SQL Report

Assigned Topic: string and dateTime and its functions

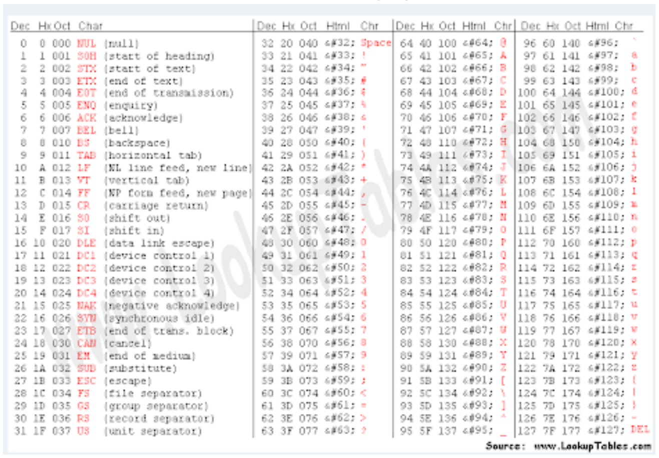
SQL String:

- A string function is a function that takes a string value as an input regardless of the data type of the returned value. There are multiple built-in string functions that can be used for string manipulation. The following are some of the most notable functions in SQL Server:

String Function Name	Description	Syntax Used	Example/Result

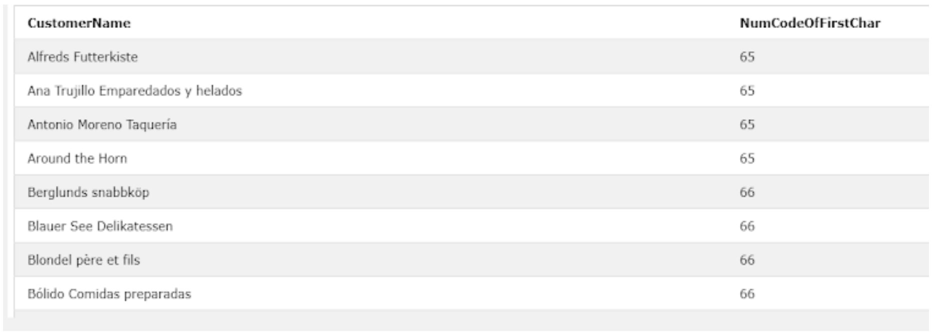
ASCII	Returns numeric value for the specific character.	`ASCII(`*`character`*`)`
		
		***If more than one character is entered, it will only return the value for the first character.
			`SELECT ASCII(CustomerName) AS NumCodeOfFirstChar FROM Customers;`
			
			`Result = prints a table of Customers Name and the ASCII char for the first letter of the customer's name.`

ASCII(American Standard Code for Information Interchange) Table:



