**Curso de Dispositivos Móveis**

**Conceitos de Banco de Dados**

**Aula 01 - Introdução a Banco de Dados**

**- Banco de Dados.**

Banco de dados é um sistema que armazena e gerencia um conjunto de dados.

Os dois principais tipos de banco de dados são: Banco de Dados Relacional e Banco de Dados Orientado a Objetos.

Banco de Dados Relacional é um modelo que faz uso da Álgebra Relacional para representar e armazenar os dados.

Banco de Dados Orientados a Objetos é um modelo que faz uso da mesma ideia de uso de objetos na OOP para representar e armazenar objetos.

Para fazer a manipulação de um banco de dados faz-se uso de um sistema capaz de gerenciá-lo, denominado SGBD (Sistema de Gerenciamento de Banco de Dados).

Exemplos de SGBD no mercado são: MySQL, PostgreSQL, Oracle, SQL Server, DB2, Access e Paradox.

Para se fazer uso de um banco de dados é necessário: hardware, software, dados e usuários.

**- Um sistema de banco de dados tem abstração em três níveis:**

\* Nível de Visão: parte do banco de dados que o usuário faz uso.

\* Nível Conceitual: defini quais dados são armazenados e o relacionamento entre eles.

\* Nível Físico: o nível de abstração mais baixo, que define de que maneira os dados são armazenados.

**- Projeto de Banco de Dados se dá em duas fases: Modelo Conceitual e Projeto Lógico.**

Modelo conceitual é composto por:

\* Entidades: define os objetos que existem e se deseja armazenar.

\* Relacionamento: são as relações entre as entidades.

Modelo lógico: Implementa os recursos, proporcionando ampla e flexível capacidade de estruturação levando em consideração as limitações dos SGBD. Onde são definidas as chaves primárias e chaves estrangeiras.

Modelo Físico: Analisa as características e recursos necessários para armazenar e manipular as estruturas de dados.

**Aula 02 - Introdução à Linguagem SQL e Sistemas Gerenciadores de Banco de Dados**

**- Modelo Relacional.**

O Modelo Relacional representa dos dados de um Banco de Dados com uma coleção de relações e seus relacionamentos.

As relações dos modelos relacionais são comumente chamadas de tabelas. Cada relação (tabela) tem um nome e um conjunto de atributos.

Tabela é onde são armazenados um conjunto de dados relacionadas em um modelo relacional.

Atributos ou Campos são as informações existentes em uma tabela.

**- Domínio.**

Todo atributo deve ter um domínio definindo os valores possíveis que um atributo pode receber. Defini então, o tipo de dado e o formato de dado que um atributo pode armazenar.

**- Tuplas ou Registros.**

São os valores armazenados em uma tabela. Uma tabela sem tuplas é uma tabela vazia.

**- Usando o MySQL Workbench:**

\* File – New Model

\* Add Diagram

\* Ícone Place new table

\* Dar um nome para a tabela e definir seus atributos

- **Chave Primária.**

Entidades podem ter o atributo Chave Primária. Sendo chave primária ela:

\* O registro dela deve ser único dentro da tabela.

\* Não pode ser nula (null).

\* Identifica de maneira única o registro.

**- Chave Estrangeira.**

Serve para indicar que determinado registro de uma tabela tem dependência com a chave primária de outra tabela. Isso permite manter a integridade das relações.

**- SQL .**

SQL é uma linguagem declarativa utilizada nos SGDB que permite a manipulação e pesquisa dos dados dentro de um sistema relacional.

As palavras-chaves utilizada em SQL são classificadas como:

\* Manipulação de Dados (DML): inserir, apagar e atualizar dados (insert – delete - update).

\* Definição de Dados (DDL): definir e apagar tabelas (create - drop).

\* Controle de Dados (DCL): dá acesso, revoga acesso para manipular os dados (grant - revoke).

\* Transação de Dados (DTL): transação de dados (start transction).

\* Consulta de Dados (DQL): consulta de dados (select).

