Aula de Banco de Dados

Criando procedure de bakup

1ª Etapa

create table backup\_tabelas.cargos select \* from itau.cargos;

create table backup\_tabelas.funcionarios select \* from itau.funcionarios;

create table backup\_tabelas.clientes select \* from itau.clientes;

create table backup\_tabelas.marcas select \* from itau.marcas;

create table backup\_tabelas.produtos select \* from itau.produtos;

2ª Etapa

Criar arquivo para excluir tabelas

drop table backup\_tabelas.cargos;

drop table backup\_tabelas.funcionarios;

drop table backup\_tabelas.clientes;

drop table backup\_tabelas.marcas;

drop table backup\_tabelas.produtos;

3ª Etapa é construir a procedure

Inserindo dados na tabela de musica, gravadora e CD

=========================================

insert into itau.gravadora

(nome\_gravadora,site)

values

("Sony","www.sony.com.br");

insert into itau.musicas

(nome\_musica,tempo\_duracao)

values

("Minha prima","00:02:50");

insert into itau.cd

(nome\_cd,codigo\_musica,codigo\_gravadora)

values

("Lavo ta novo",1,1);

Trazer o relacionamento em um select com os campos abaixo:

=========================================================

select f.nomefuncionario,c.nomecargo,c.valorcargo

from itau.cargos as c

inner join itau.funcionarios as f

on(c.codcargo=f.codcargo);

Inner join permite mostrar os registros nos quais os campos estão com igualde de relacionamento de

registros

=========================================================

select f.nomefuncionario,c.nomecargo,c.valorcargo

from itau.cargos as c

left join itau.funcionarios as f

on(c.codcargo=f.codcargo);

Left join ermite mostrar os registros nos quais os campos estão com igualde de relacionamento de

registros e os campos da tabela da esquerda que estiverem sem registros ele mostra como num,

indicando um futuro relacionamento.

Como fazer o relacionamento de tabelas com o SQL nativo

create table itau.cargos(

codcargo char(2) not null,

nomecargo varchar(50)not null,

valorcargo decimal(8,2) not null,

primary key (codcargo)

);

create table itau.funcionarios(

matricula int not null auto\_increment,

nomefuncionario varchar(50) not null,

codcargo char(2) not null,

primary key(matricula),

foreign key (codcargo) references cargos(codcargo)

);

insert into itau.cargos

(codcargo,nomecargo,valorcargo)

values

("C1","Analista de Sistemas",5890.99),

("C2","DBA",7000.00),

("C3","Analista de Suporte",4500.00),

("C4","Gerente Financeiro",6000.00);

insert into itau.funcionarios

(nomefuncionario,codcargo)

values

("Viviane Cristina","C1"),

("Rodney Azevedo","C2"),

("Roberto Lopes","C2"),

("Vitor Lucas","C4"),

("Tarciano Pereira","C3");

Comando select fazendo o relacionamento das tabelas Produtos x Marca

========================================================

select p.nome\_produto,m.nome\_marca

from itau.marcas as m,

itau.produtos as p

where (p.cod\_marca=m.cod\_marca) order by nome\_produto;

Inserindo produtos na tabela de produtos

==================================

insert into itau.produtos

(nome\_produto,categoria\_produto,cod\_marca)

values

("Batedeira","Eletrodoméstico",1),

("Cafeteira","Eletrodoméstico",1),

("Geladeira","Eletrodoméstico",2),

("Fogão","Eletrodoméstico",2),

("TV","Eletrônico",3),

("Microondas","Eletrodoméstico",2),

("MicroSystem","Eletrônico",3),

("Telefone sem Fio","Eletrônico",4);

Criando relacionamento entre tabelas utilizando o DBDesigner

================================================

Tabela de Produtos x Tabela de Marcas

CREATE TABLE `Produtos` (

`cod\_produto` int NOT NULL AUTO\_INCREMENT,

`nome\_produto` varchar(50) NOT NULL,

`categoria\_produto` varchar(50) NOT NULL,

`cod\_marca` int NOT NULL,

PRIMARY KEY (`cod\_produto`)

);

CREATE TABLE `Marcas` (

`cod\_marca` int NOT NULL AUTO\_INCREMENT,

`nome\_marca` varchar(50) NOT NULL,

PRIMARY KEY (`cod\_marca`)

);

ALTER TABLE `Produtos` ADD CONSTRAINT `Produtos\_fk0` FOREIGN KEY (`cod\_marca`) REFERENCES `Marcas`(`cod\_marca`);

Inserindo registros na Tabela de Marcas

insert into itau.marcas

(nome\_marca)

values

("Walita"),

("Brastemp"),

("LG"),

("Sony"),

("Cadence");

Utilização do Having dentro do Group By

=================================

SELECT

nome\_vendedor,

sum(valor\_unitario\_produto\*quantidade\_produtos) as Total\_Vendido,

cidade\_venda

FROM

itau.vendas

GROUP BY nome\_vendedor,cidade\_venda having sum(valor\_unitario\_produto\*quantidade\_produtos)>5000;

OBS.: O having é somente utilizado quando queremos fazer um filtro de condição dentro de um agrupamento.

