**CURSO T-SQL – ALURA:**

**SQL Server: T-SQL com Microsoft SQL Server 2017**

**<https://cursos.alura.com.br/course/tsql-com-sql-server-2017/task/41175>**

**EXERCÍCIOS E EXEMPLOS DO CURSO**

**01. INTRODUÇÃO AO T-SQL.**

**02. ELEMENTOS DO T-SQL.**

**2.1. Atribuindo valores às variáveis.**

Crie 4 variáveis com as características abaixo:

* Nome: Cliente. Tipo: Caracteres com 10 posições. Valor: João
* Nome: Idade. Tipo: Inteiro. Valor: 10
* Nome: DataNascimento. Tipo: Data. Valor: 10/01/2007
* Nome: Custo. Tipo: Número com casas decimais. Valor: 10,23

Construa um script que declare estas variáveis, atribua valores a elas e exiba-as na saída do SQL Server Management Studio.

### Opinião do instrutor

**Opinião do instrutor**

Um exemplo de script é:

DECLARE @Cliente VARCHAR(10)

DECLARE @Idade INT

DECLARE @DataNascimento DATE

DECLARE @Custo FLOAT

SET @Cliente = 'João'

SET @Idade = 10

SET @DataNascimento = '20170110'

SET @Custo = 10.23

PRINT @Cliente

PRINT @Idade

PRINT @DataNascimento

PRINT @Custo

**2.2. Usando SELECT para atribuir valores.**

Crie uma variável chamada **NUMNOTAS** e atribua a ela o número de notas fiscais do dia 01/01/2017. Mostre na saída do script o valor da variável.

### Opinião do instrutor

Um exemplo de script é:

DECLARE @NUMNOTAS INT

SELECT @NUMNOTAS = COUNT(\*) FROM [NOTAS FISCAIS]

WHERE DATA = '20170101'

PRINT @NUMNOTAS

**2.3. Testando o número de notas fiscais.**

Crie um script que, baseado em uma data, conte o número de notas fiscais. Se houver mais de 70 notas, exiba a mensagem "Muita nota". Se não, exiba a mensagem "Pouca nota". Exiba também o número de notas.

### Opinião do instrutor

Um exemplo de script é:

DECLARE @DATANOTA DATE

DECLARE @NUMNOTAS INT

SET @DATANOTA = '20170102'

SELECT @NUMNOTAS = COUNT(\*) FROM [NOTAS FISCAIS]

WHERE DATA = @DATANOTA

IF @NUMNOTAS > 70

PRINT 'Muita nota'

ELSE

PRINT 'Pouca nota'

PRINT @NUMNOTAS

**2.4. Testando o número de notas fiscais usando SELECT.**

Baseado no script de resposta do exercício anterior:

DECLARE @DATANOTA DATE

DECLARE @NUMNOTAS INT

SET @DATANOTA = '20170102'

SELECT @NUMNOTAS = COUNT(\*) FROM [NOTAS FISCAIS]

WHERE DATA = @DATANOTA

IF @NUMNOTAS > 70

PRINT 'Muita nota'

ELSE

PRINT 'Pouca nota'

PRINT @NUMNOTAS

Em vez de testar com a variável @NUMNOTAS, use a própria consulta SQL na condição de teste.

### Opinião do instrutor

Um exemplo de script é:

DECLARE @DATANOTA DATE

SET @DATANOTA = '20170102'

IF (SELECT COUNT(\*) FROM [NOTAS FISCAIS]

WHERE DATA = @DATANOTA) > 70

PRINT 'Muita nota'

ELSE

PRINT 'Pouca nota'

**2.5. Número de notas para diversos dias.**

Sabendo que a função abaixo soma um dia a uma data:

SELECT DATEADD(DAY, 1, @DATA)

Faça um script que, a partir do dia 01/01/2017, conte o número de notas fiscais até o dia 10/01/2017. Imprima a data e o número de notas fiscais.

Dicas:

* Declare variáveis do tipo DATE: DATAINICIAL e DATAFINAL;
* Faça um *loop* testando se a data inicial é menor que a data final;
* Imprima a data e o número de notas na saída do Management Studio. Não esqueça de converter as variáveis para VARCHAR;
* Acrescente um dia à data.

### Opinião do instrutor

Um exemplo de script é:

DECLARE @DATAINICIAL DATE

DECLARE @DATAFINAL DATE

DECLARE @NUMNOTAS INT

SET @DATAINICIAL = '20170101'

SET @DATAFINAL = '20170110'

WHILE @DATAINICIAL <= @DATAFINAL

BEGIN

SELECT @NUMNOTAS = COUNT(\*) FROM [NOTAS FISCAIS]

WHERE DATA = @DATAINICIAL

PRINT CONVERT(VARCHAR(10), @DATAINICIAL) + ' --> '

+ CONVERT(VARCHAR(10), @NUMNOTAS)

SELECT @DATAINICIAL = DATEADD(DAY, 1, @DATAINICIAL)

END

2.6. **Número de notas salvas em uma tabela.**

Continue evoluindo o script da resposta do exercício anterior. Agora, inclua o dia e o número de notas em uma tabela.

