

Use Stack Overflow for Teams at work to find answers in a private and secure environment. Get your first 10 users free. [Sign up.](#)

# How do I view the SQL generated by the Entity Framework?

Asked 10 years ago   Active 6 months ago   Viewed 278k times



How do I view the SQL generated by entity framework ?

567

(In my particular case I'm using the mysql provider - if it matters)



entity-framework

ado.net



224

edited Oct 6 '16 at 14:55



Sr Julien

380 1 7 25

asked Sep 11 '09 at 19:33



nos

182k 45 335 439

- 1 [This](#) article from MSDN Magazine describes some profiling options for Entity Framework 4 – [Arve](#) Feb 8 '11 at 20:37
- 2 The linked "duplicate" question is for LINQ to SQL, so its not actually a duplicate. – [jrummell](#) Jul 11 '11 at 15:18
- 2 When running under debugger, IntelliTrace shows SQL queries made, albeit without their results. – [ivan\\_pozdeev](#) Mar 7 '14 at 10:24

If you're interested in seeing the SQL just while development, you can use [LINQPad](#). When you run a LINQ query in the results there will be an SQL tab which shows the executed SQL statement. For mySQL you'll have to install a driver. I don't have a mySQL database available, but it should work. – [gligoran](#) Apr 24 '15 at 12:19

## 18 Answers



You can do the following:

445

```
IQueryable query = from x in appEntities
                    where x.id = 32
                    select x
```

By using our site, you acknowledge that you have read and understand our [Cookie Policy](#), [Privacy Policy](#), and our [Terms of Service](#).



or in EF6:

```
var sql = ((System.Data.Entity.Core.Objects.ObjectQuery)query)
    .ToString();
```

That will give you the SQL that was generated.

edited Mar 3 '16 at 20:22



Chris Moschini

27.6k 15 124 167

answered Sep 11 '09 at 19:42



Nick Berardi

46.7k 13 106 133

20 You won't get SQL for queries ending with .Single(), .Count(), .Any(), etc. that way. – [springy76](#) Feb 27 '13 at 12:31

21 That is because after running .Single() your object is no more IQueryable I guess. – [Suhaz](#) Jun 19 '13 at 7:39

10 with EF6, I could get it only with reflection. but first, I had to convert result to System.Data.Entity.Infrastructure.DbQuery<T> , then get internal property InternalQuery as (System.Data.Entity.Internal.Linq.InternalQuery<T> ) , and only then, use ToString() – [itsho](#) Jul 8 '14 at 11:20

8 add reference to System.Data.Entity, System.Data.Objects.ObjectQuery exist in the above dll – [Mahesh](#) Oct 22 '14 at 7:05

41 In EF6 you can just do result.ToString() – [Scott Chamberlain](#) Sep 2 '15 at 18:42



For those using Entity Framework 6 and up, if you want to view the output SQL in Visual Studio (like I did) you have to use the new logging/interception functionality.

887



Adding the following line will spit out the generated SQL (along with additional execution-related details) in the Visual Studio output panel:

```
using (MyDatabaseEntities context = new MyDatabaseEntities())
{
    context.Database.Log = s => System.Diagnostics.Debug.WriteLine(s);
    // query the database using EF here.
}
```

More information about logging in EF6 in this nifty blog series: <http://blog.oneunicorn.com/2013/05/08/ef6-sql-logging-part-1-simple->

By using our site, you acknowledge that you have read and understand our [Cookie Policy](#), [Privacy Policy](#), and our [Terms of Service](#).

