp2* to *string_exp1*. The resulting string is DBMS-dependent. For example, if the column represented by *string_exp1* contained a NULL value, DB2 would return NULL but SQL Server would return the non-NULL string.
DIFFERENCE(*string_exp1*,*string_exp2*) (ODBC 2.0)	Returns an integer value that indicates the difference between the values returned by the SOUNDEX function for *string_exp1* and *string_exp2*.
INSERT(*string_exp1*, *start*, *length*, *string_exp2*) (ODBC 1.0)	Returns a character string where *length* characters have been deleted from *string_exp1*, beginning at *start*, and where *string_exp2* has been inserted into *string_exp,* beginning at *start*.
LCASE(*string_exp*) (ODBC 1.0)	Returns a string equal to that in *string_exp*, with all uppercase characters converted to lowercase.
LEFT(*string_exp*, *count*) (ODBC 1.0)	Returns the leftmost *count* characters of *string_exp*.
LENGTH(*string_exp*) (ODBC 1.0)	Returns the number of characters in *string_exp,* excluding trailing blanks. **LENGTH** only accepts strings. Therefore will implicitly convert *string_exp* to a string, and return the length of this string (not the internal size of the datatype).
LOCATE(*string_exp1*, *string_exp2*[, *start*]) (ODBC 1.0)	Returns the starting position of the first occurrence of *string_exp1* within *string_exp2*. The search for the first occurrence of *string_exp1* begins with the first character position in *string_exp2* unless the optional argument, *start*, is specified. If *start* is specified, the search begins with the character position indicated by the value of *start*. The first character position in *string_exp2* is indicated by the value 1. If *string_exp1* is not found within *string_exp2*, the value 0 is returned. If an application can call the LOCATE scalar function with the *string_exp1*, *string_exp2*, and *start* arguments, the driver returns SQL_FN_STR_LOCATE when **SQLGetInfo** is called with an *Option* of SQL_STRING_FUNCTIONS. If the application can call the LOCATE scalar function with only the *string_exp1* and *string_exp2* arguments, the driver returns SQL_FN_STR_LOCATE_2 when **SQLGetInfo** is called with an *Option* of SQL_STRING_FUNCTIONS. Drivers that support calling the LOCATE function with either two or three arguments return both SQL_FN_STR_LOCATE and SQL_FN_STR_LOCATE_2.

LTRIM(*string_exp*) (ODBC 1.0)	Returns the characters of *string_exp*, with leading blanks removed.
OCTET_LENGTH(*string_exp*) (ODBC 3.0)	Returns the length in bytes of the string expression. The result is the smallest integer not less than the number of bits divided by 8. Does not work only for string data types, therefore will not implicitly convert *string_exp* to string but instead will return the (internal) size of whatever datatype it is given.
POSITION(*character_exp* IN *character_exp*) (ODBC 3.0)	Returns the position of the first character expression in the second character expression. The result is an exact numeric with an implementation-defined precision and a scale of 0.
REPEAT(*string_exp, count*) (ODBC 1.0)	Returns a character string composed of *string_exp* repeated *count* times.
REPLACE(*string_exp1*, *string_exp2*, *string_exp3*) (ODBC 1.0)	Search *string_exp1* foroccurrences of *string_exp2*, and replace with *string_exp3*.
RIGHT(*string_exp*, *count*) (ODBC 1.0)	Returns the rightmost *count* characters of *string_exp*.
RTRIM(*string_exp*) (ODBC 1.0)	Returns the characters of *string_exp* with trailing blanks removed.
SOUNDEX(*string_exp*) (ODBC 2.0)	Returns a data source–dependent character string representing the sound of the words in *string_exp*. For example, SQL Server returns a 4-digit SOUNDEX code; Oracle returns a phonetic representation of each word.
SPACE(*count*) (ODBC 2.0)	Returns a character string consisting of *count* spaces.
SUBSTRING(*string_exp*, *start*, length) (ODBC 1.0)	Returns a character string that is derived from *string_exp*, beginning at the character position specified by *start* for *length* characters.
UCASE(*string_exp*) (ODBC 1.0)	Returns a string equal to that in *string_exp*, with all lowercase characters converted to uppercase.