Dec	Hx	Oct	Char		Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr
0	0	000	NUL	(null)	32	20	040	&#32;	Space	64	40	100	&#64;	@	96	60	140	&#96;	`
1	1	001	SOH	(start of heading)	33	21	041	&#33;	!	65	41	101	&#65;	A	97	61	141	&#97;	a
2	2	002	STX	(start of text)	34	22	042	&#34;	"	66	42	102	&#66;	B	98	62	142	&#98;	b
3	3	003	ETX	(end of text)	35	23	043	&#35;	#	67	43	103	&#67;	C	99	63	143	&#99;	c
4	4	004	EOT	(end of transmission)	36	24	044	&#36;	$	68	44	104	&#68;	D	100	64	144	&#100;	d
5	5	005	ENQ	(enquiry)	37	25	045	&#37;	%	69	45	105	&#69;	E	101	65	145	&#101;	e
6	6	006	ACK	(acknowledge)	38	26	046	&#38;	&	70	46	106	&#70;	F	102	66	146	&#102;	f
7	7	007	BEL	(bell)	39	27	047	&#39;	'	71	47	107	&#71;	G	103	67	147	&#103;	g
8	8	010	BS	(backspace)	40	28	050	&#40;	(	72	48	110	&#72;	H	104	68	150	&#104;	h
9	9	011	TAB	(horizontal tab)	41	29	051	&#41;	)	73	49	111	&#73;	I	105	69	151	&#105;	i
10	A	012	LF	(NL line feed, new line)	42	2A	052	&#42;	*	74	4A	112	&#74;	J	106	6A	152	&#106;	j
11	B	013	VT	(vertical tab)	43	2B	053	&#43;	+	75	4B	113	&#75;	K	107	6B	153	&#107;	k
12	C	014	FF	(NP form feed, new page)	44	2C	054	&#44;	,	76	4C	114	&#76;	L	108	6C	154	&#108;	l
13	D	015	CR	(carriage return)	45	2D	055	&#45;	-	77	4D	115	&#77;	M	109	6D	155	&#109;	m
14	E	016	SO	(shift out)	46	2E	056	&#46;	.	78	4E	116	&#78;	N	110	6E	156	&#110;	n
15	F	017	SI	(shift in)	47	2F	057	&#47;	/	79	4F	117	&#79;	O	111	6F	157	&#111;	o
16	10	020	DLE	(data link escape)	48	30	060	&#48;	0	80	50	120	&#80;	P	112	70	160	&#112;	p
17	11	021	DC1	(device control 1)	49	31	061	&#49;	1	81	51	121	&#81;	Q	113	71	161	&#113;	q
18	12	022	DC2	(device control 2)	50	32	062	&#50;	2	82	52	122	&#82;	R	114	72	162	&#114;	r
19	13	023	DC3	(device control 3)	51	33	063	&#51;	3	83	53	123	&#83;	S	115	73	163	&#115;	s
20	14	024	DC4	(device control 4)	52	34	064	&#52;	4	84	54	124	&#84;	T	116	74	164	&#116;	t
21	15	025	NAK	(negative acknowledge)	53	35	065	&#53;	5	85	55	125	&#85;	U	117	75	165	&#117;	u
22	16	026	SYN	(synchronous idle)	54	36	066	&#54;	6	86	56	126	&#86;	V	118	76	166	&#118;	v
23	17	027	ETB	(end of trans. block)	55	37	067	&#55;	7	87	57	127	&#87;	W	119	77	167	&#119;	w
24	18	030	CAN	(cancel)	56	38	070	&#56;	8	88	58	130	&#88;	X	120	78	170	&#120;	x
25	19	031	EM	(end of medium)	57	39	071	&#57;	9	89	59	131	&#89;	Y	121	79	171	&#121;	y
26	1A	032	SUB	(substitute)	58	3A	072	&#58;	:	90	5A	132	&#90;	Z	122	7A	172	&#122;	z
27	1B	033	ESC	(escape)	59	3B	073	&#59;	;	91	5B	133	&#91;	[	123	7B	173	&#123;	{
28	1C	034	FS	(file separator)	60	3C	074	&#60;	<	92	5C	134	&#92;	\	124	7C	174	&#124;	|
29	1D	035	GS	(group separator)	61	3D	075	&#61;	=	93	5D	135	&#93;	]	125	7D	175	&#125;	}
30	1E	036	RS	(record separator)	62	3E	076	&#62;	>	94	5E	136	&#94;	^	126	7E	176	&#126;	~
31	1F	037	US	(unit separator)	63	3F	077	&#63;	?	95	5F	137	&#95;	_	127	7F	177	&#127;	DEL

Source: www.LookupTables.com



CustomerName	NumCodeOfFirstChar
Alfreds Futterkiste	65
Ana Trujillo Emparedados y helados	65
Antonio Moreno Taquería	65
Around the Horn	65
Berglunds snabbköp	66
Blauer See Delikatessen	66
Blondel père et fils	66
Bólido Comidas preparadas	66