**Aula 03 - Linguagem SQL - criação, inserção e modificação de tabelas**

**- Comando para criar banco de dados.**

CREATE DATABASE nome\_do\_banco\_de\_dados;

**- Comando para acessar banco de dados.**

USE nome\_do\_banco\_de\_dados;

**- Comando para exibir banco de dados existente.**

SHOW DATABASES;

**- Comando para apagar um banco de dados.**

DROP nome\_do\_banco\_de\_dados;

**- Comando para criar uma tabela dentro do banco acessado.**

CREATE TABLE nome\_da\_tabela

( atributo 1 tipo1,

atributo 2 tipo 2,

...

atributo N tipo N

);

Ao se criar uma tabela é uma boa prática se colocar nos nomes dos atributos uma indicação de qual tabela ele pertence. Exemplo tabNome\_codigo.

**- Tipos para os atributos.**

\* Char (tamanho): sequência de caracteres com tamanho fixo pre determinado. Max de 255 carc.

\* Varchar (tamanho): sequência de caracteres de tamanho variado, o valor máximo permitido é definido em parênteses. Max de 65.535.

\* Int : tipo numérico que aceita valores inteiros.

\* Numeric (n,d): tipo numérico que aceita valores reais, onde n indica o número de dígitos e d o número de casas decimais.

\* Time: tipo tempo. Formato hora:minuto:segundo.

\* Date: tipo data. Formato ano-mês-dia.

**- Comando para exibir as tabelas de um banco de dados selecionado.**

SHOW TABLES;

**- Comando para inserir dados em uma tabela de um banco selecionado.**

INSERT INTO nome\_da\_tabela (atributo1, atributo2, ...) VALUES (valor1, valor2, ...);

Se quiser incluir mais de um registro, após o parentes do VALUES ponha “,” e novos valores.

Ao descrever os valores não numéricos como char, varchar, time e date, estes devem ser colocados entre aspas simples.

Para separação das casas decimais deve-se utilizar ponto (.) e não vírgula (,).

**- Comando para modificar um registro de uma tabela de um banco de dados selecionado.**

UPDATE nome\_da\_tabela SET atributo = valor WHERE condição;

Ex: UPDATE clientes SET cli\_email = 'silvaJose@cursosql.com' WHERE cli\_nome = 'José da Silva';

Se não for utilizada a cláusula where, a modificação será feita em toda a tabela.

Operações aritméticas são válidas nessas modificações.

**- Comando para apagar um registro.**

DELETE FROM nome\_da\_tabela WHERE condição;

No comando delete se a cláusula WHERE for omitida, apagará toda as linhas da tabela, sem apagar a tabela.

**- Comando para apagar uma tabela.**

DROP TABLE nome\_da\_tabela;

**Aula 04 - Linguagem SQL - Consulta Simples de Tabelas**

**- Comando para mostrar a estrutura de uma tabela.**

DESC nome\_da\_tabela;

**- Comando para consultar dados de uma tabela.**

SELECT atributo1, atributo2 FROM nome\_da\_tabela1, nome\_da\_tabela2, WHERE condição;

Para consultar todos os campos da tabela usa-se \*, no lugar dos atributos.

A limitação de linha a serem exibidas é condicionada na cláusula WHERE.

**- Operadores de comparação na WHERE.**

= , <> , <= , >= , < , >

Os valores de comparação são utilizados para valores alfanuméricos. Nas comparações alfanuméricas os registros serão considerados em ordem alfabética.

**- Operadores lógicos na** WHERE.

AND e OR

**- Palavras chaves para restrições na** WHERE.

BETWEEN – especifica um intervalo de valores a serem consultado.

SELECT fil\_titulo FROM filmes WHERE fil\_codigo BETWEEN 2 AND 20;

LIKE – utilizado na comparação de cadeia de caracteres. Usamos o % para indicar uma cadeia com um número indeterminado de caracteres, e \_ para indicar um único caractere.

SELECT cli\_nome FROM clientes WHERE cli\_email LIKE '%gmail.com';

IN – utilizado para verificar a existência de algumas cadeias de caracteres que são listadas entre parênteses e separados por vírgulas.

