Entity Framework Core Database First

Ingo Köster

Diplom Informatiker (FH)

Entity Framework Core installieren

- Zuerst wird der oder die gewünschten Datenbankanbieter per NuGet in der passenden Version installiert
- Hier wird nur der SQL Server verwendet
- Über „Extras“ -> „NuGet-Paket-Manager“ -> „Paket-Manager-Konsole“
  - `Microsoft.EntityFrameworkCore.SqlServer`
- Zudem werden die Entity Framework-Tools in der gleichen Version eingesetzt
  - `Microsoft.EntityFrameworkCore.Tools`

Datenbankanbieter (Auswahl)

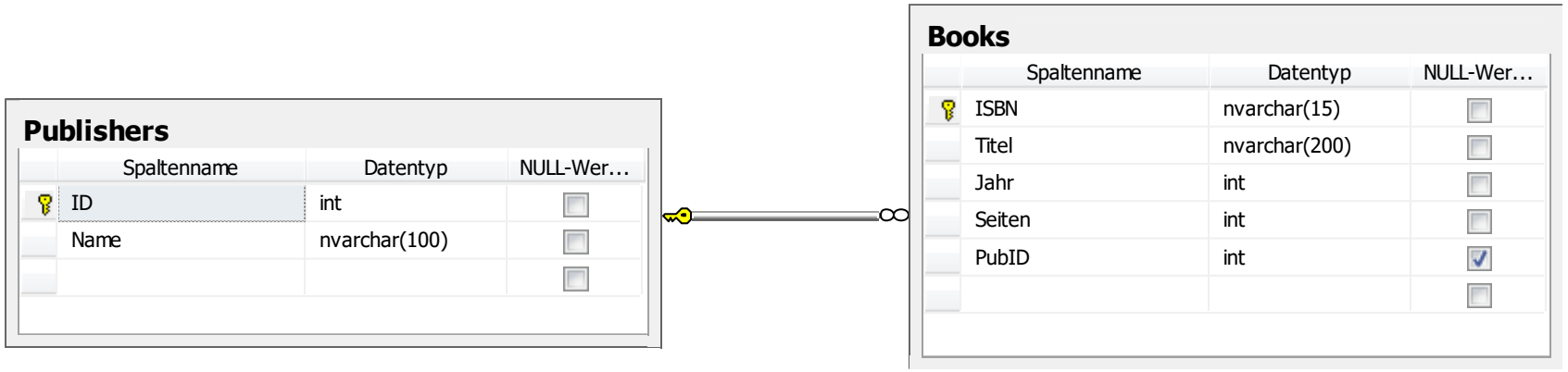
Datenbank	NuGet-Paket
SQL Server 2008 oder höher	`Microsoft.EntityFrameworkCore.SqlServer`
SQLite 3.7 oder höher	`Microsoft.EntityFrameworkCore.Sqlite`
EF Core-In-Memory-Datenbank	`Microsoft.EntityFrameworkCore.InMemory`
PostgreSQL	`Npgsql.EntityFrameworkCore.PostgreSQL`
MySQL, MariaDB	`Pomelo.EntityFrameworkCore.MySql`
MySQL	`MySql.Data.EntityFrameworkCore`
Oracle 9.2.0.4 oder höher	`Devart.Data.Oracle.EFCore`
PostgreSQL 8.0 oder höher	`Devart.Data.PostgreSql.EFCore`
Microsoft Access-Dateien	`EntityFrameworkCore.Jet`

Beispiel Datenbank für die Folien

› Bücher und Verlage müssen abgebildet werden

› Jedes Buch ist einem Verlag zugeordnet oder hat keinen Verlag

› Jedem Verlag können beliebig viele Bücher zugeordnet werden (auch keines)



Erzeugen des Modells aus der Datenbank

› Basierend auf der vorhandenen Datenbank sollen die Modell-Klassen erstellt werden

› Über die Paket-Manager-Konsole

```
Scaffold-DbContext
  "Server=.\SQLEXPRESS;Database=NetDB;Trusted_Connection=True;"
  Microsoft.EntityFrameworkCore.SqlServer
```

› Dieser Prozess erstellt die Entitätsklassen und auch eine Kontext-Klasse basierend auf dem Schema der Datenbank

Generierte Model-Klassen

```
public partial class Books
{
    public string Isbn { get; set; }
    public string Titel { get; set; }
    public int Seiten { get; set; }
    public int Jahr { get; set; }
    public int? PubId { get; set; }

    public virtual Publishers Pub { get; set; }
}
```

```
public partial class Publishers
{
    public Publishers()
    {
        Books = new HashSet<Books>();
    }

    public int PublisherId { get; set; }
    public string Name { get; set; }

    public virtual ICollection<Books> Books { get; set; }
}
```

Generierte Kontext Klasse

```
public partial class BooksContext : DbContext
{
    public BooksContext()
    {
    }

    public BooksContext(DbContextOptions<BooksContext> options)
        : base(options)
    {
    }

    public virtual DbSet<Books> Books { get; set; }
    public virtual DbSet<Publishers> Publishers { get; set; }

    protected override void OnConfiguring(DbContextOptionsBuilder optionsBuilder)...

    protected override void OnModelCreating(ModelBuilder modelBuilder)...
}
```

Daten abfragen

```
using (NetDBContext ctx = new NetDBContext()) {
  var allBooks = from book in ctx.Books
                 select book;

  allBooks.ToList().ForEach(book =>
  {
    Console.WriteLine("{0}, {1})", book.Title, book.Isbn);
  });
}
```

Optionen für `Scaffold-DbContext`

Parameter	Bedeutung
`-OutputDir`	Ausgabeverzeichnis festlegen (z.B. `-OutputDir Models`)
`-Force`	Zum Überschreiben von bereits generiertem Code
`-Context`	Name der Kontext Klasse
`-Tables`	Angabe der Tabelle (z.B. `-Tables Blog, Post, "Order Details"`)

Übersicht der Optionen
https://docs.microsoft.com/en-us/ef/core/cli/powershell#scaffold-dbcontext