CHAR	Returns the character	`CHAR(`*`code`*`)`	`SELECT CHAR(122) AS`

	based on the ASCII code entered.	***The ASCII number code to return the character for is required.	`CodeToCharacter;`
			
			`Result = z`

CHARINDEX	Searches for a substring in a string and returns the position.	`CHARINDEX(`*`substring, string, start`*`)`
		
		***If the substring is not found it will return 0. If the substring contains multiple characters the first char in the substring will be the starting point or the value that will be returned.
			`SELECT CHARINDEX('s', 'Customer') AS MatchPosition;`
			
			`Result = 3`

CONCAT	Adds two or more strings together.	`CONCAT(`*`string1, string2, ...., string_n`*`)`
		
		***The strings to add together are required.
			`SELECT CONCAT('I', ' ', 'like', ' ', 'to', ' ', 'move it,', ' ', 'move it!');`
			
			`Result = I like to move it, move it!`

Concat with +	Adds two or more strings together using the + operator.	*`string1 + string2 + string_n`*
		
		***The strings to add together are required.
			`SELECT 'SQL' + ' is' + ' fun!';`
			
			`Result = SQL is fun!`

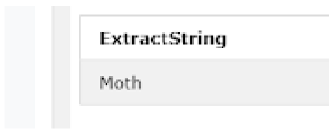
CONCAT_WS	Adds two or more strings together with a separator.	`CONCAT_WS(`*`separator, string1, string2, ...., string_n`*`)`
		
		***The separator to use and the string to add together is required.
			`SELECT CONCAT_WS('? ', 'Who', 'What', 'Where', 'When', 'Why', 'How', ' ');`
			
			`Result = Who? What? Where? When? Why? How?`

DATALENGTH	Returns the length of an expression (in bytes).	`DATALENGTH(`*`expression`*`)`
		
		***If the expression is NULL, it returns NULL.
			`SELECT DATALENGTH('YAAAAAAAAAAAAAAAAAAS');`
			
			`Result = 20`

DIFFERENCE	Compare two SOUNDEX values, and return a value.	`DIFFERENCE(`*`expression, expression`*`)`	`SELECT DIFFERENCE('Jam','James');`

	The integer value indicates the match for the two SOUNDEX values, from 0 (no similarity) to 4(strong similarities).	***Requires two expressions to be compared. Can be a constant, variable, or column.	`Result = 3`
			
			`SELECT DIFFERENCE('duck','monster');`
			
			`Result = 1`

FORMAT	Formats a value with the specified format for date/time.	`FORMAT(`*`value, format, culture`*`)`
		
		***General type conversions should use CAST() or CONVERT().
			`SELECT FORMAT(8675309000, '###-###-####');`
			
			`Result = 867 - 530 - 9000`

LEFT	Extracts a number of characters from a string (starting from left).	`LEFT(`*`string, number_of_chars`*`)`
		
		***A string and the number of characters to extract are required. If the number exceeds the number of characters in the string, it returns the string used.
			`SELECT LEFT('Mother Darling', 4) AS ExtractString;`
			
			`Result =`
			
			ExtractString
			Moth

LEN	Returns the length of a string.	`LEN(`*`string`*`)`
		
		***Trailing spaces at the end of the string are not included in total length, but the leading spaces at the start of the string are included. The string is required to return length.
			`SELECT LEN('        HiImPaul');`
			
			`Result = 17`

LOWER	Converts the text to lower-case.	`LOWER(`*`text`*`)`
		
		***The string is required to convert to lowercase.
			`SELECT LOWER('My wife TOLD ME to STOP IMPERSONATING a FLAMINGO. I HAD to put MY foot down. ');`
			
			`Result = my wife told me to stop impersonating a flamingo. i had to put my foot down.`

LTRIM	Removes leading spaces from a string.	`LTRIM(`*`string`*`)` ***The string is required to remove leading	`SELECT LTRIM('     Alright Alright Alright') AS LeftTrimmedString;`

		spaces from a string.	`Result = Alright Alright Alright`

NCHAR	Return the Unicode character based on the number code provided.	`NCHAR(`*`number_code`*`)`
		
