Faculdade: Instituto Federal Catarinense Campus: Camboriú
Curso: Sistemas para Internet
Discentes: Ademilton Amaro Mariano
Disciplina: Banco de Dados 2
Docente: Angelo Augusto Frozza

Link do repisório com arquivos originais:
https://github.com/ademilton-mariano/Centro_Exames_Pets

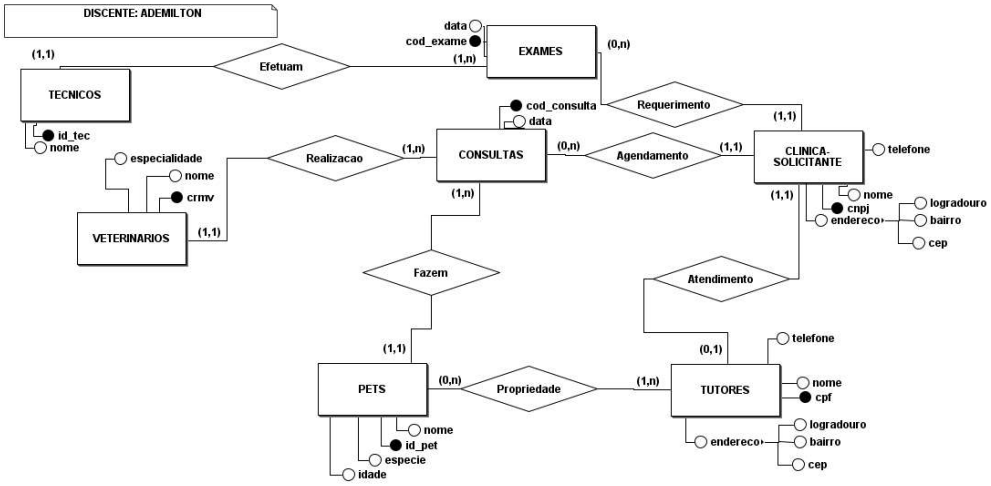
CENTRO DE CONSULTAS E EXAMES DE PETS

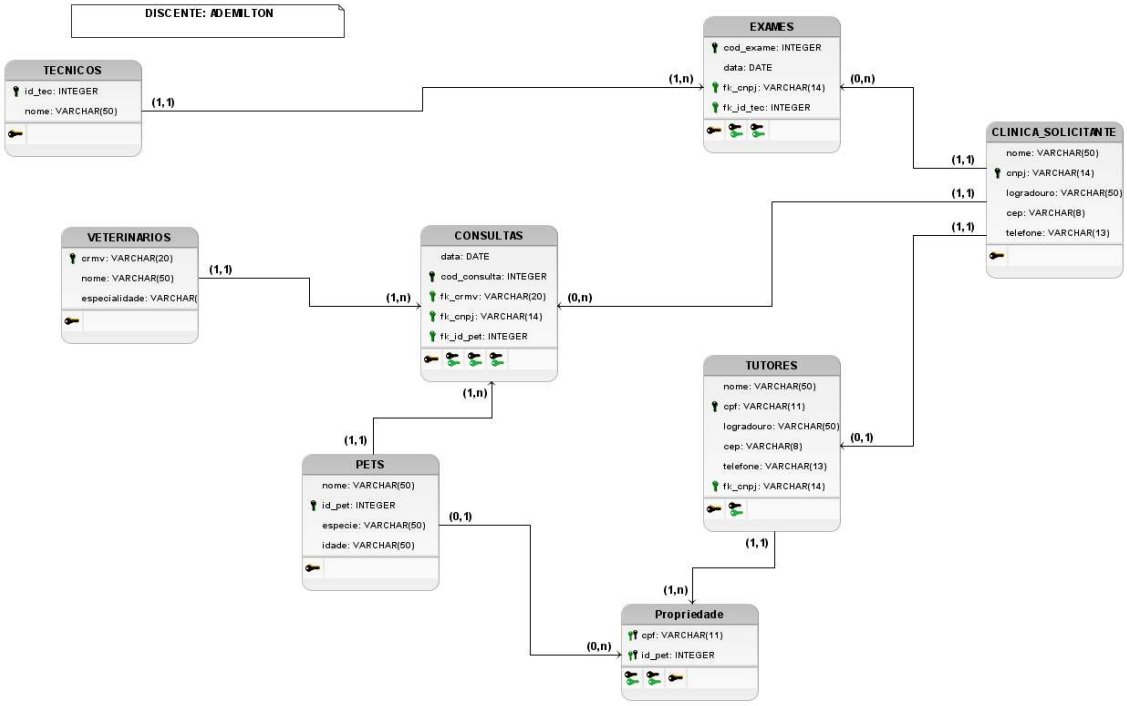
1. DESCRIÇÃO

Em um centro de consultas e exames para pets tem veterinários que realizam consultas e técnicos que efetuam exames, cada veterinário tem uma especialidade. Os médicos veterinários são identificados pelo CRMV e nome. Os Técnicos são identificados por um id e nome. Os agendamentos de consultas e exames são feitos por uma clínica solicitante, para agendamento são necessários os dados do tutor: nome, CPF, telefone e endereço. Os dados do pet também são armazenados para o agendamento: nome, idade e espécie. Um pet pode ter mais de um tutor, como um tutor pode ter mais de um pet.

2. C0NSULTAS

- Listar as consultas que uma clínica solicitou.
- Listar quais exames uma clínica solicitou.
- Mostrar por quais veterinários um pet foi atendido.
- Exibir a quais tutores o pet pertence.
- Listar quais exames um técnico efetivou.


DISCENTE: ADEMILTON
data
cod_exame
EXAMES
(1,1)
Efetuam
(1,n)
(0,n)
TECNICOS
id_tec
nome
cod_consulta
data
Requerimento
(1,1)
Realizacao
(1,n)
CONSULTAS
(0,n)
Agendamento
(1,1)
CLINICA-SOLICITANTE
telefone
especialidade
nome
crmv
VETERINARIOS
(1,1)
(1,n)
(1,1)
nome
cnpj
endereco
logradouro
bairro
cep
Fazem
Atendimento
(1,1)
(0,1)
telefone
PETS
(0,n)
Propriedade
(1,n)
TUTORES
nome
cpf
nome
id_pet
especie
idade
endereco
logradouro
bairro
cep


DISCENTE: ADEMILTON
TECNICOS
id_tec: INTEGER
nome: VARCHAR(50)
(1, 1)
(1,n)
EXAMES
cod_exame: INTEGER
data: DATE
fk_cnpj: VARCHAR(14)
fk_id_tec: INTEGER
(0,n)
(1, 1)
CLINICA_SOLICITANTE
nome: VARCHAR(50)
cnpj: VARCHAR(14)
logradouro: VARCHAR(50)
cep: VARCHAR(8)
telefone: VARCHAR(13)
(1, 1)
(1, 1)
VETERINARIOS
crmv: VARCHAR(20)
nome: VARCHAR(50)
especialidade: VARCHAR(
(1, 1)
(1,n)
CONSULTAS
data: DATE
cod_consulta: INTEGER
fk_crmv: VARCHAR(20)
fk_cnpj: VARCHAR(14)
fk_id_pet: INTEGER
(0,n)
(1,n)
(1, 1)
TUTORES
nome: VARCHAR(50)
cpf: VARCHAR(11)
logradouro: VARCHAR(50)
