Entity Framework Lab

Setup

SQL Server Developer Edition or Higher

Install SQL Server if you don't already have it on your machine

https://www.microsoft.com/en-us/sql-server/sql-server-downloads

Music Database

Create a new database named **Music**

Download the **Music.sql** and **Music-Data.sql** files from https://github.com/PaulDSheriff/VSLive-Trainings

Run the **Music.sql** script to add the tables to the Music database

Run the **Music-Data.sql** script to add data to the Music database

Lab 1: Build New Project & Get All Songs

Open a new instance of Visual Studio and create a new Console Application named **EFLinqLab**.

Right mouse-click on the EFLinqLab project and select *Manage NuGet Packages…*

Click on the *Browse* tab.

Type **Microsoft.EntityFrameworkCore.SqlServer** into the Search box and install this package into your project.

Right mouse-click on the EFLinqLab project and create a new folder named **EntityClasses**.

Right mouse-click on the **EntityClasses** folder and add a new class named **Song**.

Add the following code into this new file.

```
#nullable disable

using System.ComponentModel.DataAnnotations.Schema;

namespace Samples;

[Table("Songs", Schema = "dbo")]
public partial class Song
{
  public int SongId { get; set; }
  public string SongName { get; set; }
  public string Artist { get; set; }
  public string Album { get; set; }
  public int? GenreId { get; set; }
  public int? KindId { get; set; }
  public string TrackNumber { get; set; }
  public int? Rating { get; set; }
  public int? Year { get; set; }
  public DateTime? ReleaseDate { get; set; }
  public string Size { get; set; }
  public int? Plays { get; set; }
  public DateTime? DateAdded { get; set; }

  #region ToString Override
  public override string ToString() {
    return $"{SongName} ({SongId})";
  }
  #endregion
}
```

Right mouse-click on the EFLinqLab project and create a new folder named **DbContextClasses**.

Right mouse-click on the **DbContextClasses** folder and add a new class named **MusicDbContext**.

Add the following code into this new file.

```
#nullable disable

using Microsoft.EntityFrameworkCore;

namespace Samples;

public partial class MusicDbContext : DbContext {
  const string CONN_STRING =
"Server=Localhost;Database=Music;Integrated Security=True";

  public virtual DbSet<Song> Songs { get; set; }

  protected override void OnConfiguring(DbContextOptionsBuilder
builder) {
    base.OnConfiguring(builder);
    builder.UseSqlServer(CONN_STRING);
  }
}
```

Modify the connection string to point to your SQL Server.

Get All Songs

Open the **Program.cs** file and replace all the code in there with the following code.

