



- **Aula 010 – Linguagem SQL**
 - Alteração da estrutura de tabelas.
 - Atualização e exclusão de registros.
 - Tabelas temporárias locais e globais.
 - Cópia de partes de tabelas.
 - Exclusão de tabelas.



- Alterando a estrutura com **ALTER TABLE**
 - A estrutura de uma tabela pode ser modificada conforme haja alguma necessidade.
 - Todas as alterações são realizadas através do comando **ALTER TABLE**, seguido pela ação desejada: **ADD**, **MODIFY** ou **DROP**.



- Alterando a estrutura com **ALTER TABLE**
 - A sintaxe básica para adicionar ou modificar uma **coluna** é dada por:

ALTER TABLE Nome da Tabela

➡ {**ADD** | **MODIFY**} (Nome da Coluna **Tipo de Dados**
[{**ADD** | **MODIFY**} Nome da Coluna Tipo de Dados])

ADD – adiciona um novo campo.

MODIFY – permite alterar um campo já existente.

ALTER COLUMN – utilizado pelo SQL SERVER.



- Alterando a estrutura com **ALTER TABLE**
 - A sintaxe básica para remover uma **coluna** é dada por:

ALTER TABLE Nome da Tabela

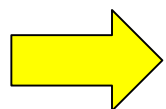
➡ **DROP** Nome da Coluna

DROP COLUMN – utilizado pelo SQL SERVER.



- Alterando a estrutura com **ALTER TABLE**
 - A sintaxe básica para adicionar uma **restrição não nomeada** é dada por:

ALTER TABLE Nome da Tabela



ADD Restrição

[**ADD** Restrição]



- Alterando a estrutura com **ALTER TABLE**
 - A sintaxe básica para adicionar uma **restrição nomeada** é dada por:

ALTER TABLE Nome da Tabela

 **ADD CONSTRAINT**

Nome da Restrição

Tipo da Restrição



- Alterando a estrutura com **ALTER TABLE**
 - A sintaxe básica para remover uma **restrição** é dada por:

ALTER TABLE Nome da Tabela

➡ **DROP CONSTRAINT**

Nome da Restrição ←



■ Alterando a estrutura com **ALTER TABLE**



Para não danificar nosso projeto, estaremos testando os comandos a seguir, utilizando uma tabela de testes.



- Alterando a estrutura com **ALTER TABLE**

-- Criação de uma tabela para testes

```
CREATE TABLE TESTE (  
    ID      INT NOT NULL,  
    Nome    CHAR(10)  
)  
GO
```



- Alterando a estrutura com **ALTER TABLE**

-- Verificando a estrutura da tabela

EXEC **sp_columns** **TESTE**

GO

 Utilizado pelo SQL SERVER

	TABLE_QUALIFI	TABLE_OWN	TABLE_NA	COLUMN_NA	DATA_TY	TYPE_NA	PRECISION
1	Aula09	dbo	TESTE	ID	4	int	10
2	Aula09	dbo	TESTE	Nome	1	char	10

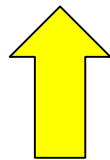


- Alterando a estrutura com **ALTER TABLE**
 - Altera o tipo de dados da
 - coluna Nome. Neste caso,
 - aumenta o tamanho do campo.

ALTER TABLE TESTE

ALTER COLUMN Nome **CHAR**(50)

GO



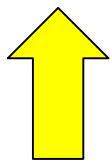
- Alterando a estrutura com **ALTER TABLE**

-- Adiciona a coluna Sexo

ALTER TABLE TESTE

ADD Sexo **CHAR**(1) NULL

GO



- Alterando a estrutura com **ALTER TABLE**

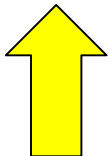
- Adiciona as colunas
- Dt_Nascimento e Peso.

ALTER TABLE TESTE

ADD Dt_Nascimento **DATE**,

Peso **DECIMAL**(5, 2)

GO



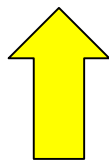
- Alterando a estrutura com **ALTER TABLE**

-- Remove a coluna Dt_Nascimento

ALTER TABLE TESTE

DROP COLUMN Dt_Nascimento

GO

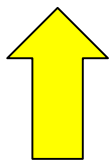


- Alterando a estrutura com **ALTER TABLE**
 - Adiciona uma restrição do
 - tipo única, na coluna Sexo.

ALTER TABLE TESTE

ADD UNIQUE (Sexo)

GO



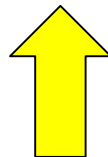
- Alterando a estrutura com **ALTER TABLE**
 - Adiciona uma restrição nomeada,
 - do tipo chave primária na
 - coluna ID.

ALTER TABLE TESTE

ADD CONSTRAINT pk_id **PRIMARY**

KEY (ID)

GO



■ Alterando a estrutura com **ALTER TABLE**

- Exibe informações sobre as restrições
- que existem na tabela.

EXEC sp_helpconstraint TESTE

GO  Utilizado pelo SQL SERVER

	Object Na						
1	TESTE						

	constraint_type	constraint_name	delete_acti	update_act	status_enab	status_for_replica...	constraint_k
1	PRIMARY KEY (clustered)	pk_id	(n/a)	(n/a)	(n/a)	(n/a)	ID
2	UNIQUE (non-clustered)	UQ__TESTE__A22D230F8687C464	(n/a)	(n/a)	(n/a)	(n/a)	Sexo



- Alterando a estrutura com **ALTER TABLE**

-- Removendo uma restrição nomeada.

```
ALTER TABLE TESTE
```

```
    DROP CONSTRAINT pk_id
```

```
GO
```



Nome da restrição



- Alterando a estrutura com **ALTER TABLE**

- Outra forma de se exibir informações
- sobre as colunas da tabela.

