INSTITUTO FEDERAL
Rio Grande do Norte | Campus Parnamirim

Modelo Físico
Linguagem SQL – Parte 05

Prof. Fábio Procópio

Tecnologia em Sistemas para Internet – Banco de Dados

Relembrando...

- Na aula passada, continuamos a estudar a **Linguagem de Consulta a Dados (DQL)** apresentando novos operadores e algumas funções de agregação como:
  1. LIKE
  2. IN e BETWEEN
  3. COUNT()
  4. MIN() e MAX()
  5. SUM() e AVG()

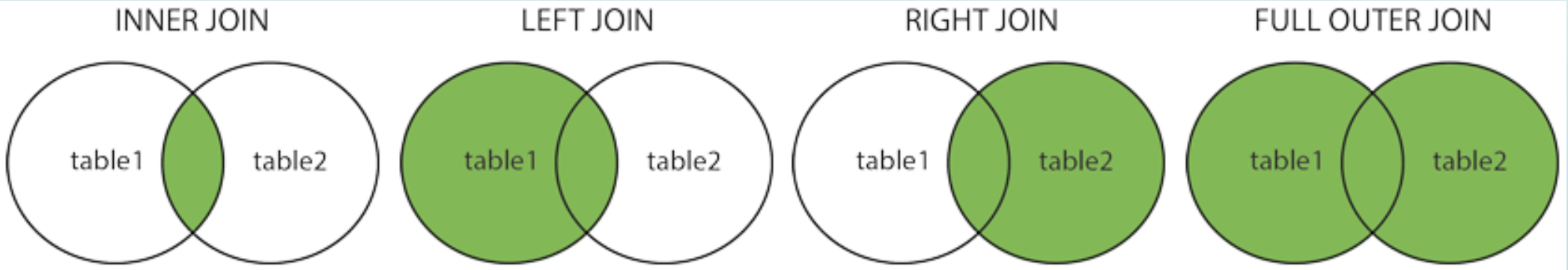
Comando DQL

- DDL – *Data Definition Language*
  - Comandos: CREATE, ALTER e DROP
- DML – *Data Manipulation Language*
  - Comandos: INSERT, UPDATE e DELETE
- **DQL – *Data Query Language***
  - Comando: SELECT
- DCL – *Data Control Language*
  - Comandos: GRANT e REVOKE
- DTL – *Data Transaction Language*
  - Comandos: COMMIT, ROLLBACK e SAVEPOINT

JOIN

- A cláusula JOIN é usada para fazer a junção entre linhas de uma tabela (auto-relacionamento), de duas ou de mais tabelas;

- Tipos de JOINs:
  - (INNER) JOIN
  - LEFT (OUTER) JOIN
  - RIGHT (OUTER) JOIN
  - FULL (OUTER) JOIN



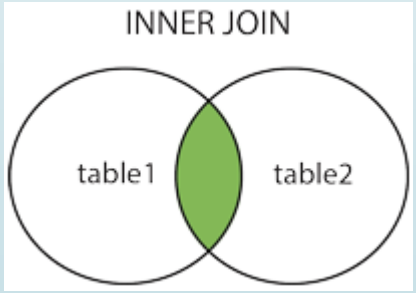
Fonte: Imagem extraída de W3SCHOOLS

INNER JOIN – 1 de 2

- Retorna registros que possuem valores correspondentes nas duas tabelas;

- Sintaxe básica:

```
SELECT <lista_de_campos>
 FROM <nome_tabela1>
 INNER JOIN <nome_tabela2>
    ON <nome_tabela1>.<nome_coluna> = <nome_tabela2>.<nome_coluna>
```

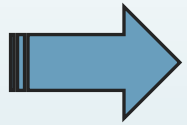


Fonte: Imagem extraída de W3SCHOOLS

INNER JOIN – 2 de 2

- Exemplo:

```
SELECT b.CoBanda, NoBanda AS 'Banda',
       NoIntegrante AS 'Nome do Integrante'
  FROM TbBanda AS b
  INNER JOIN TbIntegrante AS i
     ON b.CoBanda = i.CoBanda
    AND b.CoBanda = 1
```



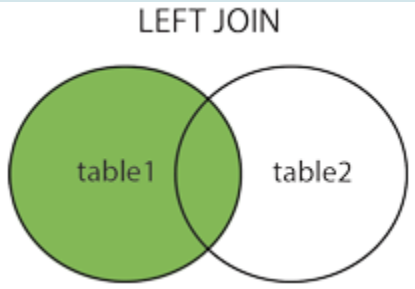
CoBanda	Banda	Nome do Integrante
1	Jota Quest	Rogério Flausino
1	Jota Quest	Marco Túlio Lara
1	Jota Quest	PJ
1	Jota Quest	Paulinho Fonseca
1	Jota Quest	Márcio Buzelin

