

Configurando uma aplicação para utilizar stored procedures em consultas

Você é membro de uma equipe de software que utiliza SQL Server, Entity Framework e consultas LINQ. Uma colega da sua equipe acaba de desenvolver uma stored procedure do SQL Server chamada `ps_Itens_Por_Cliente`, que recebe o id de um cliente e produz uma listagem detalhada de vendas por aquele cliente. Os dados dessa stored procedure deverão ser utilizados em diversas consultas LINQ. A stored procedure possui o seguinte script de criação:

```
CREATE PROCEDURE [dbo].[ps_Itens_Por_Cliente] @clienteId int = 0
AS
BEGIN

    SELECT
        i.FaixaId,
        i.ItemNotaFiscalId,
        i.NotaFiscalId,
        i.PrecoUnitario,
        i.Quantidade,
        i.PrecoUnitario* i.Quantidade As Total,
        n.DataNotaFiscal,
        f.Nome
    FROM ItemNotaFiscal i
    JOIN NotaFiscal n ON i.NotaFiscalId = n.NotaFiscalId
    JOIN Faixa f ON f.FaixaId = i.FaixaId
    WHERE n.ClienteId = @clienteId
END
```

Quais são os passos necessários para que as consultas LINQ possam utilizar essa stored procedure como origem de dados?

Selecione uma alternativa

A

1. Compilar o script da stored procedure no banco de dados SQL Server
2. Consumir a stored procedure numa consulta LINQ, passando como argumento o id do cliente, e acessando-a através do contexto do Entity Framework, como se faz com qualquer entidade do EF, assim como no seguinte exemplo:

```
int clienteId = 17;

using (var contexto = new AluraTunesEntities())
{
    var query = from p in contexto.ps_Vendas_Por_Cliente(clienteId)
                where <filtro_da_consulta>
                select p;
}
```

B

1. Criar uma nova consulta LINQ com exatamente a mesma lógica da stored procedure.
2. Utilizar essa nova consulta LINQ quando necessário

C

1. Criar uma string contendo o script da consulta da stored procedure:

```
string script = @"SELECT
    i.FaixaId.
```

```

-----,
i.ItemNotaFiscalId,
i.NotaFiscalId,
i.PrecoUnitario,
i.Quantidade,
i.PrecoUnitario* i.Quantidade As Total,
n.DataNotaFiscal,
f.Nome
FROM ItemNotaFiscal i
JOIN NotaFiscal n ON i.NotaFiscalId = n.NotaFiscalId
JOIN Faixa f ON f.FaixaId = i.FaixaId
WHERE n.ClienteId = @clienteId";

```

1. Consumir a stored procedure usando o script acima através do método `Execute()` do contexto:

```

int clienteId = 17;

using (var contexto = new AluraTunesEntities())
{
    var query = from p in contexto.Execute(script)
                where <filtro_da_consulta>
                select p;

```

D

1. Compilar o script da stored procedure no banco de dados SQL Server
2. Atualizar o modelo do entity framework a partir do banco de dados
3. Criar um método `ps_Vendas_Por_Cliente`, com o mesmo nome da stored procedure, para ser usado como proxy da stored procedure:
`public DBSet ps_Vendas_Por_Cliente(int clienteId) { }`
1. Consumir a stored procedures numa consulta LINQ, passando como argumento o id do cliente, e acessando-a através do novo método `ps_Vendas_Por_Cliente()` criado para servir como proxy da stored procedure:

```

int clienteId = 17;

using (var contexto = new AluraTunesEntities())
{
    var query = from p in ps_Vendas_Por_Cliente(clienteId)
                where <filtro_da_consulta>
                select p;

```

E

1. Compilar o script da stored procedure no banco de dados SQL Server
2. Atualizar o modelo do entity framework a partir do banco de dados
3. Consumir a stored procedures numa consulta LINQ, passando como argumento o id do cliente, e acessando-a através do `contexto` do Entity Framework, como se faz com qualquer entidade do EF, assim como no seguinte exemplo:

```

int clienteId = 17;

using (var contexto = new AluraTunesEntities())
{
    var query = from p in contexto.ps_Vendas_Por_Cliente(clienteId)
                where <filtro_da_consulta>
                select p;

```