Database Lab

CSE 3104

Session 10

ASCII:

Returns the ASCII value for the specific character.
Parameter: Character
Description: Required. The character to return the ASCII value for. If more than one character is entered, it will only return the value for the first character

```
SELECT ASCII('AUST');
```

It returns 65 as we know the ASCII value of 'A' is 65.

CHAR:

Returns the character based on the ASCII code.
Parameter: Code
Description: Required. The ASCII number code to return the character for.

```
SELECT CHAR(65) AS CodeToCharacter;
```

Return the character based on the number code 65.

CHARINDEX:

Returns the position of a substring in a string. If the substring is not found, this function returns 0. This function performs a case-insensitive search.

Parameter	Description
substring	**Required. The substring to search for**
string	**Required. The string to be searched**
start	**Optional. The position where the search will start (if you do not want to start at the beginning of *string*). The first position in *string* is 1**

.

```
SELECT CHARINDEX('OM', 'Customer') AS MatchPosition;
```
Search for "OM" in string "Customer", and return position.

SELECT CHARINDEX('us','Customer', 3) AS MatchPosition;
Search for "us" in string "Customer", and return position (start in position 3):

CONCAT:

The CONCAT() function adds two or more strings together.
Parameter: *string1, string2, string_n*
Description: Required. The strings to add together

SELECT CONCAT('Wait', ' ', 'for', ' ', 'it.', ' ', 'Legendary!!!');

Add strings together (separate each string with a space character).

CONCAT with +:

The + operator allows you to add two or more strings together.
Parameter: *string1, string2, string_n*
Description: Required. The strings to add together

SELECT 'Joey' + ' doesn't' + ' share' + ' food';

Add 4 strings together.

CONCAT_WS:

The CONCAT_WS() function adds two or more strings together with a separator.

Parameter	Description
separator	**Required. The separator to use**
string1,string2,string_n	**Required. The strings to add together**

SELECT CONCAT_WS('-', 'The', ' North', ' remembers');

Add 3 strings together with a separator between every two strings.

DATALENGTH:

Returns the number of bytes used to represent an expression.

Parameter: *expression*

Description: Required. The data type to return the length for. If *expression* is NULL, it returns NULL

```
SELECT DATALENGTH('   Winter is coming   ');
```

Return the length of an expression (in bytes) (counts both leading and trailing spaces).

DIFFERENCE:

The DIFFERENCE() function compares two SOUNDEX values, and returns an integer. The integer value indicates the match for the two SOUNDEX values, from 0 to 4.

0 indicates weak or no similarity between the SOUNDEX values. 4 indicates strong similarity or identically SOUNDEX values.

Parameter: *expression, expression*

Description: Required. Two expressions to be compared. Can be a constant, variable, or column

```
SELECT DIFFERENCE('Smith', 'Smyth');
```

Compares two SOUNDEX values, and return an integer.

LEFT:

The LEFT() function extracts a number of characters from a string (starting from left).

Parameter	Description
string	**Required. The string to extract from**

number_of_chars	**Required. The number of characters to extract. If the number exceeds the number of characters in *string*, it returns *string***

SELECT LEFT('Chandler', 4) AS ExtractString;
Extract 4 characters from a string (starting from left).

LEN:

The LEN() function returns the length of a string.
Trailing spaces at the end of the string is not included when calculating the length. However, leading spaces at the start of the string is included when calculating the length.
Parameter: *string*
Description: Required. The string to return the length for. If *string* is NULL, it returns NULL

SELECT LEN(' Walter White ');

LTRIM:

The LTRIM() function removes leading spaces from a string.
Parameter: *string*
Description: Required. The string to remove leading spaces from

SELECT LTRIM(' Thomas Shelby') AS LeftTrimmedString;

REPLACE:

The REPLACE() function replaces all occurrences of a substring within a string, with a new substring.

Parameter	Description
string	**Required. The original string**
old_string	**Required. The string to be replaced**
new_string	**Optional. The new replacement string**

```
SELECT REPLACE ('Dwight Schrute', 'Dwight', 'Michael');
SELECT REPLACE('Dexter', 'e', 'q');
```

REPLICATE:

The REPLICATE() function repeats a string a specified number of times.

Parameter	Description
string	**Required. The string to repeat**
integer	**Required. The number of times to repeat the string**

```
SELECT REPLICATE('Person_of_interest', 3);
```

REVERSE:

The REVERSE() function reverses a string and returns the result.

Parameter: *string*
Description: Required. The string to reverse

```
SELECT REVERSE ('Stranger Things');
```

STR:

The STR() function returns a number as a string.

Parameter	Description
number	**Required. The number to convert to a string**
length	**Optional. The length of the returning string. Default value is 10**
decimals	**Optional. The number of decimals to display in the returning string. Default value is 0**

```
SELECT STR(185.5);
SELECT STR(185.476, 6, 3);
```

STUFF:

The STUFF() function deletes a part of a string and then inserts another part into the string, starting at a specified position.

Parameter	Description
string	Required. The string to be modified
start	Required. The position in *string* to start to delete some characters
length	Required. The number of characters to delete from *string*
new_string	Required. The new string to insert into *string* at the *start* position

```
SELECT STUFF('SQL Tutorial!', 2, 5, ' is fun!');
```

SUBSTRING:

The SUBSTRING() function extracts some characters from a string.

Parameter	Description
string	Required. The string to extract from
start	Required. The start position. The first position in *string* is 1
length	Required. The number of characters to extract. Must be a positive number

```
SELECT SUBSTRING('VIKINGS', 1, 2) AS ExtractString;
```