Entity Framework: Code First

greenwich.edu.vn

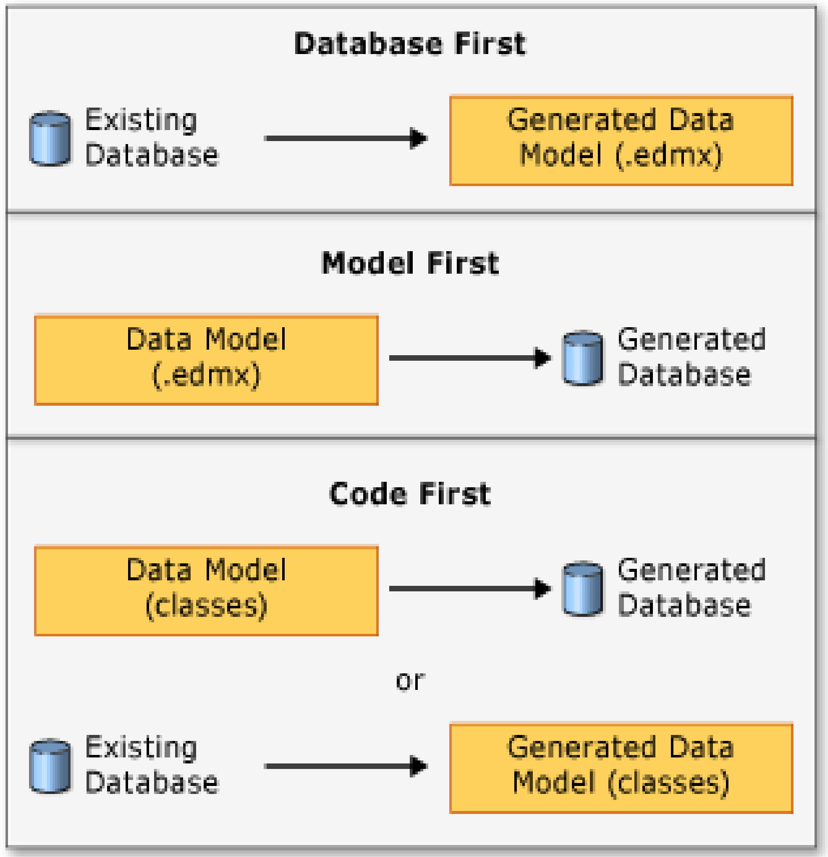
UNIVERSITY of GREENWICH

Alliance with FPT Education

Table of Contents

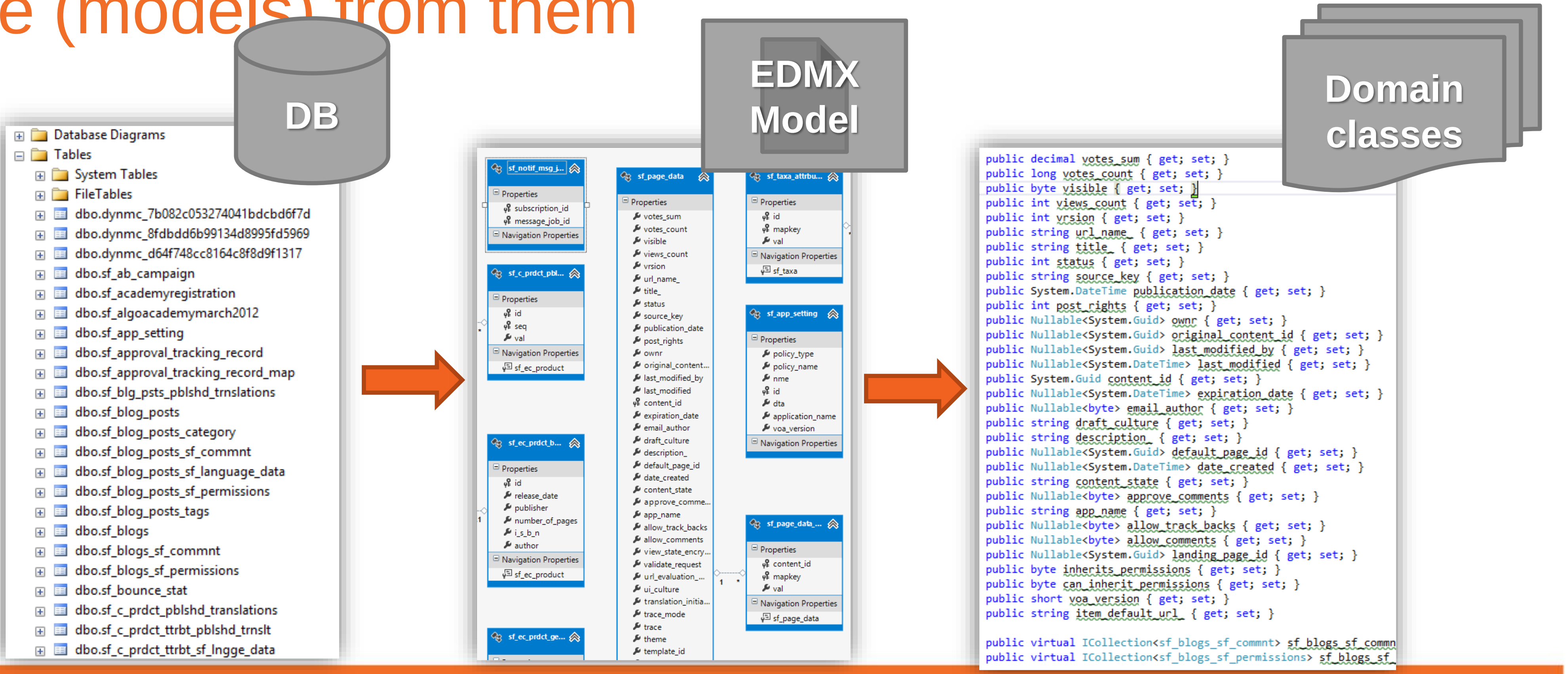
EF Modeling Workflows

- EF supports three modeling workflows:
  - Database First
    - Create models from existing DB schema (DB → C# classes)
  - Model First
    - Create models in the EF designer in VS
  - Code First
    - Write models and combine them in **`DbContex`** (C# classes → DB)



