Apostila de
MySQL

Todo material desenvolvido pela Coordenadoria do Governo
Eletrônico é resultado de um processo coletivo de produção,
que se iniciou em 2001 e que é permanente.
Agradecemos a todos que colaboraram e que queiram contribuir.

CGE COORDENADORIA DO GOVERNO ELETRÔNICO
Equipe de Treinamento Técnico

Aparecido Quesada
Adriana Tosta
Eder Moura Dourado
Simone Leal dos Santos
Thyago Akira de Morais Ribeiro
Yuri Robinson de Souza

Contato
treinamento_cge@prefeitura.sp.gov.br
telecentros@prefeitura.sp.gov.br

PALÁCIO DO ANHANGABAÚ
VIADUTO DO CHÁ Nº 15
CEP 01002-000 SÃO PAULO
TEL: 3113-8938 FAX 3113-8939

Índice

Introdução

No mundo atual existem gigantescos bancos de dados gerenciando nossas vidas. Nossa conta bancária faz parte de uma coleção imensa de contas bancárias de nosso banco. Nosso Título Eleitoral ou nosso Cadastro de Pessoa Física, certamente estão armazenados em Bancos de Dados colossais. Sabemos também que quando sacamos dinheiro no caixa eletrônico de nosso banco, nosso saldo e as movimentações existentes em nossa conta bancária já estão à nossa disposição.

Nestas situações sabemos que existe uma necessidade em se realizar o armazenamento de uma série de informações que não se encontram efetivamente isoladas umas das outras, ou seja, existe uma ampla gama de dados que se referem a relacionamentos existentes entre as informações a serem manipuladas.

Para gerenciar tantos dados como os citados são utilizados os **Sistemas de Gerenciamento de Banco de Dados (SGBD)**. Sem tais sistemas o mundo atual estaria bastante diferente de como vemos hoje. O mercado financeiro tal como vemos hoje não existiria. As praticidades como compras pela Internet, cartões de débito, caixas automáticos e mais uma infinidade de exemplos não seriam possíveis.

Popularização dos Bancos de Dados

Há poucas décadas atrás os bancos de dados eram utilizados apenas por grandes instituições, e seu uso era restrito aos grandes negócios, onde simplesmente seria impossível a operação de certas indústrias ou empresas.

Com a popularização e barateamento da plataforma IBM/PC, logo surgiram programas aplicativos para algum tipo de armazenamento de dados.

Antes desses equipamentos e softwares tornarem-se acessíveis muitas vezes eram utilizadas fichas cadastrais para armazenar dados de clientes em pequenos e médios negócios (essas fichas e armários de fichários são ainda hoje utilizados em alguns locais).

Para se entender como isso funcionava vamos usar um exemplo de uma pessoa que ao visitar uma clínica médica/ odontológica fornecia seu nome, endereço, telefone, etc. Esses dados eram então anotados nessas fichas, que eram a maneira de se guardar as informações de clientes, pacientes, estudantes, peças numa oficina mecânica, etc..

Exemplo de ficha cadastral:

Nº de cadastro: 19.000/2

Nome: Fulano Beltrano Santos ..... RG Nº:165.956.401/26
Endereço: Rua Alpha, Bairro Cariru Nº 308
Cidade: Ituporanga-SP ..... Fone: (79)5555-5555

Problema encontrado: Paciente apresentava constirpação.

Quando era necessário saber o número de telefone de um cliente chamado "Raoni Guimarães Villar de Pinho" era preciso abrir o fichário, encontrar a letra R e ir olhando aos poucos onde havia o nome Raoni. Se fosse preciso saber, em uma escola, se o aluno "Artur Magno Horta de Abreu" havia repetido alguma série, era preciso verificar as fichas que continham os históricos escolares, depois procurar pelo histórico do aluno e ainda procurar nessa ficha se o aluno havia repetido alguma série. No caso de uma farmácia, saber quais produtos custavam mais que R$ 5,00 seria extremamente difícil, e nada disso era feito.

Atualmente os bancos de dados estão por toda parte, desde a farmácia da esquina até a lojinha do posto de gasolina.

Sistema de Gerenciamento de Banco de Dados?

Um SGBD - Sistema de Gerenciamento de Banco de Dados é uma coleção de programas que permitem ao usuário definir, construir e manipular Bancos de Dados para as mais diversas finalidades.

Um SGBD deve possuir as seguintes características:

CONTROLE DE REDUNDÂNCIAS

A redundância consiste no armazenamento de uma mesma informação em locais diferentes, provocando inconsistências. Em um Banco de Dados as informações só se encontram armazenadas em um único local, não existindo duplicação descontrolada dos dados. Quando existem replicações dos dados, estas são decorrentes do processo de armazenagem típica do ambiente Cliente-Servidor, totalmente sob controle do Banco de Dados.

COMPARTILHAMENTO DOS DADOS

O SGBD deve incluir software de controle de concorrência ao acesso dos dados, garantindo em qualquer tipo de situação a escrita/leitura de dados sem erros.

CONTROLE DE ACESSO

O SGDB deve dispor de recursos que possibilitem selecionar a autoridade de cada usuário. Assim um usuário poderá realizar qualquer tipo de acesso, outros poderão ler alguns dados e atualizar outros e outros ainda poderão somente acessar um conjunto restrito de dados para escrita e leitura.

INTERFACEAMENTO

Um Banco de Dados deverá disponibilizar formas de acesso gráfico, em linguagem natural, em SQL ou ainda via menus de acesso, não sendo uma "caixa-preta" somente sendo passível de ser acessada por aplicações.

ESQUEMATIZAÇÃO

Um Banco de Dados deverá fornecer mecanismos que possibilitem a compreensão do relacionamento existentes entre as tabelas e de sua eventual manutenção.

CONTROLE DE INTEGRIDADE

Um Banco de Dados deverá impedir que aplicações ou acessos pelas interfaces possam comprometer a integridade dos dados.

CÓPIAS DE SEGURANÇA

O SGBD deverá apresentar facilidade para recuperar falhas de hardware e software, através da existência de arquivos de "pré-imagem" ou de outros recursos automáticos, exigindo minimamente a intervenção de pessoal técnico.

Em certos casos pode ocorrer de um SGBD não obedecer uma ou outra regra das vistas acima, mas ainda assim continuar sendo considerado um SGBD. Porém alguns "Bancos de Dados" atualmente comercializados não são SGBD reais, justamente por não atenderem algumas dessas características.

Existem vários tipos de bancos de dados (hierárquico, orientado ao objeto, em redes), nós utilizaremos nesse curso um SGBD Relacional, o MySQL.

Exemplo de uma tabela de um Banco de Dados Relacional:

Nome	*RG	CIC	Depto.	RG Supervisor	Salário
João Luiz	10101010	11111111	1	NULO	3000
Fernando	20202020	22222222	2	10101010	2500
Ricardo	30303030	33333333	2	10101010	2300
Jorge	40404040	44444444	2	20202020	4200
Renato	50505050	55555555	3	20202020	1300

Quem utiliza os Banco de Dados?

