

# Namespace DynamicQueryBuilder

## Classes

### [DynamicQueryBuilderModule](#)

Module for adding Dynamic Query Builder services.

# Class DynamicQueryBuilderModule

Namespace: [DynamicQueryBuilder](#)

Assembly: DynamicQueryBuilder.dll








Module for adding Dynamic Query Builder services.

```
public static class DynamicQueryBuilderModule
```

## Inheritance

[object](#)  ← DynamicQueryBuilderModule

## Inherited Members

[object.Equals\(object\)](#)  , [object.Equals\(object, object\)](#)  , [object.GetHashCode\(\)](#)  ,  
[object.GetType\(\)](#)  , [object.MemberwiseClone\(\)](#)  , [object.ReferenceEquals\(object, object\)](#)  ,  
[object.ToString\(\)](#) 

## Methods

### AddDynamicQueryBuilder(IServiceCollection, DynamicQueryBuilderSettings)

Adds Dynamic Query Builder services to the service collection.

```
public static IServiceCollection AddDynamicQueryBuilder(this IServiceCollection  
services, DynamicQueryBuilderSettings settings)
```

## Parameters

**services** [IServiceCollection](#) 

The service collection.

**settings** [DynamicQueryBuilderSettings](#)

A [DynamicQueryBuilderSettings](#) object containing the connection string and database driver definition. Example usage:

```
var settings = new DynamicQueryBuilderSettings
{
    ConnectionString = "example",
    Driver = DatabaseDriver.POSTGRESQL
};
builder.Services.AddDynamicQueryBuilder(settings);
```

## Returns

[ICollection](#)

The service collection.

# Namespace DynamicQueryBuilder.Builders

## Classes

### [QueryBuilder](#)

Implements the IQueryBuilder interface to build SQL queries fluently.

# Class QueryBuilder


Namespace: [DynamicQueryBuilder.Builders](#)

Assembly: DynamicQueryBuilder.dll

Implements the IQueryBuilder interface to build SQL queries fluently.

```
public class QueryBuilder : IQueryBuilderSetup, IQueryBuilder
```








## Inheritance

[object](#)  ← QueryBuilder

## Implements

[IQueryBuilderSetup](#), [IQueryBuilder](#)

## Inherited Members

[object.Equals\(object\)](#)  , [object.Equals\(object, object\)](#)  , [object.GetHashCode\(\)](#)  ,  
[object.GetType\(\)](#)  , [object.MemberwiseClone\(\)](#)  , [object.ReferenceEquals\(object, object\)](#)  ,  
[object.ToString\(\)](#) 

## Methods

### Columns(params string[])

Specifies the columns to select in the query.

```
public IQueryBuilder Columns(params string[] columns)
```

## Parameters

**columns** [string](#) []

An array of column names.

## Returns

[IQueryBuilder](#)

The current IQueryBuilder instance.

# Create()

Factory method to start the builder.

```
public static IQueryableSetup Create()
```

Returns

[IQueryBuilderSetup](#)

A [QueryBuilder](#) stance.

# FilterBy(string, ComparisonOperators, string)

Adds a filter condition to the query using a comparison operator with AND.

```
public IQueryable FilterBy(string column, ComparisonOperators @operator,  
string value)
```

Parameters

column [string](#)<sup>↗</sup>

The column name.

operator [ComparisonOperators](#)

The comparison operator.

value [string](#)<sup>↗</sup>

The value for the filter.

Returns

[IQueryBuilder](#)

The current IQueryable instance.

## FilterBy(string, InclusionOperators, IEnumerable<string>)

Adds a filter condition for inclusion operators (IN, NOT IN) with AND.

```
public IQueryBuilder FilterBy(string column, InclusionOperators @operator,
    IEnumerable<string> values)
```

### Parameters

column [string](#)

The column name.

operator [InclusionOperators](#)

The inclusion operator.

values [IEnumerable](#) <[string](#)>

The values for the filter.

### Returns

[IQueryBuilder](#)

The current IQueryBuilder instance.

## FilterBy(string, NullOperators)

Adds a filter condition for null checks with AND.

```
public IQueryBuilder FilterBy(string column, NullOperators @operator)
```

### Parameters

column [string](#)

The column name.

operator [NullOperators](#)

The null operator.

Returns

[IQueryBuilder](#)

The current IQueryBuilder instance.

