



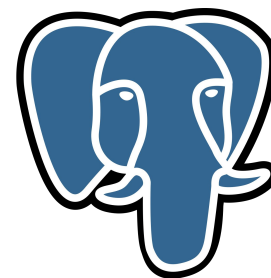
Banco de dados com PostgreSQL

Aula 2



Os comandos DML (Data Manipulation Language) são utilizados para manipular as linha dentro de uma tabela, seja pra inserir, editar ou excluir os itens de uma linha. Os comandos DML são os seguintes:

- INSERT
- UPDATE
- DELETE



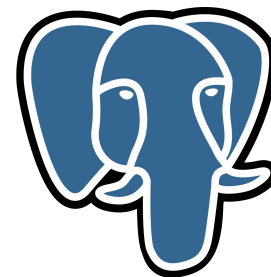


INSERT

Utilizado para inserir dados dentro de uma tabela.

```
INSERT INTO products VALUES (1, 'Cheese', 9.99);
```

```
INSERT INTO products VALUES (1, 'Cheese');
```



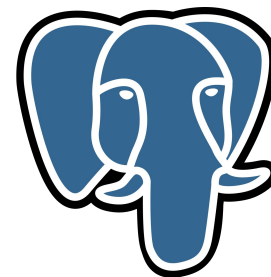


UPDATE

Para alterar informações em linhas já inseridas nas tabelas, utilizamos o comando update.

```
UPDATE products SET price = 10 WHERE price = 5;
```

```
UPDATE products SET price = price * 1.10;
```



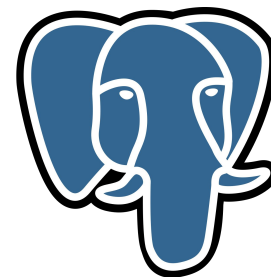


DELETE

Para remover linhas de uma tabela, utilizamos o comando DELETE.

```
DELETE FROM products WHERE price = 10;
```

```
DELETE FROM products;
```





Na tabela de raças, adicione as seguintes raças:

- Cachorro: Dushround
- Gato: Angorá
- Pássaro: Canário





Cadastre três donos de animais, após cadastrar os donos, cadastre seus respectivos animais:

- Epaminondas: Pluto, dushround
- Mariovalda: Pitty, Canário
- Terêncio: Maninho, Angorá





É necessário atualizar o cadastro de raças, deixando o tipo de animal no plural. Veja:

- Cachorro -> Cachorros;
- Gato -> Gatos;
- Pássaro -> Pássaros;

Após isso, o PetShop não atenderá mais passarinhos, apague todos os animais desse tipo da base de dados.

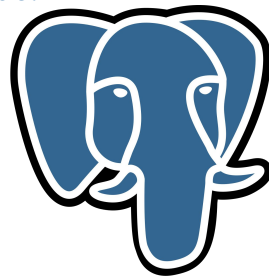




A DQL (Data Query Language) é o subconjunto responsável por comandos de consulta aos dados armazenados. Dentro dele, encontramos apenas o comando Select.

- SELECT

Esse comando é um dos mais importantes da SQL, pois é ele quem possibilita a consulta a dados de uma tabela. De modo geral, o Select recupera dados de determinado lugar. Os dados recuperados pelo Select são armazenados em uma nova tabela, chamada conjunto de resultados. É um comando que tem a possibilidade de ser estruturado de forma a fazer consultas mais simples ou mais complexas.



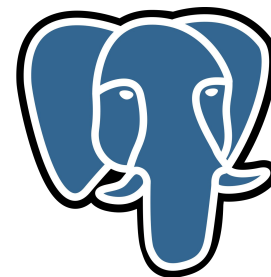


SELECT

Executando uma consulta simples

```
SELECT COLUNA1, COLUNA2, COLUNA3 FROM TABELA
```

```
SELECT * FROM TABELA
```





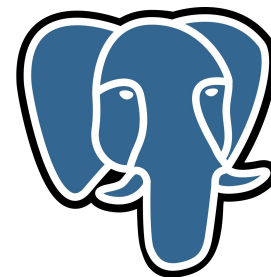
WHERE

Executando uma consulta simples

```
SELECT * FROM TABELA WHERE ID = 3
```

```
SELECT * FROM TABELA WHERE PRECO >= 15
```

```
SELECT * FROM TABELA WHERE NOME <> 'Teste'
```

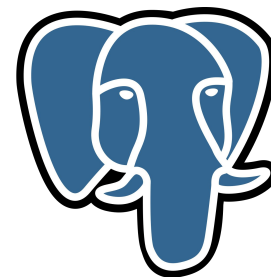




BETWEEN

A cláusula Between retorna todos os valores dentro de um determinado intervalo

```
SELECT * FROM TABELA WHERE PRECO BETWEEN 15 AND 25
```



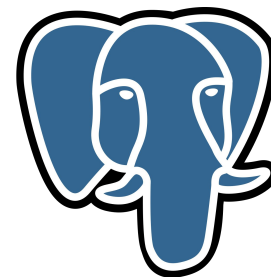


IN

O comando IN retorna os valores que estiverem dentro do intervalo

```
SELECT * FROM TABELA WHERE ID IN (1, 2, 3)
```

```
SELECT * FROM TABELA WHERE ID NOT IN (7, 14, 21)
```



