



BANCO DE DADOS

UDF

*FUNÇÕES DEFINIDAS
PELO USUÁRIO*

**Prof. Fabiano Papaiz
IFRN**

UDF – FUNÇÕES DEFINIDAS PELO USUÁRIO

- Além das funções já disponibilizadas pelo SQL Server, como LTRIM, SUBSTRING ou GETDATE, este SGBD também permite que nós criemos nossas próprias funções
- Estas funções são conhecidas como **UDF**, sigla do inglês *User Defined Functions* (Funções Definidas pelo Usuário)
- No SQL Server, as UDFs podem ou não receber parâmetros de entrada e devem obrigatoriamente retornar algum valor, o qual pode ser de 3 tipos (descritos no próximo *slide*)

UDF – FUNÇÕES DEFINIDAS PELO USUÁRIO

○ Retorno *Scalar*:

- Retorna um único valor, como um inteiro, uma *string*, uma data etc

○ Retorno *Inline Table-Valued*:

- Retorna um conjunto de linhas e colunas, gerado a partir de um único comando *SELECT* em tabelas ou *views* do BD
- Similar a utilização de *views*, mas com a diferença de que as funções podem receber parâmetros de entrada

○ Retorno *Multi-statement Table-valued*:

- Define explicitamente a estrutura da tabela que será retornada, com seus nomes de colunas e seus tipos de dados
- Utilizada quando não podemos gerar o retorno a partir de um único *SELECT*, sendo necessário executar 2 ou mais *SELECTs* em sequência para gerar os dados que serão retornados

UDF – FUNÇÕES DEFINIDAS PELO USUÁRIO

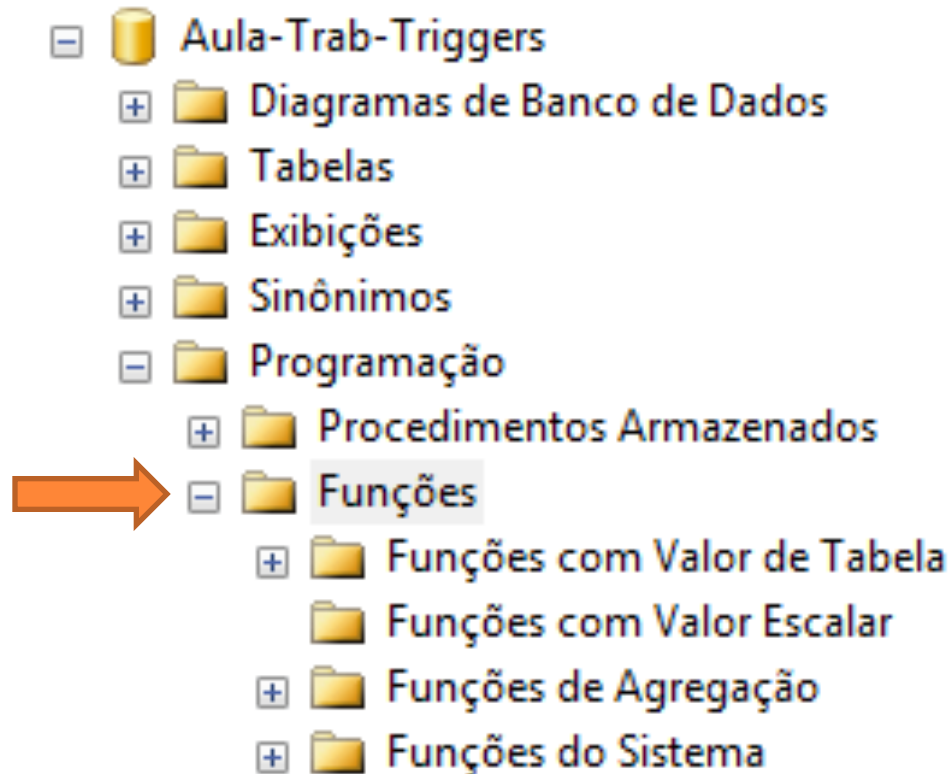
- Sintaxe de criação de uma UDF

```
create function <nome_da_funcao> (<parametros_de_entrada>)  
returns <tipo_de_retorno>  
as  
begin  
    <codigo_da_funcao>  
end
```


