Curso: Engenharia de Software – 3ª fase

Disciplina: Banco de Dados

Professora: Edicarsia Barbiero Pillon

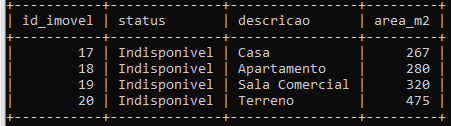
Estudantes: Edson Borges, Heberton Ayrton Cabral, Wuelliton Christian dos Santos

**Comandos – Banco de Dados**

**1**- Mostre o id\_imovel, status , descricao, area\_m2 dos imóveis que não estão sendo anunciados(status “Indisponivel”).

select id\_imovel, status, descricao, area\_m2 from

imovel where status=”Indisponivel”;



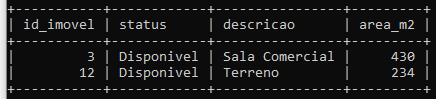
**2**- Mostre o id\_imovel, status, descricao, area\_m2 dos imóveis que estão no estado do Pará(estado=Para).

select id\_imovel, status, descricao, area\_m2 from imovel i inner join

endereco e on i.cod\_endereco=e.codigo\_endereco inner join

cidade c on e.cod\_cidade=c.codigo\_cidade where

estado=”Para”;

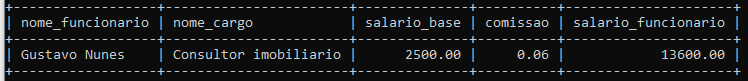


**3**- Mostre o nome do funcioario, nome do cargo, salario base do cargo, comissão e salario do funcionario que fez transacao onde o numero do contrato é igual a 2.

select f.nome\_funcionario, c.nome\_cargo, c.salario\_base, f.comissao, f.salario\_funcionario from funcionario f inner join cargo c on f.cod\_cargo=c.codigo\_cargo inner join

transacao t on f.codigo\_funcionario=t.cod\_funcionario where

num\_contrato=2;



**4**- Crie uma view com as informações do id\_imovel, status, descricao, area\_m2, nome\_cidade, estado, bairro e numero de todos os imóveis, ordenado por id\_imovel.

create view info\_imovel (id\_imovel, status, descricao, area\_m2, nome\_cidade, estado, bairro, numero) as

select i.id\_imovel, i.status, i.descricao, i.area\_m2, c.nome\_cidade, c.estado, e.bairro, e.numero

from imovel i inner join

endereco e on i.cod\_endereco=e.codigo\_endereco inner join

cidade c on e.cod\_cidade=c.codigo\_cidade order by id\_imovel;



**5**- Crie uma view com as informações do id\_transacao, nome\_funcionario, data\_transacao, tipo\_pagamento, id\_imovel, num\_contrato e valor\_final, ordenado por id\_transacao.

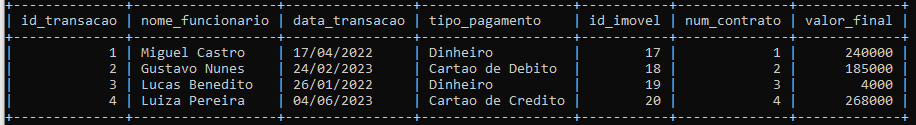
create view info\_transacao (id\_transacao, nome\_funcionario, data\_transacao, tipo\_pagamento, id\_imovel, num\_contrato, valor\_final) as

select t.id\_transacao, f.nome\_funcionario, t.data\_transacao, fp.tipo\_pagamento, i.id\_imovel, t.num\_contrato, t.valor\_final from transacao t inner join

funcionario f on t.cod\_funcionario=f.codigo\_funcionario inner join

forma\_pagamento fp on t.id\_pagm=fp.id\_pagamento inner join

imovel i on t.id\_imv=i.id\_imovel order by id\_transacao;



**6**- Crie um procedimento para quando eu digitar o nome da tabela mostre todas as informações contidas nela.

delimiter &&

create procedure select\_table (in tabela varchar(20))

begin

if (tabela=”anuncio” or tabela=”apartamento” or tabela=”cargo” or tabela=”casa” or tabela=”cidade” or tabela=”cliente” or tabela=”endereco” or tabela=”forma\_pagamento” or tabela=”foto” or tabela=”funcionario” or tabela=”historico” or tabela=”imobiliaria\_cliente” or tabela=”imovel” or tabela=”proprietario” or tabela=”sala\_comercial” or tabela=”terreno” or tabela=”transacao” or tabela=”usuario”) then