LEFT (OUTER) JOIN – 1 de 2

- Retorna todos os registros existentes na tabela da esquerda, além dos que também correspondem aos registros a tabela da direita;

- Sintaxe básica:

```
SELECT <lista_de_campos>
 FROM <nome_tabela1>
    LEFT JOIN <nome_tabela2>
    ON <nome_tabela1>.<nome_coluna> = <nome_tabela2>.<nome_coluna>
```



Fonte: Imagem extraída de W3SCHOOLS

LEFT (OUTER) JOIN – 2 de 2

- Exemplo:

```
SELECT b.CoBanda, NoBanda AS 'Banda',
       NoIntegrante AS 'Nome do Integrante'
  FROM TbBanda AS b
    LEFT JOIN TbIntegrante AS i
     ON b.CoBanda = i.CoBanda
```

CoBanda	Banda	Nome do Integrante
1	Jota Quest	Rogério Flausino
1	Jota Quest	Marco Túlio Lara
1	Jota Quest	PJ
1	Jota Quest	Paulinho Fonseca
1	Jota Quest	Márcio Buzelin
2	Skank	Samuel Rosa
2	Skank	Henrique Portugal
2	Skank	Lelo Zaneti
2	Skank	Haroldo Ferretti
3	Paralamas do Sucesso	Herbert Vianna
3	Paralamas do Sucesso	Bi Ribeiro
3	Paralamas do Sucesso	João Barone
4	Capital Inicial	Dinho Ouro Preto
4	Capital Inicial	Fê Lemos
4	Capital Inicial	Flávio Lemos
4	Capital Inicial	Yves Passarel
5	Roupa Nova	Paulinho
5	Roupa Nova	Serginho Herval
5	Roupa Nova	Nando
5	Roupa Nova	Kiko
5	Roupa Nova	Ricardo Feghali:
5	Roupa Nova	Cleberson Horsth
6	Titãs	NULL

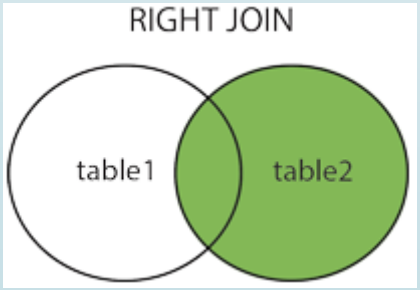
Mesmo sem existir uma correspondência entre TbBanda e TbIntegrante, para CoBanda igual a 6, LEFT JOIN permitiu o retorno de um registro da tabela TbBanda, que é Titãs.

RIGHT (OUTER) JOIN – 1 de 2

- Retorna todos os registros existentes na tabela da direita, além dos que também correspondem aos registros a tabela da esquerda;

- Sintaxe básica:

```
SELECT <lista_de_campos>
 FROM <nome_tabela1>
  RIGHT JOIN <nome_tabela2>
    ON <nome_tabela1>.<nome_coluna> = <nome_tabela2>.<nome_coluna>
```



Fonte: Imagem extraída de W3SCHOOLS

RIGHT (OUTER) JOIN – 2 de 2

- Exemplo:

```
SELECT b.CoBanda, NoBanda AS 'Banda',
       NoIntegrante AS 'Nome do Integrante'
  FROM TbBanda AS b
    RIGHT JOIN TbIntegrante AS i
     ON b.CoBanda = i.CoBanda
```

CoBanda	Banda	Nome do Integrante
1	Jota Quest	Rogério Flausino
1	Jota Quest	Marco Túlio Lara
1	Jota Quest	PJ
1	Jota Quest	Paulinho Fonseca
1	Jota Quest	Márcio Buzelin
2	Skank	Samuel Rosa
2	Skank	Henrique Portugal
2	Skank	Lelo Zaneti
2	Skank	Haroldo Ferretti
3	Paralam...	Herbert Vianna
3	Paralam...	Bi Ribeiro
3	Paralam...	João Barone
4	Capital I...	Dinho Ouro Preto
4	Capital I...	Fê Lemos
4	Capital I...	Flávio Lemos
4	Capital I...	Yves Passarel
5	Roupa N...	Paulinho
5	Roupa N...	Serginho Herval
5	Roupa N...	Nando
5	Roupa N...	Kiko
5	Roupa N...	Ricardo Feghali:
5	Roupa N...	Cleberson Horsth

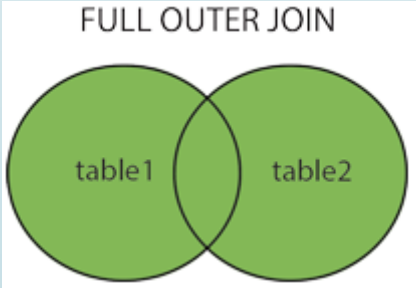
Como não existem registros em TbIntegrante (tabela do lado direito) que não haja uma correspondência em TbBanda (se isso ocorresse haveria uma "quebra" na regra de integridade referencial do modelo de banco de dados), o retorno da consulta funciona como um INNER JOIN.

