CONTAINS

**(** { *column\_name* | **(***column\_list***)** | **\*** }

**,** **'**< contains\_search\_condition >**'**

                    [ **,** LANGUAGE *language\_term* ]

**)**

< contains\_search\_condition > ::=

    { < simple\_term >

    | < prefix\_term >

    | < generation\_term >

    | < proximity\_term >

    | < weighted\_term >

    }

    | { **(** < contains\_search\_condition > **)**

    [ { < AND > | < AND NOT > | < OR > } ]

< contains\_search\_condition > [ ...*n* ]

    }

< simple\_term > ::=

*word* | **"** *phrase* **"**

< prefix term > ::=

     { **"***word* **\*** **"** | **"***phrase* **\*"** }

< generation\_term > ::=

     FORMSOF **(** { INFLECTIONAL | THESAURUS } **,** < simple\_term > [ **,**...*n* ] **)**

< proximity\_term > ::=

     { < simple\_term > | < prefix\_term > }

     { { NEAR | ~ }

     { < simple\_term > | < prefix\_term > }

     } [ ...*n* ]

< weighted\_term > ::=

     ISABOUT

**(** { {

  < simple\_term >

  | < prefix\_term >

  | < generation\_term >

  | < proximity\_term >

  }

   [ WEIGHT **(** *weight\_value* **)** ]

   } [ **,**...*n* ]

**)**

< AND > ::=

     { AND | & }

< AND NOT > ::=

     { AND NOT | & !}

< OR > ::=

     { OR | | }

***CONTAINS - OR Logic***

In this query the terms 'nut', 'screw' or 'washer' are returned.  As a note, the '\*' in the first parameter of the CONTAINS statement indicates that all columns in the table registered with Full Text Search should be searched in this query.

|  |
| --- |
| USE AdventureWorks; GO SELECT \* FROM Production.Product WHERE CONTAINS(\*, '"\*nut\*" OR "\*screw\*" OR "\*washer\*"'); GO |

***CONTAINS - AND Logic***

In the first example query just the Name column is queried for the terms 'flat' and 'washer'.  In the second example, the term 'nut' is searched without 'hex', returning results such as 'chainring nuts', 'lock nuts', etc.  These are both examples of two independent terms being included in the search criteria.

|  |
| --- |
| -- AND Logic USE AdventureWorks; GO SELECT ProductID, [Name] FROM Production.Product WHERE CONTAINS([Name], '"flat" AND "washer"'); GO  -- AND NOT Logic USE AdventureWorks; GO SELECT ProductID, [Name] FROM Production.Product WHERE CONTAINS([Name], '"nut" AND NOT "hex"'); GO |

In this example, the only difference is that the phrase 'flat washer' is queried as opposed to two separate terms.

|  |
| --- |
| USE AdventureWorks; GO SELECT ProductID, [Name] FROM Production.Product WHERE CONTAINS([Name], '"flat washer"'); GO |

As a note for these two specific queries with the original data set, these two queries return the same results, but with a different data set, that may not be the case because the two independent terms could be more prevalent than the single phrase (combination of both terms).

***CONTAINS - Prefix***

In this example, the prefix 'chain' is queried which returns the word 'chain', but also matches on a single word like 'chainring' and the phrase 'chainring bolts'.

|  |
| --- |
| USE AdventureWorks; GO SELECT ProductID, [Name] FROM Production.Product WHERE CONTAINS([Name], '"chain\*"'); GO |

***CONTAINS - Proximity***

In the first example, the NEAR keyword is used to find the word 'men' near 'shorts'.  In the second example, three words are used.  As a point of reference, the order of the words does not seem to generate different result sets.

|  |
| --- |
| -- 2 terms USE AdventureWorks; GO SELECT ProductID, [Name] FROM Production.Product WHERE CONTAINS([Name], 'men NEAR shorts'); GO  -- 3 terms USE AdventureWorks; GO SELECT ProductID, [Name] FROM Production.Product WHERE CONTAINS([Name], 'XL NEAR men NEAR shorts'); GO |

***CONTAINS - Inflection and Thesaurus***

The first example is inflectional which means that the various forms of 'shift' will be returned, such as 'shifted', 'shifting', etc. The second example is based on the SQL Server Thesaurus setup in the SQL\_Server\_install\_path\Microsoft SQL Server\MSSQL.1\MSSQL\FTDATA\ directory.  These are a set of XML files that can be customized with synonyms for terms specific to your application.

|  |
| --- |
| -- INFLECTIONAL USE AdventureWorks; GO SELECT Description FROM Production.ProductDescription WHERE CONTAINS(Description, ' FORMSOF (INFLECTIONAL, shift) '); GO  -- THESAURUS USE AdventureWorks; GO SELECT Description FROM Production.ProductDescription WHERE CONTAINS(Description, ' FORMSOF (THESAURUS, wash) '); GO |

***CONTAINS - Weight***

In this example, the weight of the terms 'nut', 'bolt' and 'washer' are selected based on the weight in that order.

|  |
| --- |
| USE AdventureWorks; GO SELECT ProductID, [Name] FROM Production.Product WHERE CONTAINS([Name], 'ISABOUT (nut weight (.8),  bolt weight (.4), washer weight (.2) )' ); GO |

***CONTAINS - Variable***

In the two examples below, variable strings are passed into the CONTAINS command.  Based on my testing, as long as the strings have the proper syntax then just about anything can be passed with the variable, which makes the programming a little easier from front end search interfaces.

|  |
| --- |
| -- Example 1 USE AdventureWorks; GO DECLARE @Parm1 varchar(50) SET @Parm1 = 'XL NEAR men NEAR shorts' SELECT ProductID, [Name] FROM Production.Product WHERE CONTAINS([Name], @Parm1); GO  -- Example 2 USE AdventureWorks; GO DECLARE @Parm1 varchar(50) SET @Parm1 = '"XL" OR "men" OR "shorts"' SELECT ProductID, [Name] FROM Production.Product WHERE CONTAINS([Name], @Parm1); GO |