DML – JOINS, FUNÇÕES PARA STRING E DATA

ANTONIO DAVID VINISKI
antonio.david@pucpr.br
PUCPR

TIPOS DE JOIN

LINGUAGEM DE MANIPULAÇÃO DE DADOS

- O **JOIN** é uma cláusula muito importante da linguagem SQL cuja função é combinar linhas de diferentes tabelas de acordo com as relações existentes entre as colunas dessas tabelas.
- Existem basicamente duas categorias de **JOINS** no SQL, **INNER** e **OUTER**, cada uma possuindo alguns tipos especiais.
  - **INNER**
    - **NATURAL JOIN**
    - **INNER JOIN** (ou apenas **JOIN**)
  - **OUTER**
    - **LEFT OUTER JOIN** (ou apenas **LEFT JOIN**)
    - **RIGHT OUTER JOIN** (ou apenas **RIGHT JOIN**)
    - **FULL OUTER JOIN**(ou apenas **OUTER JOIN**)

NATURAL JOIN

LINGUAGEM DE MANIPULAÇÃO DE DADOS

- O **JOIN NATURAL** considera apenas os pares de *tuplas* com o mesmo valor nos atributos que aparecem nos esquemas de ambas as relações (campo com mesmo nome).

```
SELECT * FROM pessoa, cliente WHERE pessoa.id = cliente.id;
```

```
SELECT * FROM pessoa NATURAL JOIN cliente;
```

INNER JOIN

LINGUAGEM DE MANIPULAÇÃO DE DADOS

- O **INNER JOIN** retorna apenas as linhas das tabelas que sejam comuns entre si, ou seja, as linhas em ambas as tabelas que possuam o campo de relacionamento com o mesmo valor.
- Na junção **INNER JOIN** especificamos o relacionamento entre as tabelas pela cláusula **ON**.

```
SELECT pessoa.*, cliente.*, comanda.*
FROM pessoa
INNER JOIN cliente ON cliente.id = pessoa.id
INNER JOIN comanda ON comanda.id_cliente = cliente.id;
```

LEFT JOIN

LINGUAGEM DE MANIPULAÇÃO DE DADOS

- O **LEFT JOIN** é usado para retornar todos os registros da tabela esquerda, além dos registros da tabela à direita que têm valores em comum com a tabela esquerda.

```
SELECT cliente.*, comanda.*
FROM cliente
LEFT JOIN comanda ON cliente.id_ cliente = cliente.id;
```

RIGHT JOIN

LINGUAGEM DE MANIPULAÇÃO DE DADOS

- O **RIGHT JOIN** é usado para retornar todos os registros da tabela direita, além dos registros da tabela à esquerda que têm valores em comum com a tabela direita.

```
SELECT pagamento_funcionario.*, registro.*
FROM pagamento_funcionario
RIGHT JOIN registro ON registro.id_comissao = pagamento_funcionario.id;
```

FULL JOIN

LINGUAGEM DE MANIPULAÇÃO DE DADOS

- O **FULL JOIN**, também conhecido como **FULL OUTER JOIN**, retorna todos os dados de ambas as tabelas quando há uma relação entre elas.

```
SELECT pagamento_funcionario.*, registro.*
FROM pagamento_funcionario
FULL JOIN registro ON registro.id_comissao = pagamento_funcionario.id;
```

FULL OUTER JOIN NÃO EXISTE NO MYSQL

FULL JOIN – UNION MYSQL

LINGUAGEM DE MANIPULAÇÃO DE DADOS

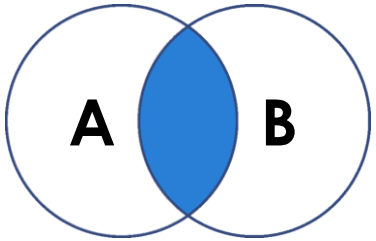
```
(SELECT pagamento_funcionario.*, registro.*
FROM pagamento_funcionario
LEFT JOIN registro ON registro.id_comissao = pagamento_funcionario.id)
UNION
(SELECT pagamento_funcionario.*, registro.*
FROM pagamento_funcionario
RIGHT JOIN registro ON registro.id_comissao = pagamento_funcionario.id);
```

EXPRESSÃO CASE

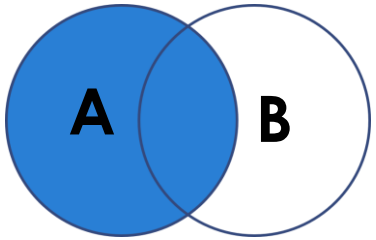
LINGUAGEM DE MANIPULAÇÃO DE DADOS

- O SQL oferece uma cláusula que permite estabelecer condições sobre os campos e especificar o retorno deles com base nos seus valores: a expressão **CASE**.
- Exemplo:
  - Precisamos definir o quando um garçom precisa receber de comissão pelos pedidos realizados.
    - considerando a comissão de 5% para vendas < R$ 300,00 e 10% para vendas > = R$ 300,00.