## FilterBy(string, RangeOperators, string, string)

Adds a filter condition for range checks (BETWEEN, NOT BETWEEN) with AND.

```
public IQueryBuilder FilterBy(string column, RangeOperators @operator, string start, string end)
```

Parameters

column [string](#) 

The column name.

operator [RangeOperators](#)

The range operator.

start [string](#) 

The start value of the range.

end [string](#) 

The end value of the range.

Returns

[IQueryBuilder](#)

The current IQueryBuilder instance.

## FilterBy(string, StringOperators, string)



Adds a filter condition using a string operator (LIKE, ILIKE) with AND.

```
public IQueryBuilder FilterBy(string column, StringOperators @operator,  
string pattern)
```

## Parameters

column [string](#)

The column name.

operator [StringOperators](#)

The string operator for filtering.

pattern [string](#)

The pattern to match.

## Returns

[IQueryBuilder](#)

The current IQueryBuilder instance.

## FilterBy(string, string, string)

Adds a filter condition to the query using a raw string operator with AND.

```
public IQueryBuilder FilterBy(string column, string @operator, string value)
```

## Parameters

column [string](#)

The column name.

operator [string](#)

The operator as a string.

value [string](#)

The value for the filter.

Returns

[IQueryBuilder](#)

The current IQueryBuilder instance.

## FromTable(string)

Specifies the table from which to select data.

```
public IQueryBuilder FromTable(string table)
```

Parameters

table [string](#)

The name of the table.

Returns

[IQueryBuilder](#)

The current IQueryBuilder instance.

## GenerateFormattedSql()

Generates a formatted SQL query with proper line breaks and indentation.

```
public string GenerateFormattedSql()
```

Returns

[string](#)

A SQL query string.

Exceptions

[InvalidOperationException](#)↗

Thrown when the table is undefined or not present in the provided database information.

## GenerateSql()

Generates the complete SQL query string based on the specified clauses.

```
public string GenerateSql()
```

Returns

[string](#)↗

A SQL query string.

Exceptions

[InvalidOperationException](#)↗

Thrown when the table is undefined or not present in the provided database information.

## GroupBy(params string[])

Specifies the GROUP BY clause for the query.

```
public IQueryBuilder GroupBy(params string[] columns)
```

Parameters

columns [string](#)↗[]

An array of column names to group by.

Returns

[IQueryBuilder](#)

The current IQueryBuilder instance.

## Join(JoinOperators, string, string)

Adds a JOIN clause to the query.

```
public IQueryBuilder Join(JoinOperators type, string table, string condition)
```

### Parameters

type [JoinOperators](#)

The type of join represented by a [JoinOperators](#) enum.

table [string](#)

The table to join.

condition [string](#)

The condition on which to join.

### Returns

[IQueryBuilder](#)

The current IQueryBuilder instance.

## OrFilterBy(string, ComparisonOperators, string)

Adds a filter condition to the query using a comparison operator with OR.

```
public IQueryBuilder OrFilterBy(string column, ComparisonOperators @operator,  
string value)
```

### Parameters

column [string](#)

The column name.

operator [ComparisonOperators](#)

The comparison operator.

value [string](#)

The value for the filter.

Returns

[IQueryBuilder](#)

The current IQueryBuilder instance.

## OrFilterBy(string, InclusionOperators, IEnumerable<string>)

Adds a filter condition for inclusion operators (IN, NOT IN) with OR.

```
public IQueryBuilder OrFilterBy(string column, InclusionOperators @operator,
    IEnumerable<string> values)
```

Parameters

column [string](#)

The column name.

operator [InclusionOperators](#)

The inclusion operator.

values [IEnumerable](#) <[string](#)>

The values for the filter.

Returns

[IQueryBuilder](#)

The current IQueryBuilder instance.

## OrFilterBy(string, NullOperators)

Adds a filter condition for null checks with OR.

```
public IQueryBuilder OrFilterBy(string column, NullOperators @operator)
```

## Parameters

column [string](#)

The column name.

operator [NullOperators](#)

The null operator.

## Returns

[IQueryBuilder](#)

The current IQueryBuilder instance.

## OrFilterBy(string, RangeOperators, string, string)

Adds a filter condition for range checks (BETWEEN, NOT BETWEEN) with OR.

```
public IQueryBuilder OrFilterBy(string column, RangeOperators @operator, string  
start, string end)
```

## Parameters

column [string](#)

The column name.

operator [RangeOperators](#)

The range operator.

start [string](#)

The start value of the range.

end [string](#)

The end value of the range.

