VEM SER

Banco de Dados Oracle

Junção de Tabelas

Sumário

- UPDATE
- DELETE
- JOIN
- UNION
- EXISTS

UPDATE – Atualizar

```
UPDATE [ tabela ]
SET
  [ coluna_1 ] = [ novo_valor_1 ],
  [ coluna_2 ] = [ novo_valor_2 ]
WHERE
  [ condicao-de-busca ]
```

```
UPDATE
  produtos
SET
  descrição = 'Resma de ofício com 500 folhas',
  preco = 18.50
WHERE
  id = 1 OR preco = 17.50
```

- https://www.devmedia.com.br/sql-update/41185

DELETE – Excluir

- **LEMBRE SEMPRE DE UTILIZAR UMA CONDIÇÃO NO WHERE** para não excluir todos os dados da tabela.

```
DELETE
  FROM table_name
 WHERE condition;
```

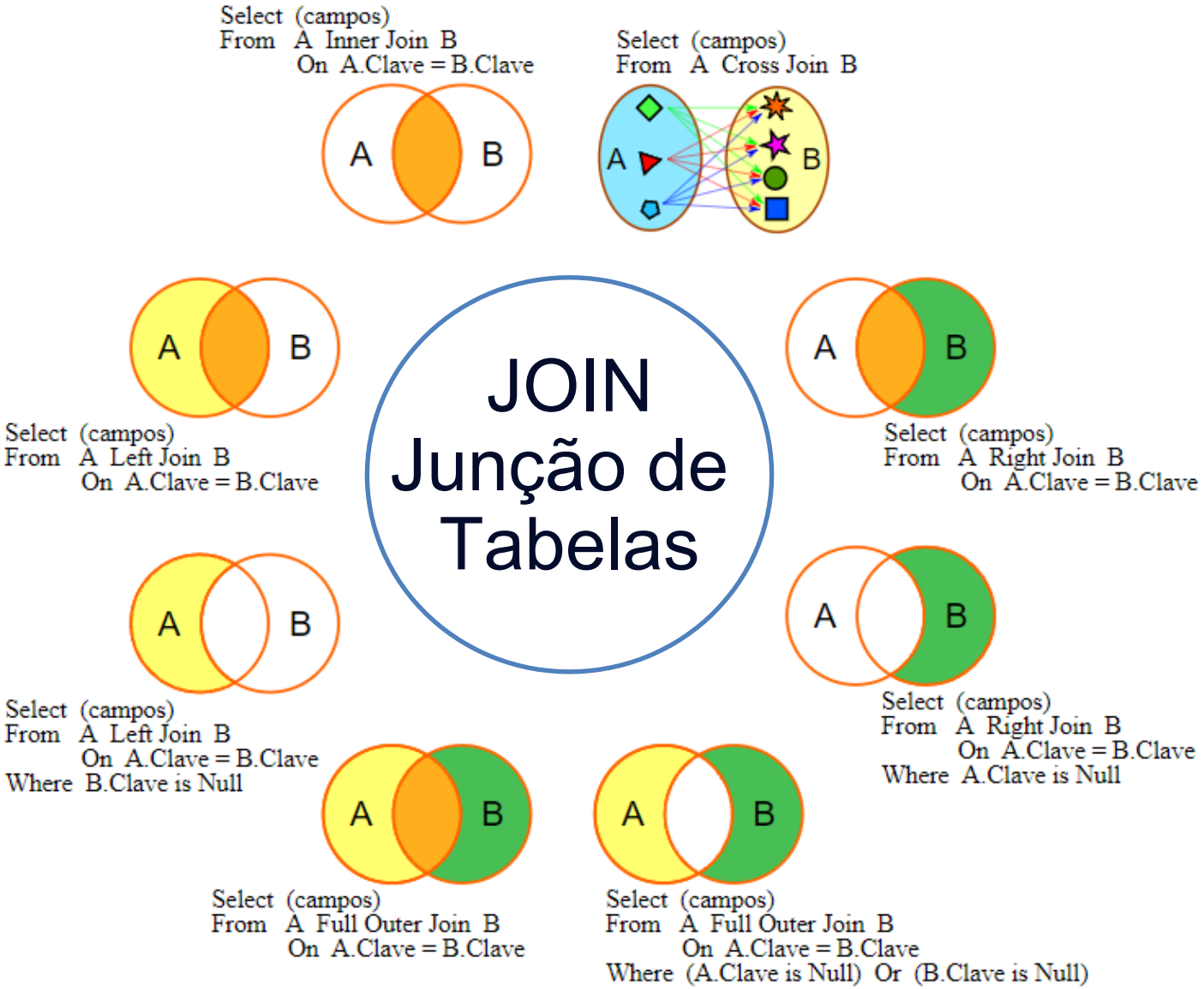
```
DELETE
  FROM produtos
WHERE id = 1
```

```
DELETE
  FROM produtos
WHERE nome like '%bala%'
```