USUÁRIOS

Para um grande banco de dados, existe um grande número de pessoas envolvidas, desde o projeto, uso até manutenção.

ADMINISTRADOR DE BANCO DE DADOS (DBA)

Em um ambiente de banco de dados, o recurso primário é o banco de dados por si só e o recurso secundário é o SGBD e os

softwares relacionados. A administração destes recursos cabe ao Administrador de Banco de Dados, o qual é responsável pela autorização de acesso ao banco de dados e pela coordenação e monitoração de seu uso.

PROJETISTA DE BANCO DE DADOS

O Projetista de Banco de Dados é responsável pela identificação dos dados que devem ser armazenados no banco de dados, escolhendo a estrutura correta para representar e armazenar dados. Muitas vezes, os projetistas de banco de dados atuam como "staff" do DBA, assumindo outras responsabilidades após a construção do banco de dados. É função do projetista também avaliar as necessidades de cada grupo de usuários.

USUÁRIOS FINAIS

Existem basicamente três categorias de usuários finais que são os usuários finais do banco de dados, fazendo consultas, atualizações e gerando documentos:

- usuários casuais: acessam o banco de dados casualmente, mas que podem necessitar de diferentes informações a cada acesso; utilizam sofisticadas linguagens de consulta para especificar suas necessidades;
- usuários novatos ou paramétricos: utilizam porções pré-definidas do banco de dados, utilizando consultas preestabelecidas que já foram exaustivamente testadas;
- usuários sofisticados: são usuários que estão familiarizados com o SGBD e realizam consultas complexas.

ANALISTAS DE SISTEMAS E PROGRAMADORES DE APLICAÇÕES

Os analistas determinam os requisitos dos usuários finais e desenvolvem especificações para transações que atendam estes requisitos, e os programadores implementam estas especificações

como programas, testando, depurando, documentando e dando manutenção no mesmo. É importante que, tanto analistas quanto programadores, estejam a par dos recursos oferecidos pelo SGBD.

Banco de Dados Relacional

O Modelo de Dados relacional representa os dados contidos em um Banco de Dados através de relações. Cada relação é uma tabela.

Veja o exemplo:

Tabela DEPARTAMENTO		
Nome	Número*	RG Gerente
Contabilidade	*1*	*10101010*
Engenharia Civil	*2*	*30303030*
Engenharia Mecânica	*3*	*20202020*
Engenharia Elétrica	*4*	*40202020*

Há 3 campos na tabela DEPARTAMENTO (nome, número e RG Gerente), sendo que o campo Número é chave primária (impede que existam 2 registros iguais no banco de dados). A primeira linha possuí valores "Contabilidade", "1" e "10101010".

Os nomes das tabelas e dos campos são de fundamental importância para nossa compreensão entre o que estamos armazenando, onde estamos armazenando e qual a relação existente entre os dados armazenados.

Cada registro de nossa relação será chamada de linha e cada coluna de nossa relação será chamada de *ATRIBUTO*.

O esquema de uma relação, nada mais são que os campos (colunas) existentes em uma tabela. Já a instância da relação

consiste no conjunto de valores que cada atributo assume em um determinado instante. Portanto, os dados armazenados no Banco de Dados, são formados pelas instâncias das relações.

As relações não podem ser duplicadas (não podem existir dois estados do Pará, no conjunto de estados brasileiros, por exemplo), a ordem de entrada de dados no Banco de Dados não deverá ter qualquer importância para as relações, no que concerne ao seu tratamento.

Chamaremos de Chave Primária ao Atributo que definir um registro, dentre uma coleção de registros.

TABELA Cidade	TABELA Estado
*CidCodi**	*EstCodi**
CidNome	*EstNome*
*EstCod**	

CidCodi e EstCodi, são chaves primárias respectivamente das tabelas Cidade e Estado, enquanto EstCodi é chave estrangeira na tabela de cidades. É precisamente por este campo (atributo, ou coluna), que será estabelecida a relação entre as tabelas Cidade->Estado.

EXERCÍCIOS:

1.Defina o seguinte termo: Sistema de Gerenciamento de Banco de Dados.

2.Quais as vantagens da utilização de um sistema de bancos de dados?

3.Descreva as características gerais de um sistema gerenciador de base de dados.

4.Descrever o modelo relacional de dados.

5. Em se tratando de Bancos de Dados Relacionais, qual a função da Chave Primária em uma tabela?

O MySQL

O MySQL foi originalmente desenvolvido pela empresa sueca TCX, que necessitava de um servidor de banco de dados que operasse com grandes escalas de dados rapidamente sem exigir caríssimas plataformas de hardware. No início eles utilizavam o mSQL, mas depois de alguns testes chegaram à conclusão que o mSQL não era rápido nem flexível o suficiente para as necessidades existentes.

Características do MySQL

As principais características do MySQL são:

- O servidor de banco de dados MySQL é extremamente rápido, confiável, e fácil de usar. O Servidor MySQL também tem um conjunto de recursos muito práticos desenvolvidos com a cooperação dos próprios usuários.

- O Servidor MySQL foi desenvolvido originalmente para lidar com bancos de dados muito grandes de maneira muito mais rápida que as soluções existentes, e tem sido usado em ambientes de produção de alta demanda por vários anos de maneira bem sucedida. Apesar de estar em constante desenvolvimento, o Servidor MySQL oferece hoje um rico e proveitoso conjunto de funções. A conectividade, velocidade, e segurança fazem com que o MySQL seja altamente adaptável para acessar bancos de dados na Internet.

- MySQL é um Sistema de Gerenciamento de Bancos de Dados relacional.

Um banco de dados relacional armazena dados em tabelas separadas em vez de colocar todos os dados em um só local. Isso

proporciona velocidade e flexibilidade. SQL é a linguagem padrão mais comum usada para acessar bancos de dados e é definida pelo Padrão ANSI/ISO SQL. (O padrão SQL vem evoluindo desde 1986 e existem diversas versões disponibilizadas)..

- MySQL é um software cujo código fonte é aberto.

Código fonte aberto significa dizer que é possível para qualquer um usar e modificar o programa. Qualquer pessoa pode fazer download do MySQL pela Internet e usá-lo sem pagar nada (o MySQL só é cobrado em alguns poucos casos). Se você quiser pode estudar o código fonte e alterá-lo para adequá-lo às suas necessidades. O MySQL usa a GPL (GNU General Public License - Licenca Pública Geral GNU) http://www.fsf.org/licenses, para definir o que você pode e não pode fazer com o software em diferentes situações. Se voce sentir desconforto com a GPL ou precisar embutir o MySQL em uma aplicação comercial você pode adquirir a versão comercial licenciada.

- O MySQL suporta diferentes plataformas, tais como: Windows, Linux, FreeBSD, Unix, entre outros.

- O MySQL possuí suporte a múltiplos processadores.