## Returns

### [IQueryBuilder](#)

The current IQueryBuilder instance.

## OrFilterBy(string, StringOperators, string)

Adds a filter condition using a string operator (LIKE, ILIKE) with OR.

```
public IQueryBuilder OrFilterBy(string column, StringOperators @operator,  
string pattern)
```

## Parameters

column [string](#)

The column name.

operator [StringOperators](#)

The string operator for filtering.

pattern [string](#)

The pattern to match.

## Returns

### [IQueryBuilder](#)

The current IQueryBuilder instance.

## OrFilterBy(string, string, string)

Adds a filter condition to the query using a raw string operator with OR.

```
public IQueryBuilder OrFilterBy(string column, string @operator, string value)
```

## Parameters

`column` [string](#)

The column name.

`operator` [string](#)

The operator as a string.

`value` [string](#)

The value for the filter.

Returns

[IQueryBuilder](#)

The current IQueryBuilder instance.

## OrderBy(string, OrderDirection)

Specifies the ORDER BY clause for a single column with a specified direction.

```
public IQueryBuilder OrderBy(string column, OrderDirection direction)
```

Parameters

`column` [string](#)

The column name to order by.

`direction` [OrderDirection](#)

The direction of ordering.

Returns

[IQueryBuilder](#)

The current IQueryBuilder instance.

## OrderBy(params string[])



Specifies the ORDER BY clause for the query using default ordering.

```
public IQueryBuilder OrderBy(params string[] columns)
```

## Parameters

`columns` [string](#)[]

An array of column names to order by.

## Returns

[IQueryBuilder](#)

The current IQueryBuilder instance.

## SetLimit(int)

Limits the number of records returned by the query.

```
public IQueryBuilder SetLimit(int limit)
```

## Parameters

`limit` [int](#)

The maximum number of records to return.

## Returns

[IQueryBuilder](#)

The current IQueryBuilder instance.

## Setup(Dictionary<string, List<string>>)

Configures the query builder with the required database metadata. This method must be called before proceeding with any query building operations.

```
public IQueryable Setup(Dictionary<string, List<string>> databaseInfo)
```

## Parameters

**databaseInfo** [Dictionary](#) <[string](#), [List](#) <[string](#)>>

A dictionary where each key is a table name (as a string) and the associated value is a list of column names (as strings) for that table.

## Returns

[IQueryBuilder](#)

An instance of [IQueryBuilder](#) that provides the full set of query-building methods.

# Namespace DynamicQueryBuilder.Drivers

## Classes

### [DriversFactory](#)

Factory class for creating database connections and metadata queries.

# Class DriversFactory

Namespace: [DynamicQueryBuilder.Drivers](#)

Assembly: DynamicQueryBuilder.dll








Factory class for creating database connections and metadata queries.

```
public static class DriversFactory
```

## Inheritance

[object](#)  ← DriversFactory

## Inherited Members

[object.Equals\(object\)](#)  , [object.Equals\(object, object\)](#)  , [object.GetHashCode\(\)](#)  ,  
[object.GetType\(\)](#)  , [object.MemberwiseClone\(\)](#)  , [object.ReferenceEquals\(object, object\)](#)  ,  
[object.ToString\(\)](#) 

## Methods

### GetDatabaseConnection(DatabaseDriver, string)

Get a database connection.

```
public static IDbConnection GetDatabaseConnection(DatabaseDriver driver,  
string connectionString)
```

## Parameters

**driver** [DatabaseDriver](#)

The database driver.

**connectionString** [string](#) 

The connection string.

## Returns

[IDbConnection](#) 

A database connection.

## GetMetadataQuery(DatabaseDriver)

Get the metadata query for a database driver.

```
public static string GetMetadataQuery(DatabaseDriver driver)
```

### Parameters

**driver** [DatabaseDriver](#)

The database driver.

### Returns

[string](#) 

The metadata query.

# Namespace DynamicQueryBuilder. Interfaces.Builders

## Interfaces

### [IQueryBuilder](#)

Defines a builder for constructing SQL queries.

### [IQueryBuilderSetup](#)

Provides the initial setup functionality for the dynamic query builder. This interface enforces that the necessary database metadata is provided before any query construction methods become available.