Numeric Functions

Function	Description
ABS(*numeric_exp*) (ODBC 1.0)	Returns the absolute value of *numeric_exp*.
ACOS(*float_exp*) (ODBC 1.0)	Returns the arccosine of *float_exp* as an angle, expressed in radians.
ASIN(*float_exp*) (ODBC 1.0)	Returns the arcsine of *float_exp* as an angle, expressed in radians.
ATAN(*float_exp*) (ODBC 1.0)	Returns the arctangent of *float_exp* as an angle, expressed in radians.
ATAN2(*float_exp1*, *float_exp2*) (ODBC 2.0)	Returns the arctangent of the *x* and *y* coordinates, specified by *float_exp1* and *float_exp2*, respectively, as an angle, expressed in radians.
CEILING(*numeric_exp*) (ODBC 1.0)	Returns the smallest integer greater than or equal to *numeric_exp*. The return value is of the same data type as the input parameter.
COS(*float_exp*) (ODBC 1.0)	Returns the cosine of *float_exp*, where *float_exp* is an angle expressed in radians.
COT(*float_exp*) (ODBC 1.0)	Returns the cotangent of *float_exp*, where *float_exp* is an angle expressed in radians.
DEGREES(*numeric_exp*) (ODBC 2.0)	Returns the number of degrees converted from *numeric_exp* radians.
EXP(*float_exp*) (ODBC 1.0)	Returns the exponential value of *float_exp*.
FLOOR(*numeric_exp*) (ODBC 1.0)	Returns the largest integer less than or equal to *numeric_exp*. The return value is of the same data type as the input parameter.
LOG(*float_exp*) (ODBC 1.0)	Returns the natural logarithm of *float_exp*.
LOG10(*float_exp*) (ODBC 2.0)	Returns the base 10 logarithm of *float_exp*.
MOD(*integer_exp1*, *integer_exp2*) (ODBC 1.0)	Returns the remainder (modulus) of *integer_exp1* divided by *integer_exp2*.
PI( ) (ODBC 1.0)	Returns the constant value of pi as a floating-point value.
POWER(*numeric_exp*, *integer_exp*) (ODBC 2.0)	Returns the value of *numeric_exp* to the power of *integer_exp*.
RADIANS(*numeric_exp*) (ODBC 2.0)	Returns the number of radians converted from *numeric_exp* degrees.
RAND([*integer_exp*]) (ODBC 1.0)	Returns a random floating-point value using *integer_exp* as the optional seed value.
ROUND(*numeric_exp*, *integer_exp*) (ODBC 2.0)	Returns *numeric_exp* rounded to *integer_exp* places right of the decimal point. If *integer_exp* is negative, *numeric_exp* is rounded to |*integer_exp*| places to the left of the decimal point.
SIGN(*numeric_exp*) (ODBC 1.0)	Returns an indicator of the sign of *numeric_exp*. If *numeric_exp* is less than zero, –1 is returned. If *numeric_exp* equals zero, 0 is returned. If *numeric_exp* is greater than zero, 1 is returned.
SIN(*float_exp*) (ODBC 1.0)	Returns the sine of *float_exp*, where *float_exp* is an angle expressed in radians.
SQRT(*float_exp*) (ODBC 1.0)	Returns the square root of *float_exp*.
TAN(*float_exp*) (ODBC 1.0)	Returns the tangent of *float_exp*, where *float_exp* is an angle expressed in radians.
TRUNCATE(*numeric_exp*, *integer_exp*) (ODBC 2.0)	Returns *numeric_exp* truncated to *integer_exp* places right of the decimal point. If *integer_exp* is negative, *numeric_exp* is truncated to |*integer_exp*| places to the left of the decimal point.