		***The number code in the Unicode standard to return the character.
			`SELECT NCHAR(33) AS NumberCodeToUnicode;`
			
			`Result = !`

PANTINDEX	Return the position of a pattern in a string.	`PATINDEX(%`*`pattern`*`%, `*`string`*`)`
		
		***The pattern must be wrapped in % sign. If the pattern is not found, this function returns 0. The search is case-insensitive and the first position in the string is 1.
			`SELECT PATINDEX('%CanIGetAnAmen%', '111CanIGetAnAmen?111');`
			
			`Result = 4`

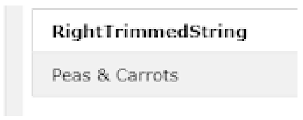
QUOTENAME	Returns a Unicode string with delimiters added to make the string a valid SQL Server delimited identifier.	`QUOTENAME(`*`string`*`, `*`quote_char`*`)`
		
		***A string of Unicode character data is required.
			`SELECT QUOTENAME('abcdef');`
			
			`Result = [abcdef]`
			
			`SELECT QUOTENAME('abcdef', '()');`
			
			`Result = (abcdef)`

REPLACE	Replaces all occurrences of a substring within a string, with a new substring.	`REPLACE(`*`string`*`, `*`old_string`*`, `*`new_string`*`)`
		
		***The original string, string to be replaced, and a new replacement string is required. The search is case-insensitive.
			`SELECT REPLACE('Napoleon give me your tots!', 'me', 'us');`
			
			`Result = Napoleon give us your tots!`

REPLICATE	Repeats a string a specified number of times.	`REPLICATE(`*`string`*`, `*`integer`*`)`
		
		***The string to repeat and the number of times to repeat the string is required.
			`SELECT REPLICATE('Hello Gorgeous! ', 3);`
			
			`Result = Hello Gorgeous! Hello Gorgeous! Hello Gorgeous!`

REVERSE	Reverses a string and returns the result.	REVERSE(*string*)
		
		***The string to reverse is required.
			SELECT REVERSE('No T no shade no Pink Lemonade!');
			
			Result = !edanomeL kniP on edahs on T oN

RIGHT	Extracts 3 characters from a string (starting from right).	RIGHT(*string*, *number_of_chars*)
		
		***The string to extract from and the number of characters to extract is required.
			SELECT RIGHT('Moms spaghetti', 5) AS ExtractString;
			
			Result = hetti

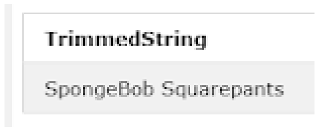
RTRIM	Removes trailing spaces from a string (starting from right).	RTRIM(*string*) ***The string to remove trailing spaces from is required.	SELECT RTRIM('Peas & Carrots ') AS RightTrimmedString; Result =  RightTrimmedString Peas & Carrots

SOUNDEX	Evaluates the similarity of two strings, and returns a four-character code based on how the string sounds when spoken.	SOUNDEX(*expression*)
		
		***The expression to evaluate is required. It can be a constant, variable, or column.
			SELECT SOUNDEX('RuPaul'), SOUNDEX('Arugula');
			
			Result = A624

Soundex code construction: ● The first character of the code is the first character of the string, converted to uppercase. ● The second through fourth characters of the code are numbers that represent the letters in the expression. ● The letters A, E, I, O, U, H, W, and Y are ignored unless they are the first letter of the string. ● Zeroes are added at the end if necessary to produce a four-character code. Other examples: Sure = S600, Water = W360, Coffee = C100, Flavor = F416, Hour = H600			

SPACE	Returns a string of the specified number of space characters.	SPACE(*number*)
		
		***The number of spaces to be returned is required.
			SELECT
			    first_name + SPACE(1) + last_name full_name
			FROM
			    sales.customers
			ORDER BY
			    first_name,