Database First Modeling Workflow

- Create models as database tables and then generate code (models) from them

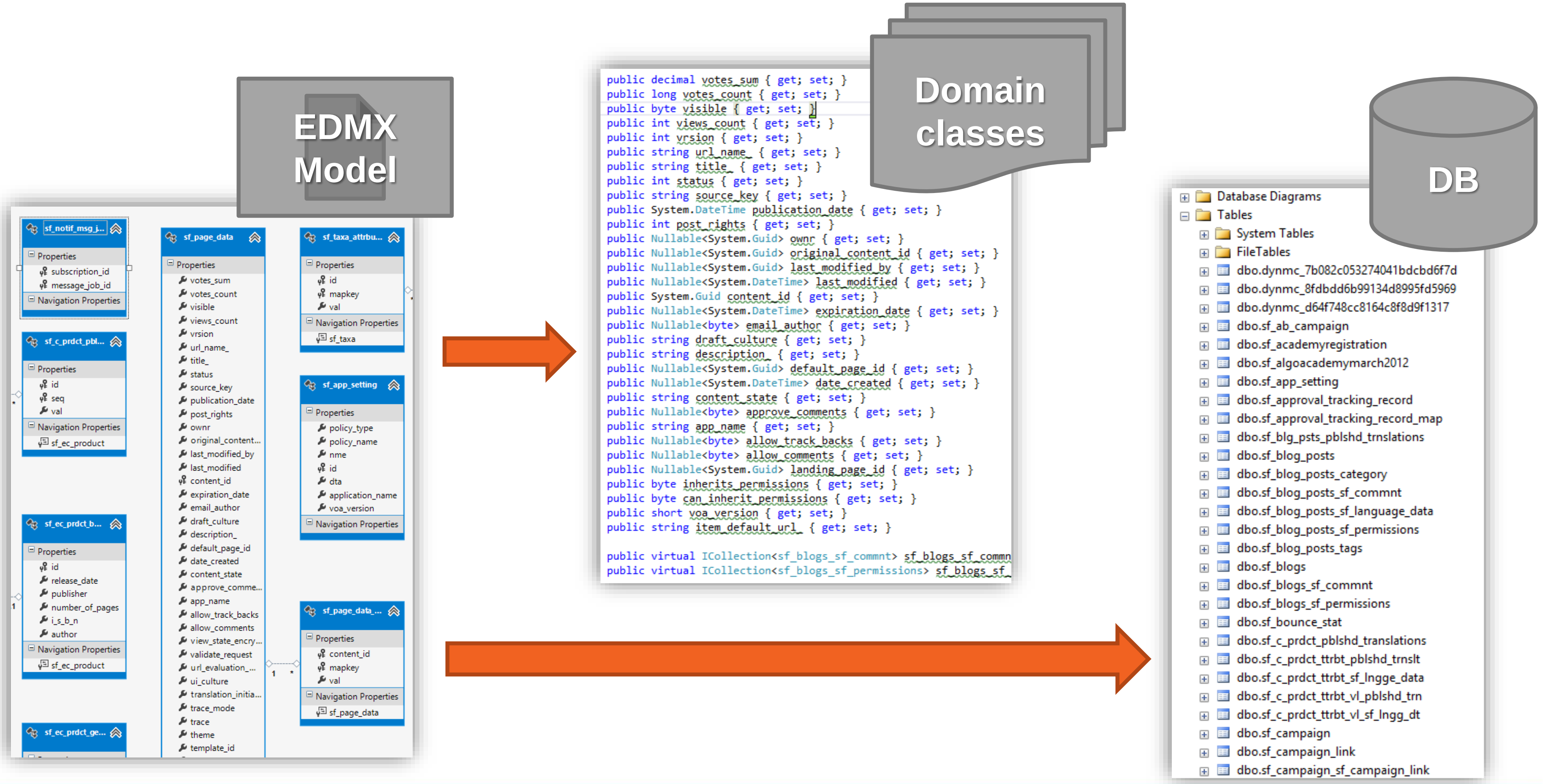


```
Database Diagrams
Tables
  System Tables
  FileTables
  dbo.dynmc_7b082c053274041bdcbd6f7d
  dbo.dynmc_8fdbdd6b99134d8995fd5969
  dbo.dynmc_d64f748cc8164c8f8d9f1317
  dbo.sf_ab_campaign
  dbo.sf_academyregistration
  dbo.sf_algoacademymarch2012
  dbo.sf_app_setting
  dbo.sf_approval_tracking_record
  dbo.sf_approval_tracking_record_map
  dbo.sf_blg_psts_pblshd_trnslations
  dbo.sf_blog_posts
  dbo.sf_blog_posts_category
  dbo.sf_blog_posts_sf_commnt
  dbo.sf_blog_posts_sf_language_data
  dbo.sf_blog_posts_sf_permissions
  dbo.sf_blog_posts_tags
  dbo.sf_blogs
  dbo.sf_blogs_sf_commnt
  dbo.sf_blogs_sf_permissions
  dbo.sf_bounce_stat
  dbo.sf_c_prdct_pblshd_translations
  dbo.sf_c_prdct_ttrbt_pblshd_trnslt
  dbo.sf_c_prdct_ttrbt_sf_lngge_data
```

```csharp
public decimal votes_sum { get; set; }
public long votes_count { get; set; }
public byte visible { get; set; }
public int views_count { get; set; }
public int vrsion { get; set; }
public string url_name_ { get; set; }
public string title_ { get; set; }
public int status { get; set; }
public string source_key { get; set; }
public System.DateTime publication_date { get; set; }
public int post_rights { get; set; }
public Nullable<System.Guid> ownr { get; set; }
public Nullable<System.Guid> original_content_id { get; set; }
public Nullable<System.Guid> last_modified_by { get; set; }
public Nullable<System.DateTime> last_modified { get; set; }
public System.Guid content_id { get; set; }
public Nullable<System.DateTime> expiration_date { get; set; }
public Nullable<byte> email_author { get; set; }
public string draft_culture { get; set; }
public string description_ { get; set; }
public Nullable<System.Guid> default_page_id { get; set; }
public Nullable<System.DateTime> date_created { get; set; }
public string content_state { get; set; }
public Nullable<byte> approve_comments { get; set; }
public string app_name { get; set; }
public Nullable<byte> allow_track_backs { get; set; }
public Nullable<byte> allow_comments { get; set; }
public Nullable<System.Guid> landing_page_id { get; set; }
public byte inherits_permissions { get; set; }
public byte can_inherit_permissions { get; set; }
public short voa_version { get; set; }
public string item_default_url_ { get; set; }

public virtual ICollection<sf_blogs_sf_commnt> sf_blogs_sf_commn
public virtual ICollection<sf_blogs_sf_permissions> sf_blogs_sf_
```