cep: VARCHAR(8)
telefone: VARCHAR(13)
fk_cnpj: VARCHAR(14)
(0, 1)
PETS
nome: VARCHAR(50)
id_pet: INTEGER
especie: VARCHAR(50)
idade: VARCHAR(50)
(0, 1)
(1, 1)
(1,n)
Propriedade
cpf: VARCHAR(11)
id_pet: INTEGER
(0,n)

```
--Discente: Ademilton Amaro Mariano
CREATE TABLE tecnicos (
  id_tecnico SERIAL PRIMARY KEY,
  nome_tecnico   VARCHAR(50) NOT NULL
);

CREATE TABLE clinica_solicitante (
  nome_clinica       VARCHAR(50) NOT NULL,
  cnpj_clinica       VARCHAR(14) PRIMARY KEY NOT NULL,
  logradouro_clinica VARCHAR(50) NOT NULL,
  cep_clinica        VARCHAR(8) NOT NULL,
  telefone_clinica   VARCHAR(13) NOT NULL
);

CREATE TABLE veterinarios (
  crmv_veterinario          VARCHAR(50) PRIMARY KEY NOT NULL,
  nome_veterinario          VARCHAR(50) NOT NULL,
  especialidade VARCHAR(50) NOT NULL
);

CREATE TABLE pets (
  nome_pet     VARCHAR(50) NOT NULL,
  id_pet   SERIAL PRIMARY KEY,
  especie  VARCHAR(50) NOT NULL,
  idade    VARCHAR(20) NOT NULL
);

CREATE TABLE exames (
  cod_exame          SERIAL PRIMARY KEY,
  data_exame               DATE NOT NULL,
  cnpj_clinica            VARCHAR(14) NOT NULL,
  id_tecnico          INTEGER NOT NULL,
  FOREIGN KEY (cnpj_clinica) REFERENCES clinica_solicitante (cnpj_clinica)
   ON DELETE CASCADE,
   FOREIGN KEY (id_tecnico) REFERENCES tecnicos (id_tecnico)
   ON DELETE CASCADE
);

CREATE TABLE tutores (
  nome_tutor                        VARCHAR(50) NOT NULL,
  cpf_tutor                         VARCHAR(11) PRIMARY KEY NOT NULL,
  logradouro_tutor                  VARCHAR(50) NOT NULL,
  cep_tutor                         VARCHAR(8) NOT NULL,
  telefone_tutor                    VARCHAR(13) NOT NULL,
  cnpj_clinica                     VARCHAR(14) NOT NULL,
  FOREIGN KEY (cnpj_clinica) REFERENCES clinica_solicitante (cnpj_clinica)
  ON DELETE CASCADE
);

CREATE TABLE consultas (
  data_consulta                   DATE NOT NULL,
```

```
  cod_consulta           SERIAL PRIMARY KEY,
  crmv_veterinario                VARCHAR(50) NOT NULL,
  cnpj_clinica               VARCHAR(14) NOT NULL,
  id_pet              INTEGER NOT NULL,
  FOREIGN KEY (crmv_veterinario) REFERENCES veterinarios (crmv_veterinario)
  ON DELETE CASCADE,
  FOREIGN KEY (cnpj_clinica) REFERENCES clinica_solicitante (cnpj_clinica)
