Examples of the LINQ Count, Sum, Min, Max and Average operators are shown below. In the examples below *_em1* is an EntityManager.

**C#**

```
[Description("This sample uses COUNT to get the number of
Orders.")]
public void LinqToEntities23() {
  var query = _em1.Orders.Count();
  Assert.IsTrue(query == 830);
}
[Description("This sample uses COUNT to get the number of Orders
placed by Customers in Mexico.")]
public void LinqToEntities24() {
  var query = _em1.Orders.Where(o => o.Customer.Address.Country
== "Mexico").Count();
  Assert.IsTrue(query == 28);
}
[Description("This sample uses COUNT to get the number of Orders
shipped to Mexico.")]
public void LinqToEntities25() {
  var query = _em1.Orders
     .Where(o => o.ShipCountry == "Mexico").Count();
  Assert.IsTrue(query == 28);
}
[Description("This sample uses SUM to find the total freight over all
Orders.")]
public void LinqToEntities26() {
  var query = _em1.Orders.Select(o => o.Freight).Sum();
  Assert.IsTrue(query == 64942.69M);
}
[Description("This sample uses SUM to find the total number of units
on order over all Products.")]
public void LinqToEntities27() {
  var query = _em1.Products.Sum(p => p.UnitsOnOrder);
  Assert.IsTrue(query == 780);
}
[Description("This sample uses SUM to find the total number of units
on order over all Products out-of-stock.")]
public void LinqToEntities28() {
  var query = _em1.Products.Where(p => p.UnitsInStock ==
0).Sum(p => p.UnitsOnOrder);
  Assert.IsTrue(query == 70);
}
[Description("This sample uses MIN to find the lowest unit price of
any Product.")]
public void LinqToEntities29() {
  var query = _em1.Products.Select(p => p.UnitPrice).Min();
  Assert.IsTrue(query == 2.5M);
}
[Description("This sample uses MIN to find the lowest freight of any
Order.")]
public void LinqToEntities30() {
  var query = _em1.Orders.Min(o => o.Freight);
  Assert.IsTrue(query == 0.02M);
}
[Description("This sample uses MIN to find the lowest freight of any
Order shipped to Mexico.")]
public void LinqToEntities31() {
  var query = _em1.Orders.Where(o => o.ShipCountry ==
"Mexico").Min(o => o.Freight);
  Assert.IsTrue(query == 0.4M);
  var query2 = _em1.Orders.Where(o => o.ShipCountry ==
"Mexico").Select(o => o.Freight).Min();
  Assert.IsTrue(query2 == 0.4M);
}
[Description("This sample uses Min to find the Products that have
the lowest unit price " +
   "in each category, and returns the result as an anonoymous type.")]
public void LinqToEntities32() {
  var query = from p in _em1.Products
              group p by p.Category.CategoryID into g
              orderby g.Key
```

```
            select new {
              CategoryID = g.Key,
              CheapestProducts =
                 from p2 in g
                 where p2.UnitPrice == g.Min(p3 => p3.UnitPrice)
                 select p2
            };
  var r = query.ToList();
  Assert.IsTrue(r.Count() == 8);
  Assert.IsTrue(r.First().CategoryID == 1);
  Assert.IsTrue(r.First().CheapestProducts.First().UnitPrice == 4.5M);
}
[Description("This sample uses MAX to find the latest hire date of
any Employee.")]
public void LinqToEntities33() {
  var query = _em1.Employees.Select(e => e.HireDate).Max();
  Assert.IsTrue(query == new DateTime(1994, 11, 15));
}
[Description("This sample uses MAX to find the most units in stock
of any Product.")]
public void LinqToEntities34() {
  var query = _em1.Products.Max(p => p.UnitsInStock);
  Assert.IsTrue(query == 125);
}
[Description("This sample uses MAX to find the most units in stock
of any Product with CategoryID = 1.")]
public void LinqToEntities35() {
  var query = _em1.Products.Where(p => p.Category.CategoryID ==
2).Max(p => p.UnitsInStock);
  Assert.IsTrue(query == 120);
}
[Description("This sample uses MAX to find the Products that have
the " +
    "highest unit price in each category, and returns the result as an
anonoymous type.")]
public void LinqToEntities36() {
  var query = from p in _em1.Products
              group p by p.Category.CategoryID into g
              orderby g.Key
              select new {
                g.Key,
                MostExpensiveProducts =
                   from p2 in g
                   where p2.UnitPrice == g.Max(p3 => p3.UnitPrice)
                   orderby p2.UnitPrice
                   select p2
              };
  var r = query.ToList();
  Assert.IsTrue(r.Count() == 8);
  Assert.IsTrue(r.First().Key == 1);
  Assert.IsTrue(r.First().MostExpensiveProducts.First().UnitPrice ==
263.5M);
}
[Description("This sample uses AVERAGE to find the average
freight of all Orders.")]