O Programa de Banco de Dados MySQL é um sistema cliente/ servidor que consiste de um servidor SQL multi-tarefa que suporta acessos diferentes, diversos programas clientes e bibliotecas, ferramentas administrativas e diversas interfaces de programação (API's).

SQL

INTRODUÇÃO

Para se utilizar, administrar, e trabalhar com um banco de dados é utilizada uma linguagem padrão, que a maior parte dos SGBD aceitam. Essa linguagem é a SQL (Structured Query Language-Linguagem de Consulta Estruturada).

A SQL é um conjunto de declarações que são utilizadas para acessar os dados utilizando gerenciadores de banco de dados. Apesar de nem todos os gerenciadores utilizarem a SQL a maior parte deles aceitam suas declarações.

A SQL pode ser utilizada para todas as atividades relativas a um banco de dados, podendo ser utilizada pelo administrador de sistemas, pelo DBA, por programadores, sistemas de suporte à tomada de decisões e outros usuários finais.

É através dela que você irá criar tabelas, inserir dados, e utilizar o seu banco de dados.

SQL para manipulação de bancos de dados MySQL

A SQL possui comandos que são utilizados para manipular os bancos de dados, as tabelas e os registros existentes. Veja abaixo como utilizá-los.

COMANDO CREATE

Este comando permite a criação de bancos de dados ou de tabelas num banco de dados.

Sintaxe:
CREATE DATABASE < nome_db >;
onde:
nome_db: indica o nome do Banco de Dados a ser criado.

Exemplo:
CREATE DATABASE curso;

Sintaxe:
CREATE TABLE < nome_tabela > (
  nome_atributo1 < tipo > [NOT NULL],
  nome_atributo2 < tipo > [NOT NULL],

```
      .....
    nome_atributoN < tipo > [ NOT NULL ]
    PRIMARY KEY(nome_atributo)
) ;
```

onde:

nome_tabela: indica o nome da tabela a ser criada.

nome_atributo: indica o nome do campo a ser criado na tabela.

tipo: indica a definição do tipo de atributo (integer(n), char(n), ...).

PRIMARY KEY: esse é o campo utilizado para que não exista na tabela dois registros iguais. Ele mantém a integridade do banco de dados. Caso você tente inserir num banco de dados um registro com uma PRIMARY KEY já existente ele emitirá uma mensagem de erro e impedirá que o registro seja inserido.

Exemplo:

```
CREATE table alunos(
  codigo int NOT NULL AUTO_INCREMENT,
  nome VARCHAR(20) NOT NULL ,
  telefone CHAR(8) NOT NULL,
  PRIMARY KEY(codigo)
);
```

Criação de uma tabela em um banco de dados à partir da tabela dada (o asterísco determina qual campo é a chave primária):

Tabela estudantes		
nome	*numerocadastro	turma
Vinícius Bomfim	*101*	*1010*
Sophia Oliveira	*102*	*3030*
Angélica de Souza	*103*	*2020*
Antunes da Silva Porto	*303*	*2365*
Adriana Ramos	*132*	*4265*
Flávia Dias	*456*	*6574*

```
CREATE TABLE estudantes(
  numerocadastro int NOT NULL auto_increment,
  nome varchar(35) not null,
  turma int,
  primary key(numerocadastro)
);
```

COMANDO DROP

Este comando elimina a definição da tabela, seus dados e referências ou um banco de dados existente:

Sintaxe:

```
DROP TABLE  < nome_tabela > ;
DROP DATABASE <nome_banco_de_dados>;
```

Exemplo:

```
DROP TABLE alunos;
DROP DATABASE curso;
DROP TABLE estudantes;
```

COMANDO ALTER

Este comando permite inserir/eliminar atributos nas tabelas já existentes.

Sintaxe:

```
ALTER  TABLE  < nome_tabela >  ADD / DROP (
  nome_atributo1      < tipo >        [ NOT NULL ],
  nome_atributoN      < tipo >        [ NOT NULL ]
) ;
```

Manipulando dados das tabelas

COMANDO SELECT

Realiza uma seleção de informações existentes nas tabelas.
Sintaxe básica:
SELECT [DISTINCT] expressao [AS nome-atributo]
[FROM from-lista]
[WHERE condicao]
[ORDER BY attr_name1 [ASC | DESC]]

onde:
DISTINCT: Elimina linhas duplicadas na seleção.

expressao: Define os dados que queremos selecionar, normalmente uma ou mais colunas de uma tabela que está em from-lista.

AS nome-atributo: Define um alias (apelido) para o nome da coluna.

FROM: Lista das tabelas onde a pesquisa será feita.

WHERE: Condição para que um registro seja selecionado.

ORDER BY: Critério para ordenação dos registros selecionados. Utilizando ASC a ordem será crescente, utilizando DESC a ordem será decrescente.

Where como base das Restrição de linhas.

A cláusula "where" restringe a seleção de dados, de acordo com seu argumento. Contém a condição que as linhas devem obedecer a fim de serem listadas.

Ela pode comparar valores em colunas, literais, expressões aritméticas ou funções.

A seguir apresentamos operadores lógicos e complementares a serem utilizados nas expressões apresentadas em where.

Operadores lógicos

operador	significado
=	igual a
>	maior que
>=	maior que ou igual a
<	menor que
<=	menor que ou igual a

Exemplos:

SELECT cidade, estado FROM brasil WHERE populacao > 100000;

Selecionará os campos cidade e estado da tabela brasil de todos os registros que tiverem valor maior que 100.000 no campo populacao.

SELECT * FROM cidadao ORDER BY nome DESC;

Selecionará todos os campos da tabela cidadao e utilizará ordenação decrescente na seleção.

Levando em conta a tabela funcionarios abaixo, veja a utilização da cláusula SELECT.

Nome	RG	CIC	Depto.	RG Supervisor	Salário
João Luiz	10101010	11111111	1	NULO	3000
Fernando	20202020	22222222	2	10101010	2500
Ricardo	30303030	33333333	2	10101010	2300
Jorge	40404040	44444444	2	20202020	4200
Renato	50505050	55555555	3	20202020	1300
Maurício Matos	14654546	55544444	4	30303030	1450
Marlene Dias	43543463	40000000	2	20202020	8000
Antônio Andrade	12544354	50400000	3	50303030	8200
Daniela	87354546	55544468	4	30303030	1350
Jorge Ricardo	43543763	40004800	2	20202020	5100
Maria Antônia	12544754	50404500	3	50303030	800

Seleções:

-Selecionar quantas pessoas existem cadastradas:

mysql>SELECT COUNT(*) FROM funcionarios;

-Selecionar quantos funcionários existem no departamento 3:

mysql>SELECT COUNT(*) FROM funcionarios WHERE depto=3;

-Selecionar o nome e o rg dos funcionários que ganham mais que 3000 reais.

mysql>SELECT nome, rg FROM funcionarios WHERE salario>3000;

COMANDO INSERT

Adiciona um ou vários registros a uma tabela.
Sintaxe básica:
INSERT INTO destino [(campo1[, campo2[, ...]])]
VALUES (valor1[, valor2[, ...])

