Minhas colas de MySQL:

**SELECT *nomedacoluna* FROM *nomedatabela*;**

Para chamar a informação das colunas dentro de uma determinada tabela no MySQL.

**SELECT \* FROM *nomedatabela*;**

Vai selecionar TODO o conteúdo das colunas dentro da tabela.

**INSERT INTO *nomedatabela* SET *nomedacoluna = ‘informaçãodacoluna’*;**

Para inserir um conteúdo dentro das colunas das tabelas, acrescentando vírgulas depois das aspas simples para referenciar a outras colunas, esse padrão funciona apenas em bancos MySQL.

**INSERT INTO *nomedatabela (nomedacoluna, nome da coluna, nomedacoluna)* VALUES *(‘informaçãodacoluna’, ‘informaçãodacoluna’, ‘informaçãodacoluna’)*;**

Outro padrão para Inserir informações dentro de uma tabela, que funciona em todos os BD SQL.

**UPDATE *nomedatabela* SET *nomedacoluna = ‘informaçãoatualizada’ ;***

Para atualizar TODAS as colunas com uma informação, exemplo: senha = ‘999’.

Todas as senhas registradas no sistema serão 999.

**UPDATE *nomedatabela* SET *nomedacoluna = ‘informaçãoatualizada’ WHERE id = ‘1’;***

Para atualizar as informações SETADAS onde o ID é = 1.

Por isso em todos os bancos, é importante cada usuário ter um INT de ID para ser identificado, principalmente nesses casos, mas pode ser identificado pelo nome, sexo, id ou qualquer característica única, tanto para 1 quanto para 2 pessoas.

Pode ser usado também para filtrar entre Homens e Mulheres numa página por exemplo.

**UPDATE nomedatabela SET nome = ‘alexandre’ WHERE sexo = ‘M’;**

**UPDATE *nomedatabela* SET *nomedacoluna = ‘informaçãoatualizada’ WHERE id = ‘1’ OR id = ‘2’;***

OR serve para comparar e escolher os dados apresentados pelo comando.

**UPDATE *nomedatabela* SET *nomedacoluna = ‘informaçãoatualizada’ WHERE id = ‘1’ AND nome = ‘Alexandre’;***

AND serve para pegar os dados apresentados pelo comando, inclusive da pra usar OR e AND juntos para condições, exemplo: Se X informação não estiver disponível, pegue uma E outra OU aquela outra.

**DELETE FROM *nomedatabela* WHERE *id=’1’ ;***

Para deletar alguém pelo ID ou por uma etiqueta de reconhecimento.

Usando a mesma lógica de Ids apresentada acima.

**SELECT \* FROM nomedatabela WHERE nome LIKE ‘Ale%’**

Esse comando vai pegar nomes ou qualquer outra informação da coluna que seja parecida com a abreviação colocada, podendo inverter a posição do % para inicio, meio ou fim caso não saber o nome de tudo.

**SELECT \* FROM nomedatabela WHERE data BETWEEN alguma data ex: ‘2016-12-01’ AND ‘2017-02-01’;**

Esse comando vai pegar usuários ou dados de pessoas que nasceram entre essas datas, ou que comentaram algo entre essa data.

**SELECT \* FROM nomedatabela WHERE id IN (1, 2, 5, 10, 30);**

Esse comando vai pegar nomes ou qualquer outra informação que esteja entre esses parênteses, SOMENTE dos Ids ou o que quer que seja apresentado entre os parênteses.

**SELECT \* FROM nomedatabela ORDER BY data\_nascimento ASC (ou DESC)**

Esse comando serve para ordenar o banco de dados por data de nascimento crescente, do menor ao maior, e DESC para maior para menor, serve para NOME em ordem alfabética e afins, até mesmo usando Where, sempre antes do ORDER BY.

**SELECT \* FROM nomedatabela ORDER BY data\_nascimento ASC LIMIT 2;**

Esse comando serve para limitar a quantidade de informação que irá aparecer, podendo definir no LIMIT a quantidade.

**SELECT \* FROM nomedatabela LIMIT 1,2**

Esse comando serve para mostrar 2 opções pulando a primeira, o 1 serve a informação que pularemos e o 2 é a quantidade de informações apresentadas depois de pular a primeira.

**SELECT COUNT(\*) as contagem FROM nomedatrabela WHERE senha = ‘123’;**

Esse comando vai contar com número inteiro quantas pessoas dentro da tabela tem a senha igual a 123.

Para isso deve-se criar uma coluna chamada contagem dentro da tabela.

**SELECT COUNT(\*) as contagem, faixa\_salarial FROM usuarios GROUP BY faixa\_salarial;**

Serve para agrupar faixas salariais diferentes, de modo mais organizado e sem precisar ficar montando agrupamento individual para cada Where =, tudo com uma só verificação.

**SELECT nomedatabela.coluna, nomedaoutratabela.coluna FROM nomedatabela INNER JOIN nomedaoutratabela ON nomedaoutratabela.id = nomedatabela.colunadeligação;**

Serve para fazer exibir uma coluna do da tabela, a coluna de ligação é a coluna que liga o id da outra tabela na tabela principal, o INNER JOIN só vem conexões bem sucedidas, filtrando relações com sucesso.

O LEFT JOIN vem todos os resultados da tabela da direita, tendo relações bem sucedidas ou não, e o RIGHT faz a mesma coisa com as coisas da tabela do lado direita.

**SELECT usuarios.nome, (select faixas.titulo from faixas where faixas.id = usuarios.faixa\_salarial) as faixa FROM usuarios;**

Serve para subconsultas enquanto estiver consultando outra coisa.

**RELACIONAMENTO ENTRE TABELAS:**

**1:N =** Esse relacionamento significa 1 para muitos, 1 tabela pode tá relacionada a vários produtos.

**1:1 =** Esse relacionamento significa 1 para 1, 1 tabela pode tá relacionada a uma outra tabela, um relacionamento exclusivo entre ambos.

**N:N =** Esse relacionamento significa muitos para muitos, várias tabelas pode tá relacionada a vários produtos.

**DELIMITER $$**

Serve para os comandos em seguida não terminar com ;, mas sim com $$.

Depois deve-se voltar o DELIMITER para ; ao finalizar.

**CREATE FUNCTION *CONTAR*(nome VARCHAR(100))**

**RETURNS INT(10)**

**BEGIN**

DECLARE qt INT(10);

SET qt = LENGTH(nome);

RETURN qt;

**END$$**

Funcão criada para contar quantas letras tem a palavra NOME.

**SELECT nome, email, CONTAR(nome) as contagem FROM usuarios;**

Funcão criada para contar quantas letras tem a palavra NOME, assim a tabela ja é apresentada com o contabilizador funcionando corretamente.