Contents

DevForce supports **Entity SQL (ESQL) queries** with its *PassthruEsqlQuery()* class. As with all other query types, *PassthruEsqlQuery* implements the *IEntityQuery* interface.

API

There are several PassThruEsqlQuery constructor overloads:

```
public PassthruEsqlQuery(Type returnType, String esql)
public PassthruEsqlQuery(Type returnType, Type queryableType, String esql)
public PassthruEsqlQuery(Type returnType, ParameterizedEsql parameterizedEsql)
public PassthruEsqlQuery(Type returnType, Type queryableType, ParameterizedEsql parameterizedEsql)
```

```
Public Sub New(ByVal returnType As Type, ByVal esql As String)
Public Sub New(ByVal returnType As Type, ByVal parameterizedEsql As ParameterizedEsql)
End Sub
Public Sub New(ByVal returnType As Type, ByVal queryableType As Type, ByVal parameterizedEsql As ParameterizedEsql)
End Sub
```

Simple queries

The simplest calls require only a return type and a valid ESQL expression as show below:

```
var query0 = new PassthruEsqlQuery(typeof(Customer),
 "SELECT VALUE c FROM Customers AS c Where c.Country == 'Brazil'");
var result0 = query.With(_em1).Execute().Cast<Customer>();
var query1 = new PassthruEsqlQuery(typeof(SalesOrderHeader),
 "SELECT VALUE SalesOrderHeader FROM SalesOrderHeaders AS SalesOrderHeader Where SalesOrderHeader.Customer.CustomerID <
10");
var results1 = q1.With(_em1).Execute().Cast<SalesOrderHeader>();
var query2 = new PassthruEsqlQuery(typeof(SalesPerson),
  "SELECT VALUE sp FROM SalesPersons AS sp Where sp.Bonus > 2000");
var results2 = query2.With(_em1).Execute().Cast<SalesPerson>();
```

```
Dim query0 = New PassthruEsqlQuery(GetType(Customer), _
 "SELECT VALUE c FROM Customers AS c Where c.Country == 'Brazil'")
Dim result0 = query.With(_em1).Execute().Cast(Of Customer)()
Dim query1 = New PassthruEsqlQuery(GetType(SalesOrderHeader), _
 "SELECT VALUE SalesOrderHeader FROM SalesOrderHeaders AS SalesOrderHeader Where SalesOrderHeader.Customer.CustomerID <
10")
Dim results1 = q1.With(_em1).Execute().Cast(Of SalesOrderHeader)()
Dim query2 = New PassthruEsqlQuery(GetType(SalesPerson), _
 "SELECT VALUE sp FROM SalesPersons AS sp Where sp.Bonus > 2000")
Dim results2 = query2.With(_em1).Execute().Cast(Of SalesPerson)()
```

Note that because the *PassthruEsqlQuery* class is not generic all of the results from executing such a query must be cast to the appropriate result type.

Queries with parameters

Queries with parameters can also be used as shown below:

```
var param = new QueryParameter("country", "Brazil");
var paramEsql = new ParameterizedEsql(
 "SELECT VALUE c FROM Customers AS c Where c.Country > @country", param);
var query = new PassthruEsqlQuery(typeof(Customer), paramEsql);
var result1 = query.With(_em1).Execute().Cast<Customer>();
param.Value = "Germany";
var result2 = query.With(_em1).Execute().Cast<Customer>();
```

```
Dim param = New QueryParameter("country", "Brazil")
Dim paramEsql = New ParameterizedEsql( _
```

```
 "SELECT VALUE c FROM Customers AS c Where c.Country > @country", param)
Dim query = New PassthruEsqlQuery(GetType(Customer), paramEsql)
Dim result1 = query.With(_em1).Execute().Cast(Of Customer)()
param.Value = "Germany"
Dim result2 = query.With(_em1).Execute().Cast(Of Customer)()
```

Note that the value of the parameter can be changed and the same query re-executed, returning different results.

More complex queries

So far all of the queries shown have involved queries where the source type or queryable type of the query is the same as the return type. If this is not the case then we need to pass in both types to the constructor. For example:

```
var query1 = new PassthruEsqlQuery(typeof(Int32), typeof(Customer),
 "SELECT VALUE Count(c.CustomerType) FROM Customers AS c Where c.CustomerID < 10");
var result1 = query1.With(_em1).Execute().Cast<Int32>();
var query2 = new PassthruEsqlQuery(typeof(Decimal), typeof(SalesPerson),
 "SELECT VALUE Sum(sp.Bonus) FROM SalesPersons AS sp Where sp.Bonus > 2000");
var result2 = query2.With(_em1).Execute().Cast<Decimal>();
```

```
Dim query1 = New PassthruEsqlQuery(GetType(Int32), GetType(Customer), _
 "SELECT VALUE Count(c.CustomerType) FROM Customers AS c Where c.CustomerID < 10")
Dim result1 = query1.With(_em1).Execute().Cast(Of Int32)()
Dim query2 = New PassthruEsqlQuery(GetType(Decimal), GetType(SalesPerson), _
 "SELECT VALUE Sum(sp.Bonus) FROM SalesPersons AS sp Where sp.Bonus > 2000")
Dim result2 = query2.With(_em1).Execute().Cast(Of Decimal)()
```

Additional notes

When you use Entity SQL, you're responsible for formulating a query string that constitutes a valid query. If you goof, you won't know until you run it.

A *PassthruEsqlQuery* can only be executed with the *DataSourceOnly* query strategy.

DevForce does not currently support ESQL queries with anonymous projections.

Learn more

Entity SQL Quick Reference
Literals in Entity SQL
Canonical functions in Entity SQL