Model First Modeling Workflow

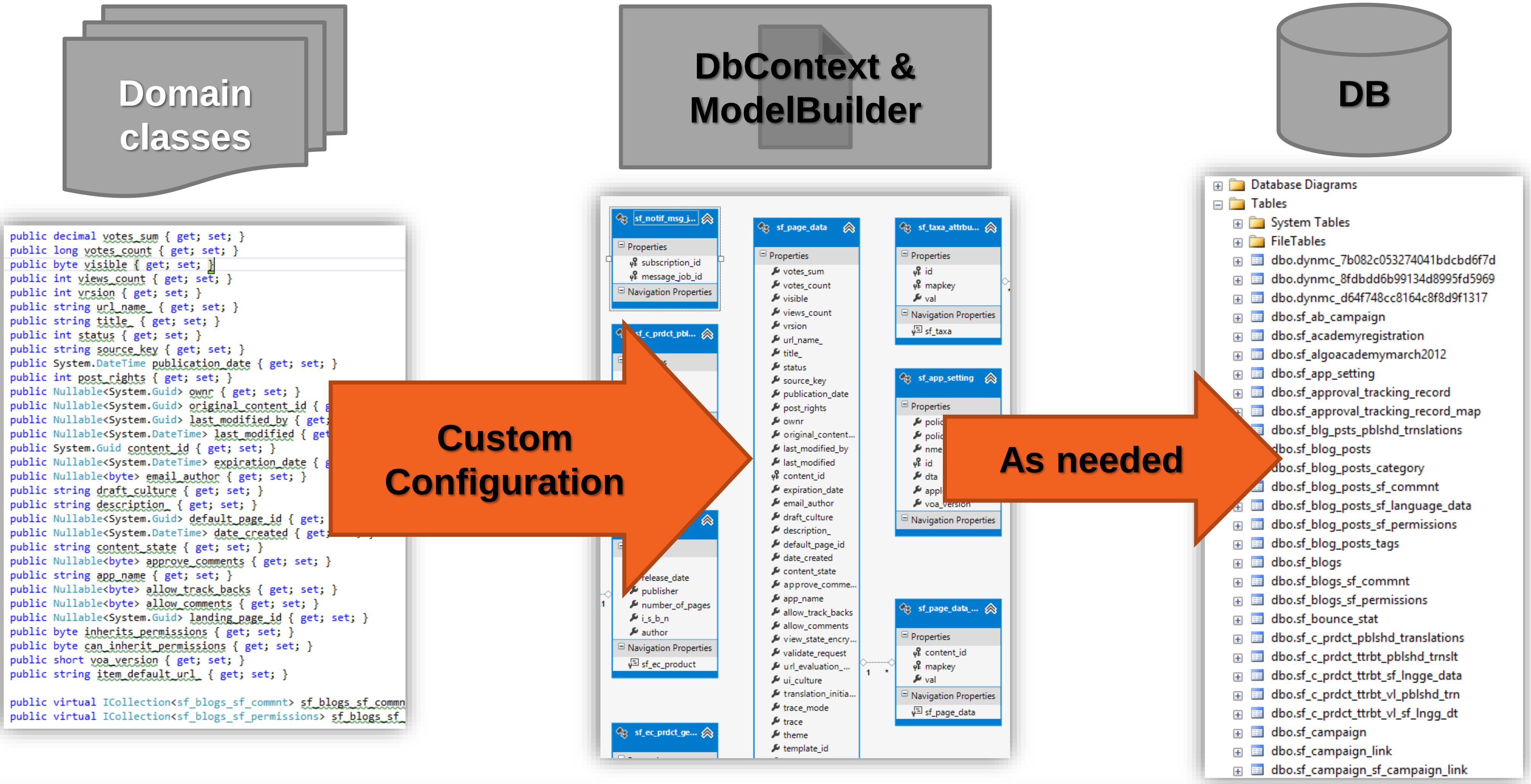


```
public decimal votes_sum { get; set; }
public long votes_count { get; set; }
public byte visible { get; set; }
public int views_count { get; set; }
public int vrsion { get; set; }
public string url_name_ { get; set; }
public string title_ { get; set; }
public int status { get; set; }
public string source_key { get; set; }
public System.DateTime publication_date { get; set; }
public int post_rights { get; set; }
public Nullable<System.Guid> ownr { get; set; }
public Nullable<System.Guid> original_content_id { get; set; }
public Nullable<System.Guid> last_modified_by { get; set; }
public Nullable<System.DateTime> last_modified { get; set; }
public System.Guid content_id { get; set; }
public Nullable<System.DateTime> expiration_date { get; set; }
public Nullable<byte> email_author { get; set; }
public string draft_culture { get; set; }
public string description_ { get; set; }
public Nullable<System.Guid> default_page_id { get; set; }
public Nullable<System.DateTime> date_created { get; set; }
public string content_state { get; set; }
public Nullable<byte> approve_comments { get; set; }
public string app_name { get; set; }
public Nullable<byte> allow_track_backs { get; set; }
public Nullable<byte> allow_comments { get; set; }
public Nullable<System.Guid> landing_page_id { get; set; }
public byte inherits_permissions { get; set; }
public byte can_inherit_permissions { get; set; }
public short voa_version { get; set; }
public string item_default_url_ { get; set; }

public virtual ICollection<sf_blogs_sf_commnt> sf_blogs_sf_commn
public virtual ICollection<sf_blogs_sf_permissions> sf_blogs_sf_
```

