Practica 4: Data Base First

Contenido

1. Crear una base de datos existente

Normalmente, su objetivo es una base de datos existente que ya está creada pero en este tutorial necesitamos crear una base de datos para obtener acceso.

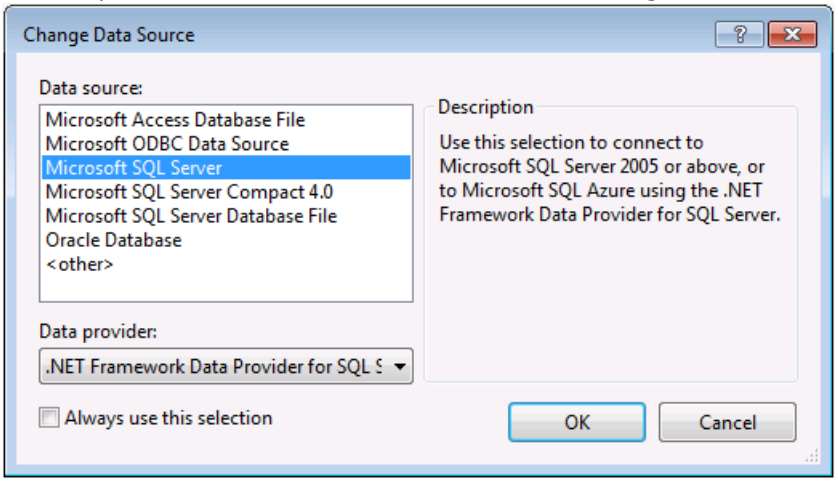
El servidor de bases de datos que se instala con Visual Studio varía en función de la versión de Visual Studio que se haya instalado:

- Si usa Visual Studio 2010, creará una base de datos de SQL Express.
- Si usa Visual Studio 2012, creará una base de datos de LocalDb.

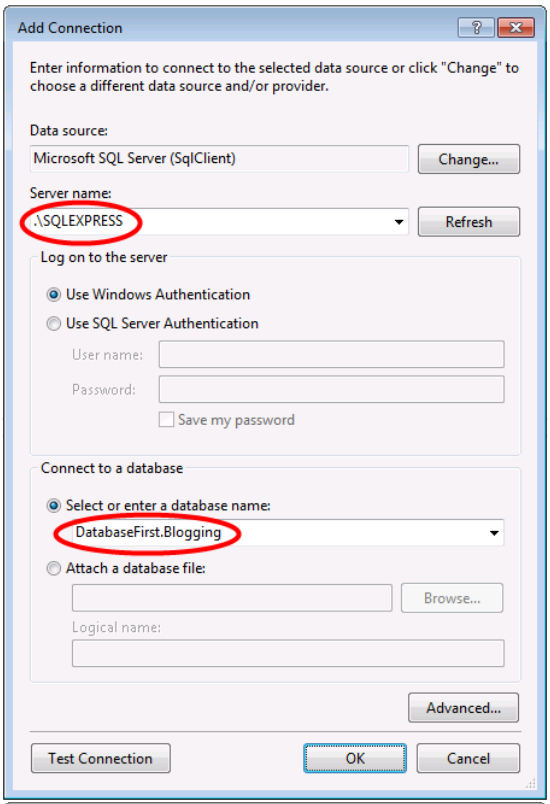
Continuemos y generemos la base de datos.