# Interface IQueryBuilder

Namespace: [DynamicQueryBuilder.Interfaces.Builders](#)

Assembly: DynamicQueryBuilder.dll

Defines a builder for constructing SQL queries.

```
public interface IQueryBuilder
```

## Methods

### Columns(params string[])

Specifies the columns to select in the query.

```
IQueryBuilder Columns(params string[] columns)
```

#### Parameters

**columns** [string](#)<sup>?</sup>[]

An array of column names.

#### Returns

[IQueryBuilder](#)

The current IQueryBuilder instance.

### FilterBy(string, ComparisonOperators, string)

Adds a filter condition to the query using a comparison operator with AND.

```
IQueryBuilder FilterBy(string column, ComparisonOperators @operator, string value)
```

#### Parameters

column [string](#)

The column name.

operator [ComparisonOperators](#)

The comparison operator.

value [string](#)

The value for the filter.

Returns

[IQueryBuilder](#)

The current IQueryBuilder instance.

## FilterBy(string, InclusionOperators, IEnumerable<string>)

Adds a filter condition for inclusion operators (IN, NOT IN) with AND.

```
IQueryBuilder FilterBy(string column, InclusionOperators @operator,
IEnumerable<string> values)
```

Parameters

column [string](#)

The column name.

operator [InclusionOperators](#)

The inclusion operator.

values [IEnumerable](#) <[string](#)>

The values for the filter.

Returns

[IQueryBuilder](#)



The current IQueryBuilder instance.

## FilterBy(string, NullOperators)

Adds a filter condition for null checks with AND.

```
IQueryBuilder FilterBy(string column, NullOperators @operator)
```

### Parameters

**column** [string](#)

The column name.

**operator** [NullOperators](#)

The null operator.

### Returns

[IQueryBuilder](#)

The current IQueryBuilder instance.

## FilterBy(string, RangeOperators, string, string)

Adds a filter condition for range checks (BETWEEN, NOT BETWEEN) with AND.

```
IQueryBuilder FilterBy(string column, RangeOperators @operator, string start,  
string end)
```

### Parameters

**column** [string](#)

The column name.

**operator** [RangeOperators](#)

The range operator.

start [string](#)

The start value of the range.

end [string](#)

The end value of the range.

Returns

[IQueryBuilder](#)

The current IQueryBuilder instance.

## FilterBy(string, StringOperators, string)

Adds a filter condition using a string operator (LIKE, ILIKE) with AND.

```
IQueryBuilder FilterBy(string column, StringOperators @operator, string pattern)
```

Parameters

column [string](#)

The column name.

operator [StringOperators](#)

The string operator for filtering.

pattern [string](#)

The pattern to match.

Returns

[IQueryBuilder](#)

The current IQueryBuilder instance.

## FilterBy(string, string, string)

Adds a filter condition to the query using a raw string operator with AND.

```
IQueryBuilder FilterBy(string column, string @operator, string value)
```

## Parameters

column [string](#)

The column name.

operator [string](#)

The operator as a string.

value [string](#)

The value for the filter.

## Returns

[IQueryBuilder](#)

The current IQueryBuilder instance.

## FromTable(string)

Specifies the table from which to select data.

```
IQueryBuilder FromTable(string table)
```

## Parameters

table [string](#)

The name of the table.

## Returns

[IQueryBuilder](#)

The current IQueryBuilder instance.

## GenerateFormattedSql()

Generates a formatted SQL query with proper line breaks and indentation.


```
string GenerateFormattedSql()
```

### Returns

[string](#) 

A SQL query string.

### Exceptions

[InvalidOperationException](#) 

Thrown when the table is undefined or not present in the provided database information.

## GenerateSql()

Generates the complete SQL query string based on the specified clauses.

```
string GenerateSql()
```

### Returns

[string](#) 

A SQL query string.

### Exceptions

[InvalidOperationException](#) 

Thrown when the table is undefined or not present in the provided database information.

## GroupBy(params string[])

Specifies the GROUP BY clause for the query.

```
IQueryBuilder GroupBy(params string[] columns)
```

## Parameters

**columns** [string](#)[]

An array of column names to group by.

## Returns

[IQueryBuilder](#)

The current IQueryBuilder instance.

# Join(JoinOperators, string, string)

Adds a JOIN clause to the query.

```
IQueryBuilder Join(JoinOperators type, string table, string condition)
```

## Parameters

**type** [JoinOperators](#)

The type of join represented by a [JoinOperators](#) enum.

**table** [string](#)

The table to join.

**condition** [string](#)

The condition on which to join.

## Returns

[IQueryBuilder](#)

The current IQueryBuilder instance.