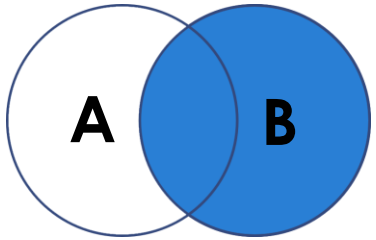
```
SELECT <fields>
FROM TableA A
INNER JOIN, TableB B
ON  A.id = B.id
```



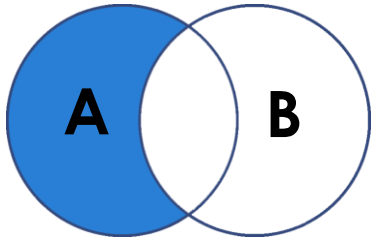

```
SELECT <fields>
FROM TableA A
LEFT JOIN, TableB B
ON  A.id = B.id
```



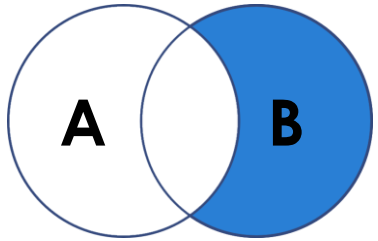

```
SELECT <fields>
FROM TableA A
RIGHT JOIN, TableB B
ON  A.id = B.id
```



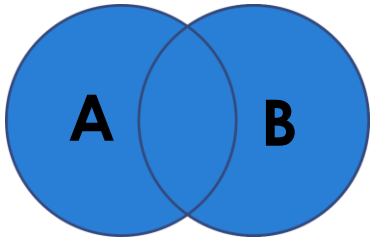

```
SELECT <fields>
FROM TableA A
LEFT JOIN, TableB B
ON  A.id = B.id
WHERE B.id IS NULL
```

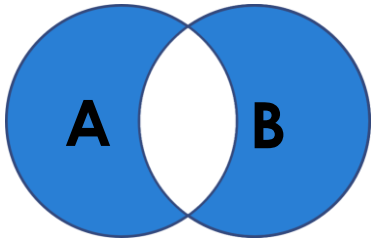



```
SELECT <fields>
FROM TableA A
RIGHT JOIN, TableB B
ON  A.id = B.id
WHERE A.id IS NULL
```




```
SELECT <fields>
FROM TableA A
FULL OUTER JOIN, TableB B
ON  A.id = B.id
```






```
SELECT <fields>
FROM TableA A
FULL OUTER JOIN, TableB B
ON  A.id = B.id
WHERE B.id IS NULL OR A.id IS NULL
```

EXPRESSÃO CASE

LINGUAGEM DE MANIPULAÇÃO DE DADOS

```
SELECT
        g.id AS "CODIGO GARÇOM",
        CASE WHEN SUM(r.valor_produto*r.quantidade) < 300.00
             THEN SUM(r.valor_produto*r.quantidade)*0.06
             ELSE SUM(r.valor_produto*r.quantidade)*0.10 END AS "PAGAMENTO"
FROM garcom AS g
JOIN registro AS r ON r.id_garcom = g.id
WHERE r.id_comissao IS NULL
GROUP BY g.id;
```

FUNÇÕES PARA MANIPULAÇÃO DE STRINGS

FUNÇÕES STRING I

LINGUAGEM DE MANIPULAÇÃO DE DADOS

- **SELECT** CONCAT('My', 'S', 'QL');

  > 'MySQL'

- **SELECT** CONCAT_WS(',', 'First name', 'Second name', 'Last Name');

  > 'First name,Second name,Last Name'.

- **SELECT** CHAR_LENGTH('ANTONIO');

  > 7

FUNÇÕES STRING II

LINGUAGEM DE MANIPULAÇÃO DE DADOS

- **SELECT** SUBSTRING('Politécnica');

  > 'técnica'

- **SELECT** SUBSTRING('Politécnica' **FROM** 4);

  > 'técnica'

- **SELECT** SUBSTRING('Politécnica', 5, 2);

  > 'té'

- **SELECT** SUBSTRING_INDEX('www.mysql.com', '.', 2);

  > 'www.mysql'

- **SELECT** SUBSTRING_INDEX('www.mysql.com', '.', -2);

  > 'mysql.com'

SUBSTRING_INDEX: Retorna a substring da string 'www.mysql.com' por exemplo antes de 2 ocorrências do delimitador. Se cont é positivo, tudo a esquerda do delimitador final (contando a partir da esquerda) é retornado. Se cont é negativo, tudo a direita do delimitador final (contando a partir da direita) é retornado.

