Aluno: 330230 – Danimar Henrique Varisa

SQL / CRUD com JDBC e DAO

1. Utilize o MySQL Workbench e crie o banco de dados chamado `unoesc_trabalho_CRUD`. Selecione o banco com o comando `USE`. Cole abaixo o comando utilizado.

```
CREATE DATABASE unoesc_trabalho_CRUD;
USE unoesc_trabalho_CRUD;
```

2. Use o comando `CREATE TABLE` para criar a tabela chamada pessoa com a estrutura a seguir. Cole abaixo o comando utilizado.

Nome	Tipo
id_pessoa	INT AUTO_INCREMENT PRIMARY KEY
nome_pessoa	VARCHAR(50) NOT NULL
data_nascimento	DATE NOT NULL
salario	DECIMAL(12, 2) NOT NULL CHECK(salario >= 0)

```
CREATE TABLE pessoa (
    id_pessoa INT(11) AUTO_INCREMENT PRIMARY KEY,
    nome_pessoa VARCHAR(50) NOT NULL,
    data_nascimento DATE NOT NULL,
    salario DECIMAL(12, 2) NOT NULL CHECK(salario >= 0)
);
```

3. Utilize o comando `INSERT` para inserir registros nesta tabela da seguinte forma, colando um exemplo de cada comando abaixo.
   a. Insira alguns registros usando a forma do comando `INSERT` que passa todos os valores mas sem especificar os campos. Para o campo `id_pessoa` passe o valor nulo de forma que o próprio MySQL gere o código.

```
INSERT INTO pessoa VALUES (null, "Danimar Henrique Varisa", '1990-05-02', 8900.00);
```

   b. Insira alguns registros usando a forma do comando `INSERT` que especifica uma lista de campos. Use a seguinte lista de campos `(nome_pessoa, data_nascimento, salario)`.

```
INSERT INTO pessoa (nome_pessoa, data_nascimento, salario) VALUES ("Jaqueline Candiago de Oliveira", '1990-07-16', 6850.00);
INSERT INTO pessoa (nome_pessoa, data_nascimento, salario) VALUES ("Daniel Varisa", '1967-09-09', 11000.00);
INSERT INTO pessoa (nome_pessoa, data_nascimento, salario) VALUES ("Diego de Melo Varisa", '2010-10-10', 900.00);
INSERT INTO pessoa (nome_pessoa, data_nascimento, salario) VALUES ("Duany Henrique Varisa", '1993-10-20',1500.00);
```

4. Execute o comando `SELECT` para listar os dados da tabela em ordem alfabética crescente por nome, depois em ordem de data de nascimento e depois por salário. Cole abaixo os comandos.

```
SELECT * FROM pessoa ORDER BY nome_pessoa ASC;
SELECT * FROM pessoa ORDER BY data_nascimento ASC;
SELECT * FROM pessoa ORDER BY salario ASC;
```

5. Modifique um registro utilizando o comando `UPDATE`. NÃO ESQUEÇA de utilizar a cláusula `WHERE` para indicar qual registro será modificado. Cole abaixo o comando.

```
UPDATE pessoa SET salario = 1800.00 WHERE id_pessoa = 4;
```

6. Com o comando `DELETE` remova o último registro inserido. NÃO ESQUEÇA de utilizar a cláusula `WHERE` para indicar qual registro será modificado. Cole abaixo o comando.

```
DELETE FROM pessoa WHERE id_pessoa = 5;
```

7. Faça uma consulta que mostre os nomes com respectivas datas de nascimento. Use a função `date_format()` com a máscara `'%d de %M de %Y'` para formatar a saída. Dê o apelido de 'data de nascimento' ao campo. Modifique a 'localização' do sistema para português do Brasil com o comando `SET lc_time_names='pt_BR'` para a consulta mostrar as datas em português. Cole abaixo os comandos.

```
SET lc_time_names='pt_BR';
SELECT nome_pessoa, date_format(data_nascimento, '%d de %M de %Y') AS 'Data de
nascimento' FROM pessoa;
```

8. Utilize a função `count()` para retornar o número de registros que existem na tabela. Cole abaixo o comando.

```
SELECT COUNT(*) AS 'Qtde pessoas' FROM pessoa;
```

9. Faça uma consulta que retorne o salário mais alto e mais baixo. Cole abaixo o comando.

```
SELECT MAX(salario), MAX(salario) FROM pessoa;
```

10. Execute uma consulta que retorne a média e o somatório dos salários. Cole abaixo o comando.

```
SELECT AVG(salario), SUM(salario) FROM pessoa;
```

11. Siga a apresentação 'aula 38c - CRUD com JDBC e DAO (26.08.2022)' e refaça o CRUD (DAO – *interface* e implementação, entidade de domínio e programa principal) mas dessa vez utilizando a tabela `pessoa`
criada no início desta lista de exercícios.