## OrFilterBy(string, ComparisonOperators, string)

Adds a filter condition to the query using a comparison operator with OR.

```
IQueryBuilder OrFilterBy(string column, ComparisonOperators @operator, string value)
```

### Parameters

**column** [string](#)

The column name.

**operator** [ComparisonOperators](#)

The comparison operator.

**value** [string](#)

The value for the filter.

### Returns

[IQueryBuilder](#)

The current IQueryBuilder instance.

## OrFilterBy(string, InclusionOperators, IEnumerable<string>)

Adds a filter condition for inclusion operators (IN, NOT IN) with OR.

```
IQueryBuilder OrFilterBy(string column, InclusionOperators @operator,  
IEnumerable<string> values)
```

### Parameters

**column** [string](#)

The column name.

**operator** [InclusionOperators](#)

The inclusion operator.

values [IEnumerable](#) <[string](#)>

The values for the filter.

Returns

[IQueryBuilder](#)

The current IQueryBuilder instance.

## OrFilterBy(string, NullOperators)

Adds a filter condition for null checks with OR.

```
IQueryBuilder OrFilterBy(string column, NullOperators @operator)
```

Parameters

column [string](#)

The column name.

operator [NullOperators](#)

The null operator.

Returns

[IQueryBuilder](#)

The current IQueryBuilder instance.

## OrFilterBy(string, RangeOperators, string, string)

Adds a filter condition for range checks (BETWEEN, NOT BETWEEN) with OR.

```
IQueryBuilder OrFilterBy(string column, RangeOperators @operator, string start,  
string end)
```

## Parameters

**column** [string](#)

The column name.

**operator** [RangeOperators](#)

The range operator.

**start** [string](#)

The start value of the range.

**end** [string](#)

The end value of the range.

## Returns

[IQueryBuilder](#)

The current IQueryBuilder instance.

## OrFilterBy(string, StringOperators, string)

Adds a filter condition using a string operator (LIKE, ILIKE) with OR.

```
IQueryBuilder OrFilterBy(string column, StringOperators @operator, string pattern)
```

## Parameters

**column** [string](#)

The column name.

**operator** [StringOperators](#)

The string operator for filtering.

**pattern** [string](#)

The pattern to match.



## Returns

[IQueryBuilder](#)

The current IQueryBuilder instance.

## OrFilterBy(string, string, string)

Adds a filter condition to the query using a raw string operator with OR.

```
IQueryBuilder OrFilterBy(string column, string @operator, string value)
```

## Parameters

column [string](#)<sup>↗</sup>

The column name.

operator [string](#)<sup>↗</sup>

The operator as a string.

value [string](#)<sup>↗</sup>

The value for the filter.

## Returns

[IQueryBuilder](#)

The current IQueryBuilder instance.

## OrderBy(string, OrderDirection)

Specifies the ORDER BY clause for a single column with a specified direction.

```
IQueryBuilder OrderBy(string column, OrderDirection direction)
```

## Parameters

column [string](#)

The column name to order by.

direction [OrderDirection](#)

The direction of ordering.

Returns

[IQueryBuilder](#)

The current IQueryBuilder instance.

## OrderBy(params string[])

Specifies the ORDER BY clause for the query using default ordering.

```
IQueryBuilder OrderBy(params string[] columns)
```

Parameters

columns [string](#)[]

An array of column names to order by.

Returns

[IQueryBuilder](#)

The current IQueryBuilder instance.

## SetLimit(int)

Limits the number of records returned by the query.

```
IQueryBuilder SetLimit(int limit)
```

Parameters

limit [int](#)

The maximum number of records to return.

Returns

[IQueryBuilder](#)

The current IQueryBuilder instance.

# Interface IQueryBuilderSetup

Namespace: [DynamicQueryBuilder.Interfaces.Builders](#)

Assembly: DynamicQueryBuilder.dll

Provides the initial setup functionality for the dynamic query builder. This interface enforces that the necessary database metadata is provided before any query construction methods become available.

```
public interface IQueryBuilderSetup
```

## Methods

### Setup(Dictionary<string, List<string>>)

Configures the query builder with the required database metadata. This method must be called before proceeding with any query building operations.

```
IQueryBuilder Setup(Dictionary<string, List<string>> databaseInfo)
```

## Parameters

**databaseInfo** [Dictionary](#) <[string](#), [List](#) <[string](#)>>

A dictionary where each key is a table name (as a string) and the associated value is a list of column names (as strings) for that table.