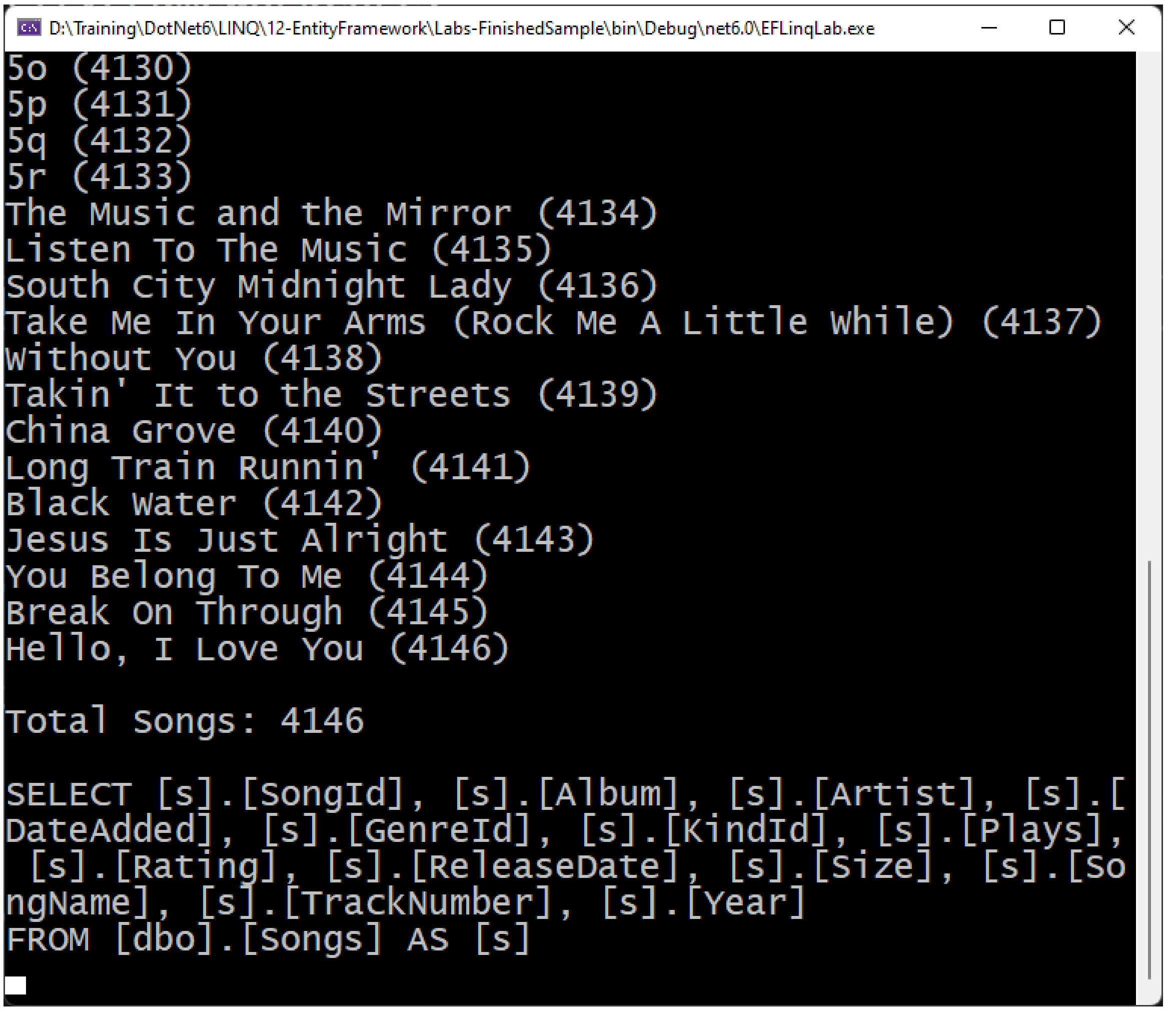
```
#nullable disable

using Microsoft.EntityFrameworkCore;
using Samples;

List<Song> list;

using (MusicDbContext db = new()) {
  // Get All Songs
  var query = (from row in db.Songs
               select row);

  // TODO: Write your query here


  list = query.ToList();

  // Display Songs
  foreach (Song row in list) {
    Console.WriteLine(row);
  }

  // Display Total Count
  Console.WriteLine();
  Console.WriteLine($"Total Songs: {list.Count}");

  // TODO: Show SQL Generated
  Console.WriteLine();
  Console.WriteLine(
    EntityFrameworkQueryableExtensions.ToQueryString(query));
}

// Pause for Results
Console.ReadKey();
```

Write the code to create a new instance of your MusicDbContext class.

Write a query to select all rows from the Songs table.

Try it Out

Run the application and your console window should look like the following:



D:\Training\DotNet6\LINQ\12-EntityFramework\Labs-FinishedSample\bin\Debug\net6.0\EFLinqLab.exe

