Capítulo 5
Entity Framework avanzado

Objetivo

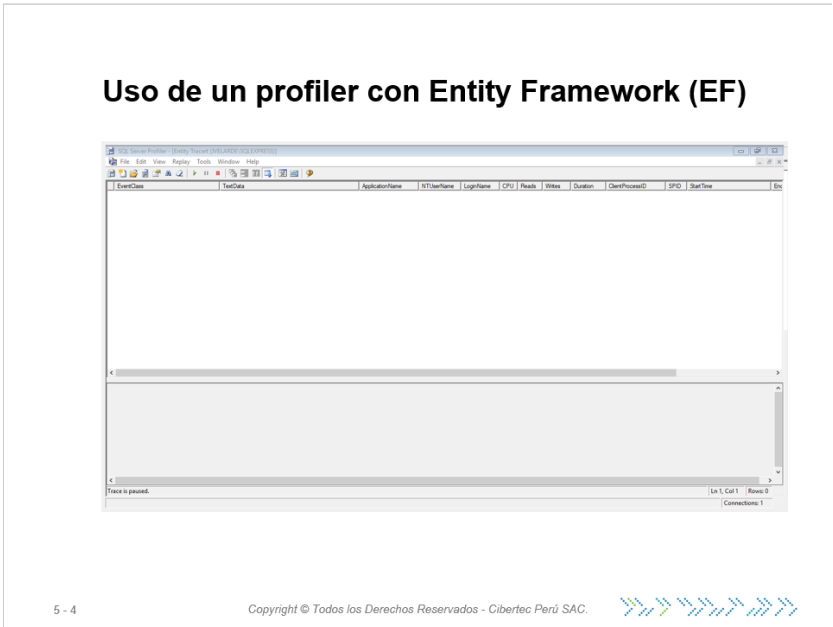
Al finalizar el capítulo, el alumno:

- Comprende el funcionamiento interno de Entity Framework.
- Ejecuta consultas SQL con Entity Framework.
- Aplica configuración aplicada para manipular el contexto de datos.

Temas

1. Uso de un profiler con EF
2. Deferred execution vs Immediate execution
3. Lazy loading vs Eager loading
4. Add/Attach
5. Revisión Entity Framework 7.0

1. Uso de un profiler con Entity Framework



Hasta el momento se conocen las ventajas que presenta el uso de Entity Framework en cuanto a la facilidad de desarrollo. Pero, podemos caer en problemas que podrían darnos la sensación de que EF no es una buena alternativa. Uno de los problemas más comunes está relacionado con la performance de EF. Nosotros utilizamos LINQ para consultar nuestro modelo de entidades y le dejamos a EF la responsabilidad de construir consultas SQL contra la base de datos y ejecutarlas, muchas veces estas consultas SQL no son las que queremos generar/ejecutar y esto es muchas veces lo que genera los problemas de performance.

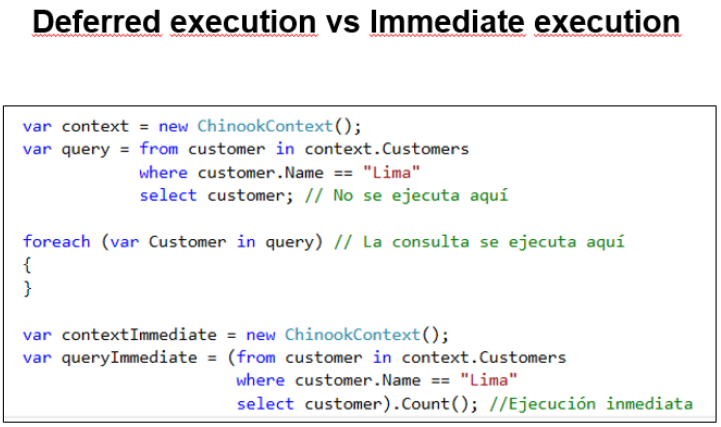
Para saber qué consultas ejecuta EF podemos utilizar productos conocidos como profilers. Estos profilers nos brindan la información de qué consultas se ejecutan en base de datos. Para SQL Server se tiene el SQL Server Profiler que viene como parte del producto. Si se utiliza el SQL Server express se puede utilizar el Express Profiler que es una aplicación independiente y que se puede descargar de manera gratuita.
El profiler necesita tener los datos de la instancia de base de datos, la base de datos y los eventos que desea monitorear. Una vez iniciado debemos ejecutar nuestra aplicación para poder ver el listado de sentencias que ejecuta EF.