- https://www.w3schools.com/sql/sql_delete.asp

Exercício #1

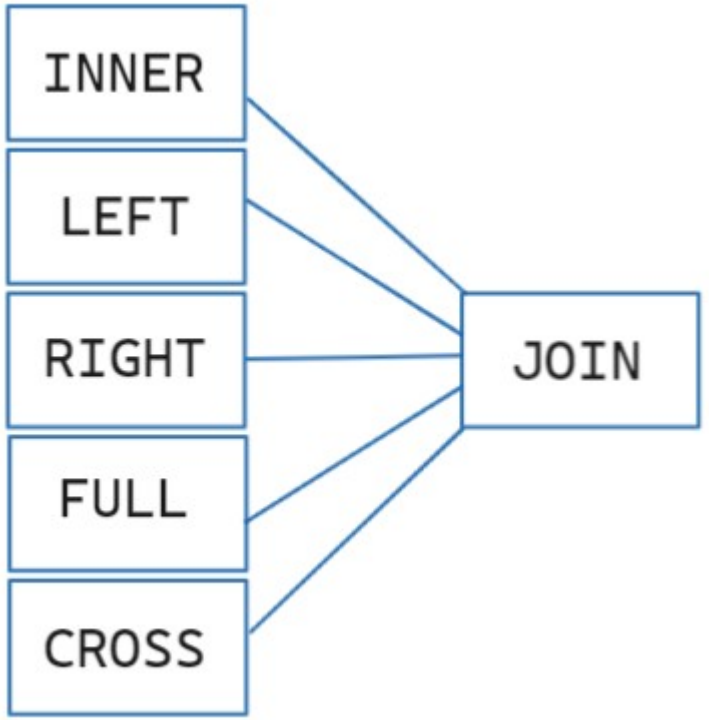
- Com as tabelas do exercício de ontem faça:
  - Atualizar o logradouro e o complemento dos endereços com id 2 e 3;
  - Atualizar o número do endereço onde id é 4 para 999999;
  - Remover o último registro da tabela endereço (utilizando a função max);
  - Remover o endereço onde o número = 999999;
  - Remover 2 registros da tabela endereço;



- https://upload.wikimedia.org/wikipedia/commons/c/c9/Joins_del_SQL.svg
- https://www.alura.com.br/artigos/join-e-seus-tipos

JOIN – Tipos de junção

- INNER JOIN
- LEFT OUTER JOIN
- RIGHT OUTER JOIN
- FULL OUTER JOIN
- CROSS JOIN
- “OUTER”s são opcionais.



- https://www.devmedia.com.br/inner-cross-left-rigth-e-full-joins/21016

INNER JOIN

INNER JOIN.sql - 1...dbTestes (sa (192))

```
SELECT F.NomeFuncionario
      ,C.NomeCargo
FROM              CARGO         AS C
   INNER JOIN     FUNCIONARIO   AS F   ON (F.CodCargo = C.CodCargo)
```

Results | Messages

	NomeFuncionario	NomeCargo
1	JOÃO	CAIXA
2	MARIA	VENDEDOR
3	CARLOS	CAIXA

- https://www.devmedia.com.br/inner-cross-left-rigth-e-full-joins/21016

LEFT JOIN

OUTER LEFT JOIN.s...Testes (sa (232))*

```
SELECT F.nomeFuncionario
      ,C.nomeCargo
FROM                      FUNCIONARIO AS F
      LEFT OUTER JOIN     CARGO       AS C  ON ( F.codCargo = C.codCargo )
```

Results | Messages

	nomeFuncionario	nomeCargo
1	JOÃO	CAIXA
2	MARIA	VENDEDOR
3	CARLOS	CAIXA
4	TADEU	NULL