  ON DELETE CASCADE,
  FOREIGN KEY (id_pet) REFERENCES pets (id_pet)
  ON DELETE CASCADE
);

CREATE TABLE propriedade (
  cpf_tutor    VARCHAR(11) NOT NULL,
  id_pet INTEGER NOT NULL,
  PRIMARY KEY (cpf_tutor, id_pet),
  FOREIGN KEY (cpf_tutor) REFERENCES tutores (cpf_tutor)
  ON DELETE CASCADE,
  FOREIGN KEY (id_pet) REFERENCES PETS (id_pet)
  ON DELETE CASCADE
  );

--foi usado o SGBD PostgreSQL




--Discente: Ademilton Amaro Mariano
--Inserindo dados

INSERT INTO tecnicos (nome_tecnico) VALUES('CARLOS');
INSERT INTO tecnicos (nome_tecnico) VALUES('DEBORA');
INSERT INTO tecnicos (nome_tecnico) VALUES('JULIA');
INSERT INTO tecnicos (nome_tecnico) VALUES('PEDRO');
INSERT INTO tecnicos (nome_tecnico) VALUES('CARLA');

INSERT INTO veterinarios VALUES('10245','VITOR','CARDIOLOGISTA');
INSERT INTO veterinarios VALUES('10345','VERA','ODONTOLOGISTA');
INSERT INTO veterinarios VALUES('10445','VERA','ORTOPEDISTA');
INSERT INTO veterinarios VALUES('10545','VERA','CLINICO GERAL');


INSERT INTO pets (nome_pet, especie, idade) VALUES('MEL','GATA','10 MESES');
INSERT INTO pets (nome_pet, especie, idade) VALUES('IGOR','GATO','8 MESES');
INSERT INTO pets (nome_pet, especie, idade) VALUES('YASMIN','CACHORRA','10 ANOS');
INSERT INTO pets (nome_pet, especie, idade) VALUES('BOLINHA','CACHORRO','5 ANOS');

INSERT INTO clinica_solicitante VALUES('PETMED','78945612309513','RUA 210,
Nº71','88210000','5547999665544');
```

```
INSERT INTO clinica_solicitante VALUES('PETLOVE','78945614596358','RUA 200,
Nº91','88210000','5547999665569');

INSERT INTO exames (data_exame, cnpj_clinica, id_tecnico)
VALUES('12/07/2020','78945612309513',1);
INSERT INTO exames (data_exame, cnpj_clinica, id_tecnico)
VALUES('12/07/2020','78945612309513',2);
INSERT INTO exames (data_exame, cnpj_clinica, id_tecnico)
VALUES('12/07/2020','78945614596358',3);
INSERT INTO exames (data_exame, cnpj_clinica, id_tecnico)
VALUES('12/09/2020','78945614596358',4);
INSERT INTO exames (data_exame, cnpj_clinica, id_tecnico)
VALUES('12/10/2020','78945612309513',1);
INSERT INTO exames (data_exame, cnpj_clinica, id_tecnico)
VALUES('10/07/2020','78945614596358',3);
INSERT INTO exames (data_exame, cnpj_clinica, id_tecnico)
VALUES('12/07/2021','78945612309513',2);