			`Last_name;` Result = Bronwyn Davies
STR	Returns a number as a string.	STR(*number*, *length*, *decimals*) ***The number to convert to a string.	`SELECT STR(185.123334);` Result = 185
STUFF	Deletes a part of a string and then inserts another part into the string, starting at a specific location.	STUFF(*string*, *start*, *length*, *new_string*) ***The string to be modified, the position in the string to start to delete some characters, the number of characters to delete from the string, and the new string to insert into the string at the start position is required.	`SELECT STUFF('Mary had a little lamb.', 1, 4, 'Rosie');` Result = Rosie had a little lamb.
SUBSTRING	Extracts some character from a string.	SUBSTRING(*string*, *start*, *length*) ***The string to extract from, the start position, and the number of characters to extract is required.	`SELECT SUBSTRING('I dance the beat to a different drummer.', 2, 4) AS ExtractString;` Result = dan
TRANSLATE	Returns a string from the first argument after the characters specified in the second argument are translated into the specified in the third argument.	TRANSLATE(*string*, *characters*, *translations*) ***The input string, the characters that should be replaced, and the new characters are required. Returns an error if characters and translators have different lengths.	`SELECT TRANSLATE('3*[2+1]/{8-4}', '[]{}', '()()');` Result = 3*(2+1)/(8-4)
TRIM	Removes the space	TRIM([*characters*	`SELECT TRIM('     SpongeBob`

	character OR other specified characters from the start or end of a string.	`FROM ]`*`string`*`)`
		
		***The string to remove spaces or characters from is required.
			`Squarepants    ') AS TrimmedString;`
			
			`Result =`
			
			TrimmedString
			SpongeBob Squarepants

UNICODE	Returns the Unicode value for the first character of the input expression	`UNICODE(character_ expression)`
		
		***Any length varchar can be placed in for the argument0
			`SELECT UNICODE('Hello World');`
			
			`Result:`
			
			`72`

UPPER	Converts a string to upper-case.	`Upper(text)`
		
		***Text is any length string to be converted to upper
			`Select UPPER('Hello World');`
			
			`Result:`
			
			`HELLO WORLD`

SQL DateTime:

- Date and time data types are used for values that contain date and time. Microsoft defines it as a date combined with a time of day with fractional seconds that is based on a 24-hour clock. The following are some of the most notable datetime functions:

Current_Timestamp	Returns the current date and time, in a 'YYYY-MM-DD hh:mm:ss.mmm' format	`CURRENT_TIMESTAMP`
		
		***No argument is needed
			`SELECT CURRENT_TIMESTAMP`
			
			`Results:`
			
			`2021-04-08 14:07:33.793`

DATEADD	Adds a time/date interval to a date and then returns the date	`DATEADD(`*`interval, number, date)`*
		
		***see below table for argument types
			`SELECT DATEADD(hour, 2, '2017/08/25');`
			
			`Result:`
			
			`2017-10-25 02:00:00.000`

DATEDIFF	Returns the difference between two dates	`DATEDIFF(interval, date1, date2)`
		
		***see below table for argument types
			`SELECT DATEDIFF(year, '2017/08/25', '2011/08/25');`
			
			`Result`
			
			`-6`

DATEFROMPARTS	Returns a date from the specified parts (year, month, and day values)	`DATEFROMPARTS(year, month, day)`
		
		***parameter values are:
		
		Year - Required. Specifies a year (4 digits)
		
		Month - Required. Specifies a month (from 1 to 12)
		
		Day - Required.
			`SELECT DATEFROMPARTS(2018, 10, 31);`
			
			`Result:`
			
			`2018-10-31`

		Specifies a day (from 1 to 31)	

DATENAME	Returns a specified part of a date as a string value.	`DATENAME(interval, date)`
		
		***see below table for parameter values
			`SELECT DATENAME(year, '2017/08/25');`
			
			`Result:`
			
			`2017`

DATEPART	Returns a specified part of a date as an integer value.	`DATEPART(interval, date)`
		