SELECT *

FROM INFORMATION_SCHEMA.COLUMNS

WHERE TABLE_NAME = 'TESTE'

GO



■ Alterando a estrutura com **ALTER TABLE**

-- Outra forma de se exibir informações específicas da tabela

```
SELECT TABLE_CATALOG      AS 'Banco de Dados',  
       TABLE_NAME         AS 'Tabela',  
       ORDINAL_POSITION    AS 'Posição',  
       COLUMN_NAME          AS 'Coluna',  
       DATA_TYPE           AS 'Tipo de Dados',  
       COLLATION_NAME       AS 'Idioma da Coluna',  
       IS_NULLABLE          AS 'Aceita Nulo?'
```

```
FROM INFORMATION_SCHEMA.COLUMNS
```

```
WHERE TABLE_NAME = 'TESTE'
```

```
GO
```



■ Alterando a estrutura com **ALTER TABLE**

	Banco de Dados	Tabela	Posição	Coluna	Tipo de Dados	Idioma da Coluna	Aceita Nulo?
1	Aula09	TESTE	1	ID	int	NULL	NO
2	Aula09	TESTE	2	Nome	char	Latin1_General_CI_AS	YES
3	Aula09	TESTE	3	Sexo	char	Latin1_General_CI_AS	YES
4	Aula09	TESTE	4	Peso	decimal	NULL	YES



Essas informações podem ser muito úteis para os administradores de Bancos de Dados.



- Alterando a estrutura com **ALTER TABLE**

-- Utilizando o comando SP_HELP

```
EXEC sp_help 'TESTE'
```

```
GO
```

- Tecla de atalho...
- Selecione o nome da tabela e tecle **ALT + F1**.



■ Alterando a estrutura com **ALTER TABLE**

	Name	Own	Type	Created_datetime						
1	TESTE	dbo	user table	2018-08-12 15:06:39.830						

	Column_na	Type	Comput	Len	Pr	Sc	Nulla	TrimTrailingBla	FixedLenNullInSou	Collation
1	ID	int	no	4	10	0	no	(n/a)	(n/a)	NULL
2	Nome	char	no	50			yes	no	yes	Latin1_General_CI_AS
3	Sexo	char	no	1			yes	no	yes	Latin1_General_CI_AS
4	Peso	decimal	no	5	5	2	yes	(n/a)	(n/a)	NULL

	Identity	Seed	Increm	Not For Replicat
1	No identity column defined.	NULL	NULL	NULL

	RowGuidCol
1	No rowguidcol column defined.

	Data_located_on_filegr
1	PRIMARY

	index_name	index_description	index_ke
1	UQ__TESTE__A22D230F8687C464	nonclustered, unique, unique key located on PRIM	Sexo

	constraint_type	constraint_name	delete_acti	update_act	status_enab	status_for_replica...	constraint_k
1	UNIQUE (non-clustered)	UQ__TESTE__A22D230F8687C464	(n/a)	(n/a)	(n/a)	(n/a)	Sexo



- Excluindo uma tabela com **DROP TABLE**

-- Exclui a tabela TESTE

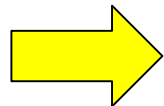
DROP TABLE TESTE

GO

-- Tenta selecionar os dados da tabela

SELECT * FROM TESTE

GO



Invalid object name 'TESTE'.



- Alterando a estrutura com **ALTER TABLE**



Agora, iremos aplicar os conhecimentos adquiridos, nas tabelas dentro de nosso projeto de estudos.



■ Alterando a estrutura com **ALTER TABLE**

- Adiciona o campo Telefone, na
- tabela FUNCIONARIOS

ALTER TABLE FUNCIONARIOS

ADD Telefone **CHAR**(10) 

GO

- Exibe todos os dados da tabela

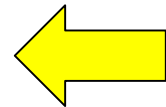
SELECT * **FROM** FUNCIONARIOS

GO



■ Alterando a estrutura com **ALTER TABLE**

ID	Nome	Sexo	Admissao	Salario	Telefone
---	-----	---	-----	-----	-----
1	Maria da Silva	F	2018-01-10	2500.00	NULL
2	Pedro Pereira	M	2015-03-25	990.00	NULL
3	Maria Cristina	F	2015-09-10	1200.00	NULL
4	Antônio Carlos	M	2015-05-15	990.00	NULL
5	Marcelo Augusto	M	2017-12-09	1900.00	NULL
6	Pedro Silva	M	2015-11-15	1050.00	NULL
7	Mônica da Silva	F	2014-10-12	3000.00	NULL
8	Tiago Lima	M	2016-05-10	1350.50	NULL
9	Maria Cristina	F	2012-09-21	1700.00	NULL
10	Maria Cristina	F	2017-10-10	1400.00	NULL



(10 row(s) affected)

SELECT * **FROM** FUNCIONARIOS

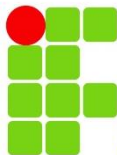


■ Alterando a estrutura com **ALTER TABLE**

-- Exibe informações da estrutura da tabela FUNCIONARIOS

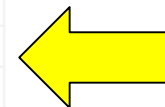
```
SELECT TABLE_CATALOG      AS 'Banco de Dados',  
       TABLE_NAME         AS 'Tabela',  
       ORDINAL_POSITION    AS 'Posição',  
       COLUMN_NAME         AS 'Coluna',  
       DATA_TYPE          AS 'Tipo de Dados',  
       COLLATION_NAME      AS 'Idioma da Coluna',  
       IS_NULLABLE        AS 'Aceita Nulo?'  
  
FROM INFORMATION_SCHEMA.COLUMNS  
  
WHERE TABLE_NAME = 'FUNCIONARIOS'  
  
GO
```





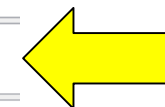
■ Alterando a estrutura com **ALTER TABLE**

	Banco de Dados	Tabela	Posição	Coluna	Tipo de Dados	Idioma da Coluna	Aceita Nulo?
1	Aula09	FUNCIONARIOS	1	ID	int	NULL	NO
2	Aula09	FUNCIONARIOS	2	Nome	varchar	Latin1_General_CI_AS	YES
3	Aula09	FUNCIONARIOS	3	Sexo	char	Latin1_General_CI_AS	YES
4	Aula09	FUNCIONARIOS	4	Admissao	date	NULL	YES
5	Aula09	FUNCIONARIOS	5	Salario	decimal	NULL	YES
6	Aula09	FUNCIONARIOS	6	Telefone	char	Latin1_General_CI_AS	YES



	Name	Own	Type	Created_datetime
1	FUNCIONARIOS	dbo	user table	2018-07-22 20:53:57.177

	Column_na	Type	Comput	Len	Pr	Sc	Nulla	TrimTrailingBla	FixedLenNullInSou	Collation
1	ID	int	no	4	10	0	no	(n/a)	(n/a)	NULL
2	Nome	varchar	no	25			yes	no	yes	Latin1_General_CI_AS
3	Sexo	char	no	1			yes	no	yes	Latin1_General_CI_AS
4	Admissao	date	no	3	10	0	yes	(n/a)	(n/a)	NULL
5	Salario	decimal	no	9	10	2	yes	(n/a)	(n/a)	NULL
6	Telefone	char	no	10			yes	no	yes	Latin1_General_CI_AS



	Identity	Seed	Increm	Not For Replicat
1	No identity column defined.	NULL	NULL	NULL

	RowGuidCol
1	No rowguidcol column defined.