Segue o script do exercício anterior:

DECLARE @DATAINICIAL DATE

DECLARE @DATAFINAL DATE

DECLARE @NUMNOTAS INT

SET @DATAINICIAL = '20170101'

SET @DATAFINAL = '20170110'

WHILE @DATAINICIAL <= @DATAFINAL

BEGIN

SELECT @NUMNOTAS = COUNT(\*) FROM [NOTAS FISCAIS]

WHERE DATA = @DATAINICIAL

PRINT CONVERT(VARCHAR(10), @DATAINICIAL) + ' --> '

+ CONVERT(VARCHAR(10), @NUMNOTAS)

SELECT @DATAINICIAL = DATEADD(DAY, 1, @DATAINICIAL)

END

### Opinião do instrutor

Um exemplo de script é:

IF OBJECT\_ID('TABELANOTAS','U') IS NOT NULL

DROP TABLE TABELANOTAS

CREATE TABLE TABELANOTAS (DATA DATE, NUMNOTAS INT)

DECLARE @DATAINICIAL DATE

DECLARE @DATAFINAL DATE

DECLARE @NUMNOTAS INT

SET @DATAINICIAL = '20170101'

SET @DATAFINAL = '20170110'

WHILE @DATAINICIAL <= @DATAFINAL

BEGIN

SELECT @NUMNOTAS = COUNT(\*) FROM [NOTAS FISCAIS]

WHERE DATA = @DATAINICIAL

INSERT INTO TABELANOTAS (DATA, NUMNOTAS)

VALUES (@DATAINICIAL, @NUMNOTAS)

SELECT @DATAINICIAL = DATEADD(DAY, 1, @DATAINICIAL)

END

SELECT \* FROM TABELANOTAS

**03. ESCREVENDO FUNÇÕES.**

**3.1. Função para obter o número de notas fiscais.**

Em exercícios anteriores, construímos um script para obter o número de notas fiscais de uma determinada data. Veja-o abaixo:

DECLARE @NUMNOTAS INT

SELECT @NUMNOTAS = COUNT(\*) FROM [NOTAS FISCAIS]

WHERE DATA = '20170101'

PRINT @NUMNOTAS

Transforme este script em uma função onde passamos a data como parâmetro e retornamos o número de notas. Chame esta função de **NumeroNotas**. Após a sua criação, teste seu uso com um SELECT.

### Opinião do instrutor

Um exemplo de script é:

CREATE FUNCTION NUMERONOTAS (@DATA DATE) RETURNS INT

AS

BEGIN

DECLARE @NUMNOTAS INT

SELECT @NUMNOTAS = COUNT(\*) FROM [NOTAS FISCAIS]

WHERE DATA = @DATA

RETURN @NUMNOTAS

END

SELECT [dbo].[NUMERONOTAS]('20170202')

3.2. **Criando uma tabela com número de notas.**

Em exercícios anteriores construímos um script para criar uma tabela com o número de notas fiscais para um período de datas. Veja o script abaixo:

IF OBJECT\_ID('TABELANOTAS','U') IS NOT NULL

DROP TABLE TABELANOTAS

CREATE TABLE TABELANOTAS (DATA DATE, NUMNOTAS INT)

DECLARE @DATAINICIAL DATE

DECLARE @DATAFINAL DATE

DECLARE @NUMNOTAS INT

SET @DATAINICIAL = '20170101'

SET @DATAFINAL = '20170110'

WHILE @DATAINICIAL <= @DATAFINAL

BEGIN

SELECT @NUMNOTAS = COUNT(\*) FROM [NOTAS FISCAIS]

WHERE DATA = @DATAINICIAL

INSERT INTO TABELANOTAS (DATA, NUMNOTAS)

VALUES (@DATAINICIAL, @NUMNOTAS)

SELECT @DATAINICIAL = DATEADD(DAY, 1, @DATAINICIAL)

END

SELECT \* FROM TABELANOTAS

Reescreva este script usando a função NumeroNotas no momento de inserir dados na tabela. Execute o SELECT para exibir os dados.

### Opinião do instrutor

Um exemplo de script é:

IF OBJECT\_ID('TABELANOTAS','U') IS NOT NULL

DROP TABLE TABELANOTAS

CREATE TABLE TABELANOTAS (DATA DATE, NUMNOTAS INT)

DECLARE @DATAINICIAL DATE

DECLARE @DATAFINAL DATE

SET @DATAINICIAL = '20170101'

SET @DATAFINAL = '20170110'

WHILE @DATAINICIAL <= @DATAFINAL

BEGIN

INSERT INTO TABELANOTAS (DATA, NUMNOTAS)

VALUES (@DATAINICIAL, [dbo].[NUMERONOTAS](@DATAINICIAL))

SELECT @DATAINICIAL = DATEADD(DAY, 1, @DATAINICIAL)

END

SELECT \* FROM TABELANOTAS

**3.3. Função para retornar o número de notas.**

Veja a consulta abaixo:

SELECT DISTINCT DATA, [dbo].[NUMERONOTAS](DATA) AS NUMERO

FROM [NOTAS FISCAIS] WHERE DATA >= ‘20170101’

AND DATA <= ‘20170110’

Ela irá retornar o número de notas fiscais entre duas datas. Transforme isto em uma função chamada **FuncTabelaNotas**, onde o resultado é a consulta acima. Lembrando que a data inicial e final serão parâmetros desta função. Depois, teste a função através de um SELECT.