set @consultar = concat(“select \* from “, tabela);

prepare smt from @consultar;

execute smt;

else

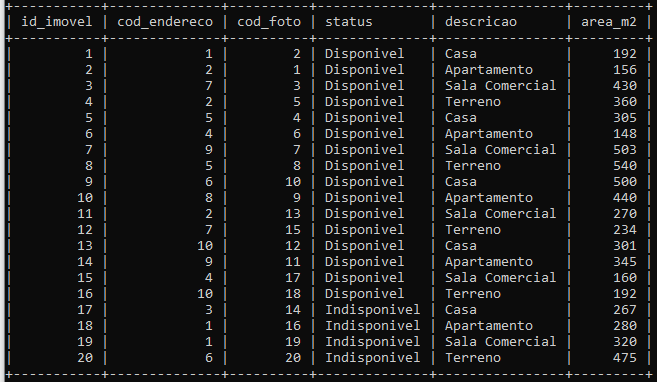
show tables;

end if;

end &&

delimiter ;

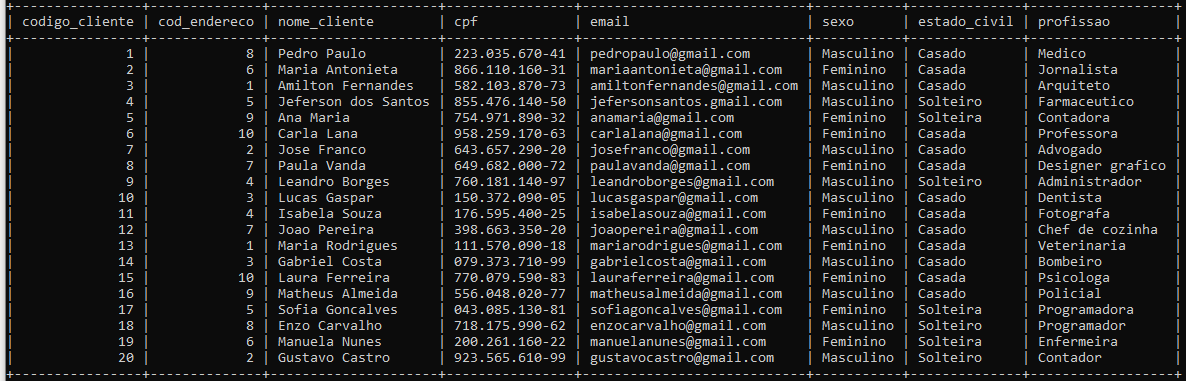
call select\_table (“imovel”);



**7**- Crie um procedimento para inserir dados na tabela cliente.

Tabela “cliente” antes da alteração

call select\_table (“cliente”);



delimiter &&

create procedure inserir\_cliente (in codigo\_cliente int(2), cod\_endereco int(2), nome\_cliente varchar(100), cpf char(15), email varchar(100), sexo varchar(20), estado\_civil varchar(20), profissao varchar(50))

begin

replace into cliente values (codigo\_cliente, cod\_endereco, nome\_cliente, cpf, email, sexo, estado\_civil, profissao);

end &&

delimiter ;

call inserir\_cliente (21, 3, "Fernando Cloriano", "449.530.690-13", "fernandocloriano@gmail.com", "Masculino", "Casado", "Jornalista");

Tabela “cliente” depois da alteração

call select\_table (“cliente”);



**8**- Crie uma trigger para quando um registro for atualizado na tabela anuncio, as informações anteriores da tabela anuncio seja passadas para a tabela historico.

Tabela “anuncio” antes da alteração

call select\_table (“anuncio”);

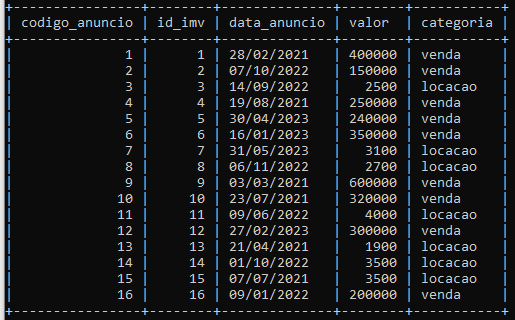
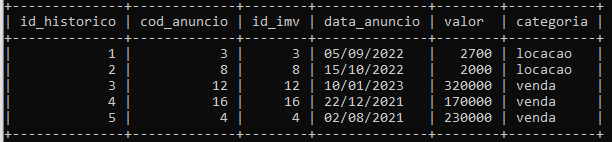