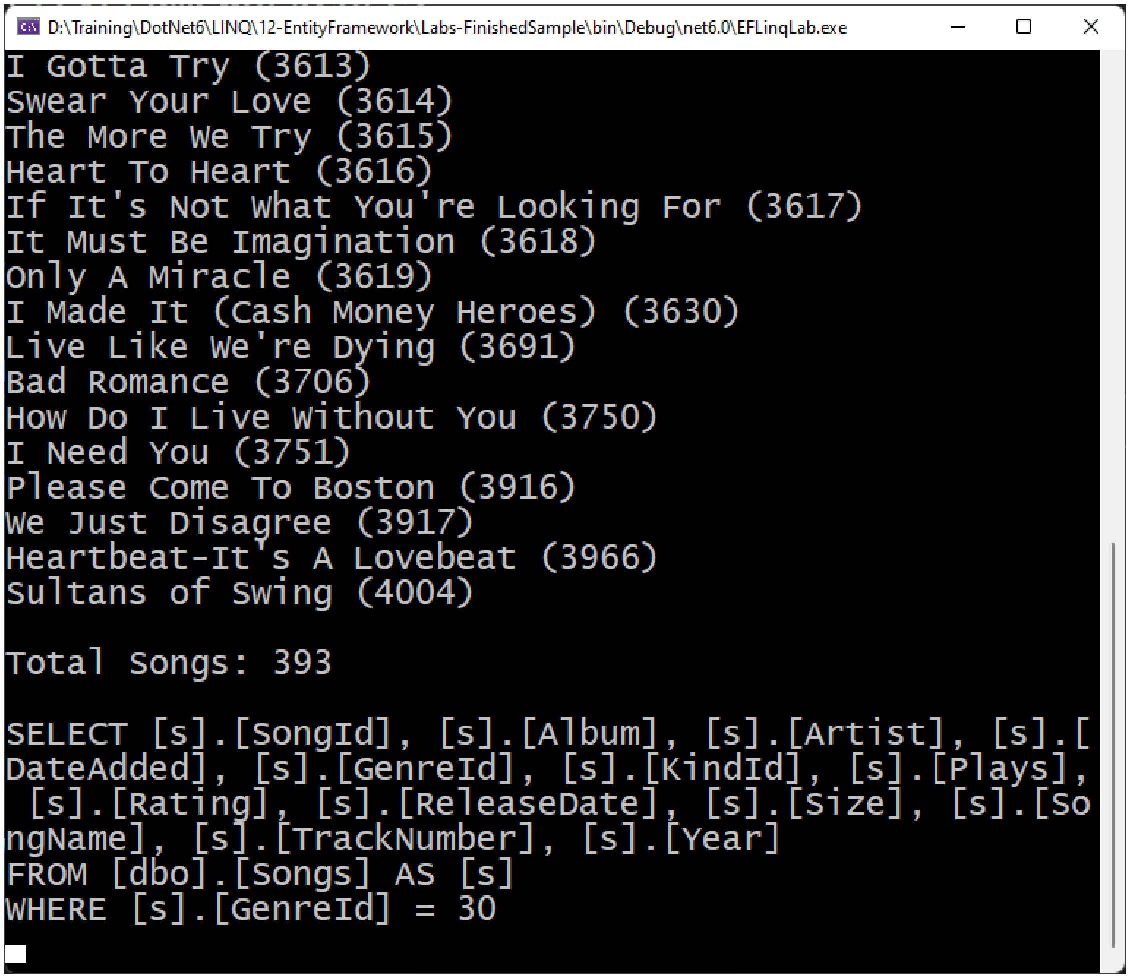
```
5o (4130)
5p (4131)
5q (4132)
5r (4133)
The Music and the Mirror (4134)
Listen To The Music (4135)
South City Midnight Lady (4136)
Take Me In Your Arms (Rock Me A Little While) (4137)
Without You (4138)
Takin' It to the Streets (4139)
China Grove (4140)
Long Train Runnin' (4141)
Black Water (4142)
Jesus Is Just Alright (4143)
You Belong To Me (4144)
Break On Through (4145)
Hello, I Love You (4146)

Total Songs: 4146

SELECT [s].[SongId], [s].[Album], [s].[Artist], [s].[
DateAdded], [s].[GenreId], [s].[KindId], [s].[Plays],
 [s].[Rating], [s].[ReleaseDate], [s].[Size], [s].[So
ngName], [s].[TrackNumber], [s].[Year]
FROM [dbo].[Songs] AS [s]
```

Lab 2: Add a Where Clause

Add a **where** clause to your query to select only those songs where the ***GenreId*** property is equal to 30.

Try it Out

Run the application and your console window should look like the following:


D:\Training\DotNet6\LINQ\12-EntityFramework\Labs-FinishedSample\bin\Debug\net6.0\EFLinqLab.exe