- Database Diagrams
- Tables
  - System Tables
  - FileTables
  - dbo.dynmc_7b082c053274041bdcbd6f7d
  - dbo.dynmc_8fdbdd6b99134d8995fd5969
  - dbo.dynmc_d64f748cc8164c8f8d9f1317
  - dbo.sf_ab_campaign
  - dbo.sf_academyregistration
  - dbo.sf_algoacademymarch2012
  - dbo.sf_app_setting
  - dbo.sf_approval_tracking_record
  - dbo.sf_approval_tracking_record_map
  - dbo.sf_blg_psts_pblshd_trnslations
  - dbo.sf_blog_posts
  - dbo.sf_blog_posts_category
  - dbo.sf_blog_posts_sf_commnt
  - dbo.sf_blog_posts_sf_language_data
  - dbo.sf_blog_posts_sf_permissions
  - dbo.sf_blog_posts_tags
  - dbo.sf_blogs
  - dbo.sf_blogs_sf_commnt
  - dbo.sf_blogs_sf_permissions
  - dbo.sf_bounce_stat
  - dbo.sf_c_prdct_pblshd_translations
  - dbo.sf_c_prdct_ttrbt_pblshd_trnslt
  - dbo.sf_c_prdct_ttrbt_sf_lngge_data
  - dbo.sf_c_prdct_ttrbt_vl_pblshd_trn
  - dbo.sf_c_prdct_ttrbt_vl_sf_lngg_dt
  - dbo.sf_campaign
  - dbo.sf_campaign_link
  - dbo.sf_campaign_sf_campaign_link

Code First Modeling Workflow



```
public decimal votes_sum { get; set; }
public long votes_count { get; set; }
public byte visible { get; set; }
public int views_count { get; set; }
public int vrsion { get; set; }
public string url_name_ { get; set; }
public string title_ { get; set; }
public int status { get; set; }
public string source_key { get; set; }
public System.DateTime publication_date { get; set; }
public int post_rights { get; set; }
public Nullable<System.Guid> ownr { get; set; }
public Nullable<System.Guid> original_content_id { g
public Nullable<System.Guid> last_modified_by { get;
public Nullable<System.DateTime> last_modified { get
public System.Guid content_id { get; set; }
public Nullable<System.DateTime> expiration_date { g
public Nullable<byte> email_author { get; set; }
public string draft_culture { get; set; }
public string description_ { get; set; }
public Nullable<System.Guid> default_page_id { get;
public Nullable<System.DateTime> date_created { get;
public string content_state { get; set; }
public Nullable<byte> approve_comments { get; set; }
public string app_name { get; set; }
public Nullable<byte> allow_track_backs { get; set; }
public Nullable<byte> allow_comments { get; set; }
public Nullable<System.Guid> landing_page_id { get; set; }
public byte inherits_permissions { get; set; }
public byte can_inherit_permissions { get; set; }
public short voa_version { get; set; }
public string item_default_url_ { get; set; }

public virtual ICollection<sf_blogs_sf_commnt> sf_blogs_sf_commn
public virtual ICollection<sf_blogs_sf_permissions> sf_blogs_sf_
```

```
⊞ Database Diagrams
⊟ Tables
    ⊞ System Tables
    ⊞ FileTables
    ⊞ dbo.dynmc_7b082c053274041bdcbd6f7d
    ⊞ dbo.dynmc_8fdbdd6b99134d8995fd5969
    ⊞ dbo.dynmc_d64f748cc8164c8f8d9f1317
    ⊞ dbo.sf_ab_campaign
    ⊞ dbo.sf_academyregistration
    ⊞ dbo.sf_algoacademymarch2012
    ⊞ dbo.sf_app_setting
    ⊞ dbo.sf_approval_tracking_record
    ⊞ dbo.sf_approval_tracking_record_map
    ⊞ dbo.sf_blg_psts_pblshd_trnslations
    ⊞ dbo.sf_blog_posts
    ⊞ dbo.sf_blog_posts_category
    ⊞ dbo.sf_blog_posts_sf_commnt
    ⊞ dbo.sf_blog_posts_sf_language_data
    ⊞ dbo.sf_blog_posts_sf_permissions
    ⊞ dbo.sf_blog_posts_tags
    ⊞ dbo.sf_blogs
    ⊞ dbo.sf_blogs_sf_commnt
    ⊞ dbo.sf_blogs_sf_permissions
    ⊞ dbo.sf_bounce_stat
    ⊞ dbo.sf_c_prdct_pblshd_translations
    ⊞ dbo.sf_c_prdct_ttrbt_pblshd_trnslt
    ⊞ dbo.sf_c_prdct_ttrbt_sf_lngge_data
    ⊞ dbo.sf_c_prdct_ttrbt_vl_pblshd_trn
    ⊞ dbo.sf_c_prdct_ttrbt_vl_sf_lngg_dt
    ⊞ dbo.sf_campaign
    ⊞ dbo.sf_campaign_link
    ⊞ dbo.sf_campaign_sf_campaign_link
```

Why Use Code First?

- Write code without having to define mappings in XML or create database models
- Define objects in POCO
  - Reuse these models and their attributes
- No base classes required
- Enables database persistence with no configuration
  - Can use automatic migrations
- Can use data annotations (`Key`, `Required`, etc.)