INSERT INTO exames (data_exame, cnpj_clinica, id_tecnico)
VALUES('12/07/2021','78945614596358',4);
INSERT INTO exames (data_exame, cnpj_clinica, id_tecnico)
VALUES('12/07/2020','78945614596358',4);

INSERT INTO consultas (data_consulta, crmv_veterinario, cnpj_clinica, id_pet)
VALUES('15/06/2019','10245','78945614596358',1);
INSERT INTO consultas (data_consulta, crmv_veterinario, cnpj_clinica, id_pet)
VALUES('15/07/2019','10545','78945614596358',2);
INSERT INTO consultas (data_consulta, crmv_veterinario, cnpj_clinica, id_pet)
VALUES('15/08/2019','10345','78945614596358',1);
INSERT INTO consultas (data_consulta, crmv_veterinario, cnpj_clinica, id_pet)
VALUES('15/06/2020','10445','78945612309513',3);
INSERT INTO consultas (data_consulta, crmv_veterinario, cnpj_clinica, id_pet)
VALUES('15/06/2019','10245','78945612309513',3);
INSERT INTO consultas (data_consulta, crmv_veterinario, cnpj_clinica, id_pet)
VALUES('10/08/2020','10345','78945612309513',4);

INSERT INTO tutores VALUES('CELSO','75315986204','RUA 100,
Nº40','88215647','5547999665987','78945614596358');
INSERT INTO tutores VALUES('CELIA','75315998412','RUA 105,
Nº80','88215650','5547999665456','78945614596358');
INSERT INTO tutores VALUES('CARLA','75315986100','RUA 600,
Nº560','88215654','5547999665568','78945612309513');
INSERT INTO tutores VALUES('ELIO','00015986204','RUA 100,
Nº70','88215647','5547999665054','78945612309513');

INSERT INTO propriedade VALUES('75315986204',1);
INSERT INTO propriedade VALUES('75315986204',2);
INSERT INTO propriedade VALUES('75315998412',1);
INSERT INTO propriedade VALUES('75315998412',2);
INSERT INTO propriedade VALUES('75315986100',3);
INSERT INTO propriedade VALUES('75315986100',4);
INSERT INTO propriedade VALUES('00015986204',3);
```

```
--d)

/*CONSULTAS
LISTAR CONSULTAS QUE UMA CLÍNICA SOLICITOU*/
/*SIMPLES*/
SELECT * FROM consultas WHERE cnpj_clinica = '78945612309513';
/*DETALHADA*/
SELECT cs.nome_clinica,
v.nome_veterinario, v.especialidade,
p.nome_pet, p.especie, p.idade,
c.data_consulta,c.cod_consulta,c.crmv_veterinario FROM consultas c
INNER JOIN clinica_solicitante cs ON c.cnpj_clinica = cs.cnpj_clinica
INNER JOIN veterinarios v ON c.crmv_veterinario = v.crmv_veterinario
INNER JOIN pets p ON p.id_pet = c.id_pet