edited Apr 3 '16 at 16:34



PussInBoots

3,665 4 35 64

answered Dec 23 '13 at 21:53



Matt Nibecker

9,084 1 9 9

- 98 This answer deserves more love (if you're using EF6+) - great debug addition, just add it in on the DbContext constructor (this.Database.Log = ...) – [keithl8041](#) Feb 7 '14 at 12:54
- 20 Make sure you are running your project in DEBUG MODE, check if the item "Debug" has selected on combobox of Output pane and also check if your debug is not redirecting to Immediate (Tools > Options > Debugging > Redirect all Output Window text to Immediate Window) – [rkawano](#) May 21 '14 at 17:41
- 4 is there a way to get this to include the variable values directly within the generated sql? Bit of a pain with the bigger ones. – [Chris](#) Jun 28 '14 at 7:02
- 17 @Matt Nibecker This does not work in EF Core. What's the alternative for EF Core? – [nam](#) Sep 16 '16 at 4:13
- 7 WARNING: I implemented this with the intention of it only running in development. When we deployed to our testing environment, we started to abruptly see memory leaks in the IIS Worker Process. After memory profiling, we realized even explicit GC wasn't collecting the entity context objects anymore (yes, they were in using statements). Removing this line returned all to normal. So, while this is a great tool, make sure you only build it into your app for development. – [Brandon Barkley](#) Aug 16 '17 at 17:35

▲ If you are using a DbContext, you can do the following to get the SQL:

77

```
var result = from i in myContext.appEntities
              select new Model
              {
                  field = i.stuff,
              };
var sql = result.ToString();
```

edited Oct 2 '13 at 22:47



Kirk Woll

63.3k 18 161 177

answered Oct 11 '12 at 18:19



Doug Clutter

2,643 2 20 24

- 12 ToString() will give you the query with variables in it, like p\_\_linq\_\_0 , instead of the final values (eg: 34563 instead of p\_\_linq\_\_0 ) – [sports](#) Feb 14 '15 at 22:15

By using our site, you acknowledge that you have read and understand our [Cookie Policy](#), [Privacy Policy](#), and our [Terms of Service](#).

Operations" to a File [here](#)

77

```
<interceptors>
  <interceptor type="System.Data.Entity.Infrastructure.Interception.DatabaseLogger,
EntityFramework">
    <parameters>
      <parameter value="C:\Temp\LogOutput.txt"/>
      <parameter value="true" type="System.Boolean"/>
    </parameters>
  </interceptor>
</interceptors>
```

answered Jul 7 '14 at 7:02



isepise

905 7 5

1 Some more info [msdn.microsoft.com/en-us/data/dn469464.aspx](https://msdn.microsoft.com/en-us/data/dn469464.aspx) – Tim Abell Nov 21 '14 at 10:55

1 Blog post on the subject [blog.oneunicorn.com/2014/02/09/...](http://blog.oneunicorn.com/2014/02/09/...) – Tim Abell Nov 21 '14 at 10:56

10 Precision, It's under : <configuration> <entityFramework> <interceptors> ... </interceptors> </entityFramework> </configuration> – Christophe P Jul 20 '16 at 9:52

Applicable for EF 6.0 and above: For those of you wanting to know more about the logging functionality and adding to the some of the answers already given.

22

Any command sent from the EF to the database can now be logged. To view the generated queries from EF 6.x, use the

DBContext.Database.Log property

## What Gets Logged

- SQL for all different kinds of commands. For example:
  - Queries, including normal LINQ queries, eSQL queries, and raw queries from methods such as `SqlQuery`.
  - Inserts, updates, and deletes generated as part of `SaveChanges`
  - Relationship loading queries such as those generated by lazy loading
  - Parameters

By using our site, you acknowledge that you have read and understand our [Cookie Policy](#), [Privacy Policy](#), and our [Terms of Service](#).

or, for async, was canceled

- **Some** indication of the result value
- **The** approximate amount of time it took to execute the command. **Note** that **this is** the time **from** sending the command to getting the result **object** back. **It** does **not** include time to read the results.