## Returns

[IQueryBuilder](#)

An instance of [IQueryBuilder](#) that provides the full set of query-building methods.

# Namespace DynamicQueryBuilder. Interfaces.Helpers

## Interfaces

### [IGetDatabaseMetaData](#)

Interface for getting database metadata.

# Interface IGetDatabaseMetaData

Namespace: [DynamicQueryBuilder.Interfaces.Helpers](#)

Assembly: DynamicQueryBuilder.dll

Interface for getting database metadata.

```
public interface IGetDatabaseMetaData
```

## Methods

### GetDatabaseTablesMetaDataAsync()

Get the metadata of the database tables.

```
Task<Dictionary<string, Dictionary<string, List<DatabaseTablesMetaData>>>>  
GetDatabaseTablesMetaDataAsync()
```

## Returns

[Task](#) <[Dictionary](#) <[string](#), [Dictionary](#) <[string](#), [List](#) <[DatabaseTablesMetaData](#)>>>>

A dictionary of schemas and tables with their metadata.

## Exceptions

[InvalidOperationException](#)

Thrown when unable to create a command.

# Namespace DynamicQueryBuilder.Models. Enums.Helpers.Databases

## Enums

### [DatabaseDriver](#)

Supported drivers to get tables metadata.

# Enum DatabaseDriver

Namespace: [DynamicQueryBuilder.Models.Enums.Helpers.Databases](#)

Assembly: DynamicQueryBuilder.dll

Supported drivers to get tables metadata.

```
public enum DatabaseDriver
```

## Fields

**MYSQL = 1**

MySQL driver.

**POSTGRESQL = 0**

PostgreSQL driver.



# Namespace DynamicQueryBuilder.Models. Enums.SqlOperators

## Classes

### [FilterClause](#)

Represents a filter clause in SQL.

## Enums

### [ArithmeticOperators](#)

Represents the arithmetic operators in SQL.

### [ComparisonOperators](#)

Represents SQL comparison operators.

### [InclusionOperators](#)

Represents SQL inclusion operators.

### [JoinOperators](#)

Represents SQL join operators.

### [LogicalOperators](#)

Represents SQL logical operators.

### [NullOperators](#)

Represents SQL null operators.

### [OrderDirection](#)

Specifies the order direction for ordering clauses.

### [RangeOperators](#)

Represents SQL range operators.

### [SetOperators](#)

Represents SQL set operators.

### [StringOperators](#)

Represents SQL string operators.

# Enum ArithmeticOperators

Namespace: [DynamicQueryBuilder.Models.Enums.SqlOperators](#)

Assembly: DynamicQueryBuilder.dll

Represents the arithmetic operators in SQL.

```
public enum ArithmeticOperators
```

## Fields

**ADDITION = 0**

Addition (+)

**DIVISION = 3**

Division (/)

**MODULO = 4**

Modulo (%) - Remainder of division

**MULTIPLICATION = 2**

Multiplication (\*)

**SUBTRACTION = 1**

Subtraction (-)

# Enum ComparisonOperators

Namespace: [DynamicQueryBuilder.Models.Enums.SqlOperators](#)

Assembly: DynamicQueryBuilder.dll

Represents SQL comparison operators.

```
public enum ComparisonOperators
```

## Fields

```
EQUAL = 0
```

Equal (=)

```
GREATER_OR_EQUAL = 4
```

Greater than or equal to (>=)

```
GREATER_THAN = 2
```

Greater than (>)

```
LESS_OR_EQUAL = 5
```

```
LESS_THAN = 3
```

```
NOT_EQUAL = 1
```

# Class FilterClause

Namespace: [DynamicQueryBuilder.Models.Enums.SqlOperators](#)

Assembly: DynamicQueryBuilder.dll

Represents a filter clause in SQL.

```
public record FilterClause : IEquatable<FilterClause>
```








## Inheritance

[object](#)  ← FilterClause

## Implements

[IEquatable](#)  <[FilterClause](#)>

## Inherited Members

[object.Equals\(object\)](#)  , [object.Equals\(object, object\)](#)  , [object.GetHashCode\(\)](#)  ,  
[object.GetType\(\)](#)  , [object.MemberwiseClone\(\)](#)  , [object.ReferenceEquals\(object, object\)](#)  ,  
[object.ToString\(\)](#) 

## Constructors

### FilterClause(LogicalOperators, string)

Represents a filter clause in SQL.

```
public FilterClause(LogicalOperators JoinOperator, string Condition)
```

## Parameters

**JoinOperator** [LogicalOperators](#)

The logical operator to join the condition.

