Aula 3 e 4
Conexão Banco com Java

APTECH® COMPUTER EDUCATION Escola de Tecnologia

Aula 3

Porta de Comunicação do MySQL -> 3306

User do MySQL -> root

Senha do MySQL -> bancodedados

Fórum sobre comunicação do Java com MySQL

http://www.devmedia.com.br/como-conectar-uma-aplicacao-java-a-um-banco-de-dados-access/28184

Adicionando o Driver para comunicação.

->botão direito em bibliotecas->driver JDBC do MySQL

Aula 3

Relembrando o passo a passo para comunicação.

1-Registrar driver JDBC

2-conexão com o banco de dados

3-comandos sql

4-processar os dados

5-fechar a conexão com banco de dados

Base da aula 3

Criar no MySQL a base de dados **Aula3_Funcionario** com as tabelas **gerente** e **vendedor.**

Na tabela **vendedor** terá o id(int, PK, NN, AI); nome (varchar(40), NN); Salario(double, NN); região(Varchar(25));

A tabela **gerente** será com os campos, id(int, PK, NN, AI); nome(varchar(40), NN); Salario(double, NN); setor(Varchar(35);

Criar 5 funcionários em cada tabela.

Comando UPDATE

Comando **UPDATE** atualiza dados em nossa tabela. Esses dados devem ser informados na hora que se cria a query.

Nesse comando deve-se tomar cuidado para que na hora de criar, sempre passar a condição que se quer atualizar, caso contrário, corre o risco de atualizar os dados de todos os registros.

Exemplo Comando UPDATE

Após iniciar a conexão com o banco de dados iremos criar as query's e prepara-las para executar.

```
//Passo 3- criar query
String query = "UPDATE alunos SET idade = (?), nota = (?)
  WHERE id = (?)";
// Passo 3 - preparar a query
stmt = conexao.prepareStatement(query);
stmt.setInt(1, 23);   //nova idade 23
stmt.setDouble(2, 6.4 );  //nova nota é 6,4
stmt.setInt(3, 2);  //atualizando os dados do aluno 2
```

Exemplo Comando UPDATE

Depois que as query's foram criadas e preparadas, iremos executa-las através do método **.executeUpdate ()**.

Exemplo:

```
//4 executar a query
stmt.executeUpdate();
```

Após executa-las, não esquecer de fechar nossa conexão com o método ***.close();***

Exemplo:

```
//passo 5 – finalizar conexão
stmt.close();
conexao.close();
```

Exercícios Comando UPDATE

1 – Criar um programa que atualize o salário e a região do vendedor 2.

2 – Mudar a região para “Norte” de todos os vendedores com salário menor que R$1.500,00.

3 – Dados o ID do gerente, aumentar em R$500,00 o salário dele.

4 – Digitar o ID do gerente. Se o salário for maior que R$1.500,00, mudar o setor para “Centro”.

Dica: salario>(?) AND id=(?)

Comando DELETE

Comando **DELETE** apaga os registros da tabela. Esses registros devem ser informados na hora que se cria a query.

Assim como o **UPDATE**, deve-se tomar cuidado para que na hora de criar, sempre passar o registro que se quer atualizar, caso contrário, corre o risco de apagar todos os registros da tabela.

Exemplo Comando DELETE

Após iniciar a conexão com o banco de dados, iremos criar as query's e prepara-las para executar.

```
//Passo 3- criar query
String query = "DELETE FROM gerente WHERE id = (?)";


//Passo 3 Preparar a query
stmt = conexao.prepareStatement(query);
stmt.setInt(1, 3);
```

Exemplo Comando DELETE

Depois que as query's foram criadas e preparadas, iremos executa-las através do mesmo método que utilizamos no **SELECT** (**.execute**()).

Exemplo:

```
//Passo 4 executar a query
stmt.execute();
```

Após executa-las, não esquecer de fechar nossa conexão com o método ***.close();***

```
Exemplo: //Passo 5 fechar conexão
stmt.close();
conexao.close();
```

Exercício Comando DELETE

1 – Criar um método que apague um vendedor pelo ID.