### Example:

```
using (var context = new BlogContext())
{
    context.Database.Log = Console.Write;

    var blog = context.Blogs.First(b => b.Title == "One Unicorn");

    blog.Posts.First().Title = "Green Eggs and Ham";

    blog.Posts.Add(new Post { Title = "I do not like them!" });

    context.SaveChangesAsync().Wait();
}
```

### Output:

```
SELECT TOP (1)
    [Extent1].[Id] AS [Id],
    [Extent1].[Title] AS [Title]
FROM [dbo].[Blogs] AS [Extent1]
WHERE (N'One Unicorn' = [Extent1].[Title]) AND ([Extent1].[Title] IS NOT NULL)
-- Executing at 10/8/2013 10:55:41 AM -07:00
-- Completed in 4 ms with result: SqlDataReader
```

```
SELECT
    [Extent1].[Id] AS [Id],
    [Extent1].[Title] AS [Title],
    [Extent1].[BlogId] AS [BlogId]
FROM [dbo].[Posts] AS [Extent1]
WHERE [Extent1].[BlogId] = @EntityKeyValue1
-- EntityKeyValue1: '1' (Type = Int32)
-- Executing at 10/8/2013 10:55:41 AM -07:00
-- Completed in 2 ms with result: SqlDataReader
```

```
UPDATE [dbo].[Posts]
```

By using our site, you acknowledge that you have read and understand our Cookie Policy, Privacy Policy, and our Terms of Service.

```
-- @1: '1' (Type = Int32)
-- Executing asynchronously at 10/8/2013 10:55:41 AM -07:00
-- Completed in 12 ms with result: 1

INSERT [dbo].[Posts]([Title], [BlogId])
VALUES (@0, @1)
SELECT [Id]
FROM [dbo].[Posts]
WHERE @@ROWCOUNT > 0 AND [Id] = scope_identity()
-- @0: 'I do not like them!' (Type = String, Size = -1)
-- @1: '1' (Type = Int32)
-- Executing asynchronously at 10/8/2013 10:55:41 AM -07:00
-- Completed in 2 ms with result: SqlDataReader
```

### To log to an external file:

```
using (var context = new BlogContext())
{
    using (var sqlLogFile = new StreamWriter("C:\\temp\\LogFile.txt"))
    {
        context.Database.Log = sqlLogFile.Write;
        var blog = context.Blogs.First(b => b.Title == "One Unicorn");
        blog.Posts.First().Title = "Green Eggs and Ham";
        context.SaveChanges();
    }
}
```

More info here: [Logging and Intercepting Database Operations](#)

edited Aug 22 '16 at 5:48

answered Aug 16 '16 at 9:44



NullReference

1,688 1 20 29

You can do the following in EF 4.1:

18

```
var result = from x in appEntities
              where x.id = 32
              select x;
```

By using our site, you acknowledge that you have read and understand our [Cookie Policy](#), [Privacy Policy](#), and our [Terms of Service](#).

That will give you the SQL that was generated.

answered Sep 1 '11 at 8:48



Capriols

189 1 2

- 
- 1 Point of fact, I believe this only works when the query returns an anonymous type. If it returns a custom type, the `ToString()` output is the namespace of that custom type. For example, if the above code was `select new CustomType { x = x.Name }`, the returned value would be something like `Company.Models.CustomType` instead of the generated SQL. – [Chad Levy](#) Sep 15 '11 at 9:57
- 
- 8 This technique produces `System.Data.Objects.ObjectQuery`1[MyProject.Models.Product]` for me. – [Carl G](#) Nov 14 '12 at 17:42
- 
- 1 @CarlG `System.Data.Objects.ObjectQuery` is not EF 4.1 (DbContext). Using DbContext it would be `System.Data.Entity.Infrastructure.DbQuery`1[MyProject.Models.Product]` which indeed outputs it's SQL on a call to `"ToString()"` – [springy76](#) Feb 27 '13 at 12:37
- 

This will give you the SQL that was generated, where, in the output window? which option from the dropdown? – [JsonStatham](#) Dec 19 '18 at 16:43

---

There are two ways:

