



# Banco de dados com PostgreSQL

## Aula 5

A partir de agora controlaremos também a caderneta de vacinação de cada um dos animais.  
Para isso:

- Crie um cadastro de Vacinas
- Cadastro de Vacinas por animal.
- Adicione uma data de nascimento no cadastro dos animais

| TB_VACINA_ANIMAL |        |
|------------------|--------|
| • ID             | Serial |
| • ANIMAL         | Int    |
| • APLICADO_EM    | DATE   |
| • APLICAR_ATE    | DATE   |
| • AQUI           | Bool   |
| • VACINA         | Int    |

| TB_VACINA       |             |
|-----------------|-------------|
| • ID            | Serial      |
| • DESCRICAO     | Varchar(50) |
| • TIPO          | Int         |
| • MES_APLICACAO | Int         |





Crie uma função chamada `gerar_vacinas`. A função deve possuir o seguinte comportamento:

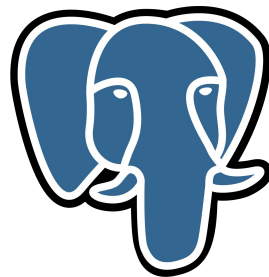
- Será passado por parâmetro o ID do animal
- A partir da raça do animal, cadastraremos as vacinas necessárias
- Caso a data de nascimento do animal não estiver cadastrada, gere um erro





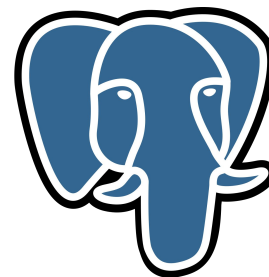
Triggers, em termos de banco de dados, são as operações realizadas de forma espontânea para eventos específicos. Quando tratamos dos eventos, estes podem ser tanto um INSERT quanto um UPDATE, ou mesmo um DELETE. Assim, podemos definir determinadas operações que serão realizadas sempre que o evento ocorrer.

Quando nos referirmos a uma operação com uma trigger, esta é conhecida por trigger de função ou trigger function. Lembre-se que trigger e função de trigger são duas coisas diferentes, onde a primeira pode ser criada utilizando a instrução CREATE TRIGGER, enquanto que a última é definida pelo comando CREATE FUNCTION. Em linhas gerais, com as triggers definimos qual tarefa executar, e com as triggers de função definimos como essa tarefa será realizada.





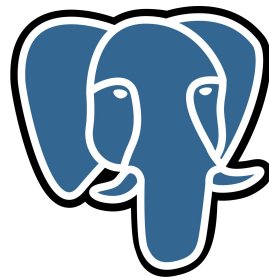
| Momento do disparo | Evento | Descrição                         |
|--------------------|--------|-----------------------------------|
| BEFORE             | INSERT | Antes de inserir um novo registro |
| AFTER              | INSERT | Após inserir um novo registro     |
| BEFORE             | UPDATE | Antes de editar um registro       |
| AFTER              | UPDATE | Depois de editar um registro      |
| BEFORE             | DELETE | Antes de excluir um registro      |
| AFTER              | DELETE | Depois de excluir um registro     |





### Criando a função que executará a regra

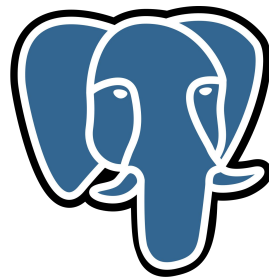
```
CREATE OR REPLACE FUNCTION funcionario_log_func()  
    RETURNS trigger  
LANGUAGE plpgsql AS  
$teste_trigger$  
BEGIN  
    INSERT INTO funcionario_funcionarios_auditoria(log_id,  
data_criacao)  
    VALUES(new.codigo_func, current_timestamp);  
  
    RETURN NEW;  
END  
$teste_trigger;
```





### Criando a função que executará a regra

```
CREATE TRIGGER log_trigger  
AFTER INSERT ON funcionarios  
FOR EACH ROW  
EXECUTE PROCEDURE funcionario_log_func();
```





Vamos criar uma tabela de configurações. Vamos chamá-la de TB\_CONFIG.


Essa tabela deve possuir as seguintes colunas:

- ID
- QTDMAXCONSULTAS

A coluna QTDMAXCONSULTAS guardará a quantidade máxima de consultas que um veterinário pode possuir no mês.







Antes de inserir uma nova consulta, será necessário verificar se o veterinário ultrapassou o limite máximo de consultas em um mês. Para isso, crie uma trigger na tabela de consultas, efetuando a verificação do parâmetro antes de inserir uma nova consulta.

Obs: Para extrair o mês e o ano de uma data, você pode utilizar as seguintes expressões:

```
EXTRACT(MONTH FROM current_timestamp)
```

```
EXTRACT(YEAR FROM current_timestamp)
```