- https://www.devmedia.com.br/inner-cross-left-rigth-e-full-joins/21016

RIGHT JOIN

OUTER RIGHT JOIN...Testes (sa (232))

```
SELECT F.nomeFuncionario
      ,C.nomeCargo
FROM                    FUNCIONARIO AS F
      RIGHT OUTER JOIN  CARGO       AS C  ON ( F.codCargo = C.codCargo )
```

Results | Messages

	nomeFuncionario	nomeCargo
1	JOÃO	CAIXA
2	CARLOS	CAIXA
3	MARIA	VENDEDOR
4	NULL	GERENTE

- https://www.devmedia.com.br/inner-cross-left-rigth-e-full-joins/21016

FULL JOIN

OUTER FULL JOIN.s...Testes (sa (232))*

```
SELECT F.nomeFuncionario
      ,C.nomeCargo
FROM                    FUNCIONARIO AS F
      FULL OUTER JOIN   CARGO       AS C  ON ( F.codCargo = C.codCargo )
```

Results | Messages

	nomeFuncionario	nomeCargo
1	JOÃO	CAIXA
2	MARIA	VENDEDOR
3	CARLOS	CAIXA
4	TADEU	NULL
5	NULL	GERENTE

- https://www.devmedia.com.br/inner-cross-left-rigth-e-full-joins/21016

CROSS JOIN

CROSS JOIN.sql - ...bTestes (sa (192))

```
SELECT F.NomeFuncionario
      ,C.NomeCargo
FROM              CARGO          AS C
   CROSS JOIN     FUNCIONARIO    AS F
```

Results | Messages

	NomeFuncionario	NomeCargo
1	JOÃO	CAIXA
2	JOÃO	VENDEDOR
3	JOÃO	GERENTE
4	MARIA	CAIXA
5	MARIA	VENDEDOR
6	MARIA	GERENTE
7	CARLOS	CAIXA
8	CARLOS	VENDEDOR
9	CARLOS	GERENTE

- https://www.devmedia.com.br/inner-cross-left-rigth-e-full-joins/21016


Vamos praticar!

Exercício #2

- Executar script “aula-03-create.sql”
- Criar scripts abaixo:
  - Fazer um cross join entre Pessoa e Contato

  - Fazer um inner join entre tabela Pessoa e Contato
  - Fazer um inner join entre tabela Pessoa, PESSOA_X_PESSOA_ENDERECO e Endereco_Pessoa
  - Fazer um inner join entre todas as tabelas (começando por pessoa)

  - Fazer um left join entre tabela Pessoa e Contato
  - Fazer um left join entre tabela Pessoa e PESSOA_X_PESSOA_ENDERECO e Endereco_Pessoa
  - Fazer um left join entre todas as tabelas (começando por pessoa)

UNION

```
SELECT <colunas> FROM table1
UNION
SELECT <colunas> FROM table2;
```

```
SELECT City FROM Customers
UNION
SELECT City FROM Suppliers
ORDER BY City;
```

EXISTS

```sql
SELECT <colunas>
FROM table_name
WHERE EXISTS
(SELECT column_name FROM table_name WHERE condition);
```

```sql
SELECT s.SupplierName
FROM Suppliers s
WHERE EXISTS (SELECT p.ProductName
                FROM Products p
               WHERE p.SupplierID = s.supplierID
                 AND Price < 20);
```

Task #1

- Criar script “task-03.sql” com a especificação abaixo :
  - Fazer um RIGHT JOIN entre tabelas:
    - Pessoa e Contato
    - Pessoa, PESSOA_X_PESSOA_ENDERECO e Endereco_Pessoa
    - Todas as tabelas (começando por pessoa)
  - Fazer um FULL JOIN entre tabelas:
    - Pessoa e Contato
    - Pessoa, PESSOA_X_PESSOA_ENDERECO e Endereco_Pessoa
    - Todas as tabelas (começando por pessoa)

- Utilizando o EXISTS, selecione as pessoas que tem endereço
- Selecione id, nome da tabela pessoa junto com id, logradouro da tabela endereço

Task #2 Grupo

- Com base no modelo ER da aula passada, gerar o script de **criação das tabelas** do projeto do time atual;
- Criar **sequences** para cada campo sequencial (os necessários) das tabelas;
- Criar script de **insert** de dados para as tabelas do time;
- Enviar o diagrama ER e os scripts atualizados no git do projeto;
- Estrutura: documentos/*.* , bd/criacao.sql, bd/dados.sql;


Let's Tech Up Together