Adicionalmente, a la opción de utilizar un profiler se puede realizar logging de las sentencias que EF genera. Esta configuración se puede hacer por medio de código utilizando la propiedad Log de nuestro DbContext actual:

```
using (var context = new BlogContext())
{
    context.Database.Log = Console.Write;
   //Código que utilice el context de EF
}
```

Lo que hace el código anterior es enviar las sentencias SQL a la consola de Visual Studio. También, se puede agregar código para que el resultado no vaya a la consola, sino a cualquier otro medio. Para más referencia ver:
http://blog.oneunicorn.com/2013/05/08/ef6-sql-logging-part-1-simple-logging/

2. Deferred execution vs Immediate execution

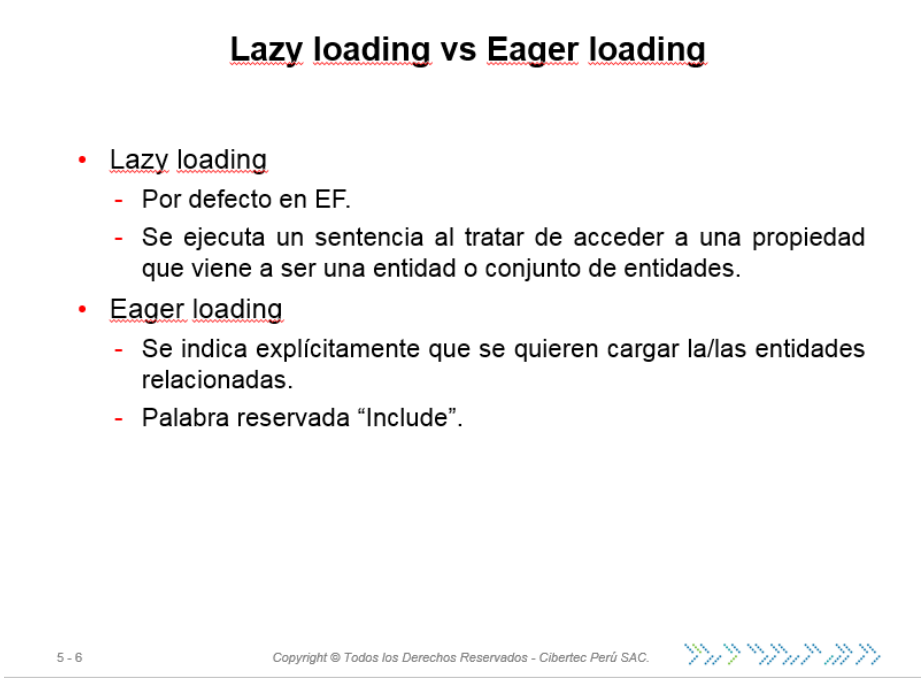


Una variable de query nunca contiene los resultados del query, solo almacena los comandos. La ejecución de una query es diferida hasta que la variable de query es consumida / iterada con un foreach loop. Este es el concepto de ejecución diferida, en otras palabras, que la ejecución ocurre tiempo después de que el query es construido.

Esto quiere decir que uno puede ejecutar una query declarada las veces que quiera. Esto es útil por ejemplo cuando se tiene información actualizándose y deseamos obtener la información más reciente.

La ejecución diferida permite que múltiples querys sean combinadas o que un query sea extendido, dándole más oportunidad de reusabilidad a los querys que declaramos.