16

1. To view the SQL that will be generated, simply call `ToTraceString()`. You can add it into your watch window and set a breakpoint to see what the query would be at any given point for any LINQ query.
2. You can attach a tracer to your SQL server of choice, which will show you the final query in all its gory detail. In the case of MySQL, the easiest way to trace the queries is simply to tail the query log with `tail -f`. You can learn more about MySQL's logging facilities in [the official documentation](#). For SQL Server, the easiest way is to use the included SQL Server profiler.

answered Sep 11 '09 at 19:37



Benjamin Pollack

18.4k 15 72 101

---

26 The `ToTraceString` of what ? – [nos](#) Sep 11 '09 at 19:48

---

The `ObjectQuery`, as Nick noted right after I posted my response. – [Benjamin Pollack](#) Sep 11 '09 at 20:55

---

2 SQL Server Profiler captures the first 4000 characters, but EF queries can be much longer than that. – [John Robertson](#) Feb 27 '14 at 20:57

---

By using our site, you acknowledge that you have read and understand our [Cookie Policy](#), [Privacy Policy](#), and our [Terms of Service](#).

14

## Simple

Override the `OnConfiguring` method of your `DbContext` class ( `YourCustomDbContext` ) [as shown here](#) to use a `ConsoleLoggerProvider`; your queries should log to the console:

```
public class YourCustomDbContext : DbContext
{
    #region DefineLoggerFactory
    public static readonly LoggerFactory MyLoggerFactory
        = new LoggerFactory(new[] {new ConsoleLoggerProvider((_, __) => true, true)});
    #endregion

    #region RegisterLoggerFactory
    protected override void OnConfiguring(DbContextOptionsBuilder optionsBuilder)
        => optionsBuilder
            .UseLoggerFactory(MyLoggerFactory); // Warning: Do not create a new
    ILoggerFactory instance each time
    #endregion
}
```

## Complex

This Complex case avoids overriding [the DbContext . OnConfiguring method](#) , which is discouraged in the docs: "This approach does not lend itself to testing, unless the tests target the full database."

This Complex case uses:

- The `IServiceCollection` in `Startup` class `ConfigureServices` method (instead of overriding the `OnConfiguring` method; the benefit is a looser coupling between the `DbContext` and the `ILoggerProvider` you want to use)
- An implementation of `ILoggerProvider` (instead of using the `ConsoleLoggerProvider` implementation shown above; benefit is our implementation shows how we would log to File (I don't see a [File Logging Provider shipped with EF Core](#)))

Like this:

```
public class Startup
{
    public void ConfigureServices(IServiceCollection services)
    {
        ...
    }
}
```

By using our site, you acknowledge that you have read and understand our [Cookie Policy](#), [Privacy Policy](#), and our [Terms of Service](#).



```

services.AddDbContext<YOUR_DB_CONTEXT>(optionsBuilder => optionsBuilder
    .UseSqlServer(connection_string)
    //Using the LoggerFactory
    .UseLoggerFactory(lf));
    ...
}

```

Here's the implementation of a `MyLoggerProvider` (and its `MyLogger` which appends its logs to a File you can configure; your EF Core queries will appear in the file.)

```

public class MyLoggerProvider : ILoggerProvider
{
    public ILogger CreateLogger(string categoryName)
    {
        return new MyLogger();
    }

    public void Dispose()
    { }

    private class MyLogger : ILogger
    {
        public bool IsEnabled(LogLevel logLevel)
        {
            return true;
        }

        public void Log<TState>(LogLevel logLevel, EventId eventId, TState state,
Exception exception, Func<TState, Exception, string> formatter)
        {
            File.AppendAllText(@"C:\temp\log.txt", formatter(state, exception));
            Console.WriteLine(formatter(state, exception));
        }

        public IDisposable BeginScope<TState>(TState state)
        {
            return null;
        }
    }
}

```

edited Jun 12 '18 at 17:26

answered Jun 1 '17 at 0:48

By using our site, you acknowledge that you have read and understand our [Cookie Policy](#), [Privacy Policy](#), and our [Terms of Service](#).