Date Time and Interval Functions

Function	Description
CURRENT_DATE( ) (ODBC 3.0)	Returns the current date.
CURRENT_TIME[(*time-precision*)] (ODBC 3.0)	Returns the current local time. The *time-precision* argument determines the seconds precision of the returned value.
CURRENT_TIMESTAMP [(*timestamp-precision*)] (ODBC 3.0)	Returns the current local date and local time as a timestamp value. The *timestamp-precision* argument determines the seconds precision of the returned timestamp.
CURDATE( ) (ODBC 1.0)	Returns the current date.
CURTIME( ) (ODBC 1.0)	Returns the current local time.
DAYNAME(*date_exp*) (ODBC 2.0)	Returns a character string containing the data source–specific name of the day (for example, Sunday through Saturday or Sun. through Sat. for a data source that uses English, or Sonntag through Samstag for a data source that uses German) for the day portion of *date_exp*.
DAYOFMONTH(*date_exp*) (ODBC 1.0)	Returns the day of the month based on the month field in *date_exp* as an integer value in the range of 1–31.
DAYOFWEEK(*date_exp*) (ODBC 1.0)	Returns the day of the week based on the week field in *date_exp* as an integer value in the range of 1–7, where 1 represents Sunday.
DAYOFYEAR(*date_exp*) (ODBC 1.0)	Returns the day of the year based on the year field in *date_exp* as an integer value in the range of 1–366.
EXTRACT(*extract-field FROM extract-source*) (ODBC 3.0)	Returns the *extract-field* portion of the *extract-source*. The *extract-source* argument is a datetime or interval expression. The *extract-field* argument can be one of the following keywords: YEAR MONTH DAY HOUR MINUTE SECOND The precision of the returned value is implementation-defined. The scale is 0 unless SECOND is specified, in which case the scale is not less than the fractional seconds precision of the *extract-source* field.
HOUR(*time_exp*) (ODBC 1.0)	Returns the hour based on the hour field in *time_exp* as an integer value in the range of 0–23.
MINUTE(*time_exp*) (ODBC 1.0)	Returns the minute based on the minute field in *time_exp* as an integer value in the range of 0–59.
MONTH(*date_exp*) (ODBC 1.0)	Returns the month based on the month field in *date_exp* as an integer value in the range of 1–12.
MONTHNAME(*date_exp*) (ODBC 2.0)	Returns a character string containing the data source–specific name of the month (for example, January through December or Jan. through Dec. for a data source that uses English, or Januar through Dezember for a data source that uses German) for the month portion of *date_exp*.
NOW( ) (ODBC 1.0)	Returns current date and time as a timestamp value.
QUARTER(*date_exp*) (ODBC 1.0)	Returns the quarter in *date_exp* as an integer value in the range of 1–4, where 1 represents January 1 through March 31.
SECOND(*time_exp*) (ODBC 1.0)	Returns the second based on the second field in *time_exp* as an integer value in the range of 0–59. Returns the timestamp calculated by adding *integer_exp* intervals of type *interval* to *timestamp_exp*. Valid values of *interval* are the following keywords: SQL_TSI_FRAC_SECOND SQL_TSI_SECOND SQL_TSI_MINUTE SQL_TSI_HOUR SQL_TSI_DAY SQL_TSI_WEEK SQL_TSI_MONTH