	Data_located_on_filegr
1	PRIMARY

SP_HELP

	index_name	index_description	index_ke
1	PK__FUNCIONA__3214EC273F1C60BC	clustered, unique, primary key located on PRIMARY	ID

	constraint_type	constraint_name	delete_acti	update_act	status_enab	status_for_replica...	constraint_k
1	PRIMARY KEY (clustered)	PK__FUNCIONA__3214EC273F1C60BC	(n/a)	(n/a)	(n/a)	(n/a)	ID



- **Atualização de dados com UPDATE**
 - Para atualizar os dados existentes em uma tabela utilizamos o comando **UPDATE**. Sua sintaxe básica é a seguinte:

UPDATE Nome da Tabela

SET Campo = **Novo Valor**

[WHERE Lista de Condições]



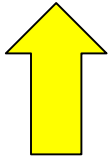
■ Atualização de dados com **UPDATE**

- Atualiza o telefone da funcionária
- 'Maria da Silva', cujo código é 1.

UPDATE FUNCIONARIOS

SET Telefone = '3668-0010'

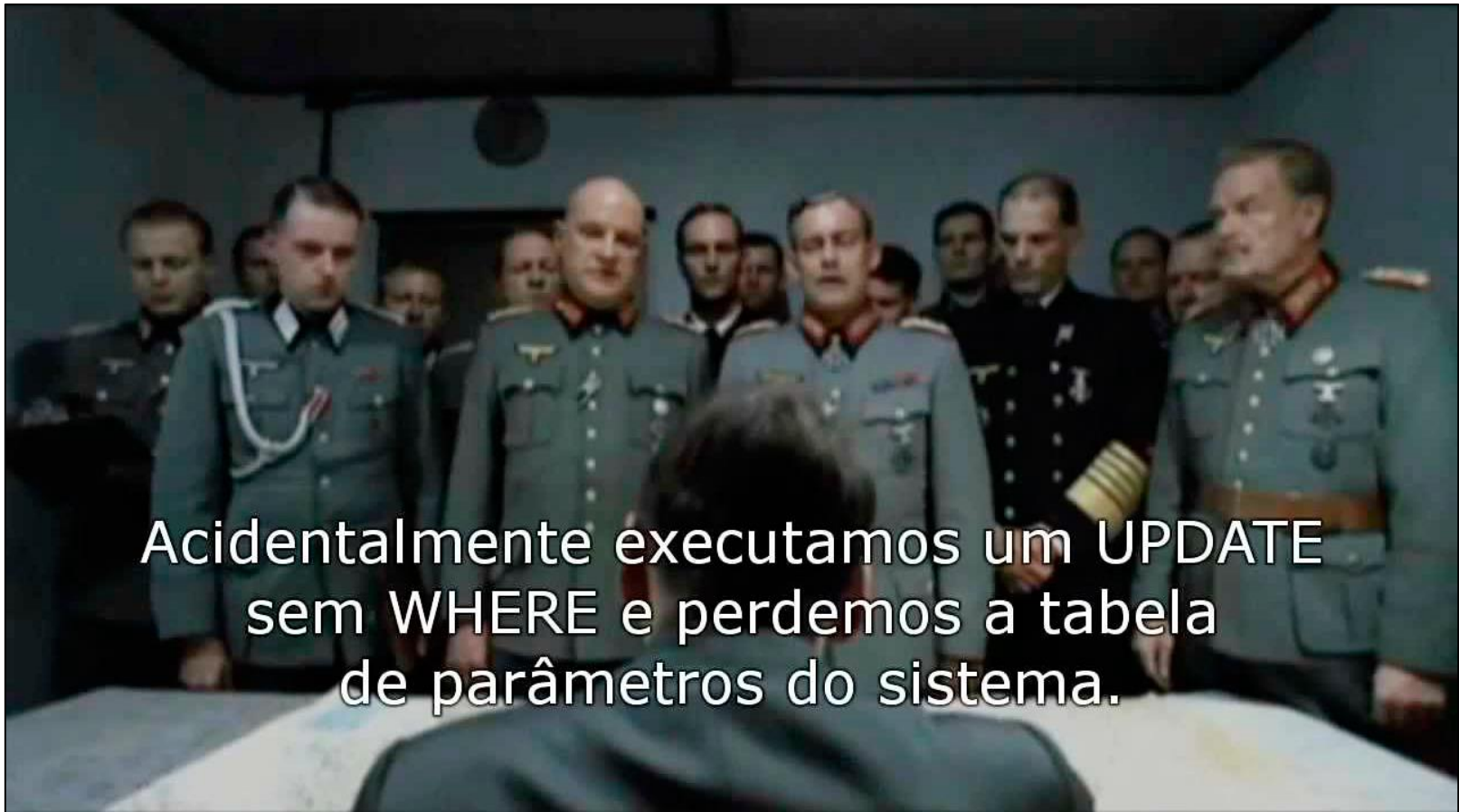
GO



Não está errado, mas...

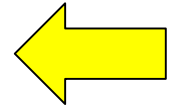


- Atualização de dados com **UPDATE**



■ Atualização de dados com **UPDATE**

ID	Nome	Sexo	Admissao	Salario	Telefone
---	-----	----	-----	-----	-----
1	Maria da Silva	F	2018-01-10	2500.00	3668-0010
2	Pedro Pereira	M	2015-03-25	990.00	3668-0010
3	Maria Cristina	F	2015-09-10	1200.00	3668-0010
4	Antônio Carlos	M	2015-05-15	990.00	3668-0010
5	Marcelo Augusto	M	2017-12-09	1900.00	3668-0010
6	Pedro Silva	M	2015-11-15	1050.00	3668-0010
7	Mônica da Silva	F	2014-10-12	3000.00	3668-0010
8	Tiago Lima	M	2016-05-10	1350.50	3668-0010
9	Maria Cristina	F	2012-09-21	1700.00	3668-0010
10	Maria Cristina	F	2017-10-10	1400.00	3668-0010



(10 row(s) affected)

SELECT * **FROM** FUNCIONARIOS



■ Atualização de dados com **UPDATE**

- Atualiza o telefone da funcionária
- 'Maria da Silva', cujo código é 1.
- Corrige os valores de volta para
- NULL.