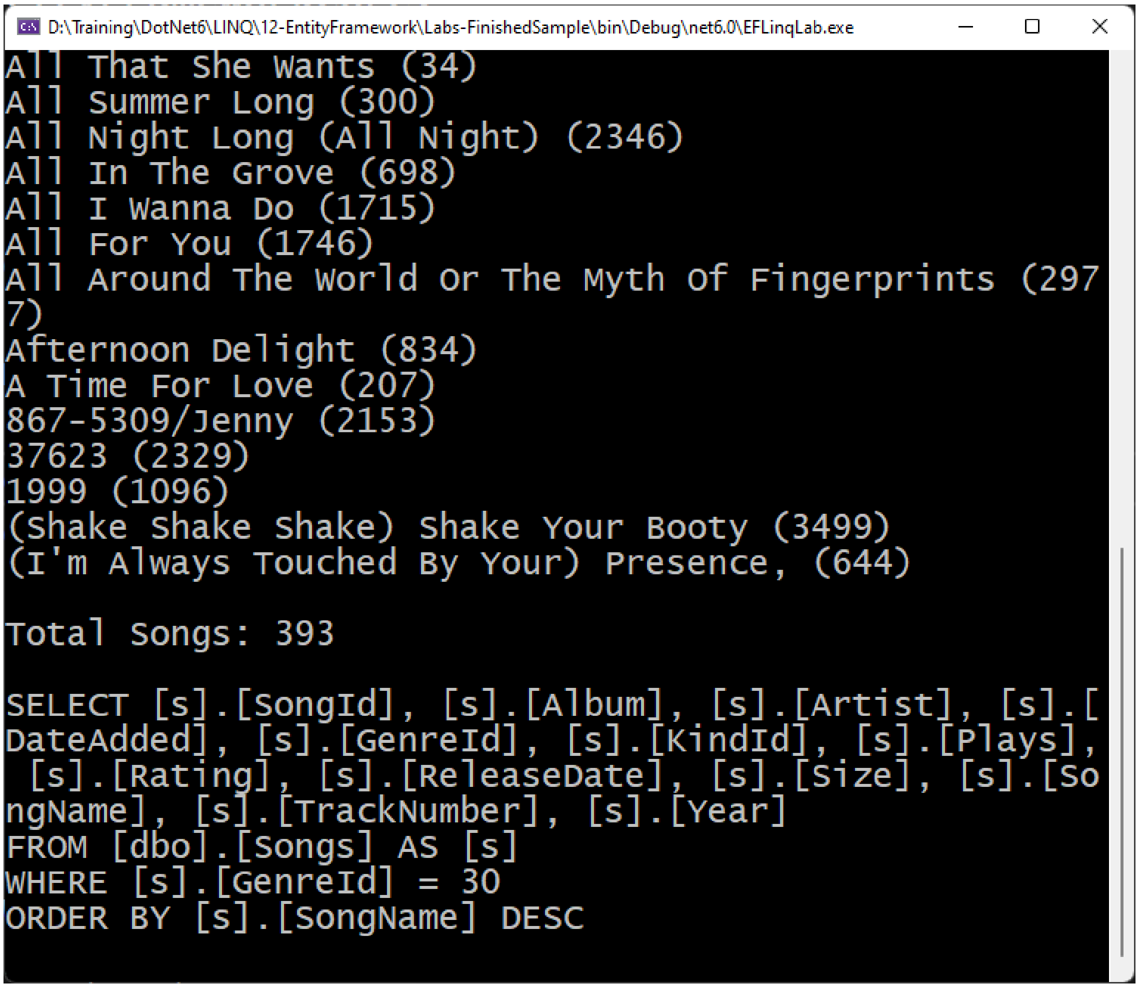
```
I Gotta Try (3613)
Swear Your Love (3614)
The More We Try (3615)
Heart To Heart (3616)
If It's Not What You're Looking For (3617)
It Must Be Imagination (3618)
Only A Miracle (3619)
I Made It (Cash Money Heroes) (3630)
Live Like We're Dying (3691)
Bad Romance (3706)
How Do I Live Without You (3750)
I Need You (3751)
Please Come To Boston (3916)
We Just Disagree (3917)
Heartbeat-It's A Lovebeat (3966)
Sultans of Swing (4004)

Total Songs: 393

SELECT [s].[SongId], [s].[Album], [s].[Artist], [s].[
DateAdded], [s].[GenreId], [s].[KindId], [s].[Plays],
 [s].[Rating], [s].[ReleaseDate], [s].[Size], [s].[So
ngName], [s].[TrackNumber], [s].[Year]
FROM [dbo].[Songs] AS [s]
WHERE [s].[GenreId] = 30
```

Lab 3: Add an Order By

Add an **order by** to the previous query to sort the songs by the ***SongName*** property in descending order.

Try it Out

Run the application and your console window should look like the following:


D:\Training\DotNet6\LINQ\12-EntityFramework\Labs-FinishedSample\bin\Debug\net6.0\EFLinqLab.exe

```
All That She Wants (34)
All Summer Long (300)
All Night Long (All Night) (2346)
All In The Grove (698)
All I Wanna Do (1715)
All For You (1746)
All Around The World Or The Myth Of Fingerprints (297
7)
Afternoon Delight (834)
A Time For Love (207)
867-5309/Jenny (2153)
37623 (2329)
1999 (1096)
(Shake Shake Shake) Shake Your Booty (3499)
(I'm Always Touched By Your) Presence, (644)

Total Songs: 393

SELECT [s].[SongId], [s].[Album], [s].[Artist], [s].[
DateAdded], [s].[GenreId], [s].[KindId], [s].[Plays],
 [s].[Rating], [s].[ReleaseDate], [s].[Size], [s].[So
ngName], [s].[TrackNumber], [s].[Year]
FROM [dbo].[Songs] AS [s]
WHERE [s].[GenreId] = 30
ORDER BY [s].[SongName] DESC
```

