Comandos básicos – Linguagem SQL

DDL – comandos para criar ou deletar

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

mysql -u root – Para iniciar o programa

CREATE DATABASE vendas – cria banco de dados “vendas”, por exemplo.

USE “nome do banco” – indica qual banco será utilizado.

CREATE TABLE “nome da tabela” - (atributoA INTEGER, atributoB VARCHAR(30), atributoC FLOAT); - cria a tabela e cria os atributos com os seus tipos

RENAME TABLE “nome da tabela antigo” to “nome da tabela novo”; – muda nome de tabelas já criadas, coloque: o nome antigo da tabela to nome novo.

ALTER TABLE ‘nome da tabela’ ADD CONSTRAINT ‘nome’ PRIMARY KEY (‘nome da coluna que será alterada’); - altera/edita atributos de uma determinada tabela.

ALTER TABLE ‘nomeDaTabela’ MODIFY COLUMN ‘atributo’ VARCHAR(50); - usando o MODIFY COLUMN, podemos alterar diretamente uma coluna, podendo editar o tipo, nome da coluna.

ALTER TABLE ‘nomeDaTabela’ ADD COLUMN ‘nome da coluna’ FLOAT AFTER ‘coluna’ – Insere uma coluna nova, informando o tipo e, até podendo especificar onde adicionar com AFTER ‘nome da coluna’

alter table tabela drop foreign key nomefk; - DROP serve para conseguir apagar algo da ESTRUTURA da tabela, por exemplo uma foreign KEY ou uma coluna.

ALTER TABLE ‘tabela ADD CONSTRAINT ‘fk\_pessoa’ FOREIGN KEY(atributo) REFERENCES tabela (atributo);

RENAME TABLE ‘nomeAntigo’ TO ‘nomeNovo’ – renomeia o nome da tabela, informando o nome anterior e o nome novo

DML – DATA MANIPULATION LANGUAGE – comandos para manipular dados

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

SELECT – seleciona apenas registros  
SELECT \* - seleciona tudo da tabela

SELECT \* FROM produto – seleciona tudo da tabela produto.

INSERT INTO ‘tabela’ (atributoA, atributoB) values (valorA, valor) – insere um registro na tabela. Os valores vem respectivamente aos atributos (na mesma ordem).

UPDATE – atualiza algo, porém é preciso usar o WHERE para especificar onde você quer fazer tal modificação. Ex:

COUNT(\*) – conta

SUM(VALOR) – somatória, exemplo: SELECT SUM(valor) AS SOMA\_2016 FROM duplicata WHERE vencimento >= '2016/01/01' AND vencimento <= '2016/12/31';

Somou os valores onde constam as datas

AS – apelido para a tabela temporária que vc vai criar

SELECT CONCAT('R$', FORMAT(AVG(valor), 2)) AS Media\_Valores\_2016 FROM duplicata WHERE YEAR(vencimento) = 2016;

Texto

Descrição gerada automaticamenteSELECT ‘tabela.atributo’, ‘tabela.atributo’, ‘tabela.atributo’ FROM ‘tabela primaria’ INNER JOIN ‘tabela secundaria’ ON ‘tabela primaria. atributo que vc quer’ = ‘tabela secundaria atributo que vc quer’; = Usamos para juntarmos informações entre outras tabelas.

Forma resumida: SELECT C.Cliente\_id, C.nome, T.Numero FROM Cliente AS C INNER JOIN Telefone AS T ON C.Cliente\_id = T.Cliente\_id;

SELECT ‘tabela.atributo’, ‘tabela.atributo’, ‘tabela.atributo’ FROM ‘tabela primaria’ LEFT JOIN ‘tabela secundaria’ ON ‘tabela primaria.atributo que vc quer’ = ‘tabela secundaria. atributo que vc quer’; = A junção das tabelas da esquerda (primeira que vc chamar) com o da direita (segunda). O da Esquerda sempre virá completa e irá comparar com o da direita

Texto

Descrição gerada automaticamente

SELECT ‘tabela.atributo’, ‘tabela.atributo’, ‘tabela.atributo’ FROM ‘tabela primaria’ RIGHT JOIN ‘tabela secundaria’ ON ‘tabela primaria.atributo que vc quer’ = ‘tabela secundaria. atributo que vc quer’; = A junção das tabelas da direita (primeira que vc chamar) com o da esquerda (segunda). O da Direita sempre virá completa e irá comparar com o da esquerda

Texto

Descrição gerada automaticamente