So...there is no beginners way of doing it? – [Juan De la Cruz](#) Feb 18 '18 at 0:07

1 @JuanDelaCruz I simplified my answer; try the simple alternative – [The Red Pea](#) Feb 18 '18 at 0:27

To have the query always handy, without changing code add this to your DbContext and check it on the output window in visual studio.

6

```
protected override void OnModelCreating(DbModelBuilder modelBuilder)
{
    Database.Log = (query) => Debug.Write(query);
}
```

Similar to @Matt Nibecker answer, but with this you do not have to add it in your current code, every time you need the query.

answered Jan 21 '17 at 18:53



[Gerrie Pretorius](#)

2,226 1 20 30

The best answer! – [AlexSC](#) Feb 19 at 11:59

Well, I am using Express profiler for that purpose at the moment, the drawback is that it only works for MS SQL Server. You can find this tool here: <https://expressprofiler.codeplex.com/>

5

answered Dec 23 '14 at 9:19



[VincentZHANG](#)

448 7 27

5

```
IQueryable query = from x in appEntities
                    where x.id = 32
                    select x;
var queryString = query.ToString();
```

By using our site, you acknowledge that you have read and understand our [Cookie Policy](#), [Privacy Policy](#), and our [Terms of Service](#).



I just tried this and it traces out the object:

Microsoft.EntityFrameworkCore.Query.Internal.EntityQueryable 1[System.Linq.IGrouping 2[System.Int32,String]] instead of the actual query.  
Am I missing something or did you forget to mention something? – [loganjones16](#) Feb 27 at 22:11

4 I am doing integration test, and needed this to debug the generated SQL statement in Entity Framework Core 2.1, so I use  
DebugLoggerProvider or ConsoleLoggerProvider like so:

```
[Fact]
public async Task MyAwesomeTest
{
    //setup log to debug sql queries
    var loggerFactory = new LoggerFactory();
    loggerFactory.AddProvider(new DebugLoggerProvider());
    loggerFactory.AddProvider(new ConsoleLoggerProvider(new
ConsoleLoggerSettings()));

    var builder = new DbContextOptionsBuilder<DbContext>();
    builder
        .UseSqlServer("my connection string") //"Server=.;Initial
Catalog=TestDb;Integrated Security=True"
        .UseLoggerFactory(loggerFactory);

    var dbContext = new DbContext(builder.Options);

    .....
```

Here is a sample output from Visual Studio console:

The screenshot shows the Visual Studio IDE with a C# file named `GetHtmlControlPermissionHandler.cs` in the `Application.API.Handlers.Form` namespace. The class inherits from `BaseHandler<GetHtmlControlPermissionRequest, HtmlControlPermissionModel>` and has a constructor that takes `PtwDbContext` and `IUserIdentity` as parameters. The `Handle` method is currently selected.