		***see below table for parameter values
			`SELECT DATEPART(year, '2017/08/25');`
			
			`Result:`
			
			`2017`

DAY	Returns the day of the month (from 1 to 31) for a specified date	`DAY(date)`
		
		***date Required. The date to return the day of the month from
			`SELECT DAY('2017/08/25');`
			
			`Result:`
			
			`DayOfMonth`
			`25`

GETDATE	Returns the current database system date and time, in a 'YYYY-MM-DD hh:mm:ss.mmm'	`GETDATE()`
		
		***No argument needed
			`SELECT GETDATE();`
			
			`Result:`
			
			`2021-04-08 14:38:26.097`

GETUTCDATE	Returns the current database system UTC date and time, in a 'YYYY-MM-DD hh:mm:ss.mm' format	`GETUTCDATE()`
		
		***No argument needed.
			`SELECT GETUTCDATE();`
			
			`Result:`
			
			`2021-04-08 13:43:14.617`

ISDATE	Checks an expression and returns 1 if it is a valid date, otherwise 0.	`ISDATE(expression)`
		
		***expression must be in the 'YYYY-MM-DD' format
			`SELECT ISDATE('2017-08-25');`
			
			`Result:`
			
			`1`

MONTH	Returns the month part for a specified date.	`MONTH(date)`
		
		***date must be in the 'YYYY-MM-DD' format
			`SELECT MONTH('2017/08/25');`
			
			`Result:`
			
			`8`

SYSDATETIME	Returns the date and time of the computer where the SQL Server is running.	`SYSDATETIME()`
		
		***No argument is needed
			`SELECT SYSDATETIME();`
			
			`Result:`
			
			`2021-04-08 14:48:28.8012549`

YEAR	Returns the year part for a specified date	`YEAR(date)`
		
		***date must be in the 'YYYY-MM-DD' format
			`SELECT YEAR('2017/08/25');`
			
			`Result:`
			
			`2017`

DATEADD - parameter values:

intervalRequired. The time/date interval to add. Can be one of the following values:
year, yyyy, yy = Year
quarter, qq, q = Quarter
month, mm, m = month
dayofyear, dy, y = Day of the year
day, dd, d = Day
week, ww, wk = Week
weekday, dw, w = Weekday
hour, hh = hour
minute, mi, n = Minute
second, ss, s = Second
millisecond, ms = Millisecond

number Required. The number of interval to add to date. Can be positive (to get dates in the future) or negative (to get dates in the past)

date Required. The date that will be modified

DATEDIFF - parameter values:

intervalRequired. The part to return. Can be one of the following values:
year, yyyy, yy = Year
quarter, qq, q = Quarter
month, mm, m = month
dayofyear = Day of the year
day, dy, y = Day
week, ww, wk = Week
weekday, dw, w = Weekday
hour, hh = hour
minute, mi, n = Minute
second, ss, s = Second
millisecond, ms = Millisecond

date1, date2 Required. The two dates to calculate the difference between

DATENAME - parameter values:

intervalRequired. The part to return. Can be one of the following values:
year, yyyy, yy = Year
quarter, qq, q = Quarter
month, mm, m = month
dayofyear = Day of the year
day, dy, y = Day
week, ww, wk = Week

weekday, dw, w = Weekday
hour, hh = hour
minute, mi, n = Minute
second, ss, s = Second
millisecond, ms = Millisecond

date	Required. The date to use

DATEPART - parameter values:
interval Required. The part to return. Can be one of the following values:
year, yyyy, yy = Year
quarter, qq, q = Quarter
month, mm, m = month
dayofyear, dy, y = Day of the year
day, dd, d = Day of the month
week, ww, wk = Week
weekday, dw, w = Weekday
hour, hh = hour
minute, mi, n = Minute
second, ss, s = Second
millisecond, ms = Millisecond

date	Required. The date to use

-

References:

SQL Server ASCII() Function
datetime (Transact-SQL) - SQL Server