A instrução INSERT INTO possuí as partes abaixo:
Destino- O nome da tabela em que os registros devem ser anexados.

campo1, campo2 - Nomes dos campos aos quais os dados devem ser inseridos.

valor1, valor2 - Valores para inserir nos campos específicados do novo registro. Cada valor é inserido no campo que corresponde à posição do valor na lista: Valor1 é inserido no campo1 do novo registro, valor2 no campo2 e assim por diante. Você deve separar os valores com uma vírgula e colocar os campos de textos entre aspas (" ").

Seguindo o exemplo da tabela funcionarios, veja a inclusão dos 2 primeiros registros:

Nome	RG	CIC	Depto.	RG Supervisor	Salário
João Luiz	10101010	11111111	1	NULO	3000
Fernando	20202020	22222222	2	10101010	2500

mysql>INSERT INTO funcionarios VALUES('joao luiz',10101010,11111111,1,'',3000);

mysql>INSERT INTO funcionarios VALUES('Fernando',20202020,22222222,2,10101010,2500);

COMANDO UPDATE

O comando UPDATE altera os valores de alguns campos de uma tabela especificada, com base em critérios específicos.

Sintaxe:

UPDATE tabela
SET campo1 = valornovo, ...
WHERE critério;

Onde:

tabela: O nome da tabela onde você quer modificar os dados.

valornovo: Uma expressão que determina o valor a ser inserido no campo do registro que será atualizado.

critério: Uma expressão que determina quais registros devem ser atualizados. Só os registros que satisfazem a expressão são atualizados.

O comando UPDATE é bastante útil quando você quer alterar muitos registros ou quando os registros que você quer alterar estão em várias tabelas. Você pode alterar vários campos ao mesmo tempo.

Utilizando a cláusula UPDATE é possível alterar os registros

da tabela funcionários, para que os funcionarios que integram o depto 3 passem a pertencer ao depto 5:

```
mysql>UPDATE funcionarios SET depto=5 WHERE depto=3;
```

Caso fosse necessário dar um aumento de 20% de salário aos funcionários que ganham menos de 3000 reais o comando seria o seguinte:

```
mysql>UPDATE funcionarios SET salario=salario+salario*0.2
WHERE salario<3000;
```

COMANDO DELETE

Remove registros de uma ou mais tabelas listadas na cláusula FROM que satisfazem a cláusula WHERE.

Sintaxe:

```
DELETE
FROM tabela
WHERE critério
```

onde:

tabela: O nome da tabela de onde os registros são excluídos.

critério: Uma expressão que determina qual registro deve ser excluído.

O comando DELETE exclui registros inteiros e não apenas dados em campos específicos. Se você quiser excluir valores de um campo específico, use o comando UPDATE que mude os valores dos campos para NULL.

Após remover os registros usando uma consulta DELETE você não poderá desfazer a operação.

Fazendo a operação:

mysql>DELETE FROM funcionarios WHERE salario>7999;

a tabela ficaria assim após o comando:

Nome	RG	CIC	Depto	RG Supervisor	Salário
João Luiz	10101010	11111111	1	NULO	3000
Fernando	20202020	22222222	2	10101010	2500
Ricardo	30303030	33333333	2	10101010	2300
Jorge	40404040	44444444	2	20202020	4200
Renato	50505050	55555555	3	20202020	1300
Maurício Matos	14654546	55544444	4	30303030	1450
Daniela	87354546	55544468	4	30303030	1350
Jorge Ricardo	43543763	40004800	2	20202020	5100
Maria Antônia	12544754	50404500	3	50303030	800

Dois registros seriam removidos, pois obedeceriam a regra declarada com a opção WHERE.

Demais Operadores	
Operador	Significado
between ... and ...	entre dois valores (inclusive)
in (....)	lista de valores
like	com um padrao de caracteres
is null	é um valor nulo

Exemplos:

```
SELECT  EMPNOME, EMPSALA
    FROM  EMP
    WHERE  EMPSALA BETWEEN  500  AND  1000;


SELECT  EMPNOME,  DEPNUME
    FROM  EMP
    WHERE  DEPNUME  IN  (10,30);
 SELECT  EMPNOME, EMPSERV
```

```
    FROM  EMP
    WHERE  EMPNOME  LIKE 'F%';

SELECT EMPNOME, EMPSERV
    FROM  EMP
    WHERE  EMPCOMI  IS  NULL;
```

O símbolo "%" pode ser usado para construir a pesquisa ("%" = qualquer sequência de nenhum até vários caracteres).

OUTROS OPERADORES:

Operador	descrição
<>!	*diferente*
not nome_coluna =	*diferente da coluna*
not nome_coluna >	*não maior que*
not between	*não entre dois valores informados*
not in	*não existente numa dada lista de valores*
not like	*diferente do padrao de caracteres informado*
is not null	*não é um valor nulo*

FUNÇÕES DE AGRUPAMENTO:

Funções Agregadas (ou de Agrupamento)	
função	retorno
avg(n)	*média do valor n, ignorando nulos*
count(expr)	*vezes que o número da expr avalia para algo nao nulo*
max(expr)	*maior valor da expr*
min(expr)	*menor valor da expr*
sum(n)	*soma dos valores de n, ignorando nulos*

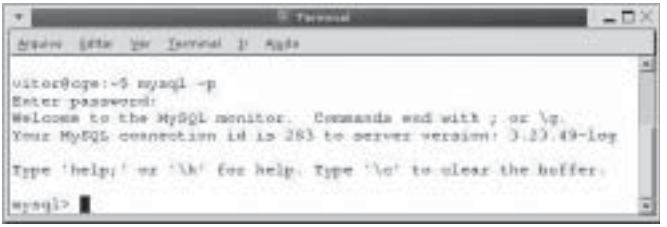
Administração do MySQL

Existem duas maneiras para administrar os bancos de dados criados no MySQL.

1. MODO TEXTO

Você deve abrir um terminal e digitar o seguinte comando: **mysql -u <usuario> -p**

Em seguida digite a senha do usuário que você indicou no comando. Caso você digite apenas mysql -p a senha pedida será a do usuário root, administrador do banco de dados. Veja a figura abaixo.



A opção **-p** indica que a senha definida para o root do MySQL deverá ser digitada obrigatoriamente e a opção **-u** é utilizada identificar o usuário.

Para ver uma lista de opções fornecidas pelo MySQL, invoque-o com -help:

shell> mysql --help

Prompt:
mysql>

O prompt diz que você está pronto para entrar com os comandos.

Algumas instalações do MySQL permitem aos usuários conectar com o servidor e continuar como anfitrião local. Se este é o caso em sua máquina, você deveria ser capaz de conectar com o servidor ao chamar o MySQL sem quaisquer opções:

```
shell> mysql
```

Depois que está conectado, você pode desconectar a qualquer momento, é só digitar QUIT no mysql> prompt:

```
mysql> QUIT
Bye
```

Você também pode desconectar por control-D.