### Opinião do instrutor

Um exemplo de script é:

CREATE FUNCTION FuncTabelaNotas (

@DATAINICIAL AS DATE,

@DATAFINAL AS DATE

) RETURNS TABLE

RETURN SELECT DISTINCT DATA, [dbo].[NUMERONOTAS](DATA) AS NUMERO

FROM [NOTAS FISCAIS] WHERE DATA >= @DATAINICIAL

AND DATA <= @DATAFINAL

SELECT \* FROM [dbo].[FuncTabelaNotas]('20170101','20170110')

**3.4. Alterando a função que lista o número de notas fiscais.**

Veja a consulta abaixo:

SELECT DATA, COUNT(\*) AS NUMERO FROM [NOTAS FISCAIS]

WHERE DATA >= '20170101' AND DATA <= '20170110'

GROUP BY DATA

Ela também retorna o número de notas entre duas datas. Modifique a função FuncTabelaNotas para que utilize esta consulta acima.

### Opinião do instrutor

Um exemplo de script é:

ALTER FUNCTION FuncTabelaNotas (

@DATAINICIAL AS DATE,

@DATAFINAL AS DATE

) RETURNS TABLE

RETURN SELECT DATA, COUNT(\*) AS NUMERO

FROM [NOTAS FISCAIS] WHERE DATA >= @DATAINICIAL AND

DATA <= @DATAFINAL

GROUP BY DATA

SELECT \* FROM [dbo].[FuncTabelaNotas]('20170101','20170110')

**04. STORED PROCEDURES**

**4.1. Acrescentando a entidade PRODUTOS na SP.**

No vídeo desta aula fizemos a *stored procedure* abaixo:

CREATE PROCEDURE BuscaPorEntidades @ENTIDADE AS VARCHAR(10)

AS

BEGIN

IF @ENTIDADE = 'CLIENTES'

SELECT [CPF] AS IDENTIFICADOR, [NOME] AS DESCRITOR,

[BAIRRO] AS BAIRRO FROM [TABELA DE CLIENTES]

ELSE IF @ENTIDADE = 'VENDEDORES'

SELECT [MATRICULA] AS IDENTIFICADOR, [NOME] AS DESCRITOR,

[BAIRRO] AS BAIRRO FROM [TABELA DE VENDEDORES]

END

Crie uma segunda stored procedure chamada **BuscaPorEntidadesCompleta** com o mesmo código da de cima, mas acrescente a entidade **PRODUTOS**. Das entidades, liste apenas os seus identificadores e os seus nomes.

### Opinião do instrutor

Um exemplo de script é:

CREATE PROCEDURE BuscaPorEntidadesCompleta @ENTIDADE AS VARCHAR(10)

AS

BEGIN

IF @ENTIDADE = 'CLIENTES'

SELECT [CPF] AS IDENTIFICADOR, [NOME] AS DESCRITOR

FROM [TABELA DE CLIENTES]

ELSE IF @ENTIDADE = 'VENDEDORES'

SELECT [MATRICULA] AS IDENTIFICADOR, [NOME] AS DESCRITOR

FROM [TABELA DE VENDEDORES]

ELSE IF @ENTIDADE = 'PRODUTOS'

SELECT [CODIGO DO PRODUTO] AS IDENTIFICADOR, [NOME DO PRODUTO]

AS DESCRITOR FROM [TABELA DE PRODUTOS]

END

EXEC BuscaPorEntidadesCompleta @ENTIDADE = 'CLIENTES'

EXEC BuscaPorEntidadesCompleta @ENTIDADE = 'VENDEDORES'

EXEC BuscaPorEntidadesCompleta @ENTIDADE = 'PRODUTOS'

**4.2. Acrescentando na SP a entidade produto.**

Na nossa empresa de suco de frutas temos 3 categorias de produtos: Garrafas, Lata ou PET. Imagine que, pela legislação, o imposto pago depende do tipo de embalagem do produto e, a cada momento, ele muda. Temos o campo IMPOSTO na tabela de notas fiscais, que determina o imposto a ser pago (alíquota sobre o faturamento). Faça uma *stored procedure* que terá como entrada de dados:

* Mês
* Ano
* Alíquota
* Tipo de Produto (Garrafas, Lata ou PET)

Ela irá modificar a alíquota para a alíquota informada na entrada da *stored procedure*, para as vendas de todas as notas fiscais no mês/ano informados, para todos os produtos do tipo informado.