Lab 4: Get Average Rating

Replace all the code in the **Program.cs** file to look like the following:

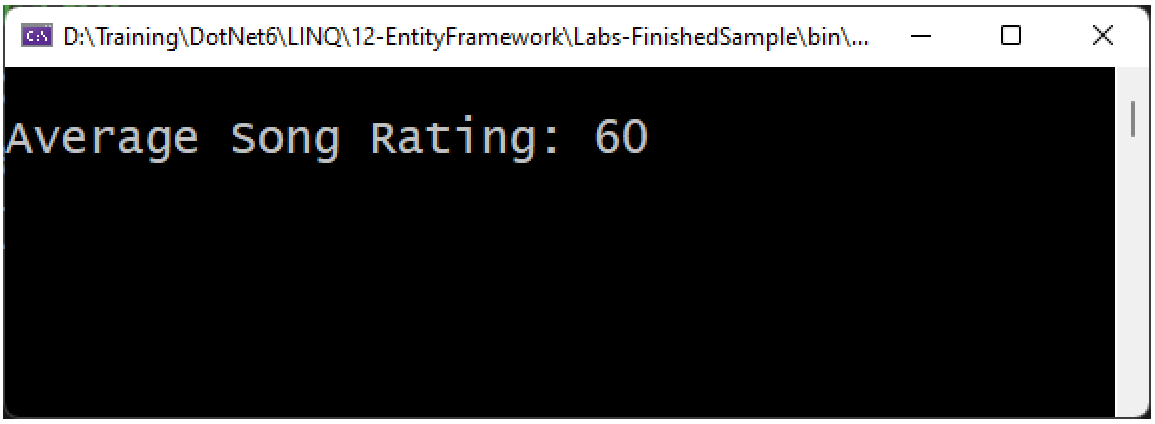
```
#nullable disable

using Samples;

using (MusicDbContext db = new()) {
  // TODO: Write a query to Average all Ratings


  // Display Average Ratings
  Console.WriteLine();
  Console.WriteLine($"Average Song Rating: {avg:n0}");
}

// Pause for Results
Console.ReadKey();
```

Add the Average() method to your query to get the average value in the ***Rating*** property.

Remove the OrderBy clause as that is not necessary when performing an aggregate function.

Try it Out

Run the application and your console window should look like the following:


D:\Training\DotNet6\LINQ\12-EntityFramework\Labs-FinishedSample\bin\...

```
Average Song Rating: 60
```