WHERE cs.nome_clinica = 'PETLOVE' AND v.nome_veterinario = 'VITOR' AND p.nome_pet=
'MEL';

/*Listar quais exames uma clínica solicitou.*/
/*SIMPLES*/
SELECT * FROM exames WHERE cnpj_clinica = '78945612309513';
/*DETALHADA*/
SELECT t.nome_tecnico,
cs.nome_clinica, cs.telefone_clinica,
e.cod_exame, e.data_exame FROM exames e
INNER JOIN clinica_solicitante cs ON cs.cnpj_clinica = e.cnpj_clinica
INNER JOIN tecnicos t ON t.id_tecnico = e.id_tecnico
WHERE cs.nome_clinica = 'PETLOVE' AND t.nome_tecnico = 'JULIA';

/*Mostrar por quais veterinários um pet foi atendido.*/

SELECT v.nome_veterinario, v.especialidade,
p.nome_pet, p.especie, p.idade FROM consultas c
INNER JOIN veterinarios v ON c.crmv_veterinario = v.crmv_veterinario
INNER JOIN pets p ON p.id_pet = c.id_pet
WHERE  v.nome_veterinario = 'VITOR' AND p.nome_pet= 'MEL';

/*Exibir a quais tutores o pet pertence.*/
/*SIMPLES*/
SELECT * FROM propriedade WHERE id_pet = 1;
/*DETALHADA*/
SELECT p.nome_pet, p.idade, p.especie,
t.nome_tutor, t.cpf_tutor FROM propriedade pr
INNER JOIN pets p ON p.id_pet = pr.id_pet
INNER JOIN tutores t ON t.cpf_tutor = pr.cpf_tutor
WHERE p.nome_pet = 'YASMIN' AND t.nome_tutor = 'CARLA';

/*Listar quais exames um técnico efetivou.*/
/*SIMPLES*/
SELECT * FROM exames WHERE id_tecnico = 1;
```

```
/*DETALHADA*/
SELECT t.nome_tecnico, t.id_tecnico,
cs.nome_clinica,
e.cod_exame, e.data_exame FROM exames e
INNER JOIN clinica_solicitante cs ON cs.cnpj_clinica = e.cnpj_clinica
INNER JOIN tecnicos t ON t.id_tecnico = e.id_tecnico
WHERE cs.nome_clinica = 'PETMED' AND t.nome_tecnico = 'DEBORA';

-- e) StoredProcedures

--Incluir Pet
CREATE OR REPLACE FUNCTION
InserePet(a_nome VARCHAR(50),
a_especie VARCHAR(50), a_idade VARCHAR(50))
RETURNS void AS $$
BEGIN
INSERT INTO pets (nome_pet, especie, idade)
VALUES (a_nome, a_especie, a_idade);
END;
$$ LANGUAGE 'plpgsql';

--Incluir Veterinario
CREATE OR REPLACE FUNCTION
InsereVeterinario(a_crmv_veterinario VARCHAR(50),
a_nome_veterinario VARCHAR(50), a_especialidade VARCHAR(50))
RETURNS void AS $$
BEGIN
INSERT INTO veterinarios (crmv_veterinario, nome_veterinario, especialidade)
VALUES (a_crmv_veterinario, a_nome_veterinario, a_especialidade);
END;
$$ LANGUAGE 'plpgsql';

-- f) Functions

/*Função dadosConsulta, recebe como parâmetro o código da consulta.
