Creating a Data Model and Database with EF Core 2

Julie Lerman
MOST TRUSTED AUTHORITY ON ENTITY FRAMEWORK
@julielerman thedatafarm.com

Module Overview

Creating a .NET Framework solution with business classes

Adding EF Core 2 and a data model

Using EF Core Migrations to create a database or database scripts

Recreating the model in a cross-platform ASP.NET Core 2 solution

Adding a many-to-many and a one-to-one relationship to the model

Setting up the Solution

Installing .NET Standard

microsoft.com/net/download

Adding EF Core with the NuGet Package Manager

Creating the Data Model With EF Core

Specifying the Data Provider and Connection String

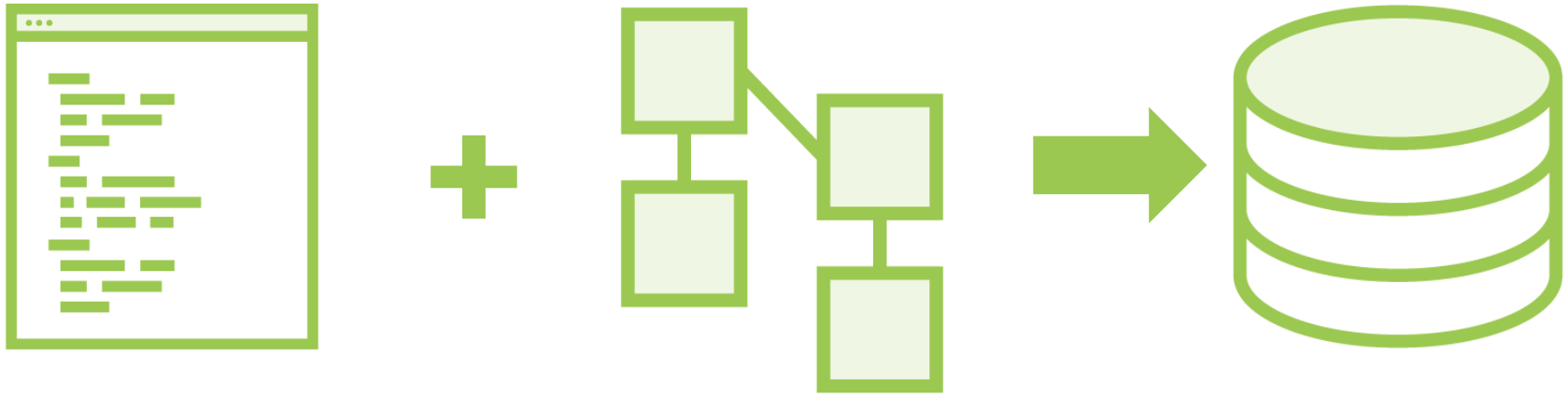
No More Database Magic

You must specify
data provider & connection string

Understanding EF Core Migrations

EF Core Basic Migrations Workflow

Mapping Your Data Model to the Database



Packages Required by Migrations in VS2017

PowerShell commands → **Microsoft.EntityFrameworkCore.Tools***

Migrations engine → **Microsoft.EntityFrameworkCore.Design**

Requires an executable project
For PMC, this must also be the startup project

*In the future, Tools will be installed with .NET Core

Adding Your First Migration

Inspecting Your First Migration

Using Migrations to Script or Directly Create the Database

Migrations Recommendation

Development database

update-database

Production database

script-migration

What If Database Does Not Exist?

update-database

API's internal code will create the database

script-migration

You must create the database before running the script

Recreating the Model in .NET Core

Adding Many-to-many and One-to-one Relationships

Many-to-many Relationship

Relational Database: Join Table

Samurais Join Table Battles

EF Core: Join Entity

Samurai Join Entity Battles

You can configure mappings that don't follow EF Core's conventions

Required vs. Optional Relationships

Samurai : SecretIdentity

(Default) Required parent

A child cannot be orphaned

Optional parent

You'll need to configure this mapping explicitly

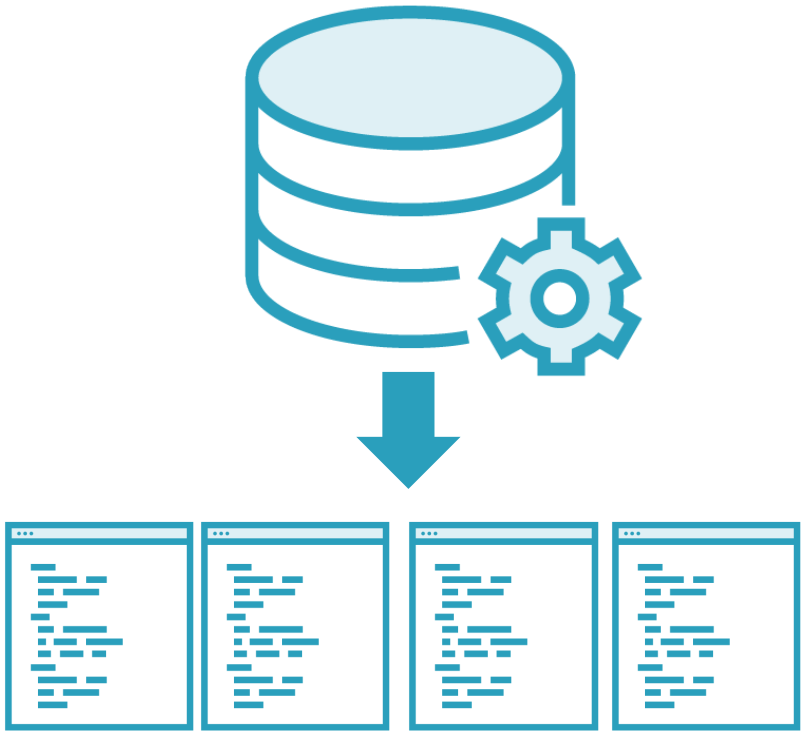
(Default) Optional child

EF Core (& database) will always allow a null child

Required child

You'll need to handle the requirement in your business logic

Reverse Engineering an Existing Database



Create DbContext & classes from database

Updating model is not currently supported

Transition to migrations is not pretty ... look for helpful link in resources

PowerShell command: scaffold-dbcontext

Review

EF Core 2 in full .NET is no different than in .NET Core

Define and update the database schema based on the model using migrations

Scaffold a head start on your classes and DbContext from an existing DB

You can configure mappings that don't follow EF Core's conventions

Various options to specify provider and connection string in your app

Resources

Entity Framework Core on GitHub github.com/aspnet/entityframework
EF Core Roadmap bit.ly/efcoreroadmap
EF Core Documentation docs.efproject.net

Announcing Entity Framework Core 2.0 bit.ly/EFCore2RTM

.NET Core Installation page www.microsoft.com/net/download/core

Entity Framework Core: Getting Started bit.ly/PS_EFCoreStart

Create Database Diagrams in SQL Server Management Studio bit.ly/2BIAahm

EF Core Power Tools Extension (model visualizer and more):
github.com/ErikEJ/SqlCeToolbox/wiki/EF-Core-Power-Tools

EF Core migrations with existing database schema and data
cmatskas.com/ef-core-migrations-with-existing-database-schema-and-data

Creating a Data Model and Database with EF Core 2

Julie Lerman
MOST TRUSTED AUTHORITY ON ENTITY FRAMEWORK
@julielerman thedatafarm.com