LINQ - QUERY OPERATORS

<https://www.tutorialspoint.com/linq/linq_query_operators.htm>

Copyright © tutorialspoint.com

A set of extension methods forming a query pattern is known as LINQ Standard Query Operators. As building blocks of LINQ query expressions, these operators offer a range of query capabilities like filtering, sorting, projection, aggregation, etc.

LINQ standard query operators can be categorized into the following ones on the basis of their functionality.

* Filtering Operators
* Join Operators
* Projection Operations
* Sorting Operators
* Grouping Operators
* Conversions
* Concatenation
* Aggregation
* Quantifier Operations
* Partition Operations
* Generation Operations
* Set Operations
* Equality
* Element Operators

**Filtering Operators**

Filtering is an operation to restrict the result set such that it has only selected elements satisfying a particular condition.

[Show Examples](https://www.tutorialspoint.com/linq/linq_filtering_operators.htm)

|  |  |  |  |
| --- | --- | --- | --- |
| **Operator** | **Description** | **C# Query Expression Syntax** | **VB Query Expression Syntax** |
| where | Filter values based on a predicate function | where | Where |
| OfType | Filter values based on their ability to be as a specified type | Not Applicable | Not Applicable |

**Join Operators**

Joining refers to an operation in which data sources with difficult to follow relationships with each other in a direct way are targeted.

[Show Examples](https://www.tutorialspoint.com/linq/linq_join_operators.htm)

|  |  |  |  |
| --- | --- | --- | --- |
| **Operator** | **Description** | **C# Query Expression Syntax** | **VB Query Expression Syntax** |
| Join | The operator join two sequences on basis of matching keys | join … in … on … equals … | From x In …, y In … Where x.a = y.a |
| GroupJoin | Join two sequences and group the matching elements | join … in … on … equals … into … | Group Join … In … On … |

**Projection Operations**

Projection is an operation in which an object is transformed into an altogether new form with only specific properties.

[Show Examples](https://www.tutorialspoint.com/linq/linq_projection_operations.htm)

|  |  |  |  |
| --- | --- | --- | --- |
| **Operator** | **Description** | **C# Query Expression Syntax** | **VB Query Expression Syntax** |
| Select | The operator projects values on basis of a transform function | select | Select |
| SelectMany | The operator project the sequences of values which are based on a transform function as well as flattens them into a single sequence | Use multiple from clauses | Use multiple From clauses |

**Sorting Operators**

A sorting operation allows ordering the elements of a sequence on basis of a single or more attributes.

[Show Examples](https://www.tutorialspoint.com/linq/linq_sorting_operators.htm)

|  |  |  |  |
| --- | --- | --- | --- |
| **Operator** | **Description** | **C# Query Expression Syntax** | **VB Query Expression Syntax** |
| OrderBy | The operator sort values in an ascending order | orderby | Order By |
| OrderByDescending | The operator sort values in a descending order | orderby ... descending | Order By ... Descending |
| ThenBy | Executes a secondary sorting in an ascending order | orderby …, … | Order By …, … |
| ThenByDescending | Executes a secondary sorting in a descending order | orderby …, … descending | Order By …, … Descending |
| Reverse | Performs a reversal of the order of the elements in a collection | Not Applicable | Not Applicable |

**Grouping Operators**

The operators put data into some groups based on a common shared attribute.

[Show Examples](https://www.tutorialspoint.com/linq/linq_grouping_operators.htm)

|  |  |  |  |
| --- | --- | --- | --- |
| **Operator** | **Description** | **C# Query Expression Syntax** | **VB Query Expression Syntax** |
| GroupBy | Organize a sequence of items in groups and return them as an IEnumerable collection of type IGrouping<key, element> | group … by -or- group … by … into … | Group … By … Into … |
| ToLookup | Execute a grouping operation in which a sequence of key pairs are returned | Not Applicable | Not Applicable |

**Conversions**

The operators change the type of input objects and are used in a diverse range of applications.