Seleciona todos os dados da consulta.
Retorna registros contendo o código da consulta,
o nome do veterinario,o nome do pet e a clinica solicitante.*/

CREATE OR REPLACE FUNCTION dadosConsulta(a_cod_consulta INTEGER)
RETURNS SETOF consulta_detalhada
AS $$
BEGIN
  RETURN QUERY
    SELECT * FROM consulta_detalhada
    WHERE cod_consulta = a_cod_consulta;
  RETURN;
END;
$$ LANGUAGE 'plpgsql';

/*Função dadosExame, recebe como parâmetro o código do exame.
```

```
Seleciona todos os dados da exame.
Retorna registros contendo o código do exame,
o nome do tecnico e a clinica solicitante.*/

CREATE OR REPLACE FUNCTION dadosExame(a_cod_exame INTEGER)
RETURNS SETOF exame_detalhado
AS $$
BEGIN
  RETURN QUERY
    SELECT * FROM exame_detalhado
    WHERE cod_exame = a_cod_exame;
  RETURN;
END;
$$ LANGUAGE 'plpgsql';

-- g) Views

--Exibir uma consulta mais detalhada
CREATE OR REPLACE VIEW consulta_detalhada
(cod_consulta, data_consulta, pet, clinica_solicitante, veterinario) AS
SELECT c.cod_consulta, c.data_consulta, p.nome_pet AS pet,
cs.nome_clinica As clinica_solicitante,
v.nome_veterinario AS veterinario  FROM consultas c
INNER JOIN pets p ON p.id_pet = c.id_pet
INNER JOIN clinica_solicitante cs ON cs.cnpj_clinica = c.cnpj_clinica
INNER JOIN veterinarios v ON c.crmv_veterinario = v.crmv_veterinario;

--Exibir exame mais detalhado
CREATE OR REPLACE VIEW exame_detalhado
(cod_exame, data_exame, clinica_solicitante, tecnico) AS
SELECT e.cod_exame, e.data_exame,
cs.nome_clinica As clinica_solicitante,
t.nome_tecnico AS tecnico  FROM exames e
INNER JOIN clinica_solicitante cs ON cs.cnpj_clinica = e.cnpj_clinica
INNER JOIN tecnicos t ON t.id_tecnico = e.id_tecnico;

-- h) Triggers

--abela para armazenar logs

CREATE TABLE logs (
  idlog SERIAL PRIMARY KEY,
  datalog TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP,
  tabela VARCHAR(100) NOT NULL,
  operacao VARCHAR(30) NOT NULL,
  dadosantigos TEXT NULL DEFAULT '',
  dadosnovos TEXT NULL DEFAULT ''
);

/* Criar a Function Trigger que verifica qual operação está sendo feita no BD
e procedendo a inclusão do registro na tabela de logs de acordo com a operação.*/
```

```
CREATE OR REPLACE FUNCTION VerificaOperacao() RETURNS TRIGGER
AS $$
DECLARE dadosantigos TEXT; dadosnovos TEXT; tabela TEXT;
BEGIN
  tabela := TG_TABLE_NAME;

  IF (TG_OP = 'UPDATE') THEN
    dadosantigos := ROW(OLD.*);
    dadosnovos := ROW(NEW.*);
    INSERT INTO logs (tabela, operacao, dadosantigos, dadosnovos)
    VALUES (tabela, 'UPDATE', dadosantigos, dadosnovos);

    RETURN NEW;
  ELSEIF (TG_OP = 'DELETE') THEN
    dadosantigos := ROW(OLD.*);

    INSERT INTO logs (tabela, operacao, dadosantigos, dadosnovos)
    VALUES (tabela, 'DELETE', dadosantigos, DEFAULT);

    RETURN OLD;
  ELSEIF (TG_OP = 'INSERT') THEN
    dadosnovos := ROW(NEW.*);

    INSERT INTO logs (tabela, operacao, dadosantigos, dadosnovos)
    VALUES (tabela, 'INSERT', DEFAULT, dadosnovos);

    RETURN NEW;

  END IF;
END;
$$ LANGUAGE 'plpgsql';

--Criação de Triggers de todas as tabelas para ser executada depois de uma
operação.