The `Output` window at the bottom shows the SQL query generated by the Entity Framework. The query is a complex join involving `WorkflowStateControls`, `HtmlControls`, `WorkflowStates`, and `Workflows` tables, filtered by tenant ID and active status. The query is executed via `Microsoft.EntityFrameworkCore.Database.Command: Executed DbCommand (191ms)` with parameters for tenant ID and workflow name.

```

SELECT TOP(2) [e].[DefaultMode]
FROM [WorkflowStateControls] AS [e]
INNER JOIN (
    SELECT [e0].*
    FROM [HtmlControls] AS [e0]
    WHERE ((([e0].[IsDeleted] = 0) AND ([e0].[TenantId] = @_ef_filter__CurrentTenantId_1)) AND ([e0].[IsActive] = 1)
) AS [t] ON [e].[HtmlControlId] = [t].[Id]
INNER JOIN (
    SELECT [e1].*
    FROM [WorkflowStates] AS [e1]
    WHERE ((([e1].[IsDeleted] = 0) AND ([e1].[TenantId] = @_ef_filter__CurrentTenantId_2)) AND ([e1].[IsActive] = 1)
) AS [t0] ON [e].[WorkflowStateId] = [t0].[Id]
INNER JOIN (
    SELECT [e2].*
    FROM [Workflows] AS [e2]
    WHERE ((([e2].[IsDeleted] = 0) AND ([e2].[TenantId] = @_ef_filter__CurrentTenantId_3)) AND ([e2].[IsActive] = 1)
) AS [t1] ON [t0].[WorkflowId] = [t1].[Id]
WHERE ((([e].[IsDeleted] = 0) AND ([e].[TenantId] = @_ef_filter__CurrentTenantId_0)) AND ([e].[IsActive] = 1)) AND ((([t1].[WorkflowName] = @_workflowName_0) AND ([t0].[StateStatusFri
Microsoft.EntityFrameworkCore.Database.Command: Executed DbCommand (191ms) [Parameters=[@_ef_filter__CurrentTenantId_1=?' (DbType = Guid), @_ef_filter__CurrentTenantId_2=
SELECT TOP(2) [e].[DefaultMode]
FROM [WorkflowStateControls] AS [e]
INNER JOIN (
    SELECT [e0].*
    FROM [HtmlControls] AS [e0]
    WHERE ((([e0].[IsDeleted] = 0) AND ([e0].[TenantId] = @_ef_filter__CurrentTenantId_1)) AND ([e0].[IsActive] = 1)
) AS [t] ON [e].[HtmlControlId] = [t].[Id]
INNER JOIN (
    SELECT [e1].*
  
```

edited Oct 13 '18 at 5:15

answered Oct 7 '18 at 8:36



Rosdi Kasim

12.9k 16 90 127

DebuggerProvider and ConsoleLoggerProvider seem to exist only in .NET Core: [docs.microsoft.com/en-us/dotnet/api/...](https://docs.microsoft.com/en-us/dotnet/api/...) – Gabriel Magana Jun 20 at 14:09

By using our site, you acknowledge that you have read and understand our [Cookie Policy](#), [Privacy Policy](#), and our [Terms of Service](#).

3

This page is the first search result when searching for a solution for any .NET Framework, so here as a public service, how it's done in EntityFrameworkCore (for .NET Core 1 & 2):

```
var someQuery = (
    from projects in _context.projects
    join issues in _context.issues on projects.Id equals issues.ProjectId into tmpMapp
    from issues in tmpMapp.DefaultIfEmpty()
    select issues
) //.ToList()
;

// string sql = someQuery.ToString();
// string sql = Microsoft.EntityFrameworkCore.IQueryableExtensions.ToSql(someQuery);
// string sql = Microsoft.EntityFrameworkCore.IQueryableExtensions1.ToSql(someQuery);
// using Microsoft.EntityFrameworkCore;
string sql = someQuery.ToSql();
System.Console.WriteLine(sql);
```

And then these extension methods (IQueryableExtensions1 for .NET Core 1.0, IQueryableExtensions for .NET Core 2.0) :

```
using System;
using System.Linq;
using System.Reflection;
using Microsoft.EntityFrameworkCore.Internal;
using Microsoft.EntityFrameworkCore.Query;
using Microsoft.EntityFrameworkCore.Query.Internal;
using Microsoft.EntityFrameworkCore.Storage;
using Remotion.Linq.Parsing.Structure;

namespace Microsoft.EntityFrameworkCore
{
    // https://stackoverflow.com/questions/1412863/how-do-i-view-the-sql-generated-by-the-entity-framework
    // http://rion.io/2016/10/19/accessing-entity-framework-core-queries-behind-the-scenes-in-asp-net-core/

    public static class IQueryableExtensions
    {
        private static readonly TypeInfo QueryCompilerTypeInfo =
            typeof(QueryCompiler).GetTypeInfo();
    }
```