[Show Examples](https://www.tutorialspoint.com/linq/linq_conversions.htm)

|  |  |  |  |
| --- | --- | --- | --- |
| **Operator** | **Description** | **C# Query Expression Syntax** | **VB Query Expression Syntax** |
| AsEnumerable | Returns the input typed as IEnumerable<T> | Not Applicable | Not Applicable |
| AsQueryable | A genericgeneric IEnumerable is converted to a genericgenericIQueryable | Not Applicable | Not Applicable |
| Cast | Performs casting of elements of a collection to a specified type | Use an explicitly typed range variable. Eg:from string str in words | From … As … |
| OfType | Filters values on basis of their , depending on their capability to be cast to a particular type | Not Applicable | Not Applicable |
| ToArray | Forces query execution and does conversion of a collection to an array | Not Applicable | Not Applicable |
| ToDictionary | On basis of a key selector function set elements into a Dictionary<TKey, TValue> and forces execution of a LINQ query | Not Applicable | Not Applicable |
| ToList | Forces execution of a query by converting a collection to a List<T> | Not Applicable | Not Applicable |
| ToLookup | Forces execution of a query and put elements into a Lookup<TKey, TElement> on basis of a key selector function | Not Applicable | Not Applicable |

**Concatenation**

Performs concatenation of two sequences and is quite similar to the Union operator in terms of its operation except of the fact that this does not remove duplicates.

[Show Examples](https://www.tutorialspoint.com/linq/linq_concatenation.htm)

|  |  |  |  |
| --- | --- | --- | --- |
| **Operator** | **Description** | **C# Query Expression Syntax** | **VB Query Expression Syntax** |
| Concat | Two sequences are concatenated for the formation of a single one sequence. | Not Applicable | Not Applicable |

**Aggregation**

Performs any type of desired aggregation and allows creating custom aggregations in LINQ.

[Show Examples](https://www.tutorialspoint.com/linq/linq_aggregation.htm)

|  |  |  |  |
| --- | --- | --- | --- |
| **Operator** | **Description** | **C# Query Expression Syntax** | **VB Query Expression Syntax** |
| Aggregate | Operates on the values of a collection to perform custom aggregation operation | Not Applicable | Not Applicable |
| Average | Average value of a collection of values is calculated | Not Applicable | Aggregate … In … Into Average |
| Count | Counts the elements satisfying a predicate function within collection | Not Applicable | Aggregate … In … Into Count |
| LonCount | Counts the elements satisfying a predicate function within a huge collection | Not Applicable | Aggregate … In … Into LongCount |
| Max | Find out the maximum value within a collection | Not Applicable | Aggregate … In … Into Max |
| Min | Find out the minimum value existing within a collection | Not Applicable | Aggregate … In … Into Min |
| Sum | Find out the sum of a values within a collection | Not Applicable | Aggregate … In … Into Sum |

**Quantifier Operations**

These operators return a Boolean value i.e. True or False when some or all elements within a sequence satisfy a specific condition.

[Show Examples](https://www.tutorialspoint.com/linq/linq_quantifier_operations.htm)

|  |  |  |  |
| --- | --- | --- | --- |
| **Operator** | **Description** | **C# Query Expression Syntax** | **VB Query Expression Syntax** |
| All | Returns a value ‘True’ if all elements of a sequence satisfy a predicate condition | Not Applicable | Aggregate … In … Into All…… |
| Any | Determines by searching a sequence that whether any element of the same satisfy a specified condition | Not Applicable | Aggregate … In … Into Any |
| Contains | Returns a ‘True’ value if finds that a specific element is there in a sequence if the sequence doe not contains that specific element , ‘false’ value is returned | Not Applicable | Not Applicable |

**Partition Operators**

Divide an input sequence into two separate sections without rearranging the elements of the sequence and then returning one of them.