CODE FIRST MAIN PARTS

Domain Classes, DbContext and DbSet

Domain Classes (Models)

- Bunch of normal C# classes (POCO)
  - May contain navigation properties

```
public class PostAnswer
{
    public int PostAnswerId { get; set; }
    public string Content { get; set; }
    public int PostId { get; set; }
    public virtual Post Post { get; set; }
}
```

Primary key

Foreign key

Virtual for lazy loading

Navigation property

Domain Classes (Models) (2)

- Another example of domain class (model)

```
public class Post
{
    private ICollection<PostAnswer> answers;
    public Post()
    {
        this.answers = new HashSet<PostAnswer>();
    }

    public virtual ICollection<PostAnswer> Answers
    {
        get { return this.answers; }
        set { this.answers = value; }
    }
    public PostType Type { get; set; }
}
```

Prevents NullReferenceException

Navigation property

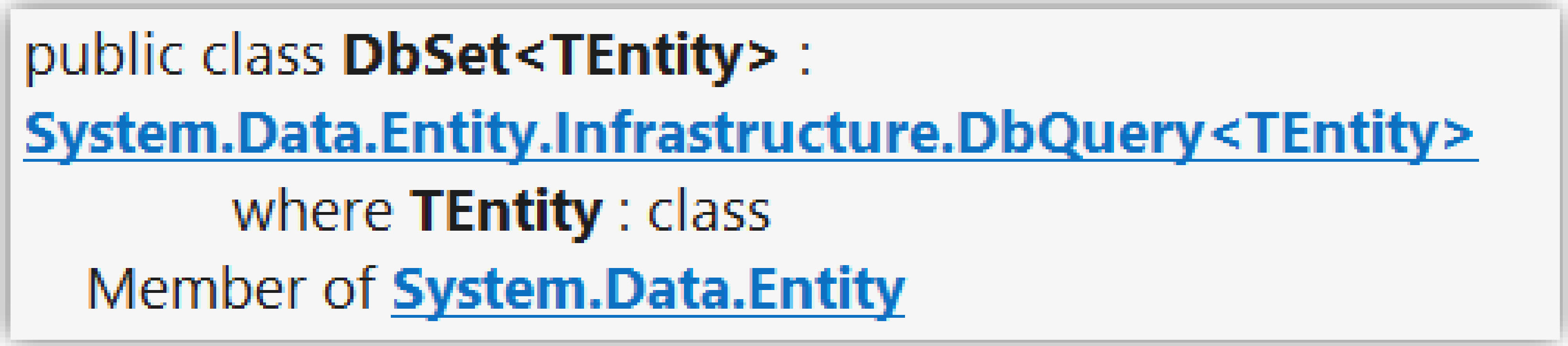
Enumeration

The DbContext Class

- A class that inherits from **`DbContext`**
  - – Manages model classes using **`DbSet<T>`** type
  - – Implements identity tracking, change tracking
  - – Provides API for CRUD operations and **LINQ-based** data access
- Recommended to be in a separate class library
  - – Don't forget to reference the Entity Framework library
    - Use the NuGet package manager
- Use several **`DbContext`** if you have too much models

DbSet Type

- Collection of single entity type
- Set operations: **`Add`**, **`Attach`**, **`Remove`**, **`Find`**
- Use with **`DbContext`** to query database



public class **DbSet<TEntity>** :
System.Data.Entity.Infrastructure.DbQuery<TEntity>
where **TEntity** : class
Member of System.Data.Entity

```
public DbSet<Post> Posts { get; set; }
```

Defining DbContext Class – Example

```
using System.Data.Entity;

using CodeFirst.Models;

public class ApplicationDbContext : DbContext
{
    public DbSet<Category> Categories { get; set; }
    public DbSet<Post> Posts { get; set; }
    public DbSet<PostAnswer> PostAnswers { get; set; }
    public DbSet<Tag> Tags { get; set; }
}
```

CRUD Operations with EF Code First

```
var db = new ForumContext();

var category = new Category { Name = "Database course" };
db.Categories.Add(category);

var post = new Post();
post.Title = "Homework Deadline";
post.Content = "Please extend the homework deadline";
post.Type = PostType.Normal;
post.Category = category;
post.Tags.Add(new Tag { Text = "homework" });
post.Tags.Add(new Tag { Text = "deadline" });

db.Posts.Add(post);

db.SaveChanges();
```

Where is My Data?

- Default **App.config** file contains link to default connection factory

```
<entityFramework>
  <defaultConnectionFactory type="System.Data.Entity.
Infrastructure.LocalDbConnectionFactory, EntityFramework">
    <parameters>
      <parameter value="v11.0" />
    </parameters>
  </defaultConnectionFactory>
</entityFramework>
```

- Server name by default:
  - **(localdb)\v11.0** or **(localdb)\MSSQLLocalDB**
- We can use VS server explorer to view database

How to Connect to SQL Server?