Tabela “historico” antes da alteração

call select\_table (“historico”);



delimiter &&

create trigger inserir\_historico

before update on anuncio

for each row

begin

insert into historico (cod\_anuncio, id\_imv, data\_anuncio, valor, categoria)

select codigo\_anuncio, id\_imv, data\_anuncio, valor, categoria from anuncio where codigo\_anuncio=old.codigo\_anuncio;

end &&

delimiter ;

update anuncio set data\_anuncio=”20/05/2023”, valor=450000, categoria=“venda” where codigo\_anuncio=1;

Tabela “anuncio” depois da alteração

call select\_table (“anuncio”);

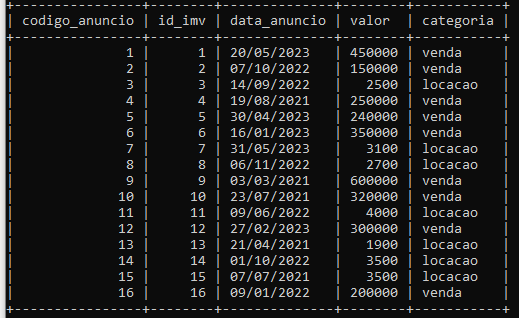
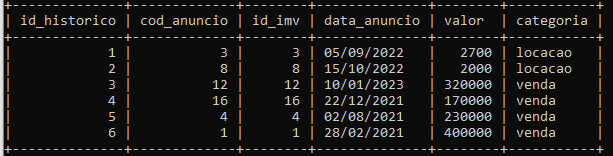


Tabela “historico” depois da alteração

call select\_table (“historico”);



**9**- Crie uma trigger para quando o registro na table “usuario” for excluida, também o mesmo aconteça na tabela “cliente”.

Tabela “usuario” antes da alteração

call select\_table (“usuario”);

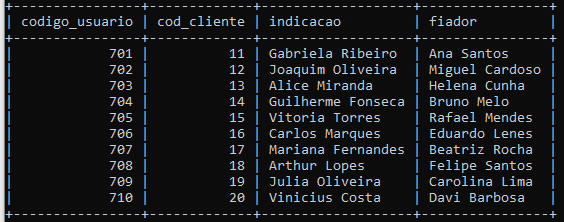
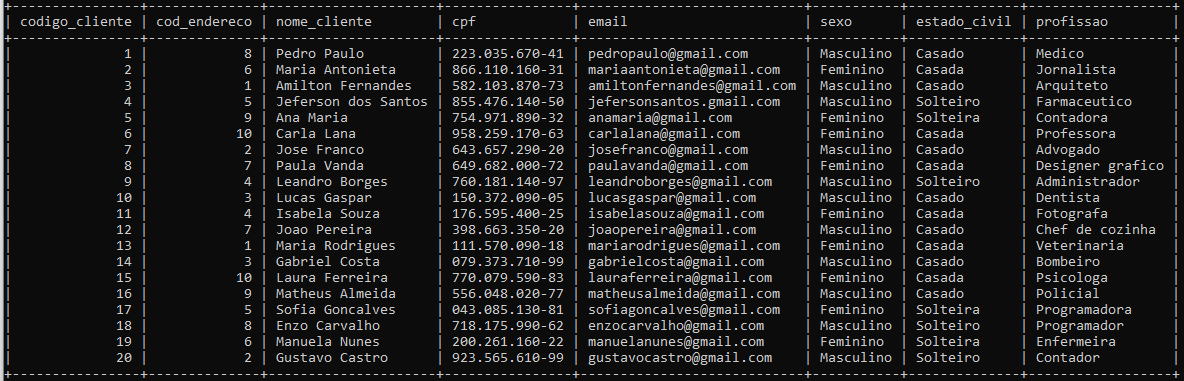


Tabela “cliente” antes da alteração

call select\_table (“cliente”);



delimiter &&

create trigger excluir\_cliente

after delete on usuario

for each row

begin

delete from cliente where codigo\_cliente=old.cod\_cliente;

end &&

delimiter ;

delete from usuario where cod\_cliente=11;

Tabela “usuario” depois da alteração

call select\_table (“usuario”);