SELECT fil\_titulo FROM filmes WHERE fil\_genero IN ('comedia', 'romance', 'acao');

IS NULL – usado para selecionar diretamente um valor NULL. Para uma ação invertida usa-se o IS NOT NULL

SELECT cli\_nome FROM clientes WHERE cli\_email IS NULL;

SELECT cli\_nome FROM clientes WHERE cli\_email IS NOT NULL;

**Aula 05 - Linguagem SQL - Consulta Avançada de Tabelas**

**- Comando para ordenar os dados de uma consulta.**

SELECT cli\_nome, cli\_email FROM clientes ORDER BY cli\_nome; // ordem crescente

SELECT cli\_nome, cli\_email FROM clientes ORDER BY cli\_nome DESC; // decrescente

**- Comando para criar um agrupamento de dados.**

SELECT fil\_preco FROM filmes GROUP BY fil\_preco;

Retorna uma tabela com os diferentes preços existentes na tabela fil\_preco.

**- Funções de agregação usada com** GROUP BY.

Essas funções retornam um único valor extraído de um conjunto de valores de determinada coluna.

COUNT(nome\_coluna) – retorna a quantidade de registro não nulos de um coluna.

SELECT COUNT(fil\_titulo) FROM filmes; // dá o total de filmes.

SELECT COUNT(DISTINCT fil\_titulo) FROM filmes; // total sem duplicados.

SUM(nome\_coluna) – retorna a soma dos valores de uma coluna.

AVG(nome\_coluna) - retorna a média dos valores de uma coluna.

MIN(coluna) – retorna o menor valor encontrado em uma coluna.

MAX(coluna) – retorna o maior valor encontrado em uma coluna.

SELECT MIN(fil\_preco), AVG(fil\_preco), MAX(fil\_preco)FROM filmes;

Exemplo: Exibir o número de pessoas que morram em cada cidade da tabela cidade do database pessoas.

SELECT pes\_cidade, COUNT(pes\_cidade) FROM pessoas GROUP BY pes\_cidade ORDER BY pes\_cidade;

Para dá um nome personalizado para uma coluna agrupada utilizamos a cláusula AS.

SELECT fil\_titulo, COUNT(fil\_titulo) AS unidades FROM filmes GROUP BY fil\_titulo;

HAVING é a clausula que se utiliza junto com o GROUP BY para determinar uma condição.

SELECT fil\_titulo AS titulo, COUNT(fil\_titulo) AS quantidade FROM filmes GROUP BY fil\_titulo HAVING COUNT(fil\_titulo)>1; // são mostrados a quantidade de filmes de cada título em uma coluna nomeada como “quantidade”, que possuam mais de uma unidade.

**- Funções Matemáticas**

Função Significado

ABS (valor) Retorna o valor absoluto (positivo) do valor informado.

FLOOR (valor) Retorna o maior número inteiro, igual ou menor ao valor informado.

ROUND (valor, n) Arredonda o valor informado para n casas decimais.

POWER (valor, p) Retorna o valor informado elevado à potência p.

**- Funções para manipulação de strings**

Função Significado

LENGTH (expressão) Retorna o número de caracteres contidos na expressão informada.

LOWER (expressão) e UPPER (expressão) Converte para minúsculo e maiúsculo a expressão informada, respectivamente.

LTRIM (expressão) e RTRIM (expressão) Remove os espaços em branco à esquerda e à direita da expressão informada, respectivamente.

SUBSTRING (expressão, início, tamanho) Extrai uma parte dos caracteres da expressão, iniciando da posição informada em início, considerando a quantidade definida em tamanho.

**- Função de data e hora**

Função Significado

CURDATE() e CURTIME() Retorna a data e hora atuais, respectivamente.

EXTRACT (parte FROM data) Retorna apenas a parte especificada de um campo de data/hora. A parte pode ser year, month, day, hour, minute etc.

DATE\_FORMAT (data, formato) Retorna a data modificando seu formato de apresentação, que pode ser: %d para dia (0-31), %m para mês (0-12), %Y para ano com quatro dígitos e %y para ano com dois dígitos.

Exemplos

Mostrar o preço médio das locações considerando apenas duas casas decimais.

SELECT DISTINCT ROUND(AVG(fil\_preco),2) FROM filmes;

Mostrar em maiúsculo os títulos de todos os filmes (sem repetição) que estão alugados.