Utilizando o Group By por Vendedores e Cidade

====================================

SELECT

nome\_vendedor,

sum(valor\_unitario\_produto\*quantidade\_produtos) as Total\_Vendido,

cidade\_venda

FROM

itau.vendas

GROUP BY nome\_vendedor,cidade\_venda order by nome\_vendedor;

Utilizando o Group By por venda de Vendedores

=====================================

SELECT

nome\_vendedor,

sum(valor\_unitario\_produto\*quantidade\_produtos) as Total\_Vendido

FROM

itau.vendas

GROUP BY nome\_vendedor order by nome\_vendedor;

Inserindo novos registros na tabela de vendas

===================================

insert into itau.vendas

(nome\_vendedor,quantidade\_produtos,nome\_produto,cidade\_venda,valor\_unitario\_produto)

values

("Cristina",100,"Mouse","Minas Gerais",29.99),

("Ricardo Alexandre",250,"Teclado","São Paulo",35.99),

("Edison",300,"Pendrive","Minas Gerais",32.99);

Utilizando o comando group by, na quebra de registros com a condição where

============================================================

SELECT

cidade\_venda, SUM(quantidade\_produtos\*valor\_unitario\_produto) AS total\_mouse

FROM

itau.vendas

WHERE

nome\_produto = 'mouse'

GROUP BY cidade\_venda;

Utilizando o comando group by, quebra de registros por grupo

===============================================

SELECT

nome\_vendedor,

sum(valor\_unitario\_produto\*quantidade\_produtos) as Total\_Vendido,

nome\_produto,

cidade\_venda

FROM

itau.vendas

GROUP BY cidade\_venda , nome\_produto order by cidade\_venda;

Criação da tabela de vendas

======================

create table itau.vendas(

id\_venda int not null auto\_increment primary key,

nome\_vendedor varchar(50) not null,

quantidade\_produtos int not null,

nome\_produto varchar(60) not null,

cidade\_venda varchar(50) not null

);

=======================

Inserindo um novo campo na tabela de vendas

alter table itau.vendas add valor\_unitario\_produto decimal(8,2) not null;

=======================

Insert dos produtos na tabela de vendas

insert into itau.vendas

(nome\_vendedor,quantidade\_produtos,nome\_produto,cidade\_venda,valor\_unitario\_produto)

values

("Paulo",100,"Mouse","São Paulo",29.99),

("Pedro",50,"Teclado","São Paulo",35.99),

("Sandra",200,"Pendrive","Rio de Janeiro",15.75),

("Cristina",300,"Impressoras","São Paulo",450.80),

("Edison",60,"Mouse","Rio de Janeiro",25.99),

("Gustavo",200,"Monitores","São Paulo",350.80),

("Gabriela",100,"Scanner","Rio de Janeiro",280.50);

Processo de Migração de Banco de Dados = Caso Unibanco p/ Itau

====================================================

Processo de migração de dados com o select na tabela do unibanco já normalizado com os campos

insert into itau.clientes

(codigo\_cliente,nome\_cliente,cpf\_cliente,endereco\_cliente,bairro\_cliente,cep\_cliente,estado\_cliente,cidade\_cliente,agencia\_cliente,conta\_corrente\_cliente,dig\_verificador\_cliente,telefone\_cliente,email\_cliente,data\_modificacao\_cliente)

select codigo\_cliente,nome\_cliente,"atualiza",endereco\_cliente,"atualiza","00000000","XX","atualiza","0191","00000","0","000000000",email\_cliente,now() from unibanco.clientes;

Criando uma nova tabela de clientes para o processo de migração

===================================================

create table itau.clientes(

codigo\_cliente int not null,

nome\_cliente varchar(50) not null,

cpf\_cliente char(11) not null,

endereco\_cliente varchar(60) not null,

bairro\_cliente varchar(40) not null,

cep\_cliente char(9) not null,

estado\_cliente char(2) not null,

cidade\_cliente varchar(40) not null,

agencia\_cliente char(4) not null,

conta\_corrente\_cliente varchar(8) not null,

dig\_verificador\_cliente int(1) not null,

telefone\_cliente char(12) not null,

email\_cliente varchar(80) not null,

data\_modificacao\_cliente datetime,

primary key (codigo\_cliente,cpf\_cliente)

);

==================================

Uso do percentual junto ao comando like

OBS.: O % permite atuar junto com o comando like como um coringa trazendo sempre as informações

contidas dentro de uma string

Exemplos:

select \* from itau.clientes where nome\_cliente like "Antonio%";

select \* from itau.clientes where nome\_cliente like "%Carlos";

select \* from itau.clientes where nome\_cliente like "%M.%";

-- Permite criar um campo na tabela

alter table itau.clientes add endereco varchar(80) not null;

-- Permite excluir um campo na tabela

alter table itau.clientes drop endereco;

-- Permite renomear um campo na tabela

alter table itau.clientes rename column endereco to endereco\_cliente;

-- Comando utilizado para criar o database.

-- create database itau;

-- Comando utilizado para por o banco de dados em uso.

-- use itau;

-- Criação do banco de dados clientes

/\*create table itau.clientes(

codigo\_cliente int not null primary key,

nome\_cliente varchar(50) not null,

idade\_cliente int(3) not null,

email\_cliente varchar(80) not null

);\*/

insert into itau.clientes

(codigo\_cliente,nome\_cliente,idade\_cliente,email\_cliente)

values

(1,"Ricardo Alexandre",47,"ricardo.a@gama.com.br"),

(2,"Antonio Carlos",22,"antonio.c@itau.com.br"),

(3,"Rogerio Esteves",19,"rogerioe@itau.com.br"),

(4,"Renata Silva Tomaz",18,"renatas@itau.com.br");