By using our site, you acknowledge that you have read and understand our Cookie Policy, Privacy Policy, and our Terms of Service.

```

        private static readonly PropertyInfo NodeTypeProviderField =
            QueryCompilerTypeInfo.DeclaredProperties.Single(x => x.Name ==
"NodeTypeProvider");

        private static readonly MethodInfo CreateQueryParserMethod =
            QueryCompilerTypeInfo.DeclaredMethods.First(x => x.Name ==
"CreateQueryParser");

        private static readonly FieldInfo DataBaseField =
            QueryCompilerTypeInfo.DeclaredFields.Single(x => x.Name == "_database");

        private static readonly PropertyInfo DatabaseDependenciesField =
            typeof(Database).GetTypeInfo().DeclaredProperties.Single(x => x.Name ==
"Dependencies");

        public static string ToSql<TEntity>(this IQueryable<TEntity> query) where
TEntity : class
        {
            if (!(query is EntityQueryable<TEntity>) && !(query is
InternalDbSet<TEntity>))
            {
                throw new ArgumentException("Invalid query");
            }

            var queryCompiler = (QueryCompiler)
QueryCompilerField.GetValue(query.Provider);
            var nodeTypeProvider = (INodeTypeProvider)
NodeTypeProviderField.GetValue(queryCompiler);
            var parser = (IQueryParser) CreateQueryParserMethod.Invoke(queryCompiler,
new object[] {nodeTypeProvider});
            var queryModel = parser.GetParsedQuery(query.Expression);
            var database = DataBaseField.GetValue(queryCompiler);
            var databaseDependencies = (DatabaseDependencies)
DatabaseDependenciesField.GetValue(database);
            var queryCompilationContext =
databaseDependencies.QueryCompilationContextFactory.Create(false);
            var modelVisitor = (RelationalQueryModelVisitor)
queryCompilationContext.CreateQueryModelVisitor();
            modelVisitor.CreateQueryExecutor<TEntity>(queryModel);
            var sql = modelVisitor.Queries.First().ToString();

            return sql;
        }
    }

```

By using our site, you acknowledge that you have read and understand our [Cookie Policy](#), [Privacy Policy](#), and our [Terms of Service](#).

```

{
    private static readonly TypeInfo QueryCompilerTypeInfo =
typeof(QueryCompiler).GetTypeInfo();

    private static readonly FieldInfo QueryCompilerField =
typeof(EntityQueryProvider).GetTypeInfo()
        .DeclaredFields
        .First(x => x.Name == "_queryCompiler");

    private static readonly PropertyInfo NodeTypeProviderField =
        QueryCompilerTypeInfo.DeclaredProperties.Single(x => x.Name ==
"NodeTypeProvider");

    private static readonly MethodInfo CreateQueryParserMethod =
        QueryCompilerTypeInfo.DeclaredMethods.First(x => x.Name ==
"CreateQueryParser");

    private static readonly FieldInfo DataBaseField =
        QueryCompilerTypeInfo.DeclaredFields.Single(x => x.Name == "_database");

    private static readonly FieldInfo QueryCompilationContextFactoryField =
typeof(Database).GetTypeInfo()
        .DeclaredFields.Single(x => x.Name == "_queryCompilationContextFactory");

    public static string ToSql<TEntity>(IQueryable<TEntity> query) where TEntity :
class
    {
        if (!(query is EntityQueryable<TEntity>) && !(query is
InternalDbSet<TEntity>))
        {
            throw new ArgumentException("Invalid query");
        }

        var queryCompiler = (IQueryCompiler)
QueryCompilerField.GetValue(query.Provider);

        var nodeTypeProvider = (INodeTypeProvider)
NodeTypeProviderField.GetValue(queryCompiler);
        var parser =
            (IQueryParser) CreateQueryParserMethod.Invoke(queryCompiler, new
object[] {nodeTypeProvider});
        var queryModel = parser.GetParsedQuery(query.Expression);
        var database = DataBaseField.GetValue(queryCompiler);
        var queryCompilationContextFactory =
            (IQueryCompilationContextFactory)