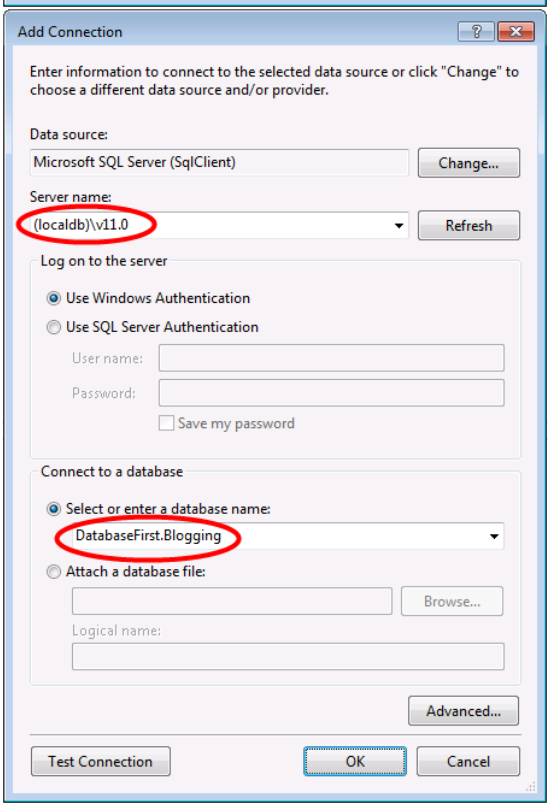
- Abra Visual Studio
- **Ver -> Explorador de servidores**
- Haga clic con el botón secundario en **Conexiones de datos -> Agregar conexión**

- Si no se ha conectado a una base de datos desde el Explorador de servidores antes, tendrá que seleccionar Microsoft SQL Server como origen de datos

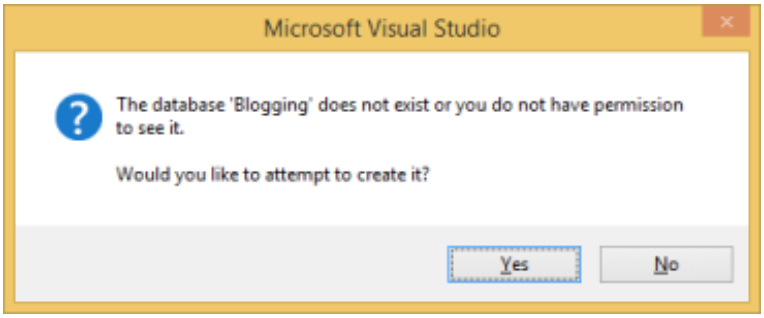


- Conéctese a LocalDb (**(localdb)\v11.0**) o a SQL Express (**.\SQLEXPRESS**), dependiendo de cuál haya instalado, y escriba **DatabaseFirst.Blogging** como nombre de la base de datos


Add Connection
Enter information to connect to the selected data source or click "Change" to choose a different data source and/or provider.
Data source:
Microsoft SQL Server (SqlClient)
Change...
Server name:
.\SQLEXPRESS
Refresh
Log on to the server
Use Windows Authentication
Use SQL Server Authentication
User name:
Password:
Save my password
Connect to a database
Select or enter a database name:
DatabaseFirst.Blogging
Attach a database file:
Browse...
Logical name:
Advanced...
Test Connection
OK
Cancel


Add Connection
Enter information to connect to the selected data source or click "Change" to choose a different data source and/or provider.
Data source:
Microsoft SQL Server (SqlClient)
Change...
Server name:
(localdb)\v11.0
Refresh
Log on to the server
Use Windows Authentication
Use SQL Server Authentication
User name:
Password:
Save my password
Connect to a database
Select or enter a database name:
DatabaseFirst.Blogging
Attach a database file:
Browse...
Logical name:
Advanced...
Test Connection
OK
Cancel

- Seleccione **Aceptar** y se le preguntará si desea crear una base de datos nueva; seleccione **Sí**



- La nueva base de datos ahora aparecerá en el Explorador de servidores, haga clic con el botón secundario en ella y seleccione **Nueva consulta**
- Copie el código SQL siguiente en la consulta, haga clic con el botón secundario en la consulta y seleccione **Ejecutar**

```
CREATE TABLE [dbo].[Blogs] (
    [BlogId] INT IDENTITY (1, 1) NOT NULL,
    [Name] NVARCHAR (200) NULL,
    [Url]  NVARCHAR (200) NULL,
    CONSTRAINT [PK_dbo.Blogs] PRIMARY KEY CLUSTERED ([BlogId] ASC)
);

CREATE TABLE [dbo].[Posts] (
    [PostId] INT IDENTITY (1, 1) NOT NULL,
    [Title] NVARCHAR (200) NULL,
    [Content] NTEXT NULL,
    [BlogId] INT NOT NULL,
    CONSTRAINT [PK_dbo.Posts] PRIMARY KEY CLUSTERED ([PostId] ASC),
    CONSTRAINT [FK_dbo.Posts_dbo.Blogs_BlogId] FOREIGN KEY ([BlogId])
REFERENCES [dbo].[Blogs] ([BlogId]) ON DELETE CASCADE
);
```