Opcional

UDF – FUNÇÕES DEFINIDAS PELO USUÁRIO

- Após uma UDF ser criada, ela ficará armazenada dentro da seguinte pasta do BD





EXEMPLOS

UDF – FUNÇÕES DEFINIDAS PELO USUÁRIO

- **EXEMPLO-1:** função *Scalar* para retornar o nome da UF a partir da sua sigla:

```
create function obterNomeDaUF ( @sigla varchar(2) )
returns varchar(100)
as
begin
    declare @nome varchar(100)
    select @nome =
        case @sigla
            when 'AC' then 'Acre'
            when 'SP' then 'Sao Paulo'
            when 'RN' then 'Rio Grande do Norte'
            --OBS: codigo para outras UF foi omitido
            else NULL
        end
    return @nome
end
```

GO



Procure sempre colocar **GO** após o código da função

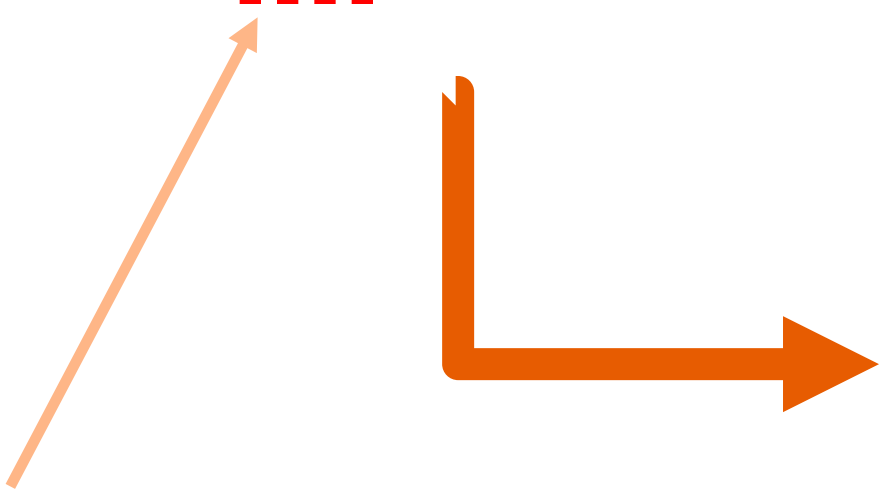
UDF – FUNÇÕES DEFINIDAS PELO USUÁRIO

- Testando a função criada:

```
select dbo.obterNomeDaUF('RN')
```

```
select dbo.obterNomeDaUF('SP')
```

```
select dbo.obterNomeDaUF('AC')
```



Resultados		Mensagens
	(Nenhum nome de coluna)	
1	Rio Grande do Norte	
	(Nenhum nome de coluna)	
1	Sao Paulo	
	(Nenhum nome de coluna)	
1	Acre	

Devemos sempre colocar **dbo.**
antes do nome de uma função

UDF – FUNÇÕES DEFINIDAS PELO USUÁRIO

- A função criada pode agora ser utilizada em:
- **SELECTs**
 - Definição dos valores das colunas (*visto no exemplo*)
 - Na cláusula *WHERE* ou *HAVING* para filtrar registros
 - `WHERE dbo.obterNomeDaUF(uf) LIKE '%Grande%'`
- **TRIGGERS**
- **STORED PROCEDURES** (*será visto mais adiante na disciplina*)
- Em outras **UDFs**

UDF – FUNÇÕES DEFINIDAS PELO USUÁRIO

- **EXEMPLO-2:** função *Scalar* para validar se uma UF é válida:

