# S112 – Banco de Dados


## Professor MSc. Eng. Márcio José de Lemos

**E-mail:** marcio.lemos@senairs.org.br

http://lattes.cnpq.br/4769158065464009




FIERGS SENAI

# Projeto de Banco de Dados

# Dado x Informação x Conhecimento

- **Dado:** Registro de um fato, cadeias numéricas ou alfanuméricas que não possuem significado associado.

**Ex.:** Matriz de valores contendo as compras efetuadas por um cliente.

# Dado x Informação x Conhecimento

- **Informação:** Dado que foi **processado** de forma a se tornar relevante para uma determinada pessoa ou organização.

Informação: dados associados ao seu significado.

**Ex.:** Matriz com valores e o significado de cada coluna.

Coluna 1= nome do cliente

Coluna 2 = valor da compra

# Dado x Informação x Conhecimento

- **Conhecimento:** Ato ou efeito de abstrair ideia ou noção de alguma coisa. **Conhecimento:** há um entendimento sobre o significado dos dados e é possível extrair conhecimento a partir dos mesmo.

**Ex.:** Valor total das compras de um cliente em um determinado período de tempo.

# Nos dias atuais, o sucesso de uma organização depende:

- Da capacidade de adquirir dados de forma correta com rapidez;
- Da capacidade de gerenciar os dados de forma efetiva;
- De utilizar os dados para agregar valor ao seu negócio.

# Definições

- **Base de Dados (BD):** É uma coleção de dados relacionados e armazenados em algum dispositivo. Genericamente pode ser qualquer conjunto de dados como, por exemplo: uma agenda com os endereços de pessoas conhecidas, uma lista de elementos, um livro, apontamentos.
- O objetivo de criar e manter uma BD são poder obter e utilizar os dados lá guardados: procurar a morada de uma determinada pessoa, saber o que foi dito nas aulas sobre um tema.

# Definições

- **Base de Dados**
- É uma coleção de dados relacionados e armazenados em algum dispositivo.
- Representação é livre
- Arquivos texto
- Às vezes provê informação

# Definições

- **Banco de Dados**
- É uma coleção de dados relacionados, os quais são fatos que podem ser gravados e que possuem um significado.
- Obrigatoriamente provém informação;
- Dados são representados segundo um padrão;
- Pressupõe um sistema de gerenciamento.

# Definições

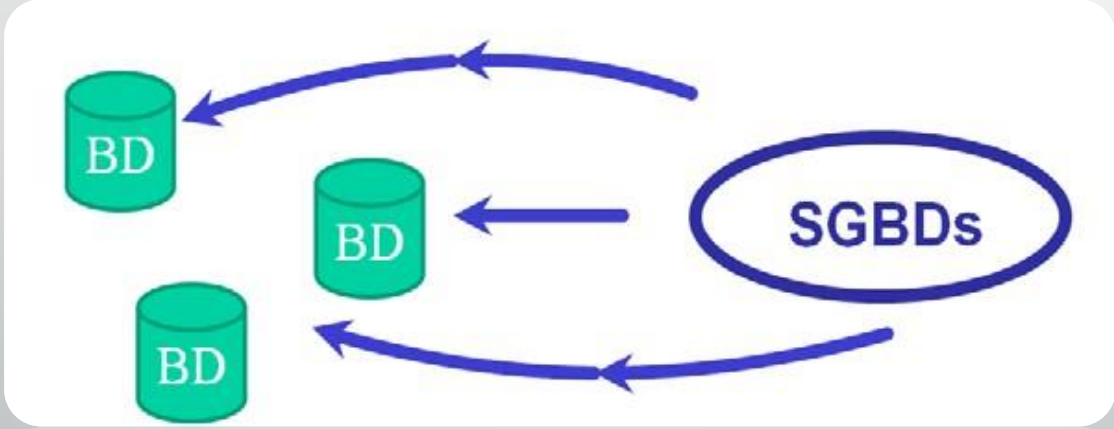
- **Gerência de Dados**
- Suporte à extração do conhecimento a partir de bancos de dados;
- Suporte à manipulação dos dados, garantindo consistência e integridade dos dados;
- SGBDs: Sistemas de Gerência de Bancos de Dados.