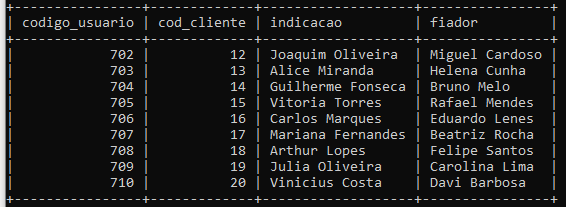
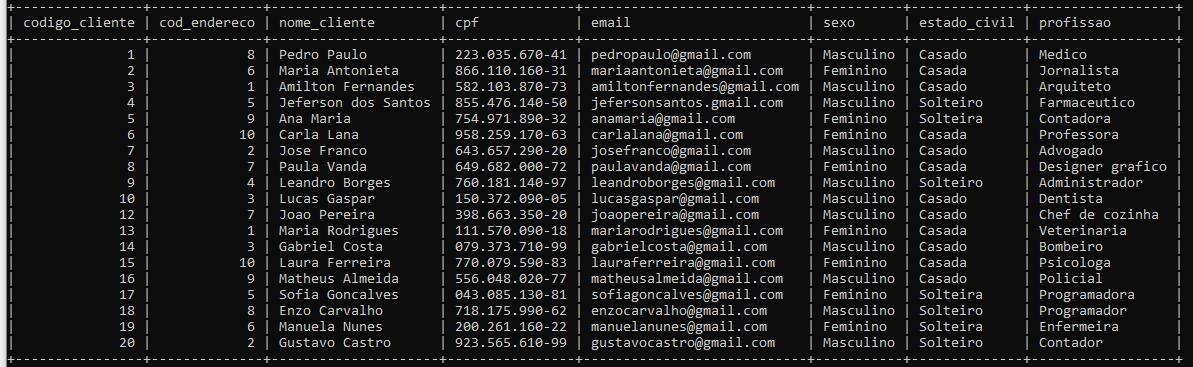


Tabela “cliente” depois da alteração

call select\_table (“cliente”);



**10**- Crie uma view com as seguintes informações nome\_cliente, cpf, estado\_civil, profissao, nome\_cidade, estado, bairro.

create view info\_cliente (nome\_cliente, cpf, estado\_civil, profissao, nome\_cidade, estado, bairro) as select cli.nome\_cliente, cli.cpf, cli.estado\_civil, cli.profissao, cid.nome\_cidade, cid.estado, e.bairro from cliente cli inner join

endereco e on cli.cod\_endereco=e.codigo\_endereco inner join

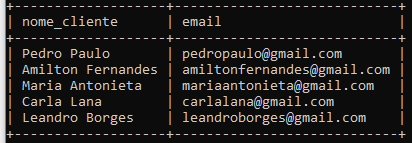
cidade cid on e.cod\_cidade=cid.codigo\_cidade;



**11**- Selecione o nome do cliente e email dos cliente que tem imovel registrado em seu nome.

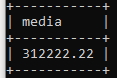
select nome\_cliente, email from cliente c inner join

imobiliaria\_cliente i on c.codigo\_cliente=i.cod\_cliente where cod\_cliente=codigo\_cliente;



**12**- selecione a media dos valores dos imóveis que estão a venda.

select round(AVG(valor),2) media from anuncio where categoria=”venda”;



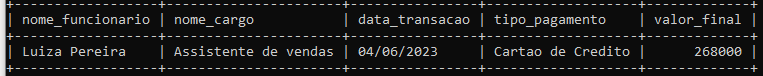
**13**- Selecione nome\_funcionario, nome\_cargo, data\_transacao, tipo\_pagamento, valor\_final do maior valor da transacao feita.

select f.nome\_funcionario, c.nome\_cargo, t.data\_transacao, fp.tipo\_pagamento, MAX(t.valor\_final) valor\_final from transacao t inner join

funcionario f on t.cod\_funcionario=f.codigo\_funcionario inner join

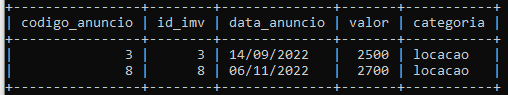
cargo c on f.cod\_cargo=c.codigo\_cargo inner join

forma\_pagamento fp on t.id\_pagm=fp.id\_pagamento where t.valor\_final=(select MAX(valor\_final) from transacao);



**14**- Selecione os anuncio que são da categoria=”locacao” em que o valor estão entre 2000 e 3000.

select codigo\_anuncio, id\_imv, data\_anuncio, valor, categoria from anuncio where categoria=”locacao” and valor between 2000 and 3000;



**15**- Selecione o nome e email dos cliente que moram no bairro com nome “Cidade Nova”.

select c.nome\_cliente, c.email from cliente c inner join

endereco e on c.cod\_endereco=e.codigo\_endereco where

bairro=”Cidade Nova”;