UPDATE FUNCIONARIOS

SET Telefone = NULL

GO



■ Atualização de dados com **UPDATE**

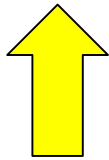
- Atualiza o telefone da funcionária
- 'Maria da Silva', cujo código é 1.

UPDATE FUNCIONARIOS

SET Telefone = '3668-0010'

WHERE ID = 1

GO

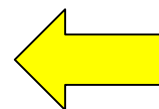


Atualiza registros específicos



■ Atualização de dados com **UPDATE**

ID	Nome	Sexo	Admissao	Salario	Telefone
---	-----	----	-----	-----	-----
1	Maria da Silva	F	2018-01-10	2500.00	3668-0010
2	Pedro Pereira	M	2015-03-25	990.00	NULL
3	Maria Cristina	F	2015-09-10	1200.00	NULL
4	Antônio Carlos	M	2015-05-15	990.00	NULL
5	Marcelo Augusto	M	2017-12-09	1900.00	NULL
6	Pedro Silva	M	2015-11-15	1050.00	NULL
7	Mônica da Silva	F	2014-10-12	3000.00	NULL
8	Tiago Lima	M	2016-05-10	1350.50	NULL
9	Maria Cristina	F	2012-09-21	1700.00	NULL
10	Maria Cristina	F	2017-10-10	1400.00	NULL



(10 row(s) affected)

SELECT * **FROM** FUNCIONARIOS



■ Atualização de dados com **UPDATE**

-- Atualiza o telefone de alguns funcionários

```
UPDATE FUNCIONARIOS SET Telefone = '3668-1550' WHERE ID = 2
```

```
UPDATE FUNCIONARIOS SET Telefone = '3664-5000' WHERE ID = 4
```

```
UPDATE FUNCIONARIOS SET Telefone = '3664-2001' WHERE ID = 5
```

```
UPDATE FUNCIONARIOS SET Telefone = '3663-9000' WHERE ID = 7
```

```
UPDATE FUNCIONARIOS SET Telefone = '3663-7000' WHERE ID = 9
```

```
UPDATE FUNCIONARIOS SET Telefone = '3662-1515' WHERE ID = 10
```

```
GO
```

-- Exibe todos os dados da tabela FUNCIONARIOS

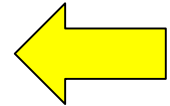
```
SELECT * FROM FUNCIONARIOS
```

```
GO
```



■ Atualização de dados com **UPDATE**

ID	Nome	Sexo	Admissao	Salario	Telefone
---	-----	----	-----	-----	-----
1	Maria da Silva	F	2018-01-10	2500.00	3668-0010
2	Pedro Pereira	M	2015-03-25	990.00	3668-1550
3	Maria Cristina	F	2015-09-10	1200.00	NULL
4	Antônio Carlos	M	2015-05-15	990.00	3664-5000
5	Marcelo Augusto	M	2017-12-09	1900.00	3664-2001
6	Pedro Silva	M	2015-11-15	1050.00	NULL
7	Mônica da Silva	F	2014-10-12	3000.00	3663-9000
8	Tiago Lima	M	2016-05-10	1350.50	NULL
9	Maria Cristina	F	2012-09-21	1700.00	3663-7000
10	Maria Cristina	F	2017-10-10	1400.00	3662-1515



(10 row(s) affected)



■ Atualização de dados com **UPDATE**

- Concede um aumento de 10% para os
- funcionários que foram admitidos
- antes de 2016.

UPDATE FUNCIONARIOS

SET Salario = Salario * 1.10

WHERE **YEAR**(Admissao) < 2016

GO



■ Atualização de dados com **UPDATE**

ID	Nome	Sexo	Admissao	Salario	Telefone
---	-----	----	-----	-----	-----
1	Maria da Silva	F	2018-01-10	2500.00	3668-0010
2	Pedro Pereira	M	2015-03-25	1089.00	3668-1550
3	Maria Cristina	F	2015-09-10	1320.00	NULL
4	Antônio Carlos	M	2015-05-15	1089.00	3664-5000
5	Marcelo Augusto	M	2017-12-09	1900.00	3664-2001
6	Pedro Silva	M	2015-11-15	1155.00	NULL
7	Mônica da Silva	F	2014-10-12	3300.00	3663-9000
8	Tiago Lima	M	2016-05-10	1350.50	NULL
9	Maria Cristina	F	2012-09-21	1870.00	3663-7000
10	Maria Cristina	F	2017-10-10	1400.00	3662-1515

(10 row(s) affected)



■ Tabelas temporárias

- Uma funcionalidade interessante do SQL Server é a possibilidade de criar **tabelas temporárias**.
- Uma tabela temporária existe apenas no escopo da sessão em que foram criadas, sendo que seu conteúdo não é armazenado de forma definitiva no banco de dados.



■ Tabelas temporárias

- A sintaxe de um objeto temporário no SQL Server utiliza o sinal **#**, para um identificar um objeto existente em apenas uma sessão e **##** para identificar um objeto existente entre outras sessões. Esse sinal é inserido na frente do nome do objeto.

CREATE TABLE **#**Nome da Tabela

CREATE TABLE **##**Nome da Tabela



■ Tabelas temporárias (local)

-- Cria a tabela temporária PART_A

```
CREATE TABLE #PART_A (  
    ID_Part      INT NOT NULL UNIQUE,  
    Nome_Part    VARCHAR(25),  
    Tel_Part     CHAR(10),  
    PRIMARY KEY (ID_Part) ←  
)
```

GO

Outra maneira de se criar uma restrição



■ Tabelas temporárias (local)

-- Insere alguns valores na tabela temporária,
-- dentro da mesma sessão

```
INSERT INTO #PART_A VALUES
```

```
(1, 'Marcelo Augusto', '3668-4545'),
```

```
(2, 'Maria Cristina', '3663-0909')
```

```
GO
```