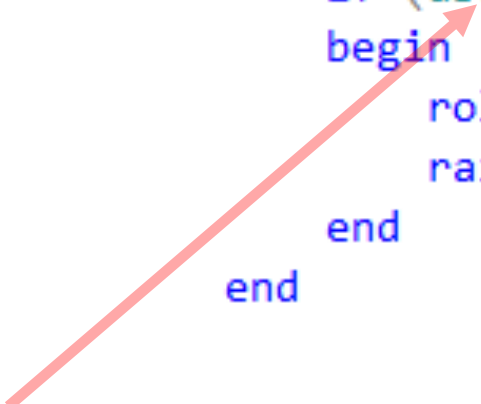
```
create function validarUF( @uf varchar(2) )
returns bit --BIT aceita somente 0 (false) ou 1 (true)
as
begin
    --declara variavel de retorno com valor 0 (false)
    declare @estah_OK bit = 0
    --remove todos os espacos em branco da UF
    set @uf = REPLACE(@uf, ' ', '')
    --verifica a uf
    if (len(@uf) = 2)
    and(@uf IN ('AC', 'AL', 'AP', 'AM', 'BA', 'CE', 'DF', 'ES', 'GO'
                , 'MA', 'MT', 'MS', 'MG', 'PA', 'PB', 'PR', 'PE', 'PI'
                , 'RJ', 'RN', 'RS', 'RO', 'RR', 'SC', 'SP', 'SE', 'TO'))
    begin
        set @estah_OK = 1
    end
    --retorna 1 se a UF for valida ou 0 se for invalida
    return @estah_OK
end
GO
```

UDF – FUNÇÕES DEFINIDAS PELO USUÁRIO

- Utilizando a função criada em uma *trigger*:

```
create trigger validar_uf on Veiculo
for insert, update
as
begin
    --guarda a uf informada
    declare @uf varchar(2)
    select  @uf = uf from inserted

    --verifica a uf
    if (dbo.validarUF(@uf) = 0)
    begin
        rollback transaction
        raiserror ('UF invalida', 0, 0)
    end
end
```



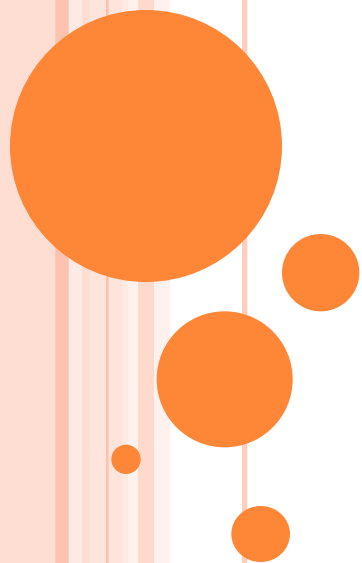
Lembre-se: devemos sempre colocar **dbo.** antes do nome de uma função

UDF – FUNÇÕES DEFINIDAS PELO USUÁRIO

- **Limitações das UDFs:**

- As UDFs possuem algumas limitações que devemos ficar atentos na hora de criá-las, sendo elas:
 1. Não podemos modificar os registros de um banco de dados (*insert*, *update* ou *delete*) através de uma UDF
 2. Uma UDF não dá suporte para a utilização de estruturas de controle de erros (*Try-Catch*) ou para a função *raiserror*
 3. Diferentemente das *Stored Procedures*, as UDFs podem retornar apenas um único conjunto de dados (*result set*)

EXERCÍCIOS



UDF – FUNÇÕES DEFINIDAS PELO USUÁRIO

- **Exercício-1:**
- Crie funções do tipo *Scalar* para realizar as seguintes validações que foram utilizadas nos exercícios sobre *trigger*:
 - Placa de um veículo
 - CPF de um cliente
- Depois altere as *triggers* dos exercícios anteriores para que passem a utilizar estas 2 funções que foram criadas

UDF – FUNÇÕES DEFINIDAS PELO USUÁRIO

- **Exercício-2:**
- ***DESAFIO !!!***
- Altere a função criada para validar um CPF e adicione nela o código necessário para verificar se o número de CPF informado se refere a um CPF válido (que realmente exista)
- Para resolver este exercício, utilize a explicação do *algoritmo gerador do CPF* disponibilizado na área do professor



FIM