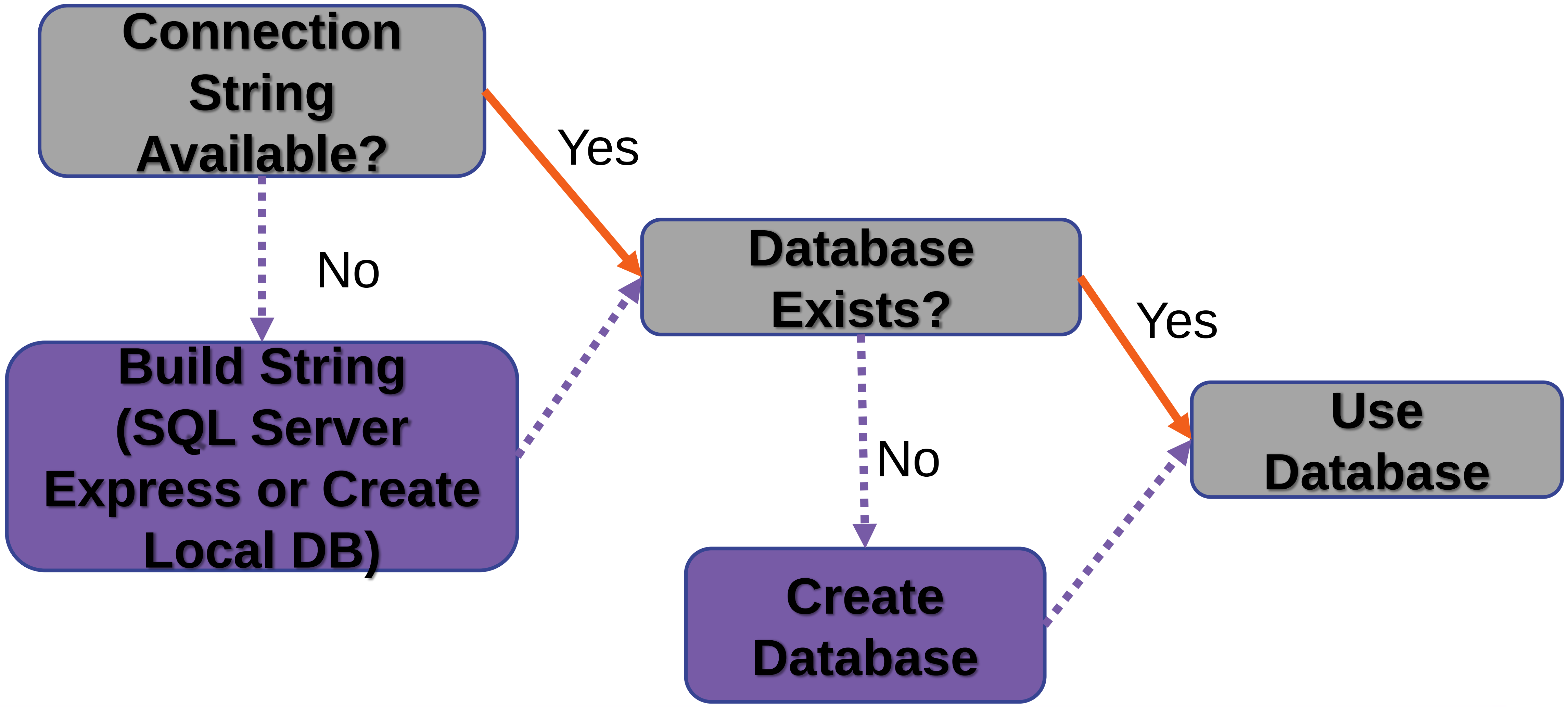
- First, create a context constructor that calls the base constructor with connection name

```
public class ApplicationDbContext : DbContext
{
    public ApplicationDbContext() : base("ForumDb") { … }
    …
}
```

- Then add the connection string in `app.config`

```
<connectionStrings>
  <add name="ForumDb" connectionString="Data Source=.;
    initial catalog=ForumDb;Integrated Security=True"
    providerName="System.Data.SqlClient" />
</connectionStrings>
```

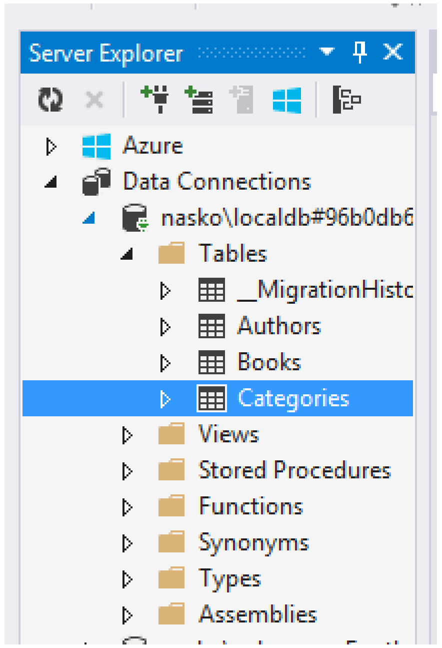
Database Connection Workflow

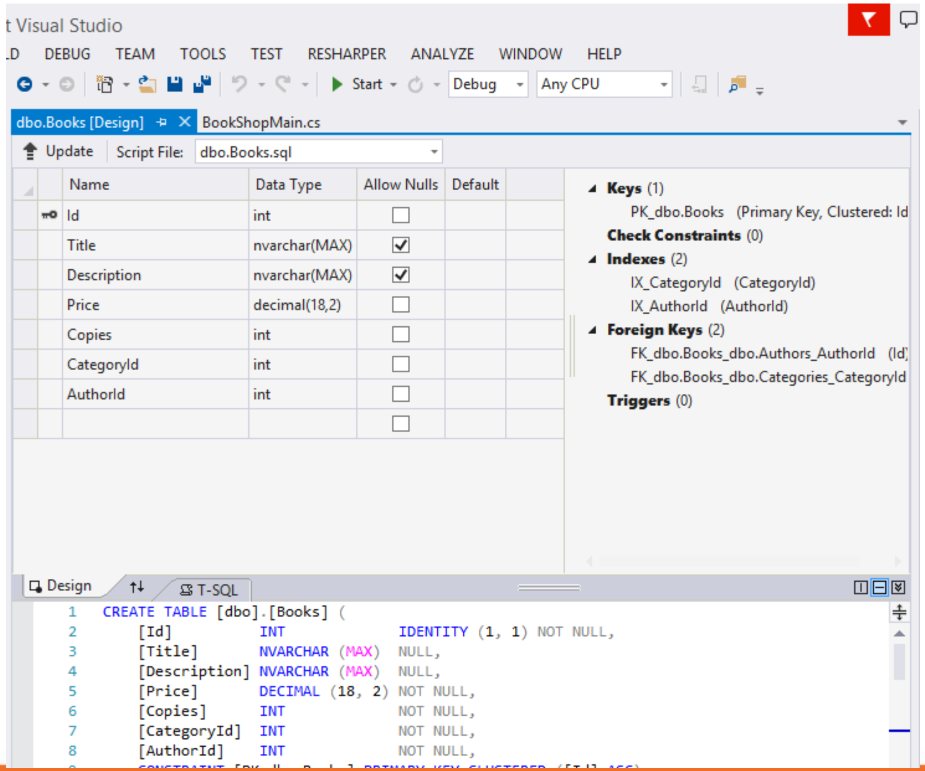


Connection String Available?
Yes
No
Build String (SQL Server Express or Create Local DB)
Database Exists?
Yes
No
Create Database
Use Database

Connecting to LocalDB in Visual Studio

- Connecting to LocalDB:
  - http://stackoverflow.com/a/21565688



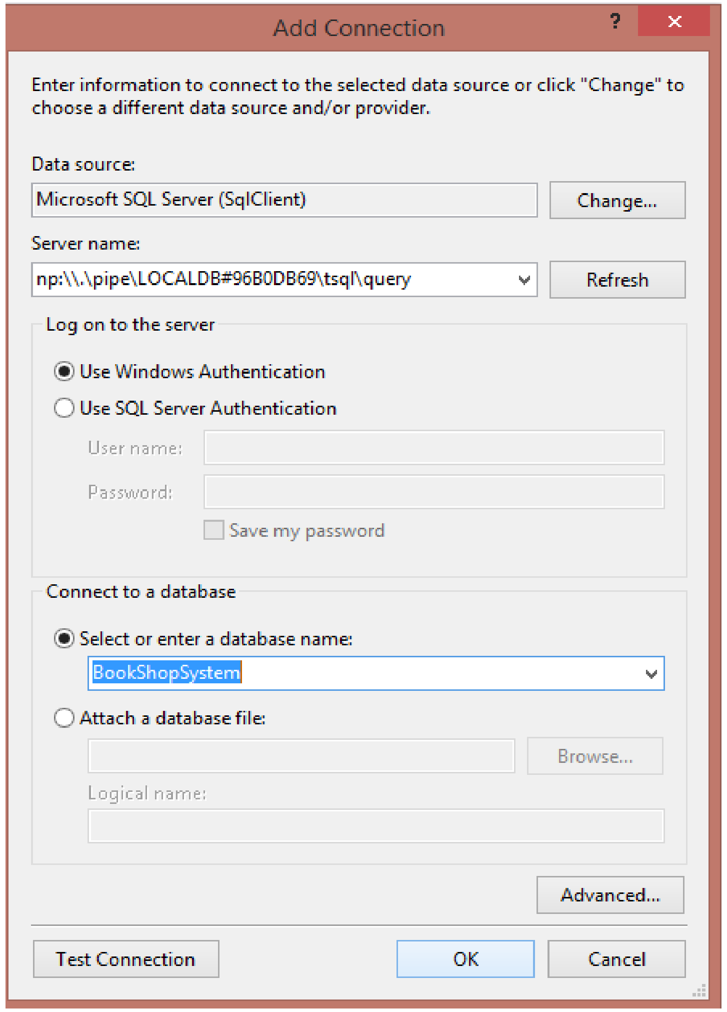


t Visual Studio
.D DEBUG TEAM TOOLS TEST RESHARPER ANALYZE WINDOW HELP
Start Debug Any CPU
dbo.Books [Design] BookShopMain.cs
Update Script File: dbo.Books.sql
Design T-SQL

Name	Data Type	Allow Nulls	Default
Id	int	☐	
Title	nvarchar(MAX)	✓	
Description	nvarchar(MAX)	✓	
Price	decimal(18,2)	☐	
Copies	int	☐	
CategoryId	int	☐	
AuthorId	int	☐	
		☐	

Keys (1)
PK_dbo.Books (Primary Key, Clustered: Id
Check Constraints (0)
Indexes (2)
IX_CategoryId (CategoryId)
IX_AuthorId (AuthorId)
Foreign Keys (2)
FK_dbo.Books_dbo.Authors_AuthorId (Id)
FK_dbo.Books_dbo.Categories_CategoryId
Triggers (0)

```
CREATE TABLE [dbo].[Books] (
    [Id]          INT             IDENTITY (1, 1) NOT NULL,
    [Title]       NVARCHAR (MAX)  NULL,
    [Description] NVARCHAR (MAX)  NULL,
    [Price]       DECIMAL (18, 2) NOT NULL,
    [Copies]      INT             NOT NULL,
    [CategoryId]  INT             NOT NULL,
    [AuthorId]    INT             NOT NULL,
```



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: np:\\.\pipe\LOCALDB#96B0DB69\tsql\query [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:] BookShopSystem
- Attach a database file: [Browse...]

Logical name:

[Advanced...]

[Test Connection] [OK] [Cancel]

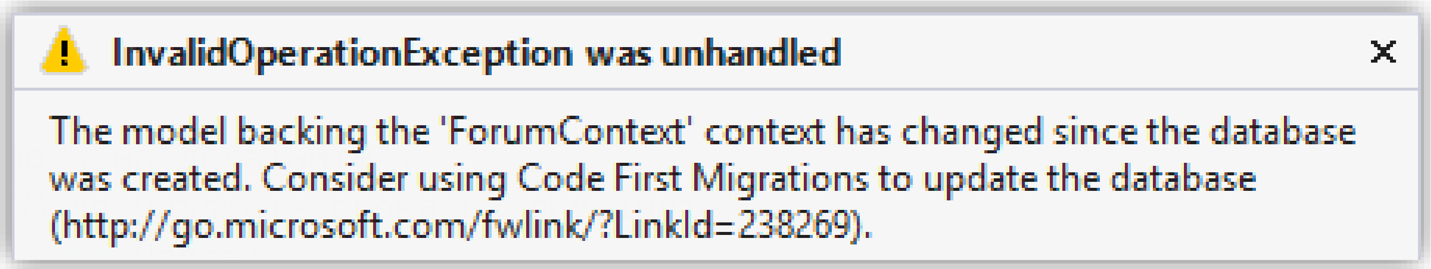
UNIVERSITY of GREENWICH

Alliance with FPT Education

USING CODE EF FIRST MIGRATIONS

Changes in Domain Classes

- What happens when we change our models?
  - Entity Framework compares our model with the model from the `__MigrationHistory` table in the DB



InvalidOperationException was unhandled

The model backing the 'ForumContext' context has changed since the database was created. Consider using Code First Migrations to update the database (http://go.microsoft.com/fwlink/?LinkId=238269).