3. Lazy loading vs Eager loading

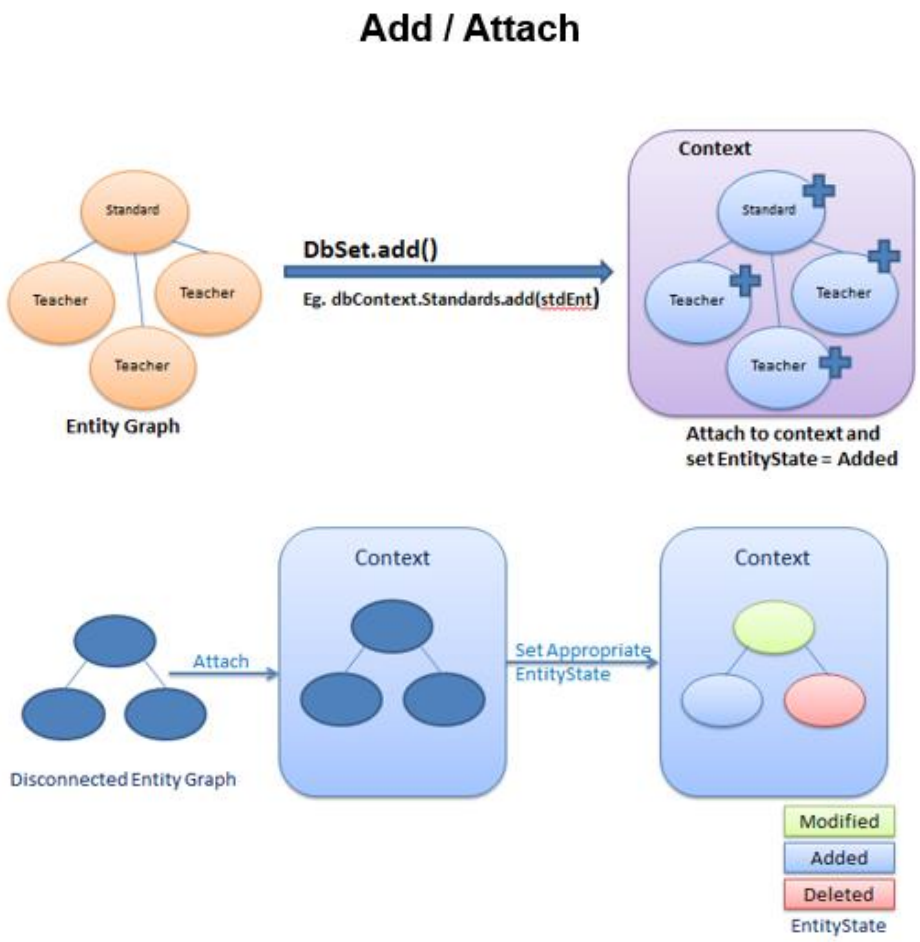


Lazy Loading es el proceso en el cual la información de una entidad o colección de entidades es cargada desde la base de datos la primera vez que una propiedad que hace referencia a esas entidades es accesada. Esto se hace para evitar traer toda la información de las entidades que tienen relación alguna con la entidad que estamos consultando. Este es el modo de trabajo que por defecto maneja EF.

Lazy loading es mucho más conveniente a la hora de desarrollar, pero por su naturaleza ejecuta una mayor cantidad de sentencias sobre base de datos por lo que su performance podría no ser la adecuada, todo está en relación con el escenario en donde deseemos aplicarlo. Por ejemplo, si tuviéramos los datos de un producto y necesitáramos la lista de materiales que conforman este producto, pero solo en algunas ocasiones, podríamos aplicar lazy loading, consiguiendo la información del producto, y si es necesario, la de sus materiales.

Eager Loading en cambio, es el proceso en el cual un query para un determinado tipo de entidad también carga las entidades relacionadas como parte del query. Se puede hacer uso de Eager Loading por medio de la palabra reservada **Include** la cual nos puede permitir incluso cargar múltiples niveles de nuestro modelo de entidades. Por ejemplo, si tuviéramos que conseguir la información de un pedido con sus detalles incluyendo nombres de productos sería recomendable usar Include para traer toda la información de todas las entidades que contienen la información que se necesita en un solo viaje a base de datos.