	SQL_TSI_QUARTER SQL_TSI_YEAR where fractional seconds are expressed in billionths of a second. For example, the following SQL statement returns the name of each employee and his or her one-year anniversary date: SELECT NAME, {fn TIMESTAMPADD(SQL_TSI_YEAR, 1, HIRE_DATE)} FROM EMPLOYEES If *timestamp_exp* is a time value and *interval* specifies days, weeks, months, quarters, or years, the date portion of *timestamp_exp* is set to the current date before calculating the resulting timestamp. If *timestamp_exp* is a date value and *interval* specifies fractional seconds, seconds, minutes, or hours, the time portion of *timestamp_exp* is set to 0 before calculating the resulting timestamp. An application determines which intervals a data source supports by calling **SQLGetInfo** with the SQL_TIMEDATE_ADD_INTERVALS option.
TIMESTAMPDIFF(*interval*, *timestamp_exp1*, *timestamp_exp2*) (ODBC 2.0)	Returns the integer number of intervals of type *interval* by which *timestamp_exp2* is greater than *timestamp_exp1*. Valid values of *interval* are the following keywords: SQL_TSI_FRAC_SECOND SQL_TSI_SECOND SQL_TSI_MINUTE SQL_TSI_HOUR SQL_TSI_DAY SQL_TSI_WEEK SQL_TSI_MONTH SQL_TSI_QUARTER SQL_TSI_YEAR where fractional seconds are expressed in billionths of a second. For example, the following SQL statement returns the name of each employee and the number of years he or she has been employed: SELECT NAME, {fn TIMESTAMPDIFF(SQL_TSI_YEAR, {fn CURDATE()}, HIRE_DATE)} FROM EMPLOYEES If either timestamp expression is a time value and *interval* specifies days, weeks, months, quarters, or years, the date portion of that timestamp is set to the current date before calculating the difference between the timestamps. If either timestamp expression is a date value and *interval* specifies fractional seconds, seconds, minutes, or hours, the time portion of that timestamp is set to 0 before calculating the difference between the timestamps. An application determines which intervals a data source supports by calling **SQLGetInfo** with the SQL_TIMEDATE_DIFF_INTERVALS option.
WEEK(*date_exp*) (ODBC 1.0)	Returns the week of the year based on the week field in *date_exp* as an integer value in the range of 1–53.
YEAR(*date_exp*) (ODBC 1.0)	Returns the year based on the year field in *date_exp* as an integer value. The range is data source–dependent.

System Functions

Function	Description
DATABASE( ) (ODBC 1.0)	Returns the name of the database corresponding to the connection handle. (The name of the database is also available by calling **SQLGetConnectOption** with the SQL_CURRENT_QUALIFIER connection option.)
IFNULL(*exp*,*value*) (ODBC 1.0)	If *exp* is null, *value* is returned. If *exp* is not null, *exp* is returned. The possible data type or types of *value* must be compatible with the data type of *exp*.
USER( ) (ODBC 1.0)	Returns the user name in the DBMS. (The user name is also available by way of **SQLGetInfo** by specifying the information type: SQL_USER_NAME.) This can be different than the login name.

Explicit Data Type Conversion Function

CONVERT(*value_exp*, *data_type***)**

SQL-92 CAST Function

CAST function is the ANSI conversion function and Convert is the ODBC conversion function. So, those functions are used to do explicit conversion.

CAST (*value_exp* AS *data_type***)**

Following are the valid data type used in the explicit conversion functions Convert and Cast

SQL_BIGINT	SQL_INTERVAL_HOUR_TO_MINUTE
SQL_BINARY	SQL_INTERVAL_HOUR_TO_SECOND
SQL_BIT	SQL_INTERVAL_MINUTE_TO_SECOND
SQL_CHAR	SQL_LONGVARBINARY
SQL_DECIMAL	SQL_LONGVARCHAR
SQL_DOUBLE	SQL_NUMERIC
SQL_FLOAT	SQL_REAL
SQL_GUID	SQL_SMALLINT
SQL_INTEGER	SQL_DATE
SQL_INTERVAL_MONTH	SQL_TIME
SQL_INTERVAL_YEAR	SQL_TIMESTAMP
SQL_INTERVAL_YEAR_TO_MONTH	SQL_TINYINT
SQL_INTERVAL_DAY	SQL_VARBINARY
SQL_INTERVAL_HOUR	SQL_VARCHAR
SQL_INTERVAL_MINUTE	SQL_WCHAR
SQL_INTERVAL_SECOND	SQL_WLONGVARCHAR
SQL_INTERVAL_DAY_TO_HOUR	SQL_WVARCHAR
SQL_INTERVAL_DAY_TO_MINUTE	
SQL_INTERVAL_DAY_TO_SECOND	