# Definições

- **SGBDs**
- Coleção de programas que permitem ao usuário definir, construir e manipular Bases de Dados para as mais diversas finalidades.

# Definições

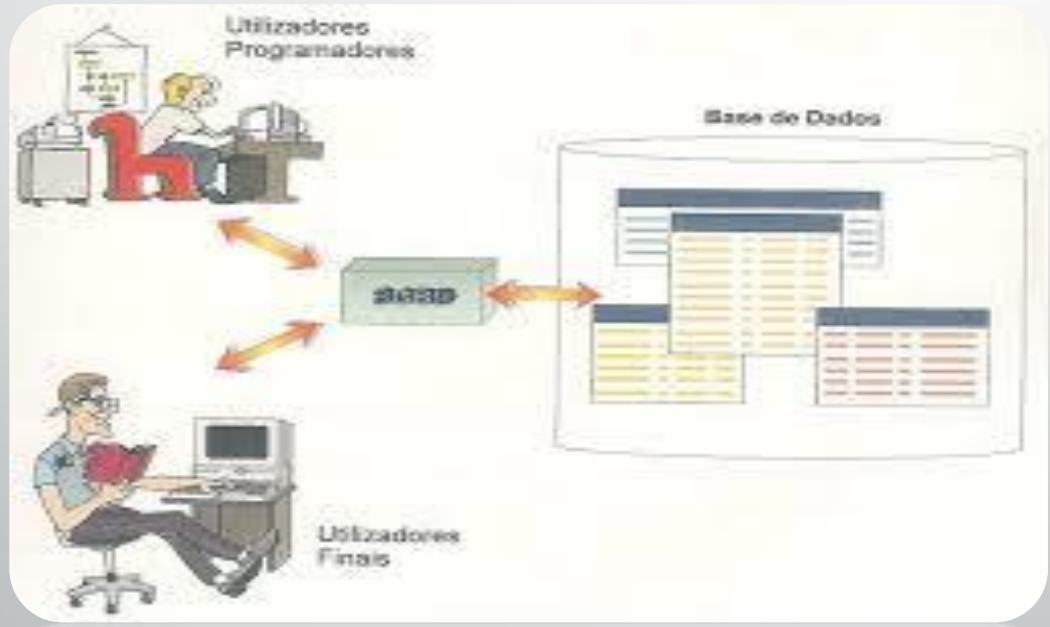
- Estas informações são armazenadas no catálogo do SGBD, o qual contém informações como a estrutura de cada arquivo, o tipo e o formato de armazenamento de cada tipo de dado, restrições, etc.



# Definições

- **Usuários**

Pessoas envolvidas, desde o projeto, uso, até a manutenção.
Dividem-se em três categorias: **causais; novatos e sofisticados**.



# Definições

- **Usuários causais**

Acessam o banco de dados casualmente, mas que podem necessitar de diferentes informações a cada acesso e utilizam sofisticadas linguagens de consulta para especificar suas necessidades.

# Definições

- **Usuários novatos**

Utilizam porções pré-definidas do banco de dados, usando consultas pré-estabelecidas que já foram exaustivamente testadas (programas).

# Definições

- **Usuários sofisticados**

São usuários que estão familiarizados com o SGBD e realizam consultas complexas.

# Definições

- **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 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, bem como da criação das estruturas, restrições e integridades, definidas no projeto.

# Modelo Relacional e Comandos DDL

- Modelo Relacional
- Criação de Tabelas, Campos e Atributos

# Modelo Relacional e Comandos DDL

- O **modelo relacional** é um modelo de dados, adequado a ser o modelo de um Sistema Gerenciador de Banco de Dados (SGBD), que se baseia no princípio em que todos os dados estão guardados em tabelas **(representação bi-dimensional de dados composta de linhas e colunas).**

# Modelo Relacional e Comandos DDL

- Tornou-se um padrão de fato para aplicações comerciais, devido a sua simplicidade e performance.