4. Add / Attach



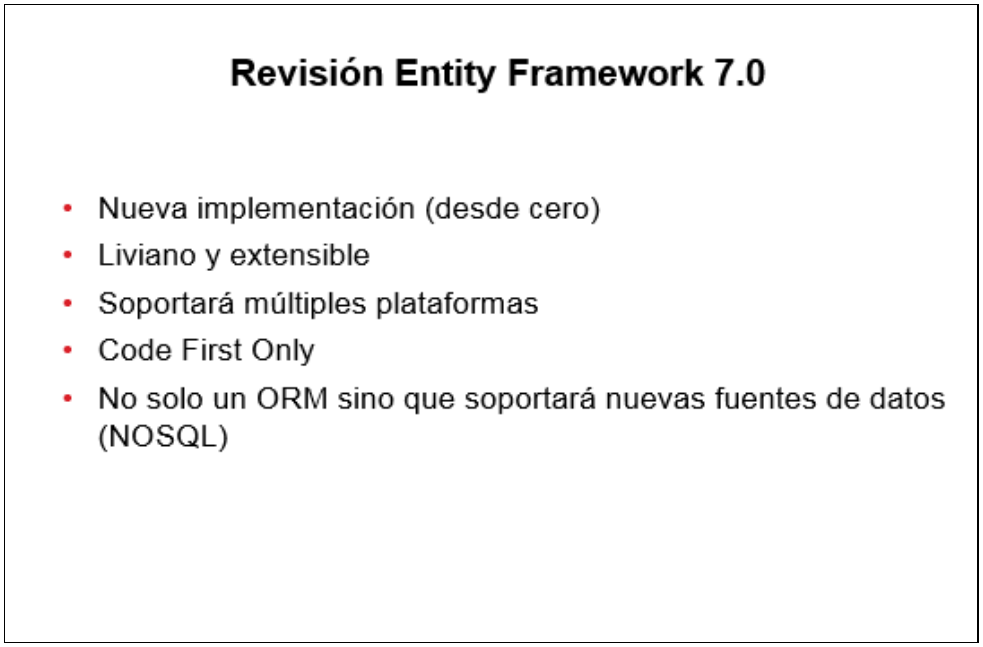
Add:

Para inserción de datos, EF maneja el método Add, con el cual podemos indicar que un objeto nuevo que ha sido creado va a ser marcado con un EntitySate de Added y será enviado a base de datos como si se tratara de un nuevo registro. Cuando el método SaveChanges es llamado, EF tomará los datos del objeto y tratará de realizar una operación de Insert.

Attach:

Por otro lado, Attach es usado para entidades que ya existen en la base de datos. El EntityState en lugar de ser seteado a Added es seteado a Unchanged, lo que significa que no ha sido cambiado desde que se le adjuntó al contexto. Se asume que estos objetos ya existen en la base de datos, cuando se llama a SaveChanges se actualiza la fila respectiva basado en su PK.

5. Revisión de Entity Framework 7



EF7 es la nueva versión de EF (que, al momento de elaborar este documento, esta como preview, sin fecha oficial de lanzamiento), que será más liviano y extensible. El equipo de desarrollo de Microsoft se ha encargado de implementar EF desde cero para la versión 7 con el propósito de brindar una versión base mucho más liviana y modular. Esto obviamente indica cambios fuertes, puede considerarse que EF hasta la versión 6 podrá usarse de manera segura ya que poseerá actualizaciones incrementales, pero con EF7 estaremos entrando a usar una nueva tecnología.

EF7 soportará múltiples plataformas, es decir, dejará de ser usado solo en Windows y podrá ser usado en el Windows Store, para Windows Phone y otras plataformas como Mono en Mac y Linux.

EF7 será Code First Only, esto quiere decir que no tendrá un visualizador de archivos .edmx los cuales tampoco se generarán. Podrá partirse de una base de datos, pero el resultado será un conjunto de clases y ya no el modelo visual edmx.

Con EF7, este producto no solo será un ORM, sino que ganará acceso a nuevas fuentes de datos como son las bases de datos NOSQL (es probable que la versión 7 de EF no tenga estas características y que sean incluidas en una siguiente versión).