2 – Dado um salário, apagar todos os gerentes que recebam mais que aquele valor.

Desafio: Criar uma interface que junte os 2 programas acima e o usuário decide se quer deletar um gerente ou um vendedor, ambos de id.

O que fazer quando temos que enviar mais de um argumento?

Às vezes nossa tabela possui muitos campos para inserir os dados, ficando muitos argumentos para passarmos ao método. Sendo assim, o Java nos permite passar o próprio objeto como argumento. Para que isso seja possível, temos que criar uma classe com os atributos iguais ao da tabela.

Exemplo enviando argumentos:

Public void cadastrarEstudante(nome, idade, rg, tel) {

Exemplo enviando o objeto:

Public void cadastrarEstudante(Estudante e1) {

Tabela dos comandos

COMANDO	FUNÇÃO	TIPO	ARGUMENTOS
INSERT	stmt.execute()	void	Elementos da tabela(nome,rg)
SELECT	stmt.executeQuery()	Objeto/valores	Elementos da consulta(id, Nome, rg)
UPDATE	stmt.executeUpdate()	void	Novo valor e elementos de alteração.
DELETE	stmt.execute()	void	Elemento que será deletado(id)

Aula 4

Porta de Comunicação do MySQL -> 3306

User do MySQL -> root

Senha do MySQL -> bancodedados

Fórum sobre comunicação do Java com MySQL

http://www.devmedia.com.br/como-conectar-uma-aplicacao-java-a-um-banco-de-dados-access/28184

Adicionando o Driver para comunicação.

->botão direito em bibliotecas->driver JDBC do MySQL

Aula 4

Relembrando o passo a passo para comunicação.

1-Registrar driver JDBC

2-conexão com o banco de dados

3-comandos sql

4-processar os dados

5-fechar a conexão com banco de dados

Base da aula 4

Criar no MySQL a base de dados **aula4_veterinario** com as tabelas: animal e remedio**.**

Na tabela **animal** terá os campos: **id**(int, PK, NN, AI); **nome** (varchar(40)); **raca**(varchar(25)); **idade**(int);

A tabela **remedio** terá os campos: **id**(int, PK, NN, AI);

nome(varchar(40)); **dosagem_ml**(double);

O que fazer quando o método precisa retornar mais de um valor?

Quando isso ocorrer, assim como podemos enviar um objeto como argumento, também podemos retornar um objeto em nosso método, para que isso seja possível, também temos que criar uma classe com os atributos iguais de nossa tabela.

Exemplo:

```
public Vendedores consultarVendedorPeloId(int id){
        Vendedores v1 = new Vendedores();
        return v1;
}
```

Aula 4 – Como organizar nosso Projeto?

Para começar organizar nosso projeto, utilizaremos a forma mais comum dos programadores, onde criamos de 4 a 5 pacotes e cada pacote é responsável por uma função.

1º Pacote:

(+)core/main -> Método onde fica as Classes Main (todas as classes principais) e toda a interface gráfica.

2º Pacote:

(+)dao -> É a classe responsável pela conexão com o banco de dados e conexão cliente/servidor.

Aula 4 – Como organizar nosso Projeto?

3º pacote:

(+)model -> São os Modelos. Classes que só possuem os atributos.

4º Pacote:

(+)util -> Pacote responsável pelas regras de negócios, métodos de cálculos, etc.

Alguns projetistas criam o **5º Pacote:**

(+)arquivos -> É o pacote onde ficam as Imagens e os arquivos utilizados no projeto.

Aula 4 – Exercícios

1º Criar um programa para cadastrar animal com os pacotes organizados.

2º Acrescentar ao programa anterior a opção para cadastrar Remédios.

3º No programa 2, criar métodos para consultar nome do animal e nome do remédio pelo id.

4º Incrementar no programa anterior os métodos para deletar animal pelo id e deletar remédio pelo nome.

5º Acrescentar o método para aumentar a dosagem do remédio em 10%.