Dica:

* Comece determinando a consulta que seleciona todas as notas fiscais dentro do mês/ano para um tipo de produto;
* Construa um comando UPDATE modificando o campo IMPOSTO de todas as notas, usando o mesmo WHERE da consulta inicial;
* Construa a *stored procedure* usando o UPDATE acima, determinando e passando os parâmetros contidos no enunciado. Nome da *stored procedure*: **AtualizaImposto**.

Consulta que determina todas as notas fiscais dentro de um mês/ano para um tipo de produto:

SELECT \* FROM [NOTAS FISCAIS] NF

INNER JOIN [ITENS NOTAS FISCAIS] INF ON

NF.NUMERO = INF.NUMERO

INNER JOIN [TABELA DE PRODUTOS] TP ON

TP.[CODIGO DO PRODUTO] = INF.[CODIGO DO PRODUTO]

WHERE MONTH(NF.DATA) = 1 AND YEAR(NF.DATA) = 2017

AND TP.EMBALAGEM = 'Lata'

Comando UPDATE para alterar o valor do campo IMPOSTO:

UPDATE NF SET NF.IMPOSTO = 0.18

FROM [NOTAS FISCAIS] NF

INNER JOIN [ITENS NOTAS FISCAIS] INF

ON NF.NUMERO = INF.NUMERO

INNER JOIN [TABELA DE PRODUTOS] TP

ON TP.[CODIGO DO PRODUTO] = INF.[CODIGO DO PRODUTO]

WHERE MONTH(NF.DATA) = 1 AND YEAR(NF.DATA) = 2017

AND TP.EMBALAGEM = 'Lata'

Construção da *stored procedure*:

CREATE PROCEDURE AtualizaImposto @MES AS INT, @ANO AS INT,

@EMBALAGEM AS VARCHAR(10), @IMPOSTO AS FLOAT

AS

UPDATE NF SET NF.IMPOSTO = @IMPOSTO FROM [NOTAS FISCAIS] NF

INNER JOIN [ITENS NOTAS FISCAIS] INF

ON NF.NUMERO = INF.NUMERO

INNER JOIN [TABELA DE PRODUTOS] TP

ON TP.[CODIGO DO PRODUTO] = INF.[CODIGO DO PRODUTO]

WHERE MONTH(NF.DATA) = @MES AND YEAR(NF.DATA) = @ANO

AND TP.EMBALAGEM = @EMBALAGEM

Testando a *stored procedure*:

EXEC AtualizaImposto @MES = 2, @ANO = 2017, @EMBALAGEM = 'PET', @IMPOSTO = 0.16

**4.3. Lista de número de notas passando uma tabela como parâmetro.**

Temos a seguinte consulta que nos retorna o número de notas fiscais por dia, onde @ListaDatas é uma variável do tipo tabela, com a lista de datas a serem exibidas:

SELECT DATA, COUNT(\*) AS NUMERO FROM [NOTAS FISCAIS]

WHERE DATA IN (SELECT DATANOTA FROM @ListaDatas)

GROUP BY DATA

Construa uma SP que retorne o número de notas fiscais por dia, baseada na lista de dias passada como parâmetro.

Dicas:

* Crie um estrutura do tipo TYPE, onde iremos inserir as datas;
* Passe este tipo para a SP como parâmetro, representando a lista de datas. O nome da SP deve ser **ListaNumeroNotas**;
* Utilize o SELECT mencionado acima, usando a variável @ListaDatas como sendo a lista de datas passada como parâmetro;
* Crie a SP;
* Inicialize algumas datas em uma variável do tipo definido no passo inicial;
* Execute a SP.

### Opinião do instrutor

Criando uma estrutura do tipo TYPE:

CREATE TYPE ListaDatas as TABLE (

DATANOTA date NOT NULL

)

Criando a SP usando este tipo como parâmetro e executando a consulta definida no enunciado:

CREATE PROCEDURE ListaNumeroNotas @ListaDatas

AS ListaDatas READONLY AS

SELECT DATA, COUNT(\*) AS NUMERO FROM [NOTAS FISCAIS]

WHERE DATA IN (SELECT DATANOTA FROM @ListaDatas)

GROUP BY DATA

Crie uma variável incluindo a lista de datas e, em seguida, execute a SP.

DECLARE @ListaDatas AS ListaDatas

INSERT INTO @ListaDatas (DATANOTA) VALUES ('20180101'),

('20180102'), ('20180103')

EXEC ListaNumeroNotas @ListaDatas=@ListaDatas

**4.4. Obtendo o número de notas fiscais pela saída de uma SP.**

Construa uma SP (nome **NumNotasSP**) cujos parâmetros são uma data, passada como valor, e o número de notas, passado como referência. Depois, faça um script onde, na variável @NUMNOTAS, some as notas do dia 01/01/2017 e 02/01/2017.

Dica: Na SP, crie uma variável auxiliar e some da variável que é passada como referência.