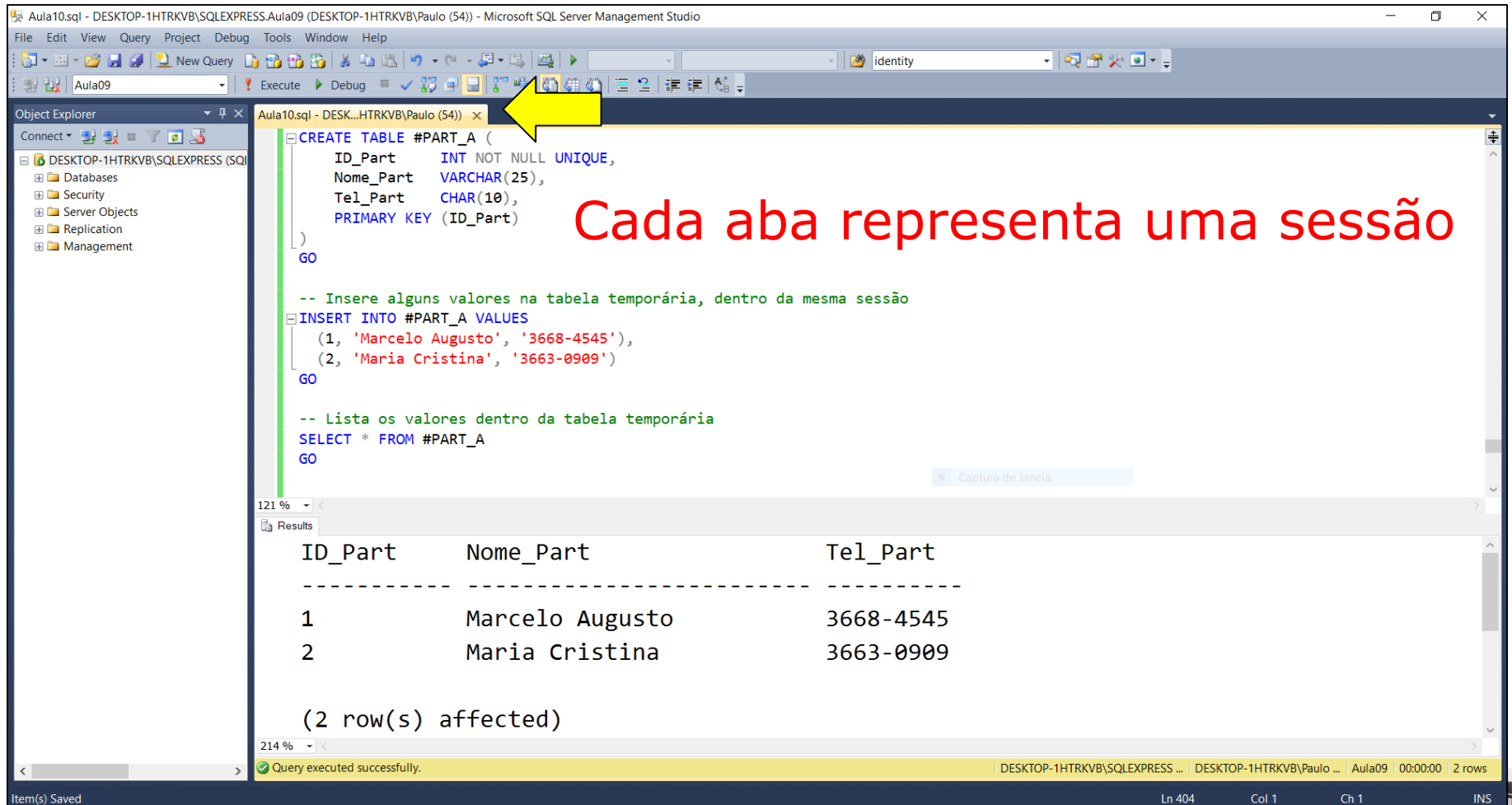
-- Lista os valores dentro da tabela temporária

```
SELECT * FROM #PART_A
```

```
GO
```



■ Tabelas temporárias (local)



A screenshot of the Microsoft SQL Server Management Studio interface. The title bar shows 'Aula10.sql - DESKTOP-1HTRKVB\SQLEXPRESS.Aula09 (DESKTOP-1HTRKVB\Paulo (54)) - Microsoft SQL Server Management Studio'. The Object Explorer on the left shows the server structure. The main query window displays the following SQL code:

```
CREATE TABLE #PART_A (  
    ID_Part    INT NOT NULL UNIQUE,  
    Nome_Part  VARCHAR(25),  
    Tel_Part   CHAR(10),  
    PRIMARY KEY (ID_Part)  
)  
GO  
  
-- Insere alguns valores na tabela temporária, dentro da mesma sessão  
INSERT INTO #PART_A VALUES  
    (1, 'Marcelo Augusto', '3668-4545'),  
    (2, 'Maria Cristina', '3663-0909')  
GO  
  
-- Lista os valores dentro da tabela temporária  
SELECT * FROM #PART_A  
GO
```