FULL (OUTER) JOIN – 1 de 2

- Retorna todos os registros quando há uma correspondência na tabela da esquerda ou na da direita;

- Sintaxe básica:

```
SELECT <lista_de_campos>
 FROM <nome_tabela1>
   FULL OUTER JOIN <nome_tabela2>
    ON <nome_tabela1>.<nome_coluna> = <nome_tabela2>.<nome_coluna>
```



Fonte: Imagem extraída de W3SCHOOLS

FULL (OUTER) JOIN – 2 de 2

- Exemplo:

```
SELECT b.CoBanda, NoBanda AS 'Banda',
       NoIntegrante AS 'Nome do Integrante'
  FROM TbIntegrante AS i
    FULL JOIN TbBanda AS b
     ON i.CoBanda = b.CoBanda
```

CoBanda	Banda	Nome do Integrante
1	Jota Quest	Rogério Flausino
1	Jota Quest	Marco Túlio Lara
1	Jota Quest	PJ
1	Jota Quest	Paulinho Fonseca
1	Jota Quest	Márcio Buzelin
2	Skank	Samuel Rosa
2	Skank	Henrique Portugal
2	Skank	Lelo Zaneti
2	Skank	Haroldo Ferretti
3	Paralamas do Sucesso	Herbert Vianna
3	Paralamas do Sucesso	Bi Ribeiro
3	Paralamas do Sucesso	João Barone
4	Capital Inicial	Dinho Ouro Preto
4	Capital Inicial	Fê Lemos
4	Capital Inicial	Flávio Lemos
4	Capital Inicial	Yves Passarel
5	Roupa Nova	Paulinho
5	Roupa Nova	Serginho Herval
5	Roupa Nova	Nando
5	Roupa Nova	Kiko
5	Roupa Nova	Ricardo Feghali:
5	Roupa Nova	Cleberson Horsth
6	Titãs	NULL

UNION – 1 de 2

- O operador UNION é usado para combinar resultados de duas ou mais instruções SELECT
  - cada instrução SELECT dentro de UNION deve ter o mesmo número de colunas
  - seleciona apenas valores distintos. Para recuperar valores repetidos, usa-se **UNION ALL**

- Sintaxe básica::

```
SELECT <lista_de_campos> FROM <nome_tabela1>
UNION [ALL]
SELECT <llista_de_campos> FROM <nome_tabela2>
```

UNION – 2 de 2

- Esta consulta retorna os códigos das bandas (sem repetição) de TbBanda e de TbIntegrante (para ver o resultado, execute a instrução SQL abaixo):

```
SELECT CoBanda FROM TbBanda
UNION
SELECT CoBanda FROM TbIntegrante
```

- Esta consulta retorna os códigos das bandas (com repetição) de TbBanda e de TbIntegrante (para ver o resultado, execute a instrução SQL abaixo):

```
SELECT CoBanda FROM TbBanda
UNION ALL
SELECT CoBanda FROM TbIntegrante
```

GROUP BY – 1 de 2

- Frequentemente é usado com funções de agregação (COUNT, MIN, MAX, AVG, SUM) para agrupar o resultado com base em uma ou mais colunas;

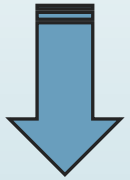
- Sintaxe básica:

```
SELECT <lista de campos>, <função de agregação>
 FROM <nome_tabela>
WHERE <critério(s) de filtragem>
GROUP BY <campos usados no agrupamento>
ORDER BY <lista de campos>
```

GROUP BY– 2 de 2

- Esta consulta retorna a quantidade de integrantes por banda:

```
SELECT CoBanda, COUNT(NoIntegrante) AS QtIntegrantes
  FROM TbIntegrante
GROUP BY CoBanda
ORDER BY CoBanda
```



CoBanda	QtIntegrantes
1	5
2	4
3	3
4	4
5	6

HAVING – 1 de 2

- A cláusula HAVING foi adicionada ao SQL porque WHERE não pode ser usado combinada a funções de agregação;