2. Crear la aplicación

Para mantener la simplicidad, vamos a generar una aplicación de consola básica que use Database First para el acceso a los datos:

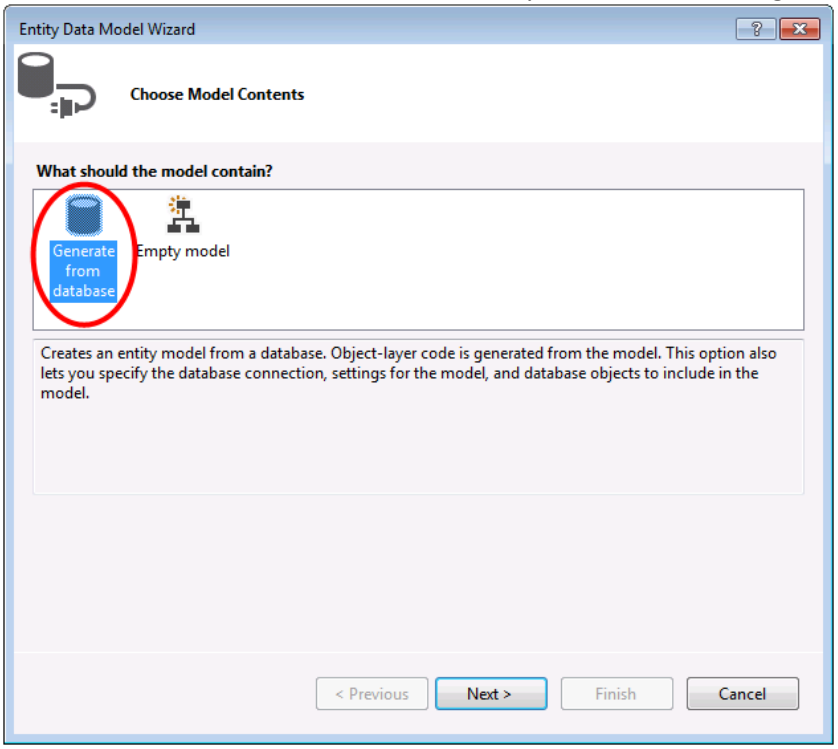
- Abra Visual Studio
- **Archivo -> Nuevo -> Proyecto**
- Seleccione **Windows** en el menú de la izquierda y **aplicación de consola**
- Escriba **DatabaseFirstSample** como nombre
- Seleccione **Aceptar**

3. Modelo de ingeniería inversa

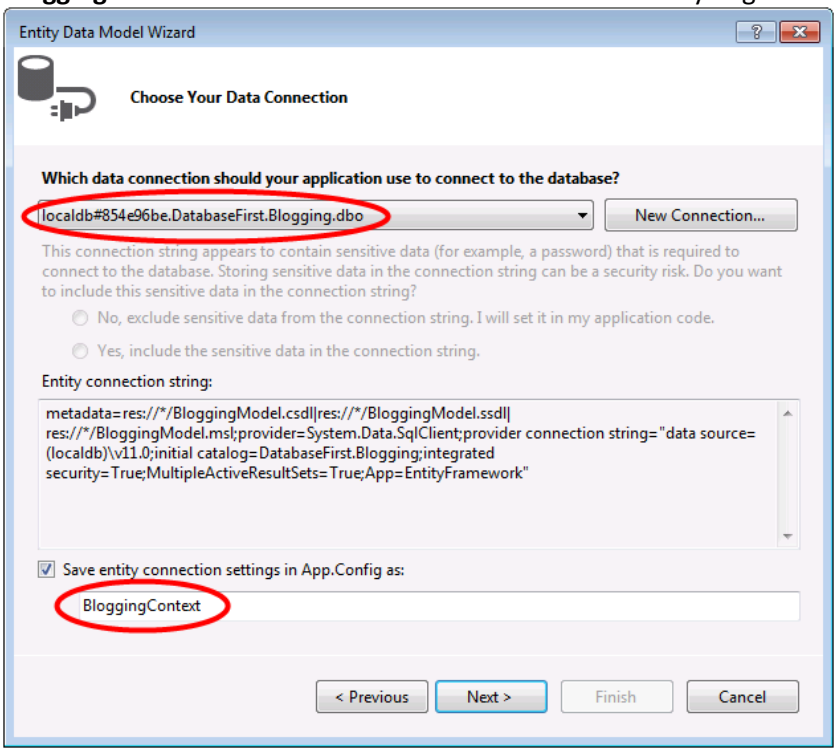
Vamos a usar Entity Framework Designer, que se incluye como parte de Visual Studio, para crear nuestro modelo.