public void LinqToEntities37() {
  var query = _em1.Orders.Select(o => o.Freight).Average();
  Assert.IsTrue(query == 78.2442M);
}
[Description("This sample uses AVERAGE to find the average unit
price of all Products.")]
public void LinqToEntities38() {
  var query = _em1.Products.Average(p => p.UnitPrice);
  Assert.IsTrue(query == 28.8663M);
}
[Description("This sample uses AVERAGE to find the average unit
price of all Products with CategoryID = 1.")]
public void LinqToEntities39() {
  var query = _em1.Products.Where(p => p.Category.CategoryID ==
1)
      .Average(p => p.UnitPrice);
```

```
  Assert.IsTrue(query == 37.9791M);
}
[Description("This sample uses AVERAGE to find the Products that have unit price higher than the average unit price of the category for each category.")]
public void LinqToEntities40() {
  var query = from p in _em1.Products
              group p by p.Category.CategoryID into g
              orderby g.Key descending
              select new {
                g.Key,
                ExpensiveProducts =
                  from p2 in g
                  where p2.UnitPrice > g.Average(p3 => p3.UnitPrice)
                  orderby p2.UnitPrice
                  select p2
              };
  var r = query.ToList();
  Assert.IsTrue(r.Count() == 8);
  Assert.IsTrue(r.First().Key == 8);
}
[Description("This sample uses AVERAGE to find the average unit price of each category.")]
public void LinqToEntities41() {
  var query = from p in _em1.Products
              group p by p.Category.CategoryID into g
              orderby g.Key descending
              select new {
                g.Key,
                Average = g.Average(p => p.UnitPrice)
              };
  Assert.IsTrue(query.ToList().First().Key == 8);
  Assert.IsTrue(query.First().Average == 20.6825M);
}
```

**VB**

```
<Description("This sample uses COUNT to get the number of Orders.")>
Public Sub LinqToEntities23()
 Dim query = _em1.Orders.Count()
  Assert.IsTrue(query = 830)
End Sub
<Description("This sample uses COUNT to get the number of Orders placed by Customers in Mexico.")>
Public Sub LinqToEntities24()
 Dim query = _em1.Orders.Where(Function(o)
o.Customer.Address.Country = "Mexico").Count()
  Assert.IsTrue(query = 28)
End Sub
<Description("This sample uses COUNT to get the number of Orders shipped to Mexico.")>
Public Sub LinqToEntities25()
 Dim query = _em1.Orders.Where(Function(o) o.ShipCountry =
"Mexico").Count()
  Assert.IsTrue(query = 28)
End Sub
<Description("This sample uses SUM to find the total freight over all Orders.")>
Public Sub LinqToEntities26()
 Dim query = _em1.Orders.Select(Function(o) o.Freight).Sum()
  Assert.IsTrue(query Is 64942.69D)
End Sub
<Description("This sample uses SUM to find the total number of units on order over all Products.")>
Public Sub LinqToEntities27()
 Dim query = _em1.Products.Sum(Function(p) p.UnitsOnOrder)
  Assert.IsTrue(query = 780)
End Sub
<Description("This sample uses SUM to find the total number of units on order over all Products out-of-stock.")>
Public Sub LinqToEntities28()
 Dim query = _em1.Products.Where(Function(p) p.UnitsInStock =
0).Sum(Function(p) p.UnitsOnOrder)
```

```
  Assert.IsTrue(query = 70)
End Sub
<Description("This sample uses MIN to find the lowest unit price of
any Product.")>
Public Sub LinqToEntities29()
 Dim query = _em1.Products.Select(Function(p) p.UnitPrice).Min()
  Assert.IsTrue(query Is 2.5D)
End Sub
<Description("This sample uses MIN to find the lowest freight of any
Order.")>
Public Sub LinqToEntities30()
 Dim query = _em1.Orders.Min(Function(o) o.Freight)
  Assert.IsTrue(query Is 0.02D)
End Sub
<Description("This sample uses MIN to find the lowest freight of any
Order shipped to Mexico.")>
Public Sub LinqToEntities31()
 Dim query = _em1.Orders.Where(Function(o) o.ShipCountry =
"Mexico").Min(Function(o) o.Freight)
  Assert.IsTrue(query Is 0.4D)
 Dim query2 = _em1.Orders.Where(Function(o) o.ShipCountry =
"Mexico").Select(Function(o) o.Freight).Min()
  Assert.IsTrue(query2 Is 0.4D)
End Sub