Na maioria dos exemplos nas seguintes seções, assumem que você está conectado ao servidor.
Isto é indicado por: mysql> prompt.

Entrando e consultando:

Neste ponto, é mais importante descobrir como emitir consultas de como criar tabelas, carregar e recuperar dados. Esta seção descreve os princípios básicos de como entrar com os comandos, usando várias perguntas.
Um comando simples pergunta ao servidor o número de sua versão e a data corrente.

```
mysql> SELECT VERSION(), CURRENT_DATE;
```

Esta consulta demonstra várias coisas sobre MySQL.
O ponto-e-vírgula nem sempre é necessário, há algumas exceções. O comando "quit" é um deles.

Utilizando outra consulta em que se pode demonstrar o uso do MySQL como uma simples calculadora:

```
mysql> SELECT SIN(PI()/4), (4+1)*5;
```

Os comandos mostrados têm estado em declarações de linhas únicas e curtas.

Você pode entrar com declarações múltiplas em uma única linha.

Somente termine cada uma com um ponto-e-vírgula.

```
mysql> SELECT VERSION(); SELECT NOW();
```

Um comando dado todo em uma única linha, assim como comandos compridos que requerem várias linhas, não têm nenhum problema. O MySQL determina que sua declaração termina por um ponto-e-vírgula, e não o fim de uma linha.

Aqui é uma simples declaração de linha múltipla:

```
mysql> SELECT
    -> USER()
    -> ,
     -> CURRENT_DATE;
```

Neste exemplo, note como o prompt muda de mysql> para -> depois que entra com a pergunta na primeira linha de uma linha múltipla. O MySQL indica que não tem uma declaração completa e fica esperando o resto da declaração.

Veja exemplos de como resolver alguns dos problemas mais comuns do MySQL.

Algum dos exemplos usam a tabela compras, insira dados com os preços de cada artigo (número de item) de cada vendedor. Supondo que cada vendedor tem um preço fixo por artigo, então (item, vendedor) é uma chave primária dos registros.

Você pode criar a tabela de exemplo como:

```
CREATE TABLE compras (
artigo INT(4) UNSIGNED ZEROFILL DEFAULT '0000' NOT NULL,
 vendedor  CHAR(20)              DEFAULT ''     NOT NULL,
 preco   DOUBLE(16,2)            DEFAULT '0.00' NOT NULL,
 PRIMARY KEY(artigo, vendedor));

INSERT INTO compras VALUES
(1,'A',3.45),(1,'B',3.99),(2,'A',10.99),(3,'B',1.45),(3,'C',1.69),
(3,'D',1.25),(4,'D',19.95);
```

Assim os dados de exemplo estarão:

SELECT * FROM compras;

artigo	vendedor	preco
1	A	3,45
1	B	3,99
2	A	10,99
3	B	1,45
3	C	1,69
3	D	1,25
4	D	19,95

Para se ter o valor máximo de uma coluna:

mysql>SELECT MAX(artigo) AS artigo FROM compras;

No MySQL (ainda não faz uma sub-seleção) somente faz isto em dois passos:

1. Obtem o valor máximo e avalia a tabela com uma declaração SELECT.
2. Usando este valor compila a pergunta real:

```
SELECT artigo, vendedor, preco
FROM   compras
WHERE  prico=19.95
```

Para selecionar o valor máximo da coluna: por grupo e por valores

```
SELECT artigo, MAX(preco) AS preco
FROM   compras
GROUP BY artigo
```

```
+---------+-------+
| artigo  |preco |
+---------+-------+
|  0001 |  3.99 |
|  0002 |10.99 |
|  0003 |  1.69 |
|  0004 |19.95 |
+---------+-------+
```

Comandos importantes no MySQL

Ainda no prompt do MySQL há comandos importantes na utilização do sistema.

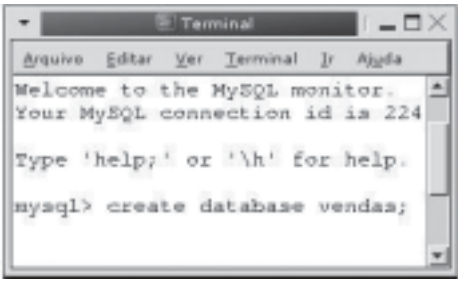
mysql>show databases;
mostra os bancos de dados existentes no MySQL.

mysql>use <bancodedados>;
seleciona <bancodedados> pra ser utilizado.

mysql>show tables;
exibe as tabelas existentes no banco de dados selecionado.

mysql>desc <tabela>
exibe uma descrição da <tabela>.

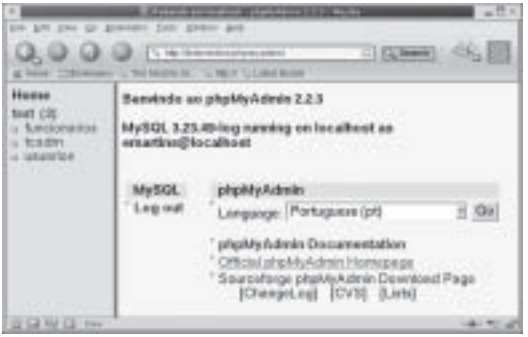
Para criar um banco de dados chamado **vendas**, você deve digitar o comando: **create database vendas**;



Utilize os comando SQL vistos anteriormente para manipular os banco de dados, sempre utilizando o ponto-e-vírgula no final de cada comando.

2.A FERRAMENTA PHPMYADMIN

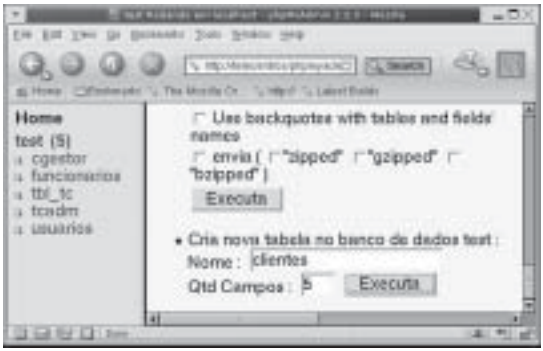
Você deve abrir o Mozilla e digitar o seguinte endereço: http://localhost/phpmyadmin



Para criar a tabela **clientes** no banco de dados **test**, você deve seguir o roteiro abaixo:

Roteiro:

1.Escolha a linguagem Portuguese (pt) e clique no botão Go.
2.Clique em test, no lado esquerdo da tela.
3.No campo "Nome" da opção "Cria nova tabela no banco de dados test:", digite: cliente
4.No campo "Qtd Campos:", digite: 5. Veja a figura abaixo:



5.Clique no botão Executa. Veja a figura abaixo:

6.Na coluna "Campo", digite os nomes dos campos.

7.Na coluna "Tipo", escolha os tipos de campos desejados.

8.Na coluna "Tamanho/Set*", digite os tamanhos dos campos.

9.Na coluna "Primary", escolha o campo que será a chave primária da sua tabela.