**Condition** [string](#) 

The condition to filter the data.

# Properties

## Condition

The condition to filter the data.

```
public string Condition { get; init; }
```

## Property Value

[string](#)

## JoinOperator

The logical operator to join the condition.

```
public LogicalOperators JoinOperator { get; init; }
```

## Property Value

[LogicalOperators](#)

# Enum InclusionOperators

Namespace: [DynamicQueryBuilder.Models.Enums.SqlOperators](#)

Assembly: DynamicQueryBuilder.dll

Represents SQL inclusion operators.

```
public enum InclusionOperators
```

## Fields

```
IN = 0
```

IN - Checks if a value belongs to a set

```
NOT_IN = 1
```

NOT IN - Checks if a value does not belong to a set

# Enum JoinOperators

Namespace: [DynamicQueryBuilder.Models.Enums.SqlOperators](#)

Assembly: DynamicQueryBuilder.dll

Represents SQL join operators.

```
public enum JoinOperators
```

## Fields

**CROSS\_JOIN = 4**

CROSS JOIN - Produces a Cartesian product of the tables

**FULL\_JOIN = 3**

FULL JOIN (or FULL OUTER JOIN) - Returns all records from both tables, with matching records when available

**INNER\_JOIN = 0**

INNER JOIN - Returns only records that have matching entries in both tables

**LEFT\_JOIN = 1**

LEFT JOIN (or LEFT OUTER JOIN) - Returns all records from the left table and the matching records from the right table

**RIGHT\_JOIN = 2**

RIGHT JOIN (or RIGHT OUTER JOIN) - Returns all records from the right table and the matching records from the left table

**SELF\_JOIN = 5**

SELF JOIN - Joins a table with itself

# Enum LogicalOperators

Namespace: [DynamicQueryBuilder.Models.Enums.SqlOperators](#)

Assembly: DynamicQueryBuilder.dll

Represents SQL logical operators.

```
public enum LogicalOperators
```

## Fields

AND = 0

Logical operator AND

NOT = 2

Logical operator NOT

OR = 1

Logical operator OR



# Enum NullOperators

Namespace: [DynamicQueryBuilder.Models.Enums.SqlOperators](#)

Assembly: DynamicQueryBuilder.dll

Represents SQL null operators.

```
public enum NullOperators
```

## Fields

```
IS_NOT_NULL = 1
```

IS NOT NULL - Checks if a value is not null

```
IS_NULL = 0
```

IS NULL - Checks if a value is null

# Enum OrderDirection

Namespace: [DynamicQueryBuilder.Models.Enums.SqlOperators](#)

Assembly: DynamicQueryBuilder.dll

Specifies the order direction for ordering clauses.

```
public enum OrderDirection
```

## Fields

**ASC = 0**

Ascending order (ASC)

**DESC = 1**

Descending order (DESC)

# Enum RangeOperators

Namespace: [DynamicQueryBuilder.Models.Enums.SqlOperators](#)

Assembly: DynamicQueryBuilder.dll

Represents SQL range operators.

```
public enum RangeOperators
```

## Fields

**BETWEEN = 0**

BETWEEN - Checks if the value is within a range

**NOT\_BETWEEN = 1**

NOT BETWEEN - Checks if the value is outside a range

# Enum SetOperators

Namespace: [DynamicQueryBuilder.Models.Enums.SqlOperators](#)

Assembly: DynamicQueryBuilder.dll

Represents SQL set operators.

```
public enum SetOperators
```

## Fields

**EXCEPT = 3**

EXCEPT - Returns records that are in the first query but not in the second

**INTERSECT = 2**

INTERSECT - Returns records common to both queries

**UNION = 0**

UNION - Combines and returns distinct results from two queries

**UNION\_ALL = 1**

UNION ALL - Combines results, including duplicates

# Enum StringOperators

Namespace: [DynamicQueryBuilder.Models.Enums.SqlOperators](#)

Assembly: DynamicQueryBuilder.dll

Represents SQL string operators.

```
public enum StringOperators
```

## Fields

**CONCAT = 2**

Concatenation of strings (|| in PostgreSQL/Oracle, + in SQL Server)

**ILIKE = 1**

ILIKE - Searches for patterns without case sensitivity (PostgreSQL)

**LIKE = 0**

LIKE - Searches for string patterns

# Namespace DynamicQueryBuilder.Models. Helpers

## Classes

### [DatabaseTablesMetaData](#)

Represents the metadata of a database table, including schema, columns, primary key, and foreign key relationships.

### [DynamicQueryBuilderSettings](#)

Configuration Object to initialize the library.

# Class DatabaseTablesMetaData

Namespace: [DynamicQueryBuilder.Models.Helpers](#)

Assembly: DynamicQueryBuilder.dll

Represents the metadata of a database table, including schema, columns, primary key, and foreign key relationships.

```
public record DatabaseTablesMetaData : IEquatable<DatabaseTablesMetaData>
```








## Inheritance

[object](#)  ← DatabaseTablesMetaData

## Implements

[IEquatable](#)  <[DatabaseTablesMetaData](#)>

## Inherited Members

[object.Equals\(object\)](#)  , [object.Equals\(object, object\)](#)  , [object.GetHashCode\(\)](#)  ,  
[object.GetType\(\)](#)  , [object.MemberwiseClone\(\)](#)  , [object.ReferenceEquals\(object, object\)](#)  ,  
[object.ToString\(\)](#) 

## Constructors

### DatabaseTablesMetaData(string, string, string?, string?, string?, string?)

Represents the metadata of a database table, including schema, columns, primary key, and foreign key relationships.

```
public DatabaseTablesMetaData(string Column, string ColumnType, string? PrimaryKey,  
string? RelatedSchema, string? RelatedTable, string? RelatedColumn)
```

## Parameters

Column [string](#) 

The name of the column in the table.

ColumnType [string](#)

The data type of the column.

PrimaryKey [string](#)

The name of the primary key column if applicable; otherwise, null.

RelatedSchema [string](#)

The schema of the related table if there is a foreign key relationship; otherwise, null.

RelatedTable [string](#)

The name of the related table if there is a foreign key relationship; otherwise, null.

RelatedColumn [string](#)

The name of the related column if there is a foreign key relationship; otherwise, null.

## Properties

### Column

The name of the column in the table.

```
public string Column { get; init; }
```

### Property Value

[string](#)

### ColumnType

The data type of the column.

```
public string ColumnType { get; init; }
```

### Property Value



[string](#)

## PrimaryKey

The name of the primary key column if applicable; otherwise, null.

```
public string? PrimaryKey { get; init; }
```

Property Value

[string](#)

## RelatedColumn

The name of the related column if there is a foreign key relationship; otherwise, null.

```
public string? RelatedColumn { get; init; }
```

Property Value

[string](#)

## RelatedSchema

The schema of the related table if there is a foreign key relationship; otherwise, null.

```
public string? RelatedSchema { get; init; }
```

Property Value

[string](#)

## RelatedTable

The name of the related table if there is a foreign key relationship; otherwise, null.

```
public string? RelatedTable { get; init; }
```

Property Value

[string](#)

# Class DynamicQueryBuilderSettings

Namespace: [DynamicQueryBuilder.Models.Helpers](#)

Assembly: DynamicQueryBuilder.dll

Configuration Object to initialize the library.

```
public record DynamicQueryBuilderSettings : IEquatable<DynamicQueryBuilderSettings>
```








## Inheritance

[object](#)  ← DynamicQueryBuilderSettings

## Implements

[IEquatable](#)  <[DynamicQueryBuilderSettings](#)>

## Inherited Members

[object.Equals\(object\)](#)  , [object.Equals\(object, object\)](#)  , [object.GetHashCode\(\)](#)  ,  
[object.GetType\(\)](#)  , [object.MemberwiseClone\(\)](#)  , [object.ReferenceEquals\(object, object\)](#)  ,  
[object.ToString\(\)](#) 

## Constructors

### DynamicQueryBuilderSettings(string, DatabaseDriver)

Configuration Object to initialize the library.

```
public DynamicQueryBuilderSettings(string connectionString,  
DatabaseDriver databaseDriver)
```

## Parameters

**ConnectionString** [string](#) 

Database connection string.

**DatabaseDriver** [DatabaseDriver](#)

Database Driver.

# Properties

## ConnectionString

Database connection string.

```
public string ConnectionString { get; init; }
```

Property Value

[string](#)

## DatabaseDriver

Database Driver.

```
public DatabaseDriver DatabaseDriver { get; init; }
```

Property Value

[DatabaseDriver](#)