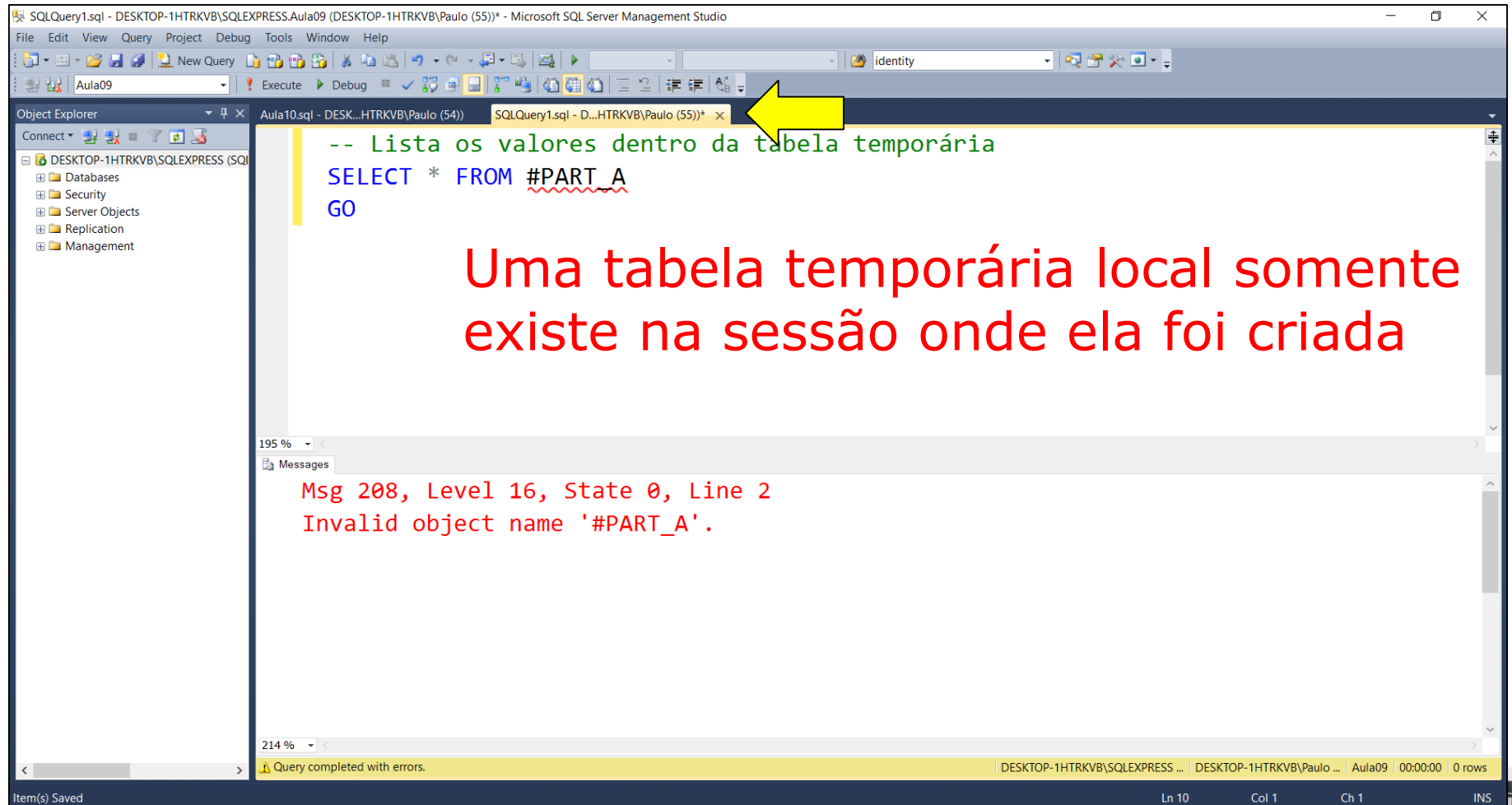
A yellow arrow points to the 'Aula10.sql' tab in the top toolbar. To the right of the code, red text reads: 'Cada aba representa uma sessão'.

The Results pane at the bottom shows the output of the SELECT query:

ID_Part	Nome_Part	Tel_Part
1	Marcelo Augusto	3668-4545
2	Maria Cristina	3663-0909

Below the table, it says '(2 row(s) affected)'. The status bar at the bottom indicates 'Query executed successfully.' and '2 rows'.

■ Tabelas temporárias (local)



SQLQuery1.sql - DESKTOP-1HTRKVB\SQLEXPRESS.Aula09 (DESKTOP-1HTRKVB\Paulo (55))* - Microsoft SQL Server Management Studio

Object Explorer

- DESKTOP-1HTRKVB\SQLEXPRESS (SQL)
- Databases
- Security
- Server Objects
- Replication
- Management

SQLQuery1.sql - D...HTRKVB\Paulo (55))*

```
-- Lista os valores dentro da tabela temporária
SELECT * FROM #PART_A
GO
```

Uma tabela temporária local somente existe na sessão onde ela foi criada

195 %

Messages

Msg 208, Level 16, State 0, Line 2
Invalid object name '#PART_A'.

214 %

Query completed with errors.

DESKTOP-1HTRKVB\SQLEXPRESS ... | DESKTOP-1HTRKVB\Paulo ... | Aula09 | 00:00:00 | 0 rows

Item(s) Saved

Ln 10 Col 1 Ch 1 INS

■ Tabelas temporárias (global)

-- Cria a tabela temporária PART_B

```
CREATE TABLE ##PART_B (  
    ID_Part      INT NOT NULL UNIQUE,  
    Nome_Part    VARCHAR(25),  
    Tel_Part     CHAR(10),  
    PRIMARY KEY (ID_Part)  
)
```

GO



■ Tabelas temporárias (global)

-- Insere alguns valores na tabela temporária,
-- dentro da mesma sessão

```
INSERT INTO ##PART_B VALUES
```

```
(1, 'Marcelo Augusto', '3668-4545'),
```

```
(2, 'Maria Cristina', '3663-0909')
```

```
GO
```