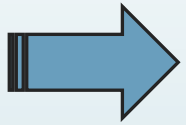
- Sintaxe básica:

```
SELECT <lista_de_campos>, <função de agregação>
 FROM <nome_tabela>
 WHERE <critério(s) de filtragem>
GROUP BY <campos usados no agrupamento>
HAVING <condição>
ORDER BY <lista de campos>
```

HAVING – 2 de 2

- Esta consulta retorna o nome das bandas que possuem mais de 4 integrantes:

```
SELECT NoBanda, COUNT(CoIntegrante) AS QtIntegrantes
 FROM TbBanda AS b
 INNER JOIN TbIntegrante AS i
    ON b.CoBanda = i.CoBanda
GROUP BY NoBanda
HAVING COUNT(CoIntegrante) > 4
ORDER BY NoBanda
```



NoBanda	QtIntegrantes
Jota Quest	5
Roupa Nova	6

EXISTS – 1 de 2

- O operador EXISTS é usado para testar a existência de qualquer registro em uma subconsulta
  - retorna true se a subconsulta retornar pelo menos um registro

- Sintaxe básica:

```
SELECT <lista_de_campos>
 FROM <nome_tabela>
WHERE EXISTS (SELECT nome_coluna
                FROM <nome_tabela>
               WHERE <critério_de_filtragem>)
```

EXISTS – 2 de 2

- Esta consulta retornará as bandas que existem em TbBanda, mas que não há registros dela em TbIntegrante:

```
SELECT NoBanda
 FROM TbBanda AS b
WHERE NOT EXISTS (SELECT *
                    FROM TbIntegrante AS i
                   WHERE b.CoBanda = i.CoBanda)
```

NoBanda
Titãs

Comandos DCL – 1 de 2

- DDL – Data Definition Language
  - Comandos: CREATE, ALTER e DROP
- DML – *Data Manipulation Language*
  - Comandos: INSERT, UPDATE e DELETE
- DQL – *Data Query Language*
  - Comando: SELECT
- **DCL – *Data Control Language***
  - Comandos: GRANT, DENY e REVOKE
- DTL – *Data Transaction Language*
  - Comandos: COMMIT, ROLLBACK e SAVEPOINT

Comandos DCL – 2 de 2

- **DCL – *Data Control Language***
  - Trata de aspectos relacionados à segurança
  - Existem instruções para controlar a autorização e direitos de acesso ao BD;
  - Exemplos:

```
GRANT INSERT, UPDATE ON TbBanda TO usrProcopio;

DENY DELETE ON TbBanda TO usrProcopio;

REVOKE INSERT, UPDATE ON TbBanda TO usrProcopio;
```

Esta instrução produzirá um **erro** porque **não existe** o usuário **usrProcopio** em nosso banco de dados.

Comandos DTL – 1 de 2

- DDL – Data Definition Language
  - Comandos: CREATE, ALTER e DROP
- DML – *Data Manipulation Language*
  - Comandos: INSERT, UPDATE e DELETE
- DQL – *Data Query Language*
  - Comando: SELECT
- DCL – *Data Control Language*
  - Comandos: GRANT e REVOKE
- **DTL – *Data Transaction Language***
  - Comandos: COMMIT, ROLLBACK e SAVEPOINT

Comandos DTL – 2 de 2

- **DTL – *Data Transaction Language***
  - Usada para gerenciar as diferentes transações de um banco de dados
  - Exemplo 1 – Transação finalizada com **sucesso** para inserção de uma banda:

```
BEGIN TRANSACTION
    INSERT INTO TbBanda(NoBanda, VaCache)
    VALUES ('Jorge Vercilo', 100000)
COMMIT TRANSACTION
```

  - Exemplo 2 – Transação finalizada com **falha** para a inserção de uma banda:

```
BEGIN TRANSACTION
    INSERT INTO TbBanda(NoBanda, VaCache)
    VALUES ('Paula Fernandes', 250000)
ROLLBACK TRANSACTION
```

Principais Referências

1) w3SCHOOLS.COM. **SQL TUTORIAL**. Disponível em: https://www.w3schools.com/sql/default.asp. Acessado em: 22 jan 2019.

2) DAVE, Pinal. **SQL SERVER – What is – DML, DDL, DCL and TCL – Introduction and Examples**. Disponível em: http://blog.sqlauthority.com/2008/01/15/sql-server-what-is-dml-ddl-dcl-and-tcl-introduction-and-examples/. Acessado em: 22 jan. 2019.