- **Proyecto -> Agregar nuevo elemento**
- Seleccione **Datos** en el menú izquierdo y después **Entity Data Model de ADO.NET**

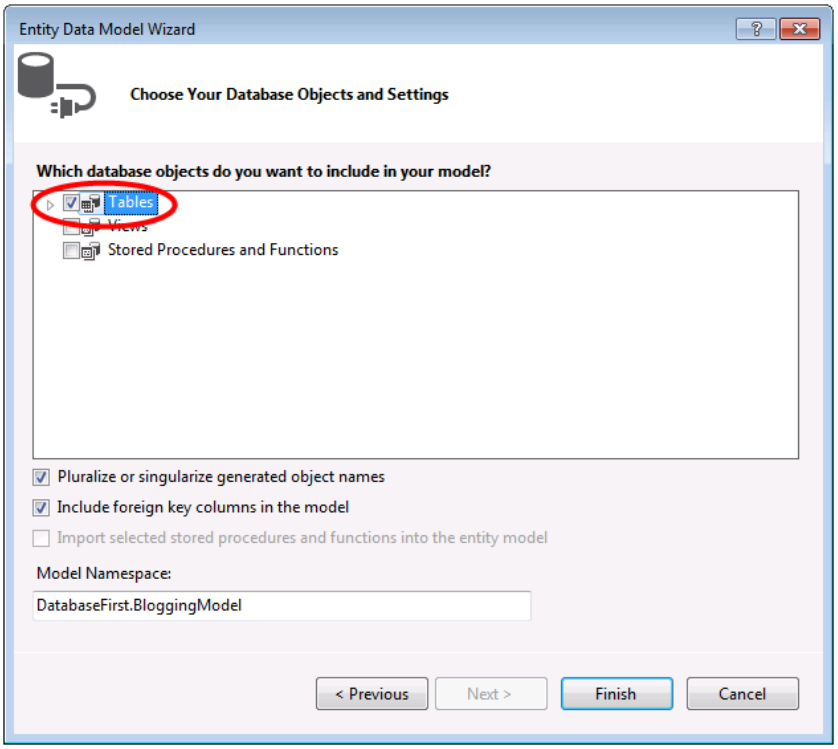
- Escriba **BloggingModel** como nombre y haga clic en **Aceptar**
- De este modo se inicia el **Asistente para Entity Data Model**
- Seleccione **Generar desde la base de datos** y, a continuación, haga clic en **Siguiente**



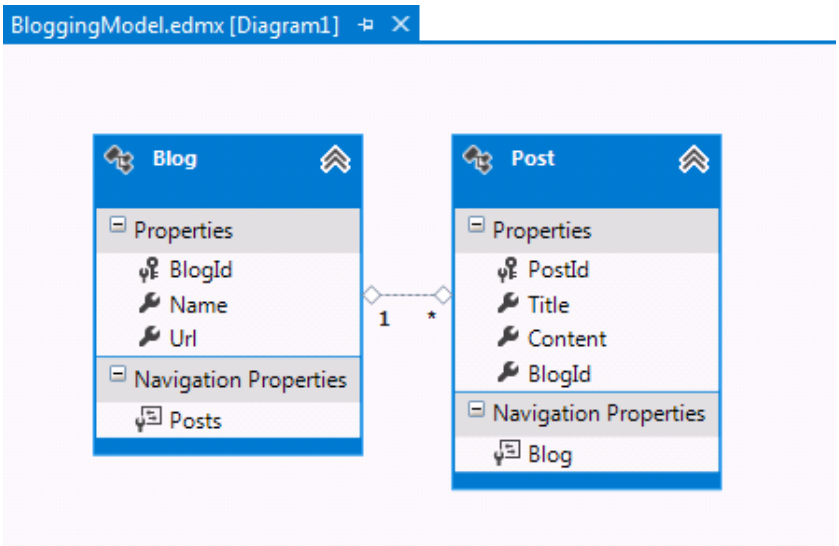
- Seleccione la conexión a la base de datos que creó en la primera sección, escriba **BloggingContext** como nombre de la cadena de conexión y haga clic en **Siguiente**



