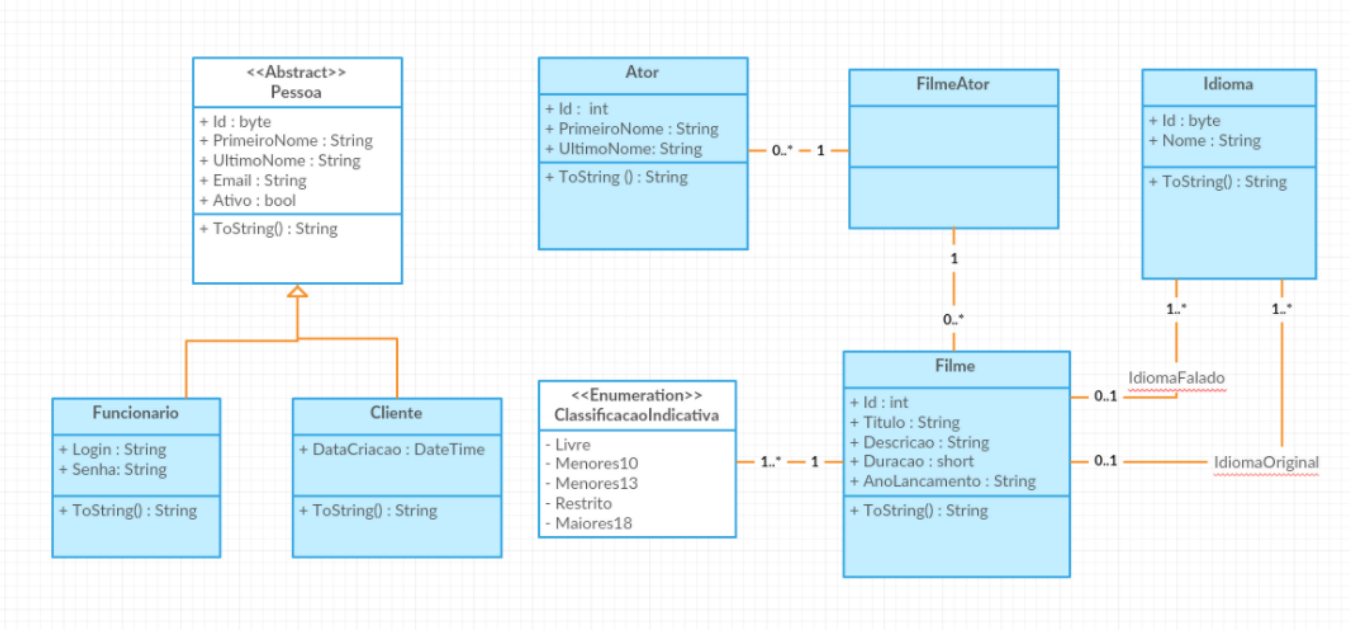
01

A entidade Filme

Transcrição

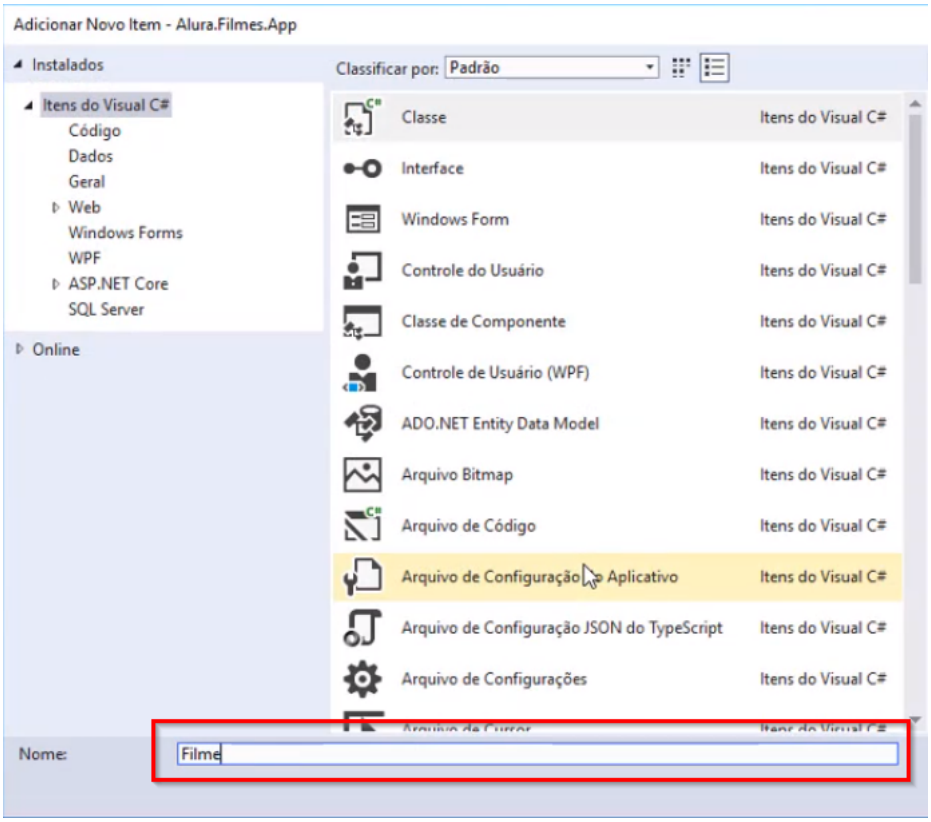
Conseguimos mapear a nossa tabela de atores. Continuaremos a aplicação mapeando a classe `Filme` na tabela `filme`. Aproveitaremos, também, para fazermos uma revisão do que já foi ensinado. Se você estiver seguro com o seu aprendizado, pular esta aula é uma opção. Você pode tentar fazer o procedimento de forma mais autônoma e depois assistir a aula para conferir se tudo está correto.



O objetivo é realizar uma listagem de filmes. Faremos um `select` na tabela `filme` e mostraremos os registros no console. Trabalharemos na classe `Program`. Criaremos um `foreach` e uma variável `filme` que irá representar o registro, e iremos percorrer uma propriedade denominada `Filmes` - que ainda não existe, pois não a criamos - .

```
namespace Alura.Filmes.App
{
    class Program
    {
        static void Main(string[] args)
        {
            using (var contexto = new AluraFilmesContexto())
            {
                contexto.LogSQLToConsole();

                forreach (var filme in contexto.Filmes)
                {
                    Console.WriteLine(filme);
                }
            }
        }
    }
}
```

Criaremos a classe `Filme`. Na área "Gerenciador de Soluções", localizada ao lado esquerdo da tela, clicaremos com o botão direito na pasta "Negocios", e selecionaremos a opção "Adicionar > Classe".



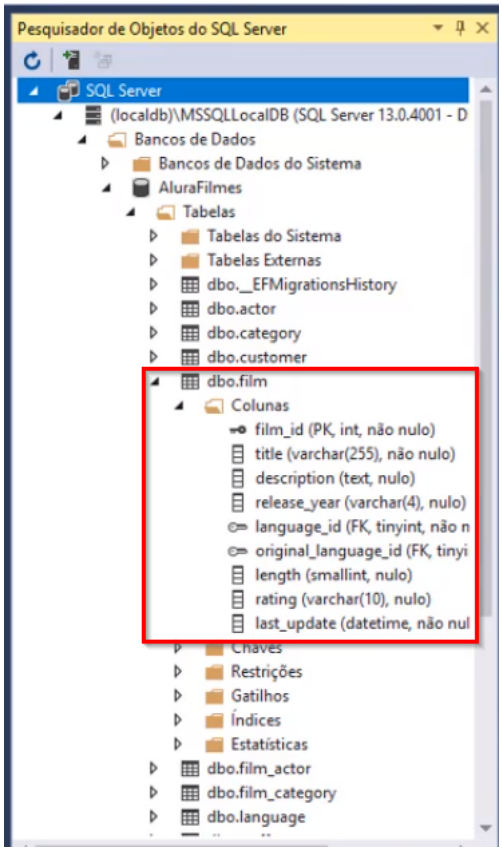
o código da nova classe `Filme` estará dessa forma:

```
using System;
using System.Collections.Genereic;
using System.Linq;
using System Text;
using System.Threading.Tasks;

namespace Alura.Filmes.App.Negocio
{
    class Filme
    {

    }
}
```

Primeiramente, removeremos as informações desnecessárias e aumentaremos a visibilidade da classe (adicionando `public`).

```
namespace Alura.Filmes.App.Negocio
{
    public class Filme
    {

    }
}
```

Analisaremos quais propriedades precisam ser criadas dentro da classe `Filme`. Para isso, iremos na área "Pesquisador de Objetos" localizada na parte esquerda da tela, e selecionaremos "SQL Server > (localdb) > Bancos de Dados do Sitema > AluraFilmes > Tabelas > dbo.film > Colunas "



Não usaremos todas as tabelas, pois algumas são de relacionamento e veremos como trabalhar com elas posteriormente. A primeira propriedade que iremos criar é a do tipo `Id`, depois criaremos outra que irá representar o título dos filmes (`Titulo`), uma do tipo `string` que representará a descrição do filme (`Descricao`), mais outra do tipo `string` que representará o ano de lançamento (`AnoLancamento`), e finalmente, uma que representará a duração do filme (`Duracao`).

```
namespace Alura.Filmes.App.Negocio
{
    public class Filme
    {
        public int Id { get; set; }
        public string Titulo { get; set; }
        public string Descricao { get; set; }
        public string AnoLancamento { get; set; }

    }
}
```

Devemos lembrar que a coluna `last_update` só existe no banco de dados pois trata-se de uma shadow property.

Trabalharemos na classe `AluraFilmesContexto`, e daremos a instrução para que o Entity gerencie a persistência de `Filme` através de uma propriedade `DbSet`. Tal propriedade será chamada de `Filmes`.

```
namespace Alura.Filmes.App.Dados
{
    public class AluraFilmesContexto : DbContext
```

```
        {
            public DbSet<Ator> Atores { get; set; }
            public DbSet<Filme> Filmes { get; set; }

            protected override void OnConfiguring(DbContextOptionsBuilder optionsBuilder)
            {
                optionsBuilder.UseSqlServer("Server=(localdb)mssqllocaldb;Database=AluraFilmes;Trusted_(
            }

            protected override void OnModelCreating(ModelBuilder modelBuilder)
            {

                modelBuilder.Entity<Ator>()
                    .ToTable("actor");

                modelBuilder.Entity<Ator>()
                    .Property(a => a.Id)
                    .HasColumnName("actor_id");

                modelBuilder.Entity<Ator>()
                    .Property(a => a.PrimieroNome)
                    .HasCoumnName("first_name")
                    .HasColumnType("varchar(45)")
                    .IsRequired();

                modelBuilder.Entity<Ator>()
                    .Property(a => a.UltimoNome)
                    .HasColumnType("varchar(45)")
                    .IsRequired();

                modelBuilder.Entity<Ator>()
                    .Property<DateTime>("last_update");
                   .HasColumnType ("datetime")
                    .HasDefaultValueSql("getdate()")
                    .IsRequired();
            }
        }
    }
```

Com isso, o nosso programa já é executável. Iremos testa-lo pressionando o atalho "F5" e receberemos uma mensagem de erro `Invalid object name 'Filmes'`. Já vimos esse erro acontecer em outro momento do curso, você se lembra o que deve ser feito para resolvermos esse problema?