### Opinião do instrutor

Vamos criar a SP passando como parâmetro a data (valor) e o número de notas (como referência):

CREATE PROCEDURE NumNotasSP

@DATA AS DATE,

@NUMNOTAS AS INT OUTPUT

AS

BEGIN

DECLARE @NUMNOTASLOCAL INT

SELECT @NUMNOTASLOCAL = COUNT(\*)

FROM [NOTAS FISCAIS] WHERE [DATA] = @DATA

SET @NUMNOTAS = @NUMNOTAS + @NUMNOTASLOCAL

END

Agora vamos chamar a SP duas vezes, uma para o dia 01/01/2017 e outra vez para 02/01/2017. Depois exibimos o valor final do número de notas:

DECLARE @DATA DATE

DECLARE @NUMNOTAS INT

SET @NUMNOTAS = 0

SET @DATA = '20170101'

EXEC NumNotasSP @DATA = @DATA, @NUMNOTAS = @NUMNOTAS OUTPUT

SET @DATA = '20170102'

EXEC NumNotasSP @DATA = @DATA, @NUMNOTAS = @NUMNOTAS OUTPUT

PRINT @NUMNOTAS

**05. TRATAMENTO DE ERROS**

**5.1. Tratando erro na inclusão de um vendedor.**

Crie a seguinte *Stored Procedure*, conforme o código abaixo:

CREATE PROCEDURE InclusaoVendedor

@MATRICULA AS VARCHAR(5), @NOME AS VARCHAR(100), @PERCENTUAL AS FLOAT,

@DATAADMISSAO AS DATE, @FERIAS AS BIT, @BAIRRO AS VARCHAR(50)

AS

BEGIN

INSERT INTO [TABELA DE VENDEDORES] (MATRICULA, NOME, [PERCENTUAL COMISSÃO], [DATA ADMISSÃO], [DE FERIAS], BAIRRO)

VALUES (@MATRICULA, @NOME, @PERCENTUAL, @DATAADMISSAO, @FERIAS, @BAIRRO)

END

Ela irá incluir um vendedor na tabela de vendedores. Depois, execute o comando:

EXEC InclusaoVendedor '00238','Pericles Alves',0.11,'20160821',0,'Santo Amaro'

Temos o erro:

Msg 2627, Level 14, State 1, Procedure InclusaoVendedor, Line 6 [Batch Start Line 192]

Violation of PRIMARY KEY constraint 'PK\_VENDEDORES'.

Cannot insert duplicate key in object 'dbo.TABELA DE VENDEDORES'.

The duplicate key value is (00238).

The statement has been terminated.

Crie uma nova SP chamada **InclusaoVendedor01**, tratando este erro usando **@@ERROR** e **@@ROWCOUNT**.

### Opinião do instrutor

Crie a SP:

CREATE PROCEDURE InclusaoVendedor01

@MATRICULA AS VARCHAR(5), @NOME AS VARCHAR(100),

@PERCENTUAL AS FLOAT,

@DATAADMISSAO AS DATE, @FERIAS AS BIT, @BAIRRO AS VARCHAR(50),

@NUMERRO INT OUTPUT, @NUMLINHA INT OUTPUT

AS

BEGIN

INSERT INTO [TABELA DE VENDEDORES] (MATRICULA, NOME,

[PERCENTUAL COMISSÃO], [DATA ADMISSÃO], [DE FERIAS], BAIRRO)

VALUES (@MATRICULA, @NOME, @PERCENTUAL, @DATAADMISSAO, @FERIAS, @BAIRRO)

SELECT @NUMERRO = @@ERROR, @NUMLINHA = @@ROWCOUNT

END

E depois execute-a:

DECLARE @NUMERRO INT

DECLARE @NUMLINHA INT

EXEC InclusaoVendedor01 '00238', 'Pericles Alves', 0.11,

'20160821', 0, 'Santo Amaro', @NUMERRO =

@NUMERRO OUTPUT, @NUMLINHA = @NUMLINHA OUTPUT

IF @NUMERRO <> 0

PRINT 'Houve um erro: ' +

CONVERT(VARCHAR(30), @NUMERRO) + ' - ' +

CONVERT(VARCHAR(30), @NUMLINHA)

**5.2. Tratando erro na inclusão de um vendedor com TRY-CATCH.**

Crie uma nova SP chamada **InclusaoVendedor02**, agora tratando o erro com TRY-CATCH.

### Opinião do instrutor

Crie a SP:

CREATE PROCEDURE InclusaoVendedor02

@MATRICULA AS VARCHAR(5), @NOME AS VARCHAR(100),

@PERCENTUAL AS FLOAT,

@DATAADMISSAO AS DATE, @FERIAS AS BIT, @BAIRRO AS VARCHAR(50),

@MENSAGEM VARCHAR(100) OUTPUT

AS

BEGIN

BEGIN TRY

INSERT INTO [TABELA DE VENDEDORES] (MATRICULA, NOME,

[PERCENTUAL COMISSÃO], [DATA ADMISSÃO], [DE FERIAS], BAIRRO)

VALUES (@MATRICULA, @NOME, @PERCENTUAL, @DATAADMISSAO, @FERIAS, @BAIRRO)

END TRY

BEGIN CATCH

SET @MENSAGEM = 'Houve um erro número: ' + CONVERT(VARCHAR(10), @@ERROR)

END CATCH

END

E depois execute-a:

DECLARE @MENSAGEM VARCHAR(30)

EXEC InclusaoVendedor02 '00238', 'Pericles Alves', 0.11, '20160821',

0, 'Santo Amaro', @MENSAGEM = @MENSAGEM OUTPUT

IF @MENSAGEM <> ''

PRINT @MENSAGEM

**5.3. Tratando erro com funções de erro.**

Crie uma nova SP chamada **InclusaoVendedor03**, agora tratando o erro com **TRY-CATCH**, exibindo todas as funções de erro na mensagem.

### Opinião do instrutor

Crie a SP:

CREATE PROCEDURE InclusaoVendedor03

@MATRICULA AS VARCHAR(5), @NOME AS VARCHAR(100),

@PERCENTUAL AS FLOAT, @DATAADMISSAO AS DATE,

@FERIAS AS BIT, @BAIRRO AS VARCHAR(50),

@MENSAGEM VARCHAR(100) OUTPUT

AS

BEGIN

BEGIN TRY

INSERT INTO [TABELA DE VENDEDORES]

(MATRICULA, NOME, [PERCENTUAL COMISSÃO],

[DATA ADMISSÃO], [DE FERIAS], BAIRRO)

VALUES (@MATRICULA, @NOME, @PERCENTUAL, @DATAADMISSAO,

@FERIAS, @BAIRRO)

END TRY

BEGIN CATCH

SET @MENSAGEM = 'Houve um erro número: ' +

CONVERT(VARCHAR(10), ERROR\_NUMBER()) + ' - '

SET @MENSAGEM = @MENSAGEM + 'Mensagem: ' +

ERROR\_MESSAGE() + ' - '

SET @MENSAGEM = @MENSAGEM + 'Grau de severidade do erro: ' +

CONVERT(VARCHAR(10), ERROR\_SEVERITY()) + ' - '

SET @MENSAGEM = @MENSAGEM + 'Estado do erro: ' +

CONVERT(VARCHAR(10), ERROR\_STATE()) + ' - '

SET @MENSAGEM = @MENSAGEM + 'Número da linha: ' +

CONVERT(VARCHAR(10), ERROR\_LINE()) + ' - '

SET @MENSAGEM = @MENSAGEM + 'Procedure: ' + ERROR\_PROCEDURE()

END CATCH

END

E depois execute-a:

DECLARE @MENSAGEM VARCHAR(MAX)

EXEC InclusaoVendedor03 '00238', 'Pericles Alves', 0.11, '20160821',

0, 'Santo Amaro', @MENSAGEM = @MENSAGEM OUTPUT

IF @MENSAGEM <> ''

PRINT @MENSAGEM

**06. CURSOR E UM EXEMPLO PRÁTICO**

**6.1. Achando o valor total do crédito.**

Crie um script usando um *cursor* para achar o valor total de todos os créditos de todos os clientes.

Dicas:

* Declare duas variáveis: uma que receba o limite de crédito do *cursor* e outra o limite de crédito total;
* Faça um *loop* no *cursor* e vá somando o valor do limite de cada cliente na variável de limite de crédito total;
* Exiba o valor total do limite.

### Opinião do instrutor

Um exemplo de script é:

DECLARE @LIMITECREDITO FLOAT

DECLARE @LIMITECREDITOACUM FLOAT

DECLARE CURSOR1 CURSOR FOR SELECT [LIMITE DE CREDITO]

FROM [TABELA DE CLIENTES]

SET @LIMITECREDITOACUM = 0

OPEN CURSOR1

FETCH NEXT FROM CURSOR1 INTO @LIMITECREDITO

WHILE @@FETCH\_STATUS = 0

BEGIN

SET @LIMITECREDITOACUM = @LIMITECREDITOACUM +

@LIMITECREDITO

FETCH NEXT FROM CURSOR1 INTO @LIMITECREDITO

END

CLOSE CURSOR1

DEALLOCATE CURSOR1

PRINT @LIMITECREDITOACUM

**6.2. Calculando o valor total do faturamento.**

Crie um script usando um *cursor* para achar o valor total do faturamento para um mês e um ano.