- Toda a Informação de um banco de dados relacional é armazenada em **Tabelas**, que na linguagem do MER, também são chamadas de Entidades.
- Por exemplo, posso ter uma Tabela “Alunos", onde seriam armazenadas informações sobre os diversos alunos.

# Modelo Relacional e Comandos DDL

- Sobre cada um dos alunos podem ser armazenadas diversas informações tais como: Nome, RG, Matricula, Rua, Bairro, Telefone, CEP, Sexo, Estado Civil, etc.

- Essas diversas **características de cada Aluno são os "Atributos"** da entidade Aluno , também chamados de campos da tabela Aluno .

# Modelo Relacional e Comandos DDL

- **Tabela Aluno:**

| Num_Matricula | Nome_Aluno | Sexo_Aluno |
|---|---|---|
| 1 | Maria | F |
| 2 | João | M |
| 3 | Pedro | M |
| 4 | Carla | F |
| 5 | Sandra | F |

- **Regras:**

Nomes de tabelas devem ser únicos no banco de dados;

De preferência a nomes curtos

# Modelo Relacional

- Considerando  a tabela Aluno:
- Ela tem três colunas Num_Matrícula, Nome_Alunoe Sexo_Aluno;
- A cada uma destas colunas damos o nome de **atributo**;
- Um nome de atributo deve ser **único** em uma tabela e dizer exatamente o tipo de informação que ele representa.

# Modelo Relacional

- **Regras:**
- Uma coluna (atributo) não segue um ordenamento especifico; ,
- Nome de uma coluna deve expressar exatamente o que armazena;
- Sempre que possível utilizar prefixos padronizados, Cod_Dept, Nome_Funcionario, Qtde_Estoque
- **NUNCA UTILIZAR ACENTOS GRÁFICOS E CARACTERES DA LÍNGUA PORTUGUESA, COMO EXEMPLO "Ç" PARA NOMEAR ATRIBUTOS E TABELAS**

# Modelo Relacional

- **A tabela Aluno possui cinco registros;**

Cada registro representa um conjunto de valores;

A este relacionamento damos o nome de registro ou linha

Cada linha da tabela é **única e possui** um atributo identificador (Num_Matrícula);

- Este atributo identificador é chamado de **chave primária.**
- **Regras:**

Em uma tabela não devem existir linhas duplicadas;

As linhas de uma tabela não seguem uma ordem especifica.

# Modelo Relacional

- A tabela Aluno possui três atributos;
- Para cada atributo existe um conjunto de valores permitidos chamado domínio daquele atributo:
- Para o atributo Num_Matrícula o domínio é o conjunto de números naturais;
- Para o atributo Nome_Aluno o domínio é qualquer nome válido;
- Enquanto que para Sexo_Aluno o domínio são os **mnemônicos Mou F.**

# SGBD

- Para a criação de banco de dados, tabelas e atributos em um SGBD, utilizaremos a linguagem SQL que é compostas de comandos de **manipulação, definição e controle de dados.**

# DDL

- Esses conjuntos de comandos de definição de dados são denominados pela sigla ***DDL(Data Definition Language*)**, que disponibiliza um conjunto de comandos para:
- criação(**CREATE**),
- alteração(**ALTER**) e
- remoção (**DROP**) de tabelas e outras estruturas.