FUNÇÕES STRING III

LINGUAGEM DE MANIPULAÇÃO DE DADOS

- **SELECT** UCASE('Antônio');
  > 'ANTÔNIO'
- **SELECT** UPPER('Antônio');
  > 'ANTÔNIO'
- **SELECT** LCASE('MYSQL');
  > 'mysql'
- **SELECT** LOWER('MYSQL');
  > 'mysql'
- **SELECT REPEAT**('MySQL', 3);
  >'MySQLMySQLMySQL'
- **SELECT REVERSE**('www.mysql.com');
  > 'moc.lqsym.www'

FUNÇÕES PARA MANIPULAÇÃO DE DATAS

FUNÇÕES DATA I

LINGUAGEM DE MANIPULAÇÃO DE DADOS

- **SELECT** DAYOFWEEK ('2022-10-05');

  > 4 (1 = Sunday, 2 = Monday, 3 = Tuesday, 4 = Wednesday, 5 = Thursday, 6 = Friday, 7 = Saturday)

- **SELECT** WEEKDAY ('2022-10-05');

  > 2 (0 = Monday, 1 = Tuesday, 2 = Wednesday, 3 = Thursday, 4 = Friday, 5 = Saturday, 6 = Sunday)

- **SELECT** DAYOFMONTH('2022-10-05');

  > 15

- **SELECT** DAYOFYEAR('2022-10-05');

  >278

- **SELECT MONTH**('2022-10-05');

  > 10

- **SELECT YEAR**('2022-10-05');

  > 2022

FUNÇÕES DATA II

LINGUAGEM DE MANIPULAÇÃO DE DADOS

- **SELECT** NOW();

  > 2022-10-03 22:53:50

- **SELECT** CURDATE();

  > 2022-10-03

- **SELECT** CURTIME();

  > 22:53:50

- **SELECT** DAYNAME('2022-10-05');

  > Wednesday

- **SELECT** MONTHNAME('2022-10-05');

  > October

FUNÇÕES DATA III

LINGUAGEM DE MANIPULAÇÃO DE DADOS

SELECT DATE_ADD(DATE, **INTERVAL** value **addunity**);

- **SELECT** DATE_ADD('2022-10-05', **INTERVAL** 3 **YEAR**);

  >2025-10-05
- **SELECT** DATE_ADD('2022-10-05', **INTERVAL** 3 **DAY**);

  >2022-10-08
- **SELECT** DATE_ADD('2022-10-05', **INTERVAL** 3 **MONTH**);

  >2023-01-05
- **SELECT** DATE_ADD('2022-10-05 12:00:00', **INTERVAL** 3 **HOUR**);

  >2022-10-05 15:00:00
- **SELECT** DATE_ADD('2022-10-05 12:00:00', **INTERVAL** 3 **MINUTE**);

  >2022-10-05 12:03:00
- **SELECT** DATE_ADD('2022-10-05 12:00:00', **INTERVAL** 45 **SECOND**);

  >2022-10-05 12:00:45

FUNÇÕES DATA IV

LINGUAGEM DE MANIPULAÇÃO DE DADOS

SELECT DATE_SUB(DATE, **INTERVAL** value **subunity**);

- **SELECT** DATE_ SUB('2022-10-05', **INTERVAL** 3 **YEAR**);

  >2019-10-05
- **SELECT** DATE_ SUB('2022-10-05', **INTERVAL** 3 **DAY**);

  >2022-10-02
- **SELECT** DATE_ SUB('2022-10-05', **INTERVAL** 3 **MONTH**);

  >2022-02-05
- **SELECT** DATE_ SUB('2022-10-05 12:00:00', **INTERVAL** 3 **HOUR**);

  >2022-10-05 09:00:00
- **SELECT** DATE_ SUB('2022-10-05 12:00:00', **INTERVAL** 3 **MINUTE**);

  >2022-10-05 11:57:00
- **SELECT** DATE_ SUB('2022-10-05 12:00:00', **INTERVAL** 45 **SECOND**);

  >2022-10-05 11:59:15

CÁLCULO DE IDADE

LINGUAGEM DE MANIPULAÇÃO DE DADOS

```
SELECT
        YEAR(FROM_DAYS(TO_DAYS(NOW())-TO_DAYS(p.data_nasc))) AS idade
FROM  pessoa AS p;
```

```
SELECT
    YEAR(FROM_DAYS(TO_DAYS(NOW())-TO_DAYS(p.data_nasc))) AS idade
FROM  pessoa AS p;
```

EXERCÍCIO - APLICAÇÃO

LINGUAGEM DE MANIPULAÇÃO DE DADOS

Nossa rede de *shoppings* tem várias filiais espalhadas pelo país. Cada filial disponibiliza diversos espaços para implantação de lojas de diferentes categorias. O espaço precisa estar cadastrado para que uma loja possa alugar e contem informações como tipo (Lojas âncoras, Megalojas, Satélites, Conveniência e serviços, Área de lazer), localização (andar), metragem, etc. Cada aluguel refere-se a um único espaço. As lojas pagam mensalmente pelo aluguel dos espaços.

1. Faça o modelo conceitual, lógico e físico.
2. Insira dados nas tabelas de forma que elas tenham 5 registros cada.
3. Realizar 10 consultas sobre os dados inseridos.