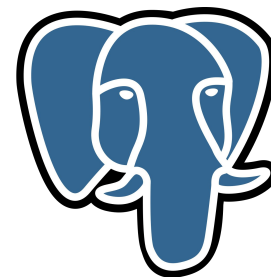


IS NULL

Os comandos IS NULL e IS NOT NULL são usados para verificar se um determinado campo está ou não nulo.

```
SELECT * FROM TABELA WHERE NOME IS NULL
```

```
SELECT * FROM TABELA WHERE NOME IS NOT NULL
```





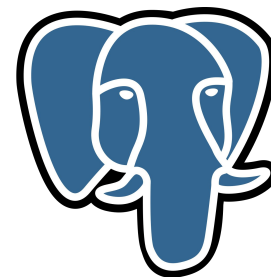
ORDER BY

A cláusula **Between** retorna todos os valores dentro de um determinado intervalo

```
SELECT * FROM TABELA ORDER BY COLUNA1
```

```
SELECT * FROM TABELA ORDER BY COLUNA2 DESC, COLUNA1 ASC
```

```
SELECT * FROM TABELA ORDER BY 1, 2 DESC, 3
```






Vamos efetuar uma consulta, listando todos os animais do petshop., listando-os na seguinte ordem:

nome, id





Uma chave estrangeira é um campo ou conjunto de campos em uma tabela que identifica uma linha em outra tabela, criando um relacionamento entre as duas tabelas. Assim, uma chave estrangeira é uma coluna em uma tabela que faz referência à chave primária de outra tabela.

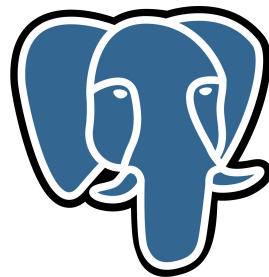
Uma tabela pode ter várias chaves estrangeiras, dependendo dos relacionamentos existentes com outras tabelas. A tabela que possui a chave estrangeira pode ser chamada de “tabela-filha”, e a tabela relacionada, onde se encontra a chave primária, pode ser definida como a “tabela-pai”.

Para definir uma chave estrangeira em PostgreSQL, usamos uma constraint de chave estrangeira – “FOREIGN KEY”. Essa restrição indica quais valores em uma coluna ou conjunto de campos na tabela-filha possuem correspondência com os valores em um campo ou conjunto de campos na tabela-pai. Desta forma, a chave estrangeira mantém o que chamamos de “integridade referencial” entre essas relações.

Criando uma tabela com foreign keys

Script de criação de uma tabela com uma foreign key

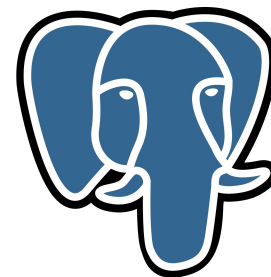
```
CREATE TABLE tbl_Livro (  
  ID_Livro int NOT NULL,  
  Nome_Livro varchar(40),  
  ID_Autor int NOT NULL,  
  ID_Editora int NOT NULL,  
  Data_Pub date,  
  Genero varchar(25),  
  Num_Paginas int,  
  PRIMARY KEY (ID_Livro),  
  FOREIGN KEY (ID_Autor) REFERENCES tbl_Autor (ID_Autor)  
);
```





Criando uma foreign key em uma tabela já existente.

```
ALTER TABLE tbl_Livro  
ADD CONSTRAINT fk_id_editora FOREIGN KEY (ID_Editora)  
REFERENCES tbl_Editora(ID_Editora);
```





Vamos continuar a aumentar a nossa base de dados. Crie uma tabela de consulta . A tabela deve possuir os seguintes campos:

- Id
- Data da consulta
- Animal que foi consultado
- Id do veterinário



Exercício 9



Insira cinco consultas diferentes, em períodos diferentes e para animais diferentes.





Crie as foreign keys das tabelas:

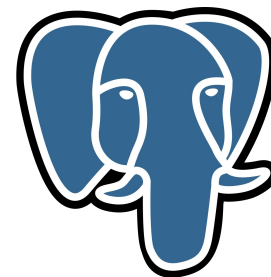
- Animal: foreign key de raça
- Animal: foreign key de pessoa, para o campo de dono





No select, podemos trabalhar com mais de uma tabela. Para isso, existem algumas formas. Abaixo a forma mais simples de fazer.

```
SELECT TABELA1.NOME, TABELA2.DESCRICAO  
FROM TABELA1, TABELA2  
WHERE TABELA1.CATEGORIA = TABELA2.ID
```



Exercício 11



Liste todas as consultas dos animais, ordenando pela data da consulta (descendente), e pelo nome do animal.

A consulta deve:

- Exibir a data da consulta
- O nome do animal
- A raça e o tipo de animal
- O nome do dono
- O nome do veterinário