12. Desafio: Faça a versão interativa do CRUD.
    a. O menu deve seguir o modelo abaixo, adaptando-o para o cadastro de pessoas:

```
========= MENU =========

1. Incluir produto
2. Alterar produto
3. Excluir produto
4. Listar produtos
5. Consultar produto
6. Finalizar programa

Escolha uma opção [1-6]:
```

    b. Estrutura do programa (classe `Principal`):
    Atributos como DAO da entidade pessoa, objetos `SimpleDateFormat` e `NumberFormat`.

```
static private Scanner sc = new Scanner(System.in);
```

    Construtor:
    - Inicializar DAO.
    - Inicializar objetos `SimpleDateFormat` e `NumberFormat`.

    Método `void adicionar()`:

- Data deve ser definida automaticamente pelo sistema e não pelo usuário.

```
java.sql.Date dataCadastro = Date.valueOf(LocalDate.now());
```

- Solicitar confirmação do usuário e em caso afirmativo instanciar um objeto `Pessoa` e usar o objeto DAO para adicioná-la no banco de dados.

Método `void alterar()`:

- Executar o método `solicitarPessoa()`. Se o resultado foi nulo sair do método, caso contrário entrar na tela de alteração, conforme modelo abaixo.

```
Id...........: 1
1. Nome produto.: Mesa Grande
2. Data cadastro: 01/01/2020
3. Quantidade...: 15
4. Preço........: R$ 15.000,50

Escolha o campo a alterar ou 5 para finalizar [1-5]:
```

- O usuário deve poder escolher qual campo modificar. Use um `switch-case` para testar a opção e solicitar o campo correto.
- No caso da data de cadastro o código abaixo pode ser utilizado para converter uma *string* em um tipo `java.sql.Date` e gerar uma exceção se for entrada uma data em formato inválido.

```
try {
        long dataHora = fd.parse(dataCadastro).getTime();
        prod.setDataCadastro(new java.sql.Date(dataHora));
        } catch (ParseException e) {
         System.out.println("Formato inválido! Formato é dd/mm/aaaa");
}
```

- Ao final das edições solicitar confirmação do usuário e em caso afirmativo usar o objeto DAO para alterar o registro no banco de dados.

Método `void excluir()`:

- Executar o método `solicitarPessoa()`. Se o resultado foi nulo sair do método.
- Caso o registro tenha sido encontrado, mostrá-lo na tela utilizando para isso o método `mostrarPessoa()`.
- Solicitar confirmação ao usuário para excluir o registro. Em caso afirmativo usar o objeto DAO para excluir o registro no banco de dados.

Método `void consultar()`:

- Executar o método `solicitarPessoa()`. Se o resultado foi nulo sair do método.
- Caso o registro tenha sido encontrado, mostrá-lo na tela utilizando para isso o método `mostrarPessoa()`.

Método `listar()`:

- Crie uma lista que receba os dados do método `listarTodos()` da DAO.
- Execute um laço de repetição (`for (Pessoa p: lista)`) e dentro do laço chame o método `mostrarProduto()` para cada objeto `p`.

Método `Pessoa solicitarPessoa()`:

- Solicita ao usuário o código da pessoa conforme modelo abaixo.

```
Digite o código do produto:
```

- Executa o método `buscarPorId()` da classe DAO para retornar uma pessoa ou nulo.

Método `mostrarPessoa(Pessoa p)`:

- Mostra os dados da pessoa, passada no parâmetro `p`, semelhante ao modelo abaixo.

```
Id...........: 1
Nome produto.: Mesa Grande
Data cadastro: 01/01/2020
Quantidade...: 15
Preço........: R$ 15.000,50
```

Método `main()`:

- Montar menu, programa só deve encerrar ao escolher a opção 6.
- Tratar a exceção de *runtime*.

13. Desafio: Faça agora a versão com *interface* gráfica utilizando Swing.