10.Clique no botão **Salva.**

EXERCÍCIOS:

1- Utilizando os dados abaixo, crie uma tabela funcionarios dentro de seu banco de dados, e realize as tarefas abaixo:

Nome	*RG	CIC	Depto	RG Supervisor	Salário
João Luiz	10101010	11111111	1	NULO	3000
Fernando	20202020	22222222	2	10101010	2500
Ricardo	30303030	33333333	2	10101010	2300
Jorge	40404040	44444444	2	20202020	4200
Renato	50505050	55555555	3	20202020	1300
Maurício Matos	14654546	55544444	4	30303030	1450
Marlene Dias	43543463	40000000	2	20202020	8000
Antônio Andrade	12544354	50400000	3	50303030	6200
Daniela	87354546	55544468	4	30303030	1350
Jorge Ricardo	43543763	40004800	2	20202020	5100
Maria Antônia	12544754	50404500	3	50303030	800

a- Insira os dados desses funcionários na tabela.

b- Crie uma instrução SQL para que o funcionário Jorge Ricardo tenha um aumento de 50% em seu salário.

c- Crie uma instrução SQL que mostre quantos funcionários ganham mais que R$ 1500.

d- Crie uma instrução SQL que mostra o nome e o salário dos funcionários que trabalham no depto 2.

e- Atualize o banco de dados para funcionarios que ganham mais que R$ 2000, descontando 15% de seus salários.

2- Dado o comando abaixo faça o desenho de como ficará essa tabela no banco de dados.

```
CREATE TABLE armazem (
item INT(6)  DEFAULT  NOT NULL AUTO_INCREMENT,
vendedor  CHAR(20)            NOT NULL,
preco  DOUBLE(16,2)           NOT NULL,
fornecedor INT(6)             NOT NULL,
PRIMARY KEY(artigo));
```

3- Dada a tabela funcionarios e o comando abaixo diga qual será o resultado da operação.

```
mysql>DELETE FROM funcionarios WHERE depto=1;
```

funcionarios

Nome	*RG	CIC	Depto	RG Supervisor	Salário
João Luiz	10101010	11111111	1	NULO	3.000,00
Fernando	20202020	22222222	2	10101010	2.500,00
Ricardo	30303030	33333333	2	10101010	2.300,00
Jorge	40404040	44444444	2	20202020	4.200,00
Renato	50505050	55555555	3	20202020	1.300,00

4- Utilizando a mesma tabela do exercício 3 crie uma instrução que apaga os funcionários que ganham menos de R$ 2000.

5- Apague todos as linhas e em seguida a tabela funcionarios.

Programas aplicativos do MySQL

A lista abaixo descreve resumidamente os programas do MySQL:

msql2mysql

Um script shell que converte programas mSQL para MySQL. Ele não lida com todos os casos, mas ele fornece um bom inicio para a conversão.

mysql

A ferramenta de linha de comando para a entrada de consultas interativamente ou a execução de consultas a partir de um arquivo no modo batch.

mysqlaccess

Um script que verifica os privilégios de acesso para uma combinação de nome de máquina, usuário e banco de dados.

mysqladmin

Utilitário para realizar operações administrativas, tais como criação ou remoção de bancos de dados, recarga das tabelas de permissões, descarga de tabelas em disco e reabertura dos arquivos log. mysqladmin também pode ser usado para exibir informações de versão, processos e estado do servidor.

mysqlbinlog

Utilitário para leitura das consultas de um log binário. Pode ser usado para recuperação de falhas com um backup antigo.

mysqldump

Descarrega um banco de dados MySQL em um arquivo como instruções SQL ou como arquivo texto separado por tabulação. Versão aprimorada do freeware escrito originalmente por Igor Romanenko.

mysqlimport

Importa arquivos texto em suas tabelas respectivas utilizando LOAD DATA INFILE.

mysqlshow

Exibe informações sobre bancos de dados, tabelas, colunas e índices.

replace

Um programa utilitário que é usado pelo msql2mysql, mas que também pode ser aplicável mais genericamente. replace altera conjuntos de caracteres. Utiliza uma máquina de estado finito para comparar strings maiores primeiro. Pode ser usada para trocar conjuntos de caracteres. Por exemplo, este comando troca a e b nos arquivos dados:

```
shell> replace a b b a -- arquivo1 arquivo2 ...
```

Tipos de campos suportados no MySQL

Os tipos de campos suportados são os tipos de dados que podem ser armazenados nos campos do banco de dados.

TINYINT

Um inteiro muito pequeno. A faixa deste inteiro com sinal é de -128 até 127. A faixa sem sinal é de 0 até 255.

BIT
BOOL
BOOLEAN

Na versão 4.0 e anteriores, este são sinônimos para TINYINT(1). A partir da versão 4.1.0, a exigência de armazenamento é um único bit (mais a exigência atual para NULL se a coluna não é especificada como NOT NULL). O sinônimo BOOLEAN foi adicionado na versão 4.1.0. Um tipo boolean verdadeiro será introduzido de acordo com o SQL-99.

SMALLINT

Um inteiro pequeno. A faixa do inteiro com sinal é de -32768 até 32767. A faixa sem sinal é de 0 a 65535.

MEDIUMINT

Um inteiro de tamanho médio. A faica com sinal é de -8388608 a 8388607. A faixa sem sinal é de 0 to 16777215.

INT

Um inteiro de tamanho normal. A faixa com sinal é de -2147483648 a 2147483647. A faixa sem sinal é de 0 a 4294967295.

INTEGER

Este é um sinônimo para INT.

BIGINT

Um inteiro grande. A faixa com sinal é de -9223372036854775808 a 9223372036854775807. A faixa sem sinal é de 0 a 18446744073709551615. Existem algumas coisas sobre campos BIGINT sobre as quias você deve estar ciente:

• Todas as operações aritiméticas são feitas usando valores BIGINT ou DOUBLE com sinal, não devemos utilizar inteiros sem sinal maiores que 9223372036854775807 (63 bits) exceto com funções ded bit! Se você fizer isto, alguns dos últimos digitos no resultado podem estar errados por causa de erros de arredondamento na conversão de BIGINT para DOUBLE. O MySQL

4.0 pode tratar BIGINT nos seguintes casos:

• Usar inteiros para armazenar grandes valores sem sinais em uma coluna BIGINT.

• Em MIN(big_int_column) e MAX(big_int_column).

• Quando usar operadores (+, -, *, etc.) onde ambos os operandos são inteiros.

• Você pode armazenar valores inteiro exatos em um campo BIGINT armazenando-os como string, como ocorre nestes casos não haverá nenhuma representação intermediaria dupla.

• `-', `+', e `*' serão utilizados em cálculos aritiméticos BIGINT quando ambos os argumentos forem valores do tipo INTEGER! Isto significa que se você multilicar dois inteiros grandes (ou obter resultados de funções que retornam inteiros) você pode obter resultados inesperados quando o resultado for maior que 9223372036854775807.