Dicas:

* Declare três variáveis: uma que recebe a quantidade, outra o preço e uma terceira que irá acumular o faturamento;
* Faça um *loop* no *cursor* e vá somando o faturamento de cada nota;
* Exiba o valor total do limite;
* Lembrando a consulta que obtém o faturamento dentro de um mês e ano:

SELECT INF.QUANTIDADE, INF.PREÇO FROM [ITENS NOTAS FISCAIS] INF

INNER JOIN [NOTAS FISCAIS] NF ON NF.NUMERO = INF.NUMERO

WHERE MONTH(NF.DATA) = 1 AND YEAR(NF.DATA) = 2017

### Opinião do instrutor

Um exemplo de script é:

DECLARE @QUANTIDADE INT

DECLARE @PRECO FLOAT

DECLARE @FATURAMENTOACUM FLOAT

DECLARE @MES INT

DECLARE @ANO INT

SET @MES = 1

SET @ANO = 2017

DECLARE CURSOR1 CURSOR FOR SELECT INF.QUANTIDADE, INF.PREÇO FROM

[ITENS NOTAS FISCAIS] INF

INNER JOIN [NOTAS FISCAIS] NF ON NF.NUMERO = INF.NUMERO

WHERE MONTH(NF.DATA) = @MES AND YEAR(NF.DATA) = @ANO

SET @FATURAMENTOACUM = 0

OPEN CURSOR1

FETCH NEXT FROM CURSOR1 INTO @QUANTIDADE, @PRECO

WHILE @@FETCH\_STATUS = 0

BEGIN

SET @FATURAMENTOACUM = @FATURAMENTOACUM + (@QUANTIDADE \* @PRECO)

FETCH NEXT FROM CURSOR1 INTO @QUANTIDADE, @PRECO

END

CLOSE CURSOR1

DEALLOCATE CURSOR1

PRINT @FATURAMENTOACUM

**6.3. Tabela com números aleatórios.**

Crie uma variável TABELA com um campo INT, use um loop para gravar 100 números aleatórios entre 0 e 1000 nesta tabela. Depois, liste esta tabela em uma consulta.

### Opinião do instrutor

Um exemplo de script é:

DECLARE @TABELA TABLE (NUMERO INT)

DECLARE @CONTADOR INT

DECLARE @CONTMAXIMO INT

SET @CONTADOR = 1

SET @CONTMAXIMO = 100

WHILE @CONTADOR <= @CONTMAXIMO

BEGIN

INSERT INTO @TABELA (NUMERO) VALUES

([dbo].[NumeroAleatorio](0,1000))

SET @CONTADOR = @CONTADOR + 1

END

SELECT \* FROM @TABELA

**6.4. Fazendo script para produto aleatório.**

No vídeo desta aula, criamos um script para obter o cliente e vendedor através da função de número aleatório. Neste exercício, crie o script para obter o produto também usando a função de número aleatório.

### Opinião do instrutor

Um exemplo de script é:

DECLARE @PRODUTO\_ALEATORIO VARCHAR(12)

DECLARE @VAL\_INICIAL INT

DECLARE @VAL\_FINAL INT

DECLARE @ALEATORIO INT

DECLARE @CONTADOR INT

SET @CONTADOR = 1

SET @VAL\_INICIAL = 1

SELECT @VAL\_FINAL = COUNT(\*) FROM [TABELA DE PRODUTOS]

SET @ALEATORIO = [dbo].[NumeroAleatorio](@VAL\_INICIAL, @VAL\_FINAL)

DECLARE CURSOR1 CURSOR FOR SELECT [CODIGO DO PRODUTO]

FROM [TABELA DE PRODUTOS]

OPEN CURSOR1

FETCH NEXT FROM CURSOR1 INTO @PRODUTO\_ALEATORIO

WHILE @CONTADOR < @ALEATORIO

BEGIN

FETCH NEXT FROM CURSOR1 INTO @PRODUTO\_ALEATORIO

SET @CONTADOR = @CONTADOR + 1

END

CLOSE CURSOR1

DEALLOCATE CURSOR1

SELECT @PRODUTO\_ALEATORIO

**6.5. Obtendo as entidades através da função.**

Crie uma consulta no Management Studio; declare três variáveis que recebem o produto, cliente e vendedor; e obtenha os valores destas variáveis através da função **EntidadeAleatoria**, criada no vídeo desta aula.

### Opinião do instrutor

Um exemplo de script é:

DECLARE @CLIENTE VARCHAR(20)

DECLARE @VENDEDOR VARCHAR(20)

DECLARE @PRODUTO VARCHAR(20)

SET @CLIENTE = [dbo].[EntidadeAleatoria]('CLIENTE')

SET @VENDEDOR = [dbo].[EntidadeAleatoria]('VENDEDOR')

SET @PRODUTO = [dbo].[EntidadeAleatoria]('PRODUTO')

PRINT @VENDEDOR

PRINT @CLIENTE

PRINT @PRODUTO

**6.6. Obtendo as entidades através da função.**

Complemente a consulta criada no exercício anterior. Agora, obtenha os dados da nota fiscal, conforme o vídeo desta aula, e inclua na tabela de notas e de itens. Execute o script diversas vezes e veja se irá ocorrer um erro. Transcreva este erro.