# Comando CREATE DATABASE

- A maioria dos SGBDs disponibiliza ferramentas que permitem a criação de Bancos de Dados, mas é possível criar o próprio Banco de Dados a partir de um comando SQL.
- A sintaxe do comando é:

  **CREATE DATABASE** nome_do_banco_de_dados

  **CREATE DATABASE** IFBA

# Comando DROP DATABASE

- O comando DROP DATABASE permite remover um determinado Banco de Dados, apagando todas as tabelas e estruturas associadas e, consequentemente, todos os dados existentes nelas
- A sintaxe do comando é:

  **DROP DATABASE** nome_do_banco_de_dados

  **DROP DATABASE** IFBA

# Comando CREATE TABLE

- O comando CREATE TABLE é o principal comando DDL da linguagem SQL. A criação de tabelas é realizada em SQL utilizando este comando.
- Sua sintaxe básica é a seguinte:

**CREATE TABLE** nome_da_tabela(Coluna1 Tipo,Coluna2 Tipo, ColunaN Tipo)

**CREATE TABLE** Empregado(Id INTEGER, Nome CHAR(50), Data_Nasc DATE, Salario FLOAT)

# Comando CREATE TABLE

**CREATE TABLE** Empregado(Id INTEGER, Nome CHAR(50), Data_Nasc DATE, Salario FLOAT)

| Id | Nome | Data_Nasc | Salario |
|---|---|---|---|
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |

# Exemplo 01

Escrever um comando de SQL que permita criar uma tabela com o nome Caixa_Postal, capaz de armazenar um inteiro de até quatro dígitos e uma string com 45 caractere.

# Exemplo 01

**CREATE TABLE** Caixa_Postal (Codigo NUMERIC, Localidade CHAR(45))

| Codigo | Localidade |
| --- | --- |
| | |
| | |
| | |

# Exemplo 02

Escrever um comando de SQL que permita criar uma tabela com o nome **Pessoa**, com o seguintes atributos: **ID, Nome, Idade, Salario, Telefone e Código Postal).**

# Exemplo 02

**CREATE TABLE** Pessoa (Codigo INTEGER, nome CHAR(45), idade INTEGER, salario NUMERIC(10,2), telefone CHAR(12), Codigo_Postal CHAR(9)

| Codigo | Nome | Idade | Salario | Telefone | Codigo_Postal |
|---|---|---|---|---|---|
| | | | | | |
| | | | | | |
| | | | | | |

# Exercício Prático para entregar no Sapin

- Escrever um comando para criar uma base de dados com o nome de **ESCOLA.**

**Criar duas tabelas:**

- A primeira tabela deverá ter o nome **DISCIPLINA** com os atributos: id, nome, carga horária, período e ementa,
- A segunda tabela deverá ter o nome **LIVROS**, com os atributos: id, isbn, nome, autor, editora, edicao, disciplina, data de aquisição

# Exercício Prático

- Ao final depois de enviar o trabalho no Sapin, exclua as duas tabelas e o banco de dados criado.

# Características das Colunas

- Para execução do comando **CREATE TABLE** é necessário indicar qual o nome da tabela e, para cada uma das colunas, o nome da coluna e o tipo de dados.

- No entanto, podem ser indicadas as características próprias de cada uma das colunas, tais como: Que valores pode admitir ?Qual o valor padrão? O campo representa um atributo identificador(chave primária)?

# Atributos NOT NULL

- **No exemplo apresentado anteriormente:**

**CREATE TABLE** Caixa_Postal (Codigo NUMERIC, Localidade CHAR(45))

Estamos admitindo que a tabela é composta por duas colunas(código e localidade) e que qualquer uma delas pode admitir valores nulos(ou seja, o usuário poderá informar dados vazios para os campos).

- Se quisermos que uma coluna não admita valores nulos, usamos a cláusula **NOT NULL**

# Valores por padrão(default)

- Caso um valor não seja inserido em uma coluna o valor padrão (default) armazenado nela é NULL. No entanto, é possível associar um outro valor default através da cláusula **DEFAULT.**
- Se quisermos por exemplo que a localidade padrão (default) se chame **Ilhéus**, então teremos que fazer o seguinte:

# Valores por padrão(default)

- **Veja  o exemplo:**

**CREATE TABