

FÁBRICA DE DESENVOLVIMENTO

FÁBRICA DE DESENVOLVIMENTO - 2018

Prof. Thiago T. I. Yamamoto

#11 - RELACIONAMENTOS



thiagoyama



thiagoyama@gmail.com

#11 - RELACIONAMENTOS

- Relacionamento um-para-um
- Tabela - Foreign Key
- Transações
- Relacionamento um-para-muitos
- Query Multiple

UM PARA UM

Vamos criar uma tabela para armazenar as informações da carteira de motorista de um cliente:

The screenshot displays the SQL Server Enterprise Designer interface. The top pane shows the 'Design' view of the 'CarteiraMotorista' table. The table has three columns: 'Clienteld' (int, NOT NULL), 'DataValidade' (datetime, NOT NULL), and 'Numero' (bigint, NULL). The 'Clienteld' column is marked as the primary key. The bottom pane shows the T-SQL script for creating the table.

Name	Data Type	Allow Nulls	Default
Clienteld	int	<input type="checkbox"/>	
DataValidade	datetime	<input type="checkbox"/>	
Numero	bigint	<input checked="" type="checkbox"/>	

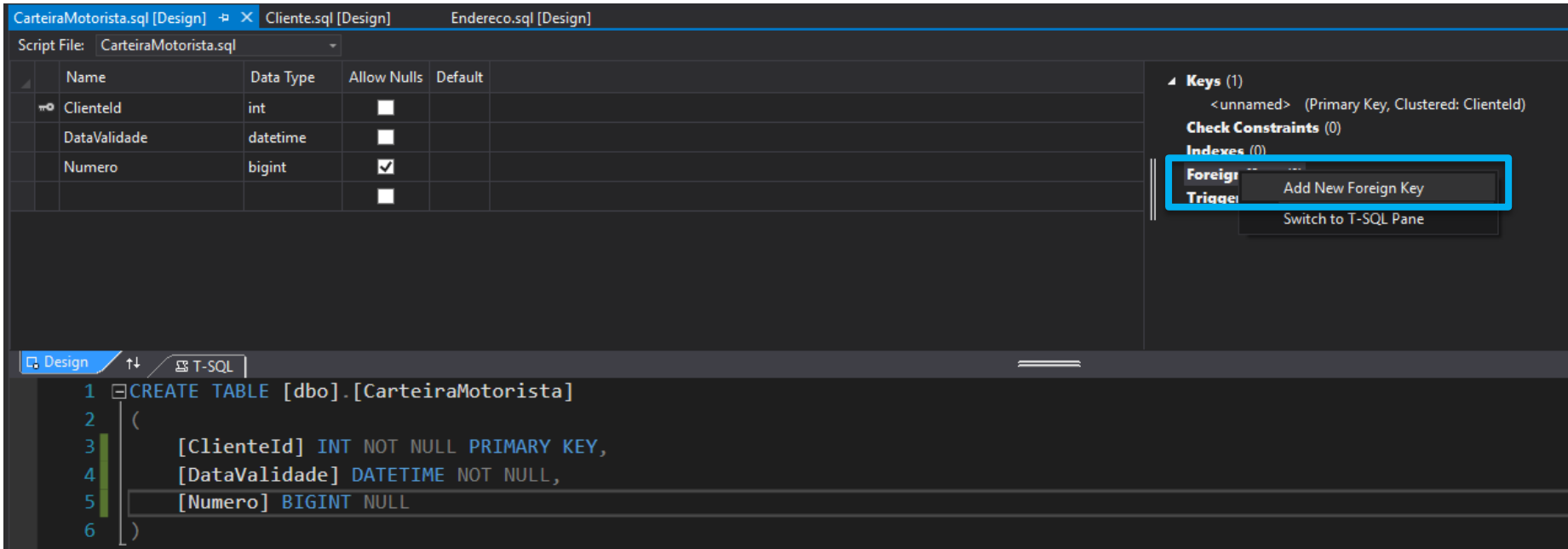
```
1 CREATE TABLE [dbo].[CarteiraMotorista]
2 (
3     [ClienteId] INT NOT NULL PRIMARY KEY,
4     [DataValidade] DATETIME NOT NULL,
5     [Numero] BIGINT NULL
6 )
```

Keys (1)
<unnamed> (Primary Key, Clustered: Clienteld)
Check Constraints (0)
Indexes (0)
Foreign Keys (0)
Triggers (0)

FOREIGN KEY - FK

Agora é preciso adicionar uma **foreign key**.

Clique com o botão direito do mouse na opção “**Foreign Key**” e depois em “**Add New Foreign Key.**”



The screenshot displays the SQL Server Enterprise Designer interface. The main window shows the design view of the 'CarteiraMotorista' table. The table structure is as follows:

Name	Data Type	Allow Nulls	Default
ClienteId	int	<input type="checkbox"/>	
DataValidade	datetime	<input type="checkbox"/>	
Numero	bigint	<input checked="" type="checkbox"/>	

The 'Keys' pane on the right shows a primary key for 'ClienteId'. The 'Foreign Key' option is highlighted in the context menu, and the 'Add New Foreign Key' button is visible.

The T-SQL pane at the bottom shows the following code:

```
1 CREATE TABLE [dbo].[CarteiraMotorista]
2 (
3     [ClienteId] INT NOT NULL PRIMARY KEY,
4     [DataValidade] DATETIME NOT NULL,
5     [Numero] BIGINT NULL
6 )
```

Para finalizar, ajuste o nome da **constraint** e as colunas e tabelas:

The screenshot displays the SQL Server Enterprise Designer interface. The top pane shows the 'Design' view of the 'CarteiraMotorista' table. The bottom pane shows the 'T-SQL' view with the following SQL code:

```
1 CREATE TABLE [dbo].[CarteiraMotorista]
2 (
3     [ClienteId] INT NOT NULL PRIMARY KEY,
4     [DataValidade] DATETIME NOT NULL,
5     [Numero] BIGINT NOT NULL,
6     CONSTRAINT [FK_CarteiraMotorista_Cliente] FOREIGN KEY ([ClienteId]) REFERENCES [Cliente]([Id])
7 )
8
```

The foreign key constraint line (line 6) is highlighted with a red rectangle. The right-hand pane shows the 'Keys' section, indicating a primary key for 'ClienteId' and a foreign key named 'FK_CarteiraMotorista_Cliente' that references the 'Id' column in the 'Cliente' table.

Para atualizar o modelo crie uma nova classe para a **Carteira de Motorista** e ajuste a classe **Cliente**:

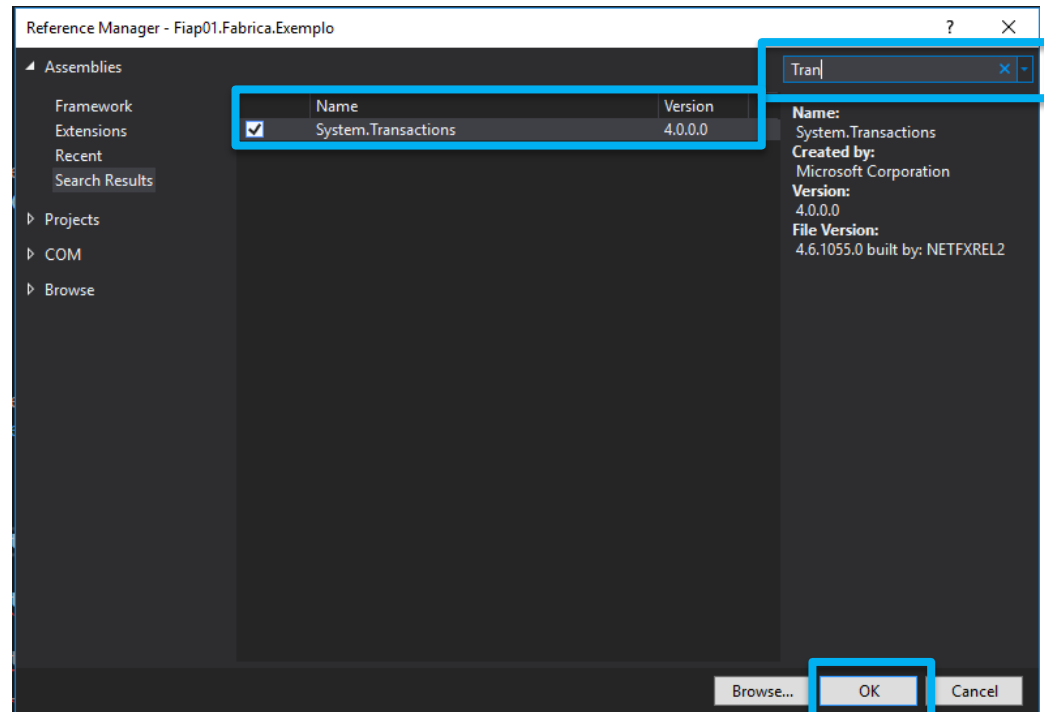
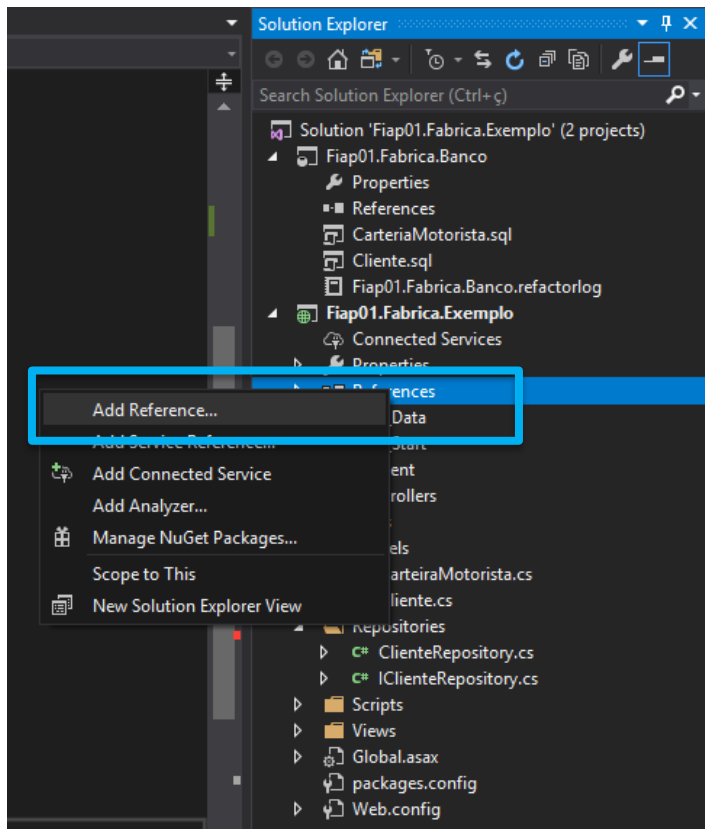
```
public class CarteiraMotorista
{
    public int Numero { get; set; }
    public DateTime DataValidade { get; set; }
    public int ClienteId { get; set; }
}
```

```
public class Cliente
{
    public int Id { get; set; }
    public string Nome { get; set; }
    public DateTime DataNascimento { get; set; }
    public bool Especial { get; set; }
    public decimal Saldo { get; set; }

    public CarteiraMotorista CarteiraMotorista { get; set; }
}
```


TRANSAÇÕES

- Podemos realizar o cadastro das duas entidades ao mesmo tempo, para isso precisamos utilizar a mesma **transação** para os dois **inserts**.
- Vamos adicionar a referencia para controlar as transações no



CADASTRANDO DUAS ENTIDADES

Cadastre o cliente e a carteira de motorista, caso exista:

```
using (var txScope = new TransactionScope())
{

    string sql1 = "INSERT INTO Cliente (Nome, DataNascimento, Saldo, Especial) VALUES (@Nome, @DataNascimento, @Saldo, @Especial);" +
        "SELECT CAST(SCOPE_IDENTITY() as int)";

    int id = db.Query<int>(sql1, cliente).Single();
    cliente.Id = id;

    if (cliente.CarteiraMotorista != null)
    {
        cliente.CarteiraMotorista.ClienteId = cliente.Id;
        string sql2 = "INSERT INTO CarteiraMotorista (Numero, DataValidade, ClienteId) VALUES (@Numero, @DataValidade, @ClienteId);";
        db.Query(sql2, cliente.CarteiraMotorista);
    }
    txScope.Complete();
}
```

LISTANDO AS DUAS ENTIDADES

Podemos realizar uma pesquisa que retorna o cliente e a carteira de motorista:

Inner Join

```
var sql = "SELECT * FROM Cliente AS CLI INNER JOIN  
CarteiraMotorista AS CM ON CLI.Id = CM.ClienteId";  
  
IList<Cliente> lista =  
    db.Query<Cliente, CarteiraMotorista, Cliente>(sql,  
        (cliente, carteira) => {  
            cliente.CarteiraMotorista = carteira;  
            return cliente; },  
        splitOn: "Id,ClienteId").ToList();
```

Chave
primárias

UM PARA MUITOS

TABELA ENDERECOS

Vamos criar uma tabela para armazenar os endereços de um cliente.

The screenshot displays the SQL Server Enterprise Designer interface. The top pane shows the 'Endereco.sql [Design]' view, which is a table design grid. The bottom pane shows the 'T-SQL' view, which contains the SQL script to create the table. The right-hand pane shows the 'Keys' section, indicating a primary key for the 'Id' column.

Name	Data Type	Allow Nulls	Default
Id	int	■	
Logradouro	varchar(50)	■	
Cep	varchar(8)	■	
Tipo	int	■	
ClienteId	int	■	

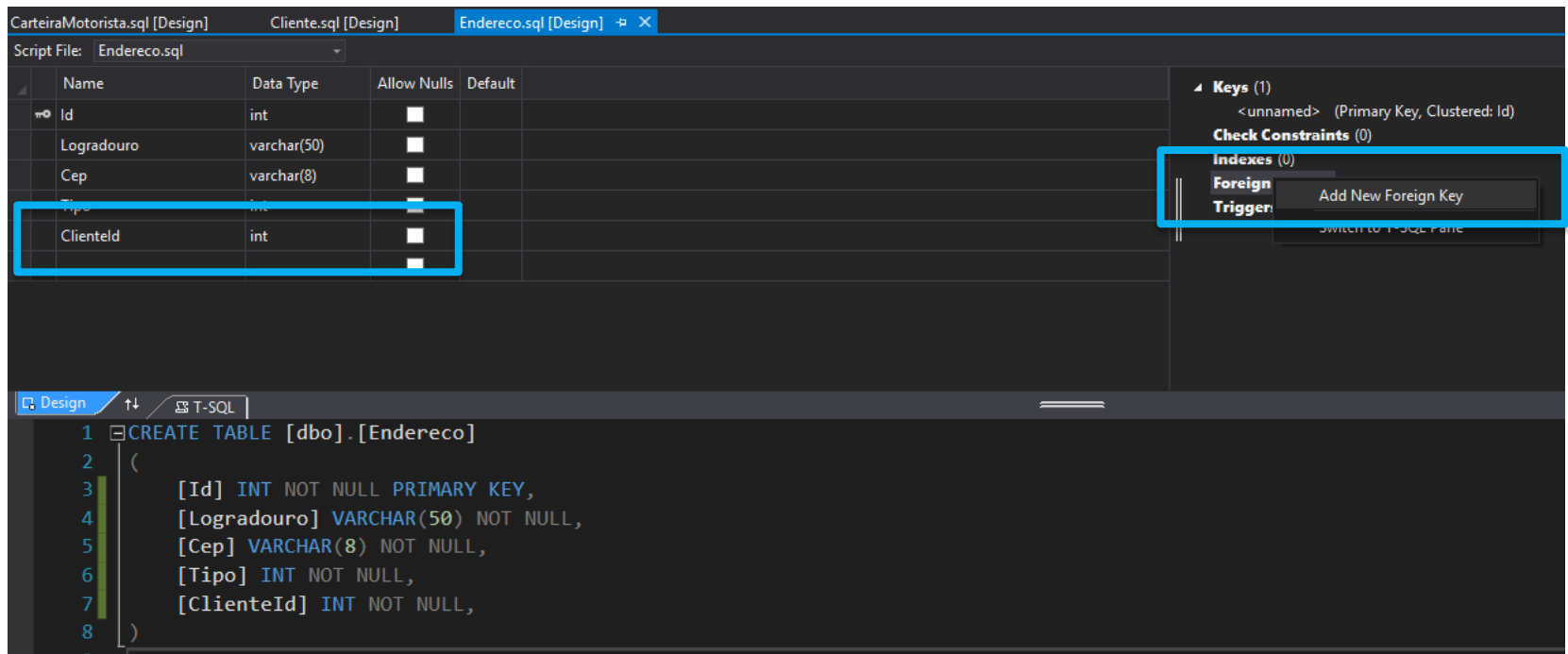
```
1 CREATE TABLE [dbo].[Endereco]
2 (
3     [Id] INT NOT NULL PRIMARY KEY,
4     [Logradouro] VARCHAR(50) NOT NULL,
5     [Cep] VARCHAR(8) NOT NULL,
6     [Tipo] INT NOT NULL,
7     [ClienteId] INT NOT NULL,
8 )
9
```

Keys (1)
<unnamed> (Primary Key, Clustered: Id)
Check Constraints (0)
Indexes (0)
Foreign Keys (0)
Triggers (0)

Agora é preciso adicionar uma **foreign key**.

Clique com o botão direito do mouse na opção “**Foreign Key**” e depois em “**Add New Foreign Key.**”

Note que criamos uma coluna para armazenar o **id** do **cliente**.



Para finalizar, ajuste o nome da constraint e as colunas e tabelas:

The screenshot displays the SQL Server Enterprise Designer interface. The top pane shows the 'Endereco' table design with the following columns:

Name	Data Type	Allow Nulls	Default
Id	int	■	
Logradouro	varchar(50)	■	
Cep	varchar(8)	■	
Tipo	int	■	
ClienteId	int	■	

The right pane shows the 'Keys' section with the following details:

- Keys (1)**
 - <unnamed> (Primary Key, Clustered: Id)
- Check Constraints (0)**
- Indexes (0)**
- Foreign Keys (1)**
 - FK_Endereco_Cliente (Id)
- Triggers (0)**

The bottom pane shows the T-SQL script for creating the table, with the foreign key constraint highlighted in a red box:

```
1 CREATE TABLE [dbo].[Endereco]
2 (
3     [Id] INT NOT NULL PRIMARY KEY,
4     [Logradouro] VARCHAR(50) NOT NULL,
5     [Cep] VARCHAR(8) NOT NULL,
6     [Tipo] INT NOT NULL,
7     [ClienteId] INT NOT NULL,
8     CONSTRAINT [FK_Endereco_Cliente] FOREIGN KEY ([ClienteId]) REFERENCES [Cliente]([Id])
9 )
```

CLASSE ENDEREÇO

A classe **Endereço** possui uma propriedade para a FK:

```
public class Endereco
{
    public int Id { get; set; }
    public string Logradouro { get; set; }
    public string Cep { get; set; }

    public TipoEndereco Tipo { get; set; }

    public int ClienteId { get; set; }
}

public enum TipoEndereco
{
    Residencial, Comercial
}
```


ATUALIZE A CLASSE CLIENTE

A classe **Cliente** terá uma propriedade que será uma lista de **Endereços**:

```
public class Cliente
{
    public int Id { get; set; }
    public string Nome { get; set; }
    public DateTime DataNascimento { get; set; }
    public bool Especial { get; set; }
    public decimal Saldo { get; set; }

    public CarteiraMotorista CarteiraMotorista { get; set; }

    public IList<Endereco> Enderecos { get; set; }
}
```

Vamos criar uma função que pesquisa um **cliente** e seus **endereços**:

```
var sql = "SELECT * FROM Cliente WHERE Id = @Id; " +
"SELECT * FROM Endereco WHERE ClienteId = @Id";

using (var results = db.QueryMultiple(sql, new { id }))
{
    var cliente = results.Read<Cliente>().SingleOrDefault();

    var enderecos = results.Read<Endereco>().ToList();

    if (cliente != null && enderecos != null)
    {
        cliente.Enderecos = enderecos;
    }
}
```

Copyright © 2018 - Prof. Me. Thiago T. I. Yamamoto

Todos direitos reservados. Reprodução ou divulgação total ou parcial deste documento é expressamente proibido sem o consentimento formal, por escrito, do Professor (autor).

*A vida vai ficando cada vez mais dura perto do topo.
Friedrich Nietzsche*