### Opinião do instrutor

Vamos incluir os dados de notas e itens na tabela:

DECLARE @CLIENTE VARCHAR(12)

DECLARE @VENDEDOR VARCHAR(12)

DECLARE @PRODUTO VARCHAR(12)

DECLARE @DATA DATE

DECLARE @NUMERO INT

DECLARE @IMPOSTO FLOAT

DECLARE @NUM\_ITENS INT

DECLARE @CONTADOR INT

DECLARE @QUANTIDADE INT

DECLARE @PRECO FLOAT

SET @DATA = '20180521'

SET @CLIENTE = [dbo].[EntidadeAleatoria]('CLIENTE')

SET @VENDEDOR = [dbo].[EntidadeAleatoria]('VENDEDOR')

SELECT @NUMERO = MAX(NUMERO) + 1 FROM [NOTAS FISCAIS]

SET @IMPOSTO = 0.18

SET @CONTADOR = 1

SET @NUM\_ITENS = [dbo].[NumeroAleatorio](2, 10)

INSERT INTO [NOTAS FISCAIS] (CPF, MATRICULA, DATA, NUMERO, IMPOSTO)

VALUES (@CLIENTE, @VENDEDOR, @DATA, @NUMERO, @IMPOSTO)

WHILE @CONTADOR <= @NUM\_ITENS

BEGIN

SET @PRODUTO = [dbo].[EntidadeAleatoria]('PRODUTO')

SET @QUANTIDADE = [dbo].[NumeroAleatorio](5, 100)

SELECT @PRECO = [PREÇO DE LISTA] FROM [TABELA DE PRODUTOS]

WHERE [CODIGO DO PRODUTO] = @PRODUTO

INSERT INTO [ITENS NOTAS FISCAIS]

(NUMERO, [CODIGO DO PRODUTO], QUANTIDADE, PREÇO)

VALUES (@NUMERO, @PRODUTO, @QUANTIDADE, @PRECO)

SET @CONTADOR = @CONTADOR + 1

END

Execute os comandos acima até ocorrer um erro de inclusão na tabela de itens. O erro que irá ocorrer é:

Msg 2627, Level 14, State 1, Line 29

Violation of PRIMARY KEY constraint 'PK\_ITENS NOTAS FISCAIS'.

Cannot insert duplicate key in object 'dbo.ITENS NOTAS FISCAIS'.

The duplicate key value is (87985, 1101035).

**6.7. Transformando o script em uma SP.**

Com o script completo, após a solução do problema da PK, transforme-o numa SP chamada **CriaNotaFiscal** onde passamos a data da nota como parâmetro. Execute 3 vezes a nova SP.

### Opinião do instrutor

Um exemplo de script é:

CREATE PROCEDURE CriaNotaFiscal

@DATA AS DATE

AS

BEGIN

DECLARE @CLIENTE VARCHAR(12)

DECLARE @VENDEDOR VARCHAR(12)

DECLARE @PRODUTO VARCHAR(12)

DECLARE @NUMERO INT

DECLARE @IMPOSTO FLOAT

DECLARE @NUM\_ITENS INT

DECLARE @CONTADOR INT

DECLARE @QUANTIDADE INT

DECLARE @PRECO FLOAT

DECLARE @LISTAPRODUTOS TABLE (PRODUTO VARCHAR(20))

DECLARE @AUXPRODUTO INT

SET @CLIENTE = [dbo].[EntidadeAleatoria]('CLIENTE')

SET @VENDEDOR = [dbo].[EntidadeAleatoria]('VENDEDOR')

SELECT @NUMERO = MAX(NUMERO) + 1 FROM [NOTAS FISCAIS]

SET @IMPOSTO = 0.18

SET @CONTADOR = 1

SET @NUM\_ITENS = [dbo].[NumeroAleatorio](2, 10)

INSERT INTO [NOTAS FISCAIS] (CPF, MATRICULA, DATA, NUMERO, IMPOSTO)

VALUES (@CLIENTE, @VENDEDOR, @DATA, @NUMERO, @IMPOSTO)

WHILE @CONTADOR <= @NUM\_ITENS

BEGIN

SET @PRODUTO = [dbo].[EntidadeAleatoria]('PRODUTO')

SELECT @AUXPRODUTO = COUNT(\*) FROM @LISTAPRODUTOS

WHERE PRODUTO = @PRODUTO

IF @AUXPRODUTO = 0

BEGIN

SET @QUANTIDADE = [dbo].[NumeroAleatorio](5, 100)

SELECT @PRECO = [PREÇO DE LISTA] FROM [TABELA DE PRODUTOS]

WHERE [CODIGO DO PRODUTO] = @PRODUTO

INSERT INTO [ITENS NOTAS FISCAIS]

(NUMERO, [CODIGO DO PRODUTO], QUANTIDADE, PREÇO)

VALUES (@NUMERO, @PRODUTO, @QUANTIDADE, @PRECO)

SET @CONTADOR = @CONTADOR + 1

END

INSERT INTO @LISTAPRODUTOS (PRODUTO) VALUES (@PRODUTO)

END

END;

EXEC [dbo].[CriaNotaFiscal] '20180521'

EXEC [dbo].[CriaNotaFiscal] '20180521'

EXEC [dbo].[CriaNotaFiscal] '20180521'