- Haga clic en la casilla junto a 'Tablas' para importar todas las tablas y haga clic en 'Finalizar'



Una vez que el proceso de ingeniería inversa se completa, el nuevo modelo se agrega al proyecto y se abre para que se vea en Entity Framework Designer. Un archivo App.config también se ha agregado al proyecto con los detalles de la conexión para la base de datos.



Pasos adicionales en Visual Studio 2010

Si está trabajando en Visual Studio 2010, hay algunos pasos adicionales que debe seguir para actualizar a la versión más reciente de Entity Framework. La actualización es

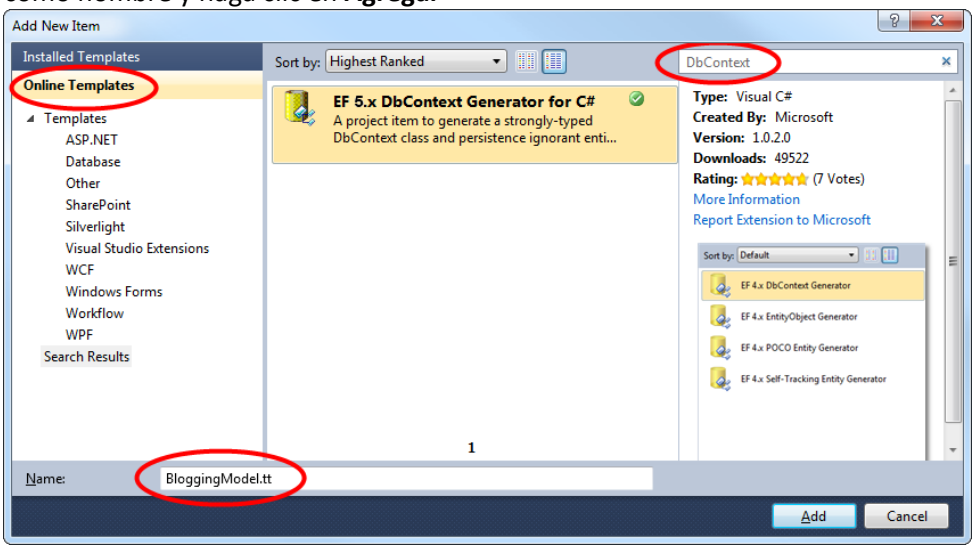
importante porque proporciona acceso a una superficie mejorada de la API, que es mucho más fácil de usar, así como a las correcciones de errores más recientes.

Primero, necesitamos obtener la versión más reciente de Entity Framework de NuGet.

- **Proyecto –> Administrar paquetes de NuGet**
  *Si no dispone de la opción **Administrar paquetes de NuGet**, debe instalar la versión más reciente de NuGet*
- Select the **Online**
- Select the **EntityFramework**
- Haga clic en **Instalar**.

A continuación, tenemos que intercambiar nuestro modelo para generar el código que usa la API DbContext, que se incluyó en las versiones posteriores de Entity Framework.