```

By using our site, you acknowledge that you have read and understand our Cookie Policy, Privacy Policy, and our Terms of Service.

```
queryCompilationContext.CreateQueryModelVisitor();
    modelVisitor.CreateQueryExecutor<TEntity>(queryModel);
    var sql = modelVisitor.Queries.First().ToString();

    return sql;
}

}

}
```

answered Feb 13 '18 at 18:58

**Stefan Steiger****48.9k** 57 287 374

I am using EF Core 2.0.1 and the above suggestion results in: System.InvalidCastException: 'Unable to cast object of type Microsoft.EntityFrameworkCore.Query.Internal.InMemoryQueryModelVisitor' to type 'Microsoft.EntityFrameworkCore.Query.RelationalQueryModelVisitor' for the line: var modelVisitor = (RelationalQueryModelVisitor) queryCompilationContext.CreateQueryModelVisitor(); – [Chris Wolf](#) Apr 18 '18 at 16:52

- 1 @ChrisWolf if you follow the original author's gist you can find somebody who [provided an updated version of that extension method](#). Worked for me. – [B12Toaster](#) Oct 5 '18 at 20:24



SQL Management Studio =&gt; Tools =&gt; SQL Server profiler

3

File =&gt; New Trace...



Use the Template =&gt; Blank

Event selection =&gt; T-SQL

Lefthandside check for: SP.StmtComplete

Column filters can be used to select a specific ApplicationName or DatabaseName

Start that profile running then trigger the query.

By using our site, you acknowledge that you have read and understand our [Cookie Policy](#), [Privacy Policy](#), and our [Terms of Service](#).





sorry thats just for SQL server, not MySQL – andrew pate Mar 6 at 17:31

In my case for EF 6+, instead of using this in the Immediate Window to find the query string:

2 `var sql = ((System.Data.Entity.Core.Objects.ObjectQuery)query).ToTraceString();`

I ended up having to use this to get the generated SQL command:

```
var sql =  
((System.Data.Entity.Infrastructure.DbQuery<>f__AnonymousType3<string,string,string,short
```

Of course your anonymous type signature might be different.

HTH.



I've just done this:

2 `IQueryable<Product> query = EntitySet.Where(p => p.Id == id);  
Debug.WriteLine(query);`

And the result shown in the **Output**:


```
[Extent1].[Name] AS [Name],
[Extent2].[Id] AS [Id1],
[Extent2].[FileName] AS [FileName],
FROM [dbo].[Products] AS [Extent1]
INNER JOIN [dbo].[PersistedFiles] AS [Extent2] ON [Extent1].[PersistedFileId] =
[Extent2].[Id]
WHERE [Extent1].[Id] = @p__linq__0
```

answered Jun 22 '16 at 4:27

 **Daniel Camargo**  
2,788 20 39

Yes, but i believe that Nobody wants to see the p\_\_linq\_\_i , but the real values – [Tom Stickel](#) Oct 2 '18 at 0:30



This way still works in EF 6 and it will be helpful if you only care about what the query structure look like. In my case the project I create the IQueryable<T> object does not have reference to System.Data.Entity nor I want to add it just for debugging purpose. So this method worked just fine. – [wctiger](#) Nov 8 '18 at 22:05

  
**2**  


For me, using EF6 and Visual Studio 2015 I entered `query` in the immediate window and it gave me the generated SQL Statement


answered Jul 21 '16 at 15:28

 **Jonas Stawski**  
3,800 4 52 98

  
**1**  


If you want to have parameter values (not only `@p__linq__0` but also their values) too, you can use [IDbCommandInterceptor](#) and add some logging to `ReaderExecuted` method.

edited Jan 27 '17 at 11:27

 **Palec**  
8,299 6 43 94

answered Jan 27 '17 at 10:12

 **jabko87**  
1,933 1 15 24

2 Can you show an example? – [Tom Stickel](#) Oct 2 '18 at 0:29

By using our site, you acknowledge that you have read and understand our [Cookie Policy](#), [Privacy Policy](#), and our [Terms of Service](#).