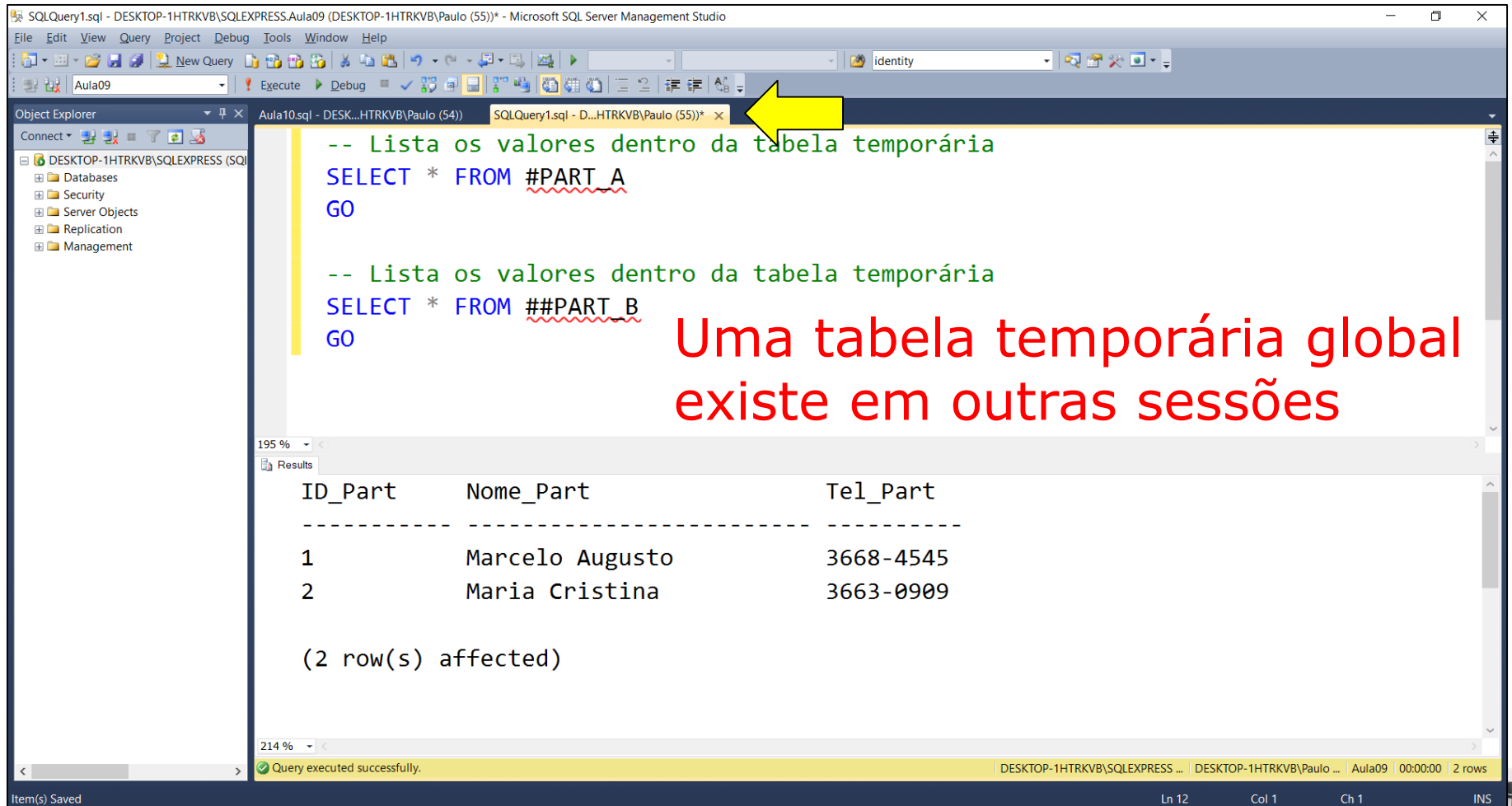
-- Lista os valores dentro da tabela temporária

```
SELECT * FROM ##PART_B
```

```
GO
```



■ Tabelas temporárias (global)



The screenshot shows the Microsoft SQL Server Management Studio interface. The query window displays two SQL queries. The first query uses a local temporary table (#PART_A), and the second query uses a global temporary table (##PART_B). A yellow arrow points to the second query. The results window shows the output of the second query, displaying two rows of data. A red text overlay states: "Uma tabela temporária global existe em outras sessões".

```
-- Lista os valores dentro da tabela temporária
SELECT * FROM #PART_A
GO

-- Lista os valores dentro da tabela temporária
SELECT * FROM ##PART_B
GO
```

ID_Part	Nome_Part	Tel_Part
1	Marcelo Augusto	3668-4545
2	Maria Cristina	3663-0909

(2 row(s) affected)

Query executed successfully.

■ Copiando partes de tabelas

- Existem duas maneiras de se copiar partes dos valores de uma tabela para dentro de outra. No **primeiro caso**, é necessário a existência de duas tabelas com uma estrutura semelhante. Se isso ocorrer, podemos utilizar o seguinte comando para copiar valores de uma tabela para outra:

```
INSERT INTO Nome_da_Tabela_Destino  
SELECT      Lista_de_Colunas  
FROM        Nome_da_Tabela_Fonte
```



■ Copiando partes de tabelas

-- Copia alguns valores da tabela FUNCIONARIOS,
-- para dentro da tabela #PART_A

```
INSERT INTO #PART_A
```

```
    SELECT    ID,  
              Nome,  
              Telefone
```

```
    FROM  FUNCIONARIOS
```

```
    WHERE ID > 2
```

Dados que serão copiados

```
GO
```



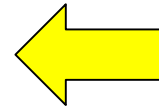
■ Copiando partes de tabelas

ID_Part	Nome_Part	Tel_Part
1	Marcelo Augusto	3668-4545
2	Maria Cristina	3663-0909
3	Maria Cristina	NULL
4	Antônio Carlos	3664-5000
5	Marcelo Augusto	3664-2001
6	Pedro Silva	NULL
7	Mônica da Silva	3663-9000
8	Tiago Lima	NULL
9	Maria Cristina	3663-7000
10	Maria Cristina	3662-1515

(10 row(s) affected)

`SELECT * FROM #PART_A`

Não se preocupe com
a ordem de inserção



- **Copiando partes de tabelas**
 - No **segundo caso**, podemos utilizar um comando para copiar além dos dados, a estrutura de uma determinada tabela. Isso permite a criação de uma nova tabela, com os mesmos campos da tabela fonte, porém, sem suas restrições:

```
SELECT Lista_de_Colunas  
INTO Nova_Tabela  
FROM Nome_da_Tabela_Fonte
```



■ Copiando partes de tabelas

- Cria uma nova tabela, utilizando como
- base os dados e a estrutura da tabela
- FUNCIONARIOS. Será criado a tabela
- FUNCIONARIOS_2

SELECT *

INTO FUNCIONARIOS_2 

FROM FUNCIONARIOS

GO



■ Copiando partes de tabelas

ID	Nome	Sexo	Admissao	Salario	Telefone
1	Maria da Silva	F	2018-01-10	2500.00	3668-0010
2	Pedro Pereira	M	2015-03-25	1089.00	3668-1550
3	Maria Cristina	F	2015-09-10	1320.00	NULL
4	Antônio Carlos	M	2015-05-15	1089.00	3664-5000
5	Marcelo Augusto	M	2017-12-09	1900.00	3664-2001
6	Pedro Silva	M	2015-11-15	1155.00	NULL
7	Mônica da Silva	F	2014-10-12	3300.00	3663-9000
8	Tiago Lima	M	2016-05-10	1350.50	NULL
9	Maria Cristina	F	2012-09-21	1870.00	3663-7000
10	Maria Cristina	F	2017-10-10	1400.00	3662-1515

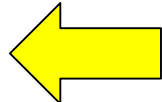
(10 row(s) affected)