FLOAT(precisão)

Um número de ponto flutuante. Não pode ser sem sinal. precisão pode ser <=24 para um número de ponto flutuante de precisão simples e entre 25 e 53 para um número de ponto flutuante de dupla-precisão. Estes tipos são como os tipos FLOAT e DOUBLE descritos logo abaixo. FLOAT(X) tem a mesma faixa que os tipos correspondentes FLOAT e DOUBLE, mas o tamanho do display e número de casas decimais é indefinido. Na versão 3.23 do MySQL, este é um verdadeiro valor de ponto flutuante. Em versões anteriores, FLOAT(precisão) sempre tem 2 casas decimais. Note que o uso de FLOAT pode trazer alguns problemas inesperados como nos cálculos já que em MySQL todos são feitos com dupla-precisão. FLOAT[(M,D)] [UNSIGNED] [ZEROFILL]

Um numero de ponto flutuante pequeno (precisão simples). Os valores permitidos estão entre -3.402823466E+38 e -1.175494351E-38, 0 e entre 1.175494351E-38 e 3.402823466E+38. Se UNSIGNED for especificado, valores negativos não são permitidos. O M é a largura do display e o D é o número de casas decimais. FLOAT sem um argumento ou FLOAT(X) onde X <=24 tende a um número de ponto flutuante de precisão simples.

DOUBLE[(M,D)]

Um número de ponto flutuante de tamanho normal (dupla-precisão). Valores permitidos estão entre -1.7976931348623157E+308 e -2.2250738585072014E-308, 0 e entre 2.2250738585072014E-308 e 1.7976931348623157E+308. Se UNSIGNED for especificado, valores negativos não são permitidos. O M é a largura do display e o D é número de casa decimais. DOUBLE sem argumento ou FLOAT(X) onde 25 <= X <= 53 são números de ponto flutuante de dupla-precisão.

DOUBLE PRECISION
REAL[(M,D)]
Estes são sinônimos para DOUBLE.

DECIMAL[(M[,D])]

Um número de ponto flutuante não empacotado. Se comporta como um campo CHAR: ``não empacotado" significa que o número é armazenado como uma string, usando um caracter para cada digito do valor. O ponto decimal e, para números negativos, o sinal de menos (`-'), não são contados em M (mas é reservado espaço para isto). Se D for 0, os valores não terão ponto decimal ou parte fracionária. A faixa máxima do valor DECIMAL é a mesma do DOUBLE, mas a faixa atual para um campo DECIMAL dado pode ser limitado pela escolha de M e D. Se UNSIGNED é especificado, valores negativos não são permitidos. Se D não for definido será considerado como 0. Se M não for definido é considerado como 10. Note que antes da versão 3.23 do MySQL o argumento M deve incluir o espaço necessário para o sinal é o ponto decimal.

DEC[(M[,D])]
NUMERIC[(M[,D])]
FIXED[(M[,D])]

Este é um sinônimo para DECIMAL. O alias FIXED foi adicionado na versão 4.1.0 para compatibilidade com outros servidores.

DATE

Uma data. A faixa suportada é entre '1000-01-01' e '9999-12-31'. MySQL mostra valores DATE no formato 'AAAA-MM-DD', mas permite a você a atribuir valores a campos DATE utilizando tanto strings quanto números.

DATETIME

Um combinação de hora e data. A faixa suportada é entre '1000-01-01 00:00:00' e '9999-12-31 23:59:59'. MySQL mostra valores DATETIME no formato 'AAAA-MM-DD HH:MM:SS', mas permite a você que atribuir valores a campos DATETIME utilizado strings ou números.

TIMESTAMP[(M)]

Um timestamp. A faixa é entre '1970-01-01 00:00:00' e algum momento no ano 2037. No MySQL 4.0 ou anteriores, os valores TIMESTAMP são exibidos nos formatos YYYYMMDDHHMMSS, YYMMDDHHMMSS, YYYYMMDD, ou YYMMDD, dependendo se M é 14 (ou não definido), 12, 8 ou 6, mas permite a você atribuir valores ao campo TIMESTAMP usando strings ou números. Um campo TIMESTAMP é util para gravar a data e a hora em uma operação de INSERT or UPDATE porque é automaticamente definido a data e a hora da operação mais recente se você próprio não especificar um valor. Você também pode definir a data e a hora atual atribuindo ao campo um valor NULL. Desde o MySQL 4.1, TIMESTAMP é retornado com um string com o formato 'YYYY-MM-DD HH:MM:SS'. Se você deseja tê-lo como um número você deve adcionar +0 a coluna timestamp. Timestamp de tamanhos diferentes não são suportados. Desde a versão 4.0.12, a opção --new pode ser usada para fazer o servidor se comportar como na versào 4.1. Um TIMESTAMP sempre é armazenado em 4 bytes. O argumento M só afeta como a coluna TIMESTAMP é exibida. Note que colunas do tipo TIMESTAMP(M) columns onde M é 8 ou 14 são apresentadas como números enquanto as outras colunas TIMESTAMP(M) são strings. Isto é apenas para assegurar que podemos eliminar e restaurar com segurança tabelas com estes tipos!

TIME

Uma hora. A faixa é entre '-838:59:59' e '838:59:59'. MySQL mostra valores TIME no formato 'HH:MM:SS', mas permite a você atribuir valores para as colunas TIME usando strings ou números.

YEAR[(2|4)]

Um ano no formato de 2 ou 4 digitos (padrão são 4 digitos). Os valores permitidos estão entre 1901 e 2155, 0000 no formato de 4 digitos, e 1970-2069 se você estiver usando o formato de 2 digitos (70-69). MySQL mostra valores YEAR no formato YYYY, mas permie atribuir valores aos campos do tipo YEAR usando strings ou números. (O tipo YEAR é novo na versão 3.22 do MySL).

CHAR(M)

Uma string de tamanho fixo que é sempre preenchida a direita com espaços até o tamanho especificado quando armazenado. A faixa de M é de 1 a 255 caracteres. Espaços extras são removidos quando o valor é recuperado. Valores CHAR são ordenados e comparados no modo caso insensitivo de acordo com o conjunto de caracteres padrão, a menos que a palavra chave BINARY seja utilizada. A partir da versão 4.1.0, se o valor M especificado é maio que 255, o tipo de coluna é convertido para TEXT. Este é um recurso de compatibilidade. NATIONAL CHAR (ou em sua forma reduzida NCHAR) é o modo SQL-99 de definir que um campo CHAR deve usar o conjunto CHARACTER padrão. Este é o padrão no MySQL. CHAR é uma simplificaçào para CHARACTER. O MySQL lhe permite criar um campo do tipo CHAR(0).Isto é muito útil quando você precisa de comptibilidade com aplicativos antigos que dependem da existência de uma coluna, mas que, na verdade, não utiliza um valor. Isto também é muito bom quando você precisa de uma coluna que só pode receber 2 valores. Um CHAR(0), que não é definido como um NOT NULL, só irá ocupar um bit e pode assumir 2 valores: NULL or "".

BIT
BOOL
VARCHAR(M)

Uma string de tamanho variável. NOTA: Espaços extras são removidos quando o caracter é armazenado (o que difere da especificação ANSI SQL). A faixa de M é de 1 a 255 characters. Valores VARCHAR são ordenados e comparados no modo caso insensitivo a menos que a palavra chave BINARY seja utilizada. A partir da versão 4.1.0, se o valor M especificado é maio que 255, o tipo de coluna é convertido para TEXT. Este é um recurso de compatibilidade. VARCHAR é uma simplificação para CHARACTER VARYING.

TINYBLOB
TINYTEXT

Um campo BLOB ou TEXT com tamanho máximo de 255 (2^8 - 1) caracteres.

BLOB
TEXT

Um campo BLOB ou TEXT com tamanho máximo de 65535 (2^16 - 1) caracteres.

MEDIUMBLOB
MEDIUMTEXT

Um campo BLOB ou TEXT com tamanho máximo de 16777215 (2^24 - 1) caracteres.

LONGBLOB
LONGTEXT

Um campo BLOB ou TEXT com tamanho máximo de 4294967295 ou 4G (2^32 - 1) caracteres. Até a versão 3.23 o protocolo cliente/servidor e tabelas MyISAM tinham um limite de 16M por pacote de transmissão/registro de tabela, a partir da versão 4.x o tamanho máximo permitido das colunas LONGTEXT ou LONGBLOB depende do tamanho máximo configurado para o pacote no protocolo cliente/servidor e da memória disponível.

ENUM('valor1','valor2',...)

Uma enumeração. Um objeto string que só pode ter um valor, selecionado da lista de valores 'valor1', 'valor2', ..., NULL ou valor especial de erro "". Um ENUM pode ter um máximo de 65535 valores diferentes.

SET('valor1','valor2',...)

Um conjunto. Um objeto string que pode ter zero ou mais valores, cada um deve ser selecionado da lista de valores 'valor1', 'valor2', ... Um SET pode ter até 64 membros.

EXERCÍCIOS:

1- Utilizando o mysql transforme a tabela à seguir em uma tabela chamada clientes dentro de seu banco de dados. Faça as tarefas abaixo:

Tabela clientes					
*Código Clientes	RG Mãe	Nome Dependente	Dt. Nascimento	Relação	Sexo
2	10101012	Edson	27/12/46	Filho	Masculino
3	10105710	Luiz	18/11/49	Filho	Masculino
4	20232020	Flávia	14/02/89	Conjuge	Feminino
5	20232020	Angelo	10/02/45	Filho	Masculino
6	30333030	Fernanda	01/05/80	Filho	Feminino
7	10101010	Jorge	27/12/76	Filho	Masculino
8	10101010	Mauro	18/11/72	Filho	Masculino
9	10101010	Larisa	14/02/79	Conjuge	Feminino
10	23202020	Antunes	10/02/45	Filho	Masculino
11	30341030	Maria	01/05/70	Filho	Feminino
12	10101341	João	27/12/76	Filho	Masculino
13	30101010	Mário	18/11/89	Filho	Masculino
14	10101010	Gisele	14/02/79	Conjuge	Feminino
15	34202020	Arnaldo	10/02/85	Filho	Masculino
16	10101010	Kátia	01/05/70	Filho	Feminino
17	10105710	José	27/12/46	Conjuge	Masculino
18	78601010	Luis	18/11/75	Filho	Masculino
19	20202032	Fabiana	14/02/89	Conjuge	Feminino
20	78601010	Anisio	10/02/94	Filho	Masculino
21	76503030	Marcela	01/05/97	Filho	Feminino

a- Insira os dados na tabela.

b- Calcule quantos clientes existem na tabela.

c- Calcule quantos clientes do sexo feminino existem na tabela.

d- Crie uma seleção em que serão exibidos os nomes e RG da mãe dos clientes que são filhos no campo relação

e- Calcule quantos clientes são do sexo masculino e filhos no campo relação.

f- Altere a data de nascimento para 15/11/1980 do cliente com código igual a 12.

g- Selecione os clientes que têm mais de 18 anos, exibindo seus nomes e sexo.

h- Selecione as clientes que são casadas, exibindo seu nome e sexo, ordenando o resultado pela data de nascimento.

i- Selecione todos os clientes e ordene a seleção em ordem decrescentes pela data de nascimento.

2- Crie um banco de dados com o nome de "empresa" no MySQL. Utilize as tabelas e as descrições a seguir como um guia para inserir suas tabelas e dados em seu banco. Ao completar essa fase passe para as questões.

Tabela EMPREGADO

Nome	*RG	CIC	Depto	RG Supervisor	Salário
João Luiz	10101010	11111111	1	**NULO**	3.000,00
Fernando	20202020	22222222	2	10101010	2.500,00
Ricardo	30303030	33333333	2	10101010	2.300,00
Jorge	40404040	44444444	2	20202020	4.200,00
Renato	50505050	55555555	3	20202020	1.300,00

Tabela DEPARTAMENTO

Nome	*Número	RG Gerente
Contabilidade	1	10101010
Engenharia Civil	2	30303030
Engenharia Mecânica	3	20202020

Tabela PROJETO

Nome	*Número	Localização
Financeiro 1	5	São Paulo
Motor 3	10	Rio Claro
Prédio Central	20	Campinas

Tabela DEPENDENTES

*RG Responsável	Nome Dependente	*Dt. Nascimento	Relação	Sexo
10101010	Jorge	27/12/86	Filho	Masculino
10101010	Luiz	18/11/79	Filho	Masculino
20202020	Fernanda	14/02/69	Conjuge	Feminino
20202020	Angelo	10/02/95	Filho	Masculino
30303030	Adreia	01/05/90	Filho	Feminino

Tabela DEPARTAMENTO_PROJETO

*Número Depto	*Número Projeto
2	5
3	10
2	20

Tabela EMPREGADO_PROJETO

*RG Empregado	*Número Projeto	Horas
20202020	5	10
20202020	10	25
30303030	5	35
40404040	20	50
50505050	20	35

a- Mostre quantos dependentes possuí o empregado de nome João Luiz.

b- Mostre o nome do responsável pelo dependente nascido em 27/12/86.

c- Mostre de qual departamento é o funcionário de rg 30303030.

d- Quantas horas o empregado Fernando trabalhou?

e- Selecione todos os funcionários que trabalham no projeto 5 exibindo seus nomes.

f- Mostre o nome do responsável pelo dependente Angelo.

g- Mostre qual o salário do responsável pelo dependente Adreia.

h- Selecione os funcionários que trabalham em São Paulo.

Bibliografia

Fundamentals of Database Systems; Ramez Elmasri, Shamkant Navathe; The Benjamin CummingsPublishing Company; 1989;

Sistema de Banco de Dados; Henry F. Korth, Abraham Silberschatz; Makro Books; 1995;

SQL Language - Oracle Reference Manual; Version 7.2;

www.mysql.com

Anotações

Anotações

Anotações

Anotações

Anotações

Anotações

Anotações

Anotações

Anotações