- By default Entity Framework only creates the database
  - EF doesn't do any schema changes after that
- Using **Code First Migrations** we can handle differences between models and database

Code First Migrations

- Enable Code First Migrations
  - Open Package Manager Console
  - Run **`Enable-Migrations`** command
    - This will create some initial jumpstart code
    - **`-EnableAutomaticMigrations`** for auto migrations
- Two types of migrations
  - Automatic migrations
    - Set **`AutomaticMigrationsEnabled = true;`**
  - Code-based (providing full control)
    - Separate C# code file for every migration

Database Migration Strategies

- **`CreateDatabaseIfNotExists`**
  - Default migration
- **`DropCreateDatabaseIfModelChanges`**
  - We lose all the data when the model changes
- **`DropCreateDatabaseAlways`**
  - Great for automated integration testing
- **`MigrateDatabaseToLatestVersion`**
  - This option uses our migrations
- Custom migrations
  - Implement **`IDatabaseInitializer`** for custom migration strategy

Use Code First Migrations

- First, enable code first migrations
- Second, we need to tell to Entity Framework to use our migrations with code (or in **`App.config`**)

```
Database.SetInitializer(
  new MigrateDatabaseToLatestVersion<ForumContext, Configuration>());
```

- We can configure automatic migration

```
public Configuration()
{
    this.AutomaticMigrationsEnabled = true;
    this.AutomaticMigrationDataLossAllowed = true;
}
```

This will allow us to delete or change properties

Seeding the Database

- During a migration we can seed the database with some data using the **`Seed()`** method

```
protected override void Seed(ForumContext context)
{
  /* This method will be called after migrating to the latest version.
     You can use the DbSet<T>.AddOrUpdate() helper extension method
     to avoid creating duplicate seed data. E.g. */

  context.Tags.AddOrUpdate(t => t.Text, new Tag { Text = "C#" });
  context.SaveChanges();
}
```

- This method will be run after each migration

UNIVERSITY of GREENWICH

Alliance with FPT Education

CONFIGURE MAPPINGS

Using Data Annotations and EF Fluent API

Configure Mappings

- Entity Framework respects mapping details from two sources:
  - **Data annotation attributes** in the models
    - E.g. `[Key]`, `[Required]`, `[MaxLength]`
    - Can be reused for validation purposes
  - **Fluent API** code mapping configuration
    - By overriding `OnModelCreating` method
    - By using custom configuration classes
- Use one approach or the other

Data Annotations

- There is a bunch of data annotation attributes
  - **System.ComponentModel.DataAnnotations**
  - Specify the primary key of the table: **[Key]**
  - For validation:
    - **[StringLength]**, **[MaxLength]**, **[MinLength]**, **[Required]**
  - Schema:
    - **[Column]**, **[Table]**, **[ComplexType]**, **[ConcurrencyCheck]**, **[Timestamp]**, **[InverseProperty]**, **[ForeignKey]**, **[DatabaseGenerated]**, **[NotMapped]**, **[Index]**
- EF 6 supports custom attributes by using custom conventions

ENTITY FRAMEWORK FLUENT API

Using the ModelBuilder

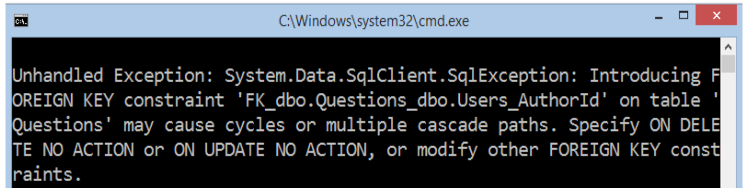
EF Fluent API for Mappings

- By overriding **DbContext.OnModelCreating** we can specify mapping configurations:

```
protected override void OnModelCreating(DbModelBuilder modelBuilder)
{
  modelBuilder.Entity<Tag>()
    .HasKey(x => x.TagId);
  modelBuilder.Entity<Tag>()
    .Property(x => Text).IsUnicode(true);
  modelBuilder.Entity<Tag>()
    .Property(x => x.Text).HasMaxLength(255);
  modelBuilder.Entity<Tag>()
    .Property(x => x.Text).IsFixedLength();
  base.OnModelCreating(modelBuilder);
}
```

Solving Cascade Delete Issue with Fluent API

- When there several cascade delete paths, EF throws an exception



```
C:\Windows\system32\cmd.exe

Unhandled Exception: System.Data.SqlClient.SqlException: Introducing F
OREIGN KEY constraint 'FK_dbo.Questions_dbo.Users_AuthorId' on table '
Questions' may cause cycles or multiple cascade paths. Specify ON DELE
TE NO ACTION or ON UPDATE NO ACTION, or modify other FOREIGN KEY const
raints.
```

- Two solutions with Fluent API:
  - Remove default cascade delete convention globally

```
modelBuilder.Conventions
        .Remove<OneToManyCascadeDeleteConvention>();
```

  - Manually configure the relation

```
modelBuilder.Entity<User>()
    .HasMany(u => u.Answers)
    .WithRequired(a => a.User)
    .WillCascadeOnDelete(false);
```

Manually Setting Many-to-Many Relation

- In some cases EF does not properly recognize the relation
  - E.g. we want a User to have many friends (other users)
  - We have to manually map the relation to a many-to-many table

```
modelBuilder.Entity<User>()
    .HasMany(u => u.Friends)
    .WithMany()
    .Map(m =>
    {
        m.MapLeftKey("UserId");
        m.MapRightKey("FriendId");
        m.ToTable("UserFriends");
    });
```

EF Fluent API Configurations

- `.Entity()`
  - – Map: Table Name, Schema
  - – Inheritance Hierarchies, Complex Types
  - – Entity -> Multiple Tables
  - – Table -> Multiple Entities
  - – Specify Key (including Composite Keys)
- `.Property()`
  - – Attributes (and Validation)
  - – Map: Column Name, Type, Order
  - – Relationships
  - – Concurrency