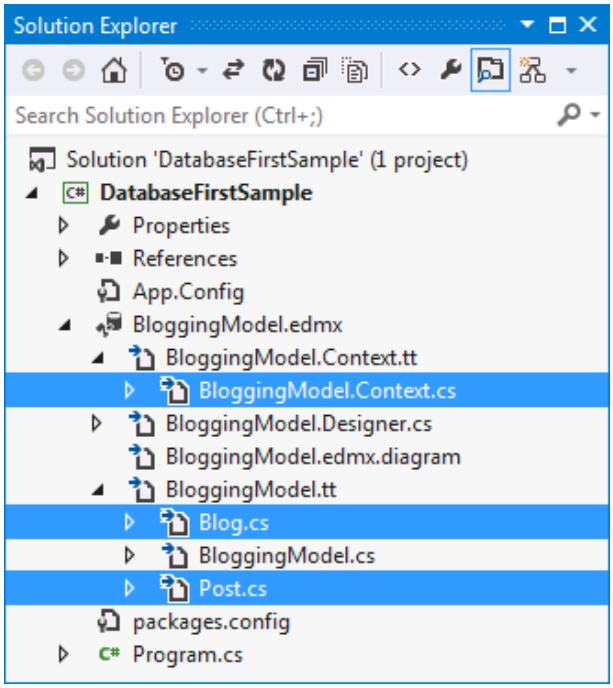
- Haga clic con el botón secundario en un punto vacío del modelo en el EF Designer y seleccione **Agregar elemento de generación de código**
- Seleccione **Plantillas en línea** en el menú de la izquierda y busque **DbContext**
- Seleccione el **Generador de DbContext de EF 5.x para C#**, escriba **BloggingModel** como nombre y haga clic en **Agregar**



4. Leer y escribir datos

Ahora que tenemos un modelo, es hora de usarlo para tener acceso a algunos datos. Las clases que vamos a usar para tener acceso a los datos se generan automáticamente según el archivo EDMX.

Esta captura de pantalla es de Visual Studio 2012, si usa Visual Studio 2010, los archivos BloggingModel.tt y BloggingModel.Context.tt estarán directamente debajo del proyecto en lugar de anidados en el archivo EDMX.



Implemente el método Main en Program.cs como se muestra a continuación. Este código crea una nueva instancia de nuestro contexto y la usa para insertar un nuevo blog. Entonces, usa una consulta LINQ para recuperar todos los blogs de la base de datos ordenados alfabéticamente por el título.

```
class Program
{
    static void Main(string[] args)
    {
        using (var db = new BloggingContext())
        {
            // Create and save a new Blog
            Console.Write("Enter a name for a new Blog: ");
            var name = Console.ReadLine();

            var blog = new Blog { Name = name };
            db.Blogs.Add(blog);
            db.SaveChanges();

            // Display all Blogs from the database
            var query = from b in db.Blogs
                        orderby b.Name
                        select b;

            Console.WriteLine("All blogs in the database:");
            foreach (var item in query)
            {
                Console.WriteLine(item.Name);
            }

            Console.WriteLine("Press any key to exit...");
            Console.ReadKey();
        }
```

```
    }
}
```

Ahora puede ejecutar la aplicación y probarla.

```
Escriba un nombre para el nuevo blog: Blog de ADO.NET
Todos los blogs de la base de datos:
Blog de ADO.NET
Presione cualquier tecla para salir.
```

5. Tratar los cambios en la base de datos

Es el momento de realizar algunos cambios en nuestro esquema de la base de datos; cuando realicemos dichos cambios, también necesitaremos actualizar nuestro modelo para reflejarlos.

El primer paso es ejecutar algunos cambios en el esquema de la base de datos. Vamos a agregar una tabla de usuarios al esquema.