SELECT * **FROM** FUNCIONARIOS_2



■ Copiando partes de tabelas

-- Exibe informações da estrutura da tabela FUNCIONARIOS_2

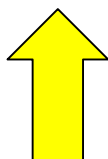
```
SELECT TABLE_CATALOG      AS 'Banco de Dados',  
       TABLE_NAME        AS 'Tabela',  
       ORDINAL_POSITION   AS 'Posição',  
       COLUMN_NAME        AS 'Coluna',  
       DATA_TYPE         AS 'Tipo de Dados',  
       COLLATION_NAME     AS 'Idioma da Coluna',  
       IS_NULLABLE       AS 'Aceita Nulo?'  
  
FROM INFORMATION_SCHEMA.COLUMNS  
  
WHERE TABLE_NAME = 'FUNCIONARIOS_2'   
  
GO
```



■ Copiando partes de tabelas

Banco de Dados	Tabela	Posição	Coluna	Tipo de Dados	Idioma da Coluna	Aceita Nulo?
Aula09	FUNCIONARIOS_2	1	ID	int	NULL	NO
Aula09	FUNCIONARIOS_2	2	Nome	varchar	Latin1_General_CI_AS	YES
Aula09	FUNCIONARIOS_2	3	Sexo	char	Latin1_General_CI_AS	YES
Aula09	FUNCIONARIOS_2	4	Admissao	date	NULL	YES
Aula09	FUNCIONARIOS_2	5	Salario	decimal	NULL	YES
Aula09	FUNCIONARIOS_2	6	Telefone	char	Latin1_General_CI_AS	YES

(6 row(s) affected)



■ Copiando partes de tabelas

-- Exibe informações sobre as restrições das
-- tabelas FUNCIONARIOS e FUNCIONARIOS_2

```
SELECT OBJECT_NAME(object_id)          AS 'Nome da Restrição',  
       SCHEMA_NAME(schema_id)         AS 'Nome do Esquema',  
       OBJECT_NAME(parent_object_id) AS 'Nome da Tabela',  
       type_desc                       AS 'Tipo de Restrição'  
FROM SYS.OBJECTS  
WHERE type_desc LIKE '%CONSTRAINT' AND  
       OBJECT_NAME(parent_object_id) IN ('FUNCIONARIOS',  
     'FUNCIONARIOS_2')  
GO
```



■ Copiando partes de tabelas

Nome da Restrição	Nome do Esquema	Nome da Tabela	Tipo de Restrição
PK__FUNCIONA__3214EC273F1C60BC	dbo	FUNCIONARIOS	PRIMARY_KEY_CONSTRAINT

(1 row(s) affected)



Somente a tabela FUNCIONARIOS
possui restrições



■ Copiando partes de tabelas

- Adiciona uma restrição nomeada do
- tipo chave primária, na coluna 'ID'
- da tabela FUNCIONARIOS_2

```
ALTER TABLE FUNCIONARIOS_2
```

```
    ADD CONSTRAINT pk_id PRIMARY KEY (ID)
```

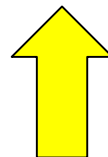
```
GO
```



■ Copiando partes de tabelas

Nome da Restrição	Nome do Esquema	Nome da Tabela	Tipo de Restrição
PK__FUNCIONA__3214EC273F1C60BC	dbo	FUNCIONARIOS	PRIMARY_KEY_CONSTRAINT
pk_id	dbo	FUNCIONARIOS_2	PRIMARY_KEY_CONSTRAINT

(2 row(s) affected)



Agora a restrição aparece na
tabela FUNCIONARIOS_2



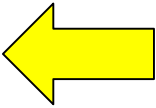
■ Removendo dados de uma tabela

- Para remover os dados de uma tabela temos os comandos **DELETE** e **TRUNCATE**. O comando **DELETE** pode ser utilizado em conjunto com a cláusula **WHERE** para remover apenas linhas que atendam a uma determinada condição. Se não for especificada nenhuma condição, todas as linhas da tabela serão removidas.
- O comando **TRUNCATE** também pode ser utilizado para remover todas as linhas de uma tabela. Ele não aceita a cláusula **WHERE** e não grava informações no arquivo de **LOG** do servidor.



- Removendo dados de uma tabela

- Sintaxe do comando **DELETE**:

DELETE FROM Nome_da_Tabela
[WHERE Condições] 

- Sintaxe do comando **TRUNCATE**:

TRUNCATE TABLE Nome_da_Tabela

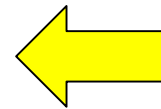


■ Removendo dados de uma tabela

- Utiliza o comando DELETE para remover
- algumas linhas da tabela FUNCIONARIOS_2

```
SELECT * FROM FUNCIONARIOS_2  
GO
```

```
DELETE FROM FUNCIONARIOS_2  
WHERE Salario < 1500  
GO
```



Condição para
remoção das linhas

```
SELECT * FROM FUNCIONARIOS_2  
GO
```



■ Removendo dados de uma tabela

ID	Nome	Sexo	Admissao	Salario	Telefone
---	-----	----	-----	-----	-----
1	Maria da Silva	F	2018-01-10	2500.00	3668-0010
5	Marcelo Augusto	M	2017-12-09	1900.00	3664-2001
7	Mônica da Silva	F	2014-10-12	3300.00	3663-9000
9	Maria Cristina	F	2012-09-21	1870.00	3663-7000

(4 row(s) affected) 

Foram removidos 6 registros



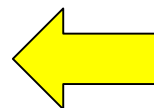
■ Removendo dados de uma tabela

-- Utiliza o comando TRUNCATE para remover
-- todos os registros da tabela FUNCIONARIOS_2

```
SELECT * FROM FUNCIONARIOS_2
```

```
GO
```

```
TRUNCATE TABLE FUNCIONARIOS_2
```



```
GO
```

```
SELECT * FROM FUNCIONARIOS_2
```

```
GO
```



■ Excluindo uma tabela

- Para excluir uma tabela do banco de dados utilizamos o comando **DROP TABLE**.
- Somente é possível excluir tabelas que não possuam restrições de integridade, as quais são implementadas através do conceito de chave estrangeira.

DROP TABLE Nome_da_Tabela



■ Excluindo uma tabela

-- Remove a tabela FUNCIONARIOS_2

DROP TABLE FUNCIONARIOS_2 

GO

-- Exibe uma listagem com o nome das tabelas

-- existentes no banco de dados em uso no momento

SELECT name AS 'Nome da Tabela'

FROM sys.tables

GO



- **Na próxima aula veremos**
 - Funções para agregação e classificação.