CREATE TRIGGER Operacao
AFTER INSERT OR UPDATE OR DELETE ON clinica_solicitante
FOR EACH ROW EXECUTE PROCEDURE VerificaOperacao();

CREATE TRIGGER Operacao
AFTER INSERT OR UPDATE OR DELETE ON consultas
FOR EACH ROW EXECUTE PROCEDURE VerificaOperacao();

CREATE TRIGGER Operacao
AFTER INSERT OR UPDATE OR DELETE ON exames
FOR EACH ROW EXECUTE PROCEDURE VerificaOperacao();

CREATE TRIGGER Operacao
AFTER INSERT OR UPDATE OR DELETE ON pets
FOR EACH ROW EXECUTE PROCEDURE VerificaOperacao();
```

```
CREATE TRIGGER Operacao
AFTER INSERT OR UPDATE OR DELETE ON tecnicos
FOR EACH ROW EXECUTE PROCEDURE VerificaOperacao();

CREATE TRIGGER Operacao
AFTER INSERT OR UPDATE OR DELETE ON tutores
FOR EACH ROW EXECUTE PROCEDURE VerificaOperacao();

CREATE TRIGGER Operacao
AFTER INSERT OR UPDATE OR DELETE ON veterinarios
FOR EACH ROW EXECUTE PROCEDURE VerificaOperacao();

CREATE TRIGGER Operacao
AFTER INSERT OR UPDATE OR DELETE ON propriedade
FOR EACH ROW EXECUTE PROCEDURE VerificaOperacao();

-- testes

INSERT INTO tecnicos (nome_tecnico) VALUES('SERAFIM');
UPDATE tecnicos SET nome_tecnico = 'JUVENAL'
WHERE id_tecnico = 6;

SELECT * FROM logs;

-- i)

 --Criando Usuarios
CREATE USER Ademilton PASSWORD '123456';
CREATE USER Alex PASSWORD '963147';
CREATE USER Fabio PASSWORD '56789'

-- Criando grupos

CREATE GROUP Atendimento;
CREATE GROUP Gerencia;

-- Adicionando usuários aos grupos

ALTER GROUP Atendimento ADD USER Alex;
ALTER GROUP Gerencia ADD USER Fabio;

--Definindo permissões

GRANT SELECT, INSERT, UPDATE ON clinica_solicitante, consultas,
exames, pets, propriedade, tutores TO GROUP Atendimento;
GRANT SELECT, INSERT, UPDATE, DELETE, RULE, REFERENCES, TRIGGER ON
clinica_solicitante, consultas,
exames, pets, propriedade, tutores, tecnicos, veterinarios TO GROUP Gerencia;

-- Foi usado o SGBD postgreSQL
```