[Show Examples](https://www.tutorialspoint.com/linq/linq_partition_operators.htm)

|  |  |  |  |
| --- | --- | --- | --- |
| **Operator** | **Description** | **C# Query Expression Syntax** | **VB Query Expression Syntax** |
| Skip | Skips some specified number of elements within a sequence and returns the remaining ones | Not Applicable | Skip |
| SkipWhile | Same as that of Skip with the only exception that number of elements to skip are specified by a Boolean condition | Not Applicable | Skip While |
| Take | Take a specified number of elements from a sequence and skip the remaining ones | Not Applicable | Take |
| TakeWhile | Same as that of Take except the fact that number of elements to take are specified by a Boolean condition | Not Applicable | Take While |

**Generation Operations**

A new sequence of values is created by generational operators.

[Show Examples](https://www.tutorialspoint.com/linq/linq_generation_operations.htm)

|  |  |  |  |
| --- | --- | --- | --- |
| **Operator** | **Description** | **C# Query Expression Syntax** | **VB Query Expression Syntax** |
| DefaultIfEmpty | When applied to an empty sequence, generate a default element within a sequence | Not Applicable | Not Applicable |
| Empty | Returns an empty sequence of values and is the most simplest generational operator | Not Applicable | Not Applicable |
| Range | Generates a collection having a sequence of integers or numbers | Not Applicable | Not Applicable |
| Repeat | Generates a sequence containing repeated values of a specific length | Not Applicable | Not Applicable |

**Set Operations**

There are four operators for the set operations, each yielding a result based on different criteria.

[Show Examples](https://www.tutorialspoint.com/linq/linq_set_operations.htm)

|  |  |  |  |
| --- | --- | --- | --- |
| **Operator** | **Description** | **C# Query Expression Syntax** | **VB Query Expression Syntax** |
| Distinct | Results a list of unique values from a collection by filtering duplicate data if any | Not Applicable | Distinct |
| Except | Compares the values of two collections and return the ones from one collection who are not in the other collection | Not Applicable | Not Applicable |
| Intersect | Returns the set of values found t be identical in two separate collections | Not Applicable | Not Applicable |
| Union | Combines content of two different collections into a single list that too without any duplicate content | Not Applicable | Not Applicable |

**Equality**

Compares two sentences enumerableenumerable and determine are they an exact match or not.

[Show Examples](https://www.tutorialspoint.com/linq/linq_equality.htm)

|  |  |  |  |
| --- | --- | --- | --- |
| **Operator** | **Description** | **C# Query Expression Syntax** | **VB Query Expression Syntax** |
| SequenceEqual | Results a Boolean value if two sequences are found to be identical to each other | Not Applicable | Not Applicable |

**Element Operators**

Except the DefaultIfEmpty, all the rest eight standard query element operators return a single element from a collection.

[Show Examples](https://www.tutorialspoint.com/linq/linq_element_operators.htm)

|  |  |  |  |
| --- | --- | --- | --- |
| **Operator** | **Description** | **C# Query Expression Syntax** | **VB Query Expression Syntax** |
| ElementAt | Returns an element present within a specific index in a collection | Not Applicable | Not Applicable |
| ElementAtOrDefault | Same as ElementAt except of the fact that it also returns a default value in case the specific index is out of range | Not Applicable | Not Applicable |
| First | Retrieves the first element within a collection or the first element satisfying a specific condition | Not Applicable | Not Applicable |
| FirstOrDefault | Same as First except the fact that it also returns a default value in case there is no existence of such elements | Not Applicable | Not Applicable |
| Last | Retrieves the last element present in a collection or the last element satisfying a specific condition | Not Applicable | Not Applicable |
| LastOrDefault | Same as Last except the fact that it also returns a default value in case there is no existence of any such element | Not Applicable | Not Applicable |
| Single | Returns the lone element of a collection or the lone element that satisfy a certain condition | Not Applicable | Not Applicable |
| SingleOrDefault | Same as Single except that it also returns a default value if there is no existence of any such lone element | Not Applicable | Not Applicable |
| DefaultIfEmpty | Returns a default value if the collection or list is empty or null | Not Applicable | Not Applicable |