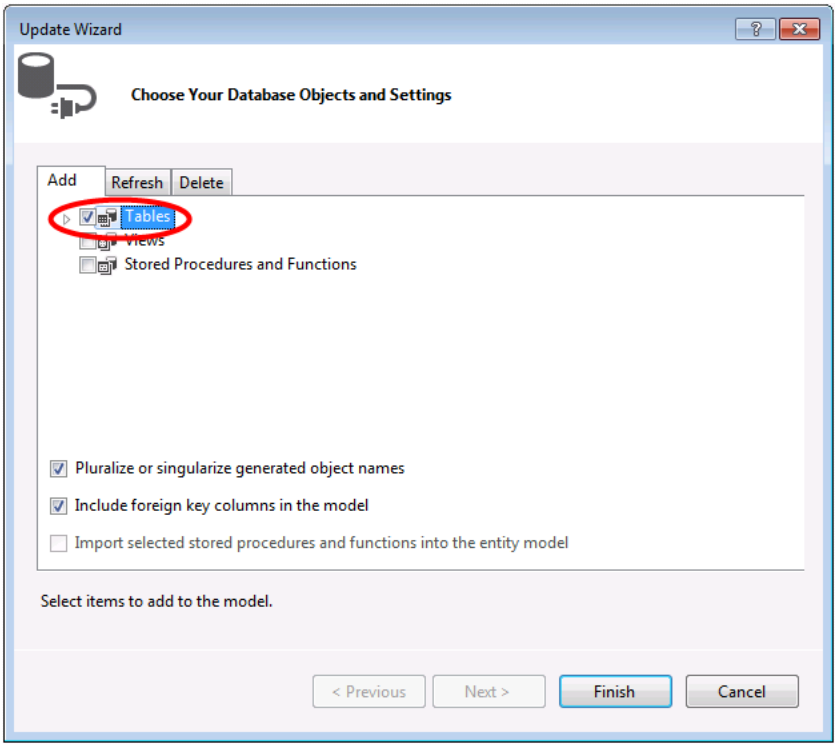
- Haga clic con el botón secundario en la base de datos **DatabaseFirst.Blogging** en el Explorador de servidores y seleccione **Nueva consulta**
- Copie el código SQL siguiente en la consulta, haga clic con el botón secundario en la consulta y seleccione **Ejecutar**

```
CREATE TABLE [dbo].[Users]
(
        [Username] NVARCHAR(50) NOT NULL PRIMARY KEY,
    [DisplayName] NVARCHAR(MAX) NULL
)
```

Ahora que se actualiza el esquema, es hora de actualizar el modelo con esos cambios.

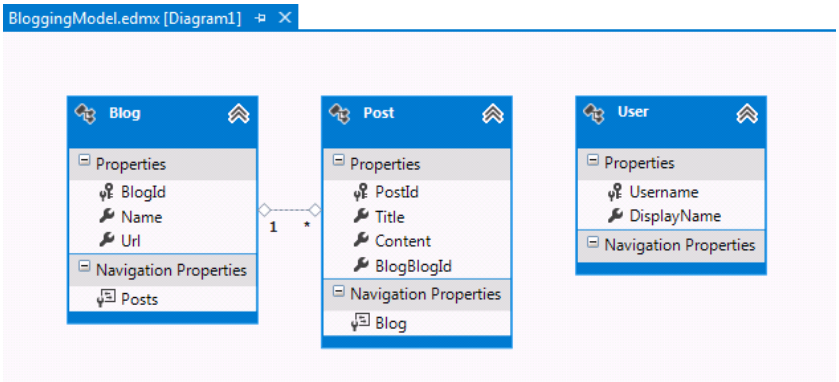
- Haga clic con el botón secundario en un punto vacío de EF Designer y seleccione "Actualizar modelo desde base de datos"; así se iniciará el Asistente para actualizar
- En la pestaña Agregar de la casilla de verificación del Asistente para actualizar junto a Tablas, esto indica que deseamos agregar una nueva tabla de esquema.

  La pestaña Actualización muestra las tablas existentes en el modelo en las que se comprobará si hay cambios durante la actualización. Las pestañas Eliminar muestran las tablas que se han quitado del esquema y también se quitarán del modelo como parte de la actualización. La información de estas dos pestañas se detecta de forma automática y se proporciona solo con fines informativos, no puede cambiar los valores.



- Haga clic en Finalizar en el Asistente para actualizar

El modelo se actualiza para incluir una nueva entidad de usuario que se asigna a la tabla de usuarios que agregamos a la base de datos.



Resumen

En este tutorial examinamos el desarrollo de Database First, que nos permitió crear un modelo en EF Designer a partir de una base de datos existente. Entonces usamos ese modelo para leer y escribir algunos datos de la base de datos. Finalmente, actualizamos el modelo para reflejar los cambios que realizamos en el esquema de la base de datos.