<Description("This sample uses Min to find the Products that have
the lowest unit price " & "in each category, and returns the result as
an anonoymous type.")>
Public Sub LinqToEntities32()
 Dim query = From p In _em1.Products
             Group p By p.Category.CategoryID Into g = Group
             Order By CategoryID
            Select New With {Key .CategoryID = CategoryID,
Key .CheapestProducts =
                From p2 In g
                Where p2.UnitPrice = g.Min(Function(p3) p3.UnitPrice)
               Select p2}
 Dim r = query.ToList()
  Assert.IsTrue(r.Count() = 8)
  Assert.IsTrue(r.First().CategoryID = 1)
  Assert.IsTrue(r.First().CheapestProducts.First().UnitPrice = 4.5D)
End Sub
<Description("This sample uses MAX to find the latest hire date of
any Employee.")>
Public Sub LinqToEntities33()
 Dim query = _em1.Employees.Select(Function(e) e.HireDate).Max()
  Assert.IsTrue(query Is New Date(1994, 11, 15))
End Sub
<Description("This sample uses MAX to find the most units in stock
of any Product.")>
Public Sub LinqToEntities34()
 Dim query = _em1.Products.Max(Function(p) p.UnitsInStock)
  Assert.IsTrue(query = 125)
End Sub
<Description("This sample uses MAX to find the most units in stock
of any Product with CategoryID = 1.")>
Public Sub LinqToEntities35()
 Dim query = _em1.Products.Where(Function(p)
p.Category.CategoryID = 2).Max(Function(p) p.UnitsInStock)
  Assert.IsTrue(query = 120)
End Sub
<Description("This sample uses MAX to find the Products that have
the " & "highest unit price in each category, and returns the result as
an anonoymous type.")>
Public Sub LinqToEntities36()
 Dim query = From p In _em1.Products
             Group p By p.Category.CategoryID Into g = Group
             Order By CategoryID
            Select New With {Key CategoryID,
Key .MostExpensiveProducts =
                From p2 In g
                Where p2.UnitPrice = g.Max(Function(p3) p3.UnitPrice)
```

```
                Order By p2.UnitPrice
                Select p2}
 Dim r = query.ToList()
  Assert.IsTrue(r.Count() = 8)
  Assert.IsTrue(r.First().Key = 1)
  Assert.IsTrue(r.First().MostExpensiveProducts.First().UnitPrice =
263.5D)
End Sub
<Description("This sample uses AVERAGE to find the average
freight of all Orders.")>
Public Sub LinqToEntities37()
 Dim query = _em1.Orders.Select(Function(o) o.Freight).Average()
  Assert.IsTrue(query Is 78.2442D)
End Sub
<Description("This sample uses AVERAGE to find the average unit
price of all Products.")>
Public Sub LinqToEntities38()
 Dim query = _em1.Products.Average(Function(p) p.UnitPrice)
  Assert.IsTrue(query Is 28.8663D)
End Sub
<Description("This sample uses AVERAGE to find the average unit
price of all Products with CategoryID = 1.")>
Public Sub LinqToEntities39()
 Dim query = _em1.Products.Where(Function(p)
p.Category.CategoryID = 1).Average(Function(p) p.UnitPrice)
  Assert.IsTrue(query Is 37.9791D)
End Sub
<Description("This sample uses AVERAGE to find the Products that
have unit price higher than the average unit price of the category for
each category.")>
Public Sub LinqToEntities40()
 Dim query = From p In _em1.Products
             Group p By p.Category.CategoryID Into g = Group
             Order By CategoryID Descending
            Select New With {Key CategoryID, Key .ExpensiveProducts
=
                From p2 In g
                Where p2.UnitPrice > g.Average(Function(p3)
p3.UnitPrice)
                Order By p2.UnitPrice
               Select p2}
 Dim r = query.ToList()
  Assert.IsTrue(r.Count() = 8)
  Assert.IsTrue(r.First().Key = 8)
End Sub
<Description("This sample uses AVERAGE to find the average unit
price of each category.")>
Public Sub LinqToEntities41()
 Dim query = From p In _em1.Products
             Group p By p.Category.CategoryID Into g = Group
             Order By CategoryID Descending
            Select New With {Key CategoryID, Key .Average =
g.Average(Function(p) p.UnitPrice)}
  Assert.IsTrue(query.ToList().First().Key = 8)
  Assert.IsTrue(query.First().Average = 20.6825D)
End Sub
```