Lab 5: Display Query using LogTo

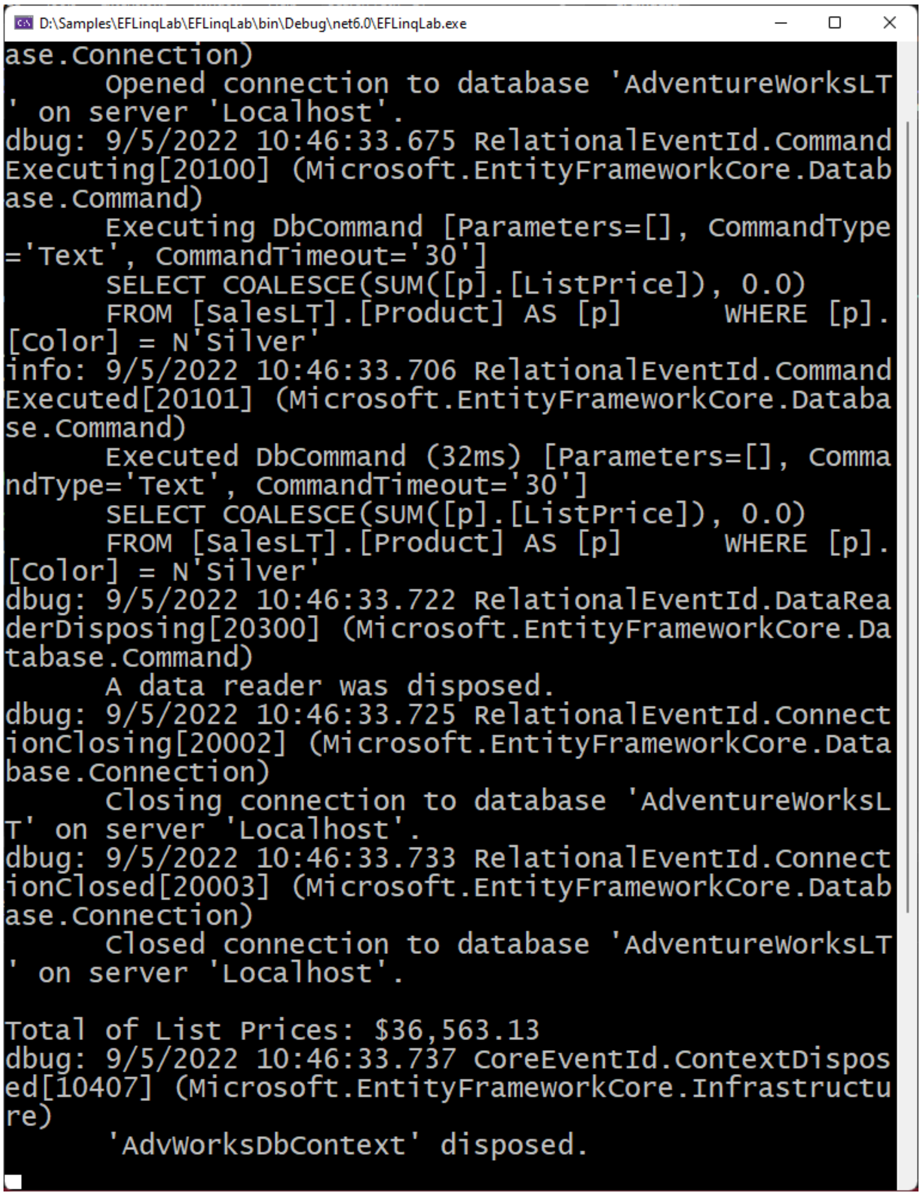
Open the **MusicDbContext.cs** file.

Add the following line of code as the last line in the OnConfiguring() method.

```
builder.LogTo(msg => Console.WriteLine(msg));
```

Try it Out

Run the application and your console window should look like the following:



D:\Samples\EFLinqLab\EFLinqLab\bin\Debug\net6.0\EFLinqLab.exe

```
ase.Connection)
      Opened connection to database 'AdventureWorksLT
' on server 'Localhost'.
dbug: 9/5/2022 10:46:33.675 RelationalEventId.Command
Executing[20100] (Microsoft.EntityFrameworkCore.Datab
ase.Command)
      Executing DbCommand [Parameters=[], CommandType
='Text', CommandTimeout='30']
      SELECT COALESCE(SUM([p].[ListPrice]), 0.0)
      FROM [SalesLT].[Product] AS [p]      WHERE [p].
[Color] = N'Silver'
info: 9/5/2022 10:46:33.706 RelationalEventId.Command
Executed[20101] (Microsoft.EntityFrameworkCore.Databa
se.Command)
      Executed DbCommand (32ms) [Parameters=[], Comma
ndType='Text', CommandTimeout='30']
      SELECT COALESCE(SUM([p].[ListPrice]), 0.0)
      FROM [SalesLT].[Product] AS [p]      WHERE [p].
[Color] = N'Silver'
dbug: 9/5/2022 10:46:33.722 RelationalEventId.DataRea
derDisposing[20300] (Microsoft.EntityFrameworkCore.Da
tabase.Command)
      A data reader was disposed.
dbug: 9/5/2022 10:46:33.725 RelationalEventId.Connect
ionClosing[20002] (Microsoft.EntityFrameworkCore.Data
base.Connection)
      Closing connection to database 'AdventureWorksL
T' on server 'Localhost'.
dbug: 9/5/2022 10:46:33.733 RelationalEventId.Connect
ionClosed[20003] (Microsoft.EntityFrameworkCore.Datab
ase.Connection)
      Closed connection to database 'AdventureWorksLT
' on server 'Localhost'.

Total of List Prices: $36,563.13
dbug: 9/5/2022 10:46:33.737 CoreEventId.ContextDispos
ed[10407] (Microsoft.EntityFrameworkCore.Infrastructu
re)
      'AdvWorksDbContext' disposed.
```

If you want to see all the log information, modify that line of code to the following:

```
builder.LogTo(msg => Debug.WriteLine(msg));
```

After running, check the Output window for the log information