

# 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, string)

Adds Dynamic Query Builder services to the service collection.

```
public static IServiceCollection AddDynamicQueryBuilder(this IServiceCollection  
services, string connectionString)
```

## Parameters

**services** [IServiceCollection](#) 

The service collection.

**connectionString** [string](#) 

The connection string.

## Returns

[IServiceCollection](#) 

The service collection.

# Namespace DynamicQueryBuilder. Databases

## Classes

### [GetDatabaseMetaData](#)

Get the metadata of the database tables.

# Class GetDatabaseMetaData

Namespace: [DynamicQueryBuilder.Databases](#)

Assembly: DynamicQueryBuilder.dll

Get the metadata of the database tables.

```
public sealed class GetDatabaseMetaData : IGetDatabaseMetaData
```







## Inheritance

[object](#)  ← GetDatabaseMetaData

## Implements

[IGetDatabaseMetaData](#)

## Inherited Members

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

## Constructors

### GetDatabaseMetaData(string)

Get the metadata of the database tables.

```
public GetDatabaseMetaData(string connectionString)
```

## Parameters

connectionString [string](#) 

## Methods

### GetDatabaseTablesMetaDataAsync(DatabaseDriver)

Get the metadata of the database tables.

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

## Parameters

**driver** [DatabaseDriver](#)

The database driver.

## 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.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.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(DatabaseDriver)

Get the metadata of the database tables.

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

## Parameters

**driver** [DatabaseDriver](#)

The database driver.

## 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. Helpers

## Classes

### [DatabaseTablesMetaData](#)

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

# 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](#)