SELECT DISTINCT UPPER(fil\_titulo) FROM filmes WHERE fil\_situacao='alugado';

Mostrar o nome e ano de nascimento de todos os clientes da locadora do sexo feminino.

SELECT cli\_nome, EXTRACT(year FROM cli\_data\_nasc) FROM clientes WHERE cli\_sexo='F';

Mostrar o nome e data de nascimento de todos os clientes da locadora no formato brasileiro dd/mm/aaaa e em ordem alfabética do nome.

SELECT cli\_nome, DATE\_FORMAT(cli\_data\_nasc,'%d %m %Y') FROM clientes

ORDER BY cli\_nome;

**Aula 06 - Linguagem SQL - Alteração de Estrutura de Tabelas e Ambientes de Múltiplas Tabelas**

- **Comando** ALTER TABLE **é utilizado para:**

Adicionar colunas com o comando ADD.

ALTER TABLE nome\_da\_tabela ADD coluna tipo, coluna tipo, ...;

Quando uma coluna é adicionada à tabela esta será posicionada na ultima posição. Para gerenciar o local onde será posicionada faz uso do comando FIRST e AFTER.

ALTER TABLE filmes ADD fil\_direcao VARCHAR(50) FIRST;

ALTER TABLE filmes ADD fil\_atores VARCHAR(200) AFTER fil\_direcao;

Excluir colunas com o comando DROP COLUMN.

ALTER TABLE nome\_da\_tabela DROP COLUMN atributo;

O uso da cláusula COLUMN é opcional. Ao se excluir uma coluna de uma tabela todos os dados nela contidos serão perdidos.

Alterar o tipo e o nome de uma coluna já existente com o comando CHANGE COLUMN.

ALTER TABLE nome\_da\_tabela CHANGE COLUMN nome\_atual novo\_nome tipo;

Quando se quer modificar apenas o tipo da coluna usa-se o comando MODIFY COLUMN.

ALTER TABLE nome\_da\_tabela MODIFY COLUMN nome\_da\_coluna tipo;

**- Comando para determinar chave primária.**

Para determinar um campo como chave primária na criação de uma tabela usamos o comando abaixo no final do comando.

PRIMARY KEY (nome\_da\_coluna\_chave\_primária)

**- Comando para determinar chave estrangeira.**

Para determinar um campo como chave estrangeira na criação de uma tabela usamos o comando abaixo no final do comando.

FOREIGN KEY (nome\_do\_coluna\_chava\_estrangeira) REFERENCE nome\_da\_tabela\_origem (nome\_da\_chave\_primaria\_na\_tabela\_origem)

- **Consultar entre tabelas com o comando** CROSS JOIN.

Neste tipo de consulta o número total de linha a serem exibidas será produtos das linhas das tabelas.

SELECT atributo1\_da\_tabela1, atributo1\_da\_tabela2, ... FROM nome\_da\_tabela1 CROSS JOIN nome\_da\_tabela2, ...;

**- Consulta entre tabelas com o comando** SELECT.

SELECT obr\_titulo AS Livros, aut\_nome AS Autor FROM autores AS a, obras AS o autores\_obra AS ao WHERE o.abr\_titulo LIKE ‘%Banco de Dados%’ AND a.aut\_codigo = ao.aut\_codigo AND o.obr\_codigo = ao.aut\_codigo;

Esta consulta solicita que sejam exibidos dados de duas colunas, obr\_titulo que vai ser exibida com o nome de Livros e aut\_nome que vai ser exibida com o nome Autor.

A consulta, para isso, faz uso de três tabelas, tabela autores que nomeada, na consulta, para “a”, a tabela obras que nomeada como “o” e a tabela autores\_obras nomeada como “ao”.

O critério da consulta delimita o retorno a todos os registros da tabela obras na coluna obr\_titulo que tenha a string “Banco de dados” em qualquer ponto do registro, desde que a coluna aut\_codido da tabela obras seja igual a coluna aut\_codigo da tabela autores\_obra e o campo obr\_codigo da tabela obras sejam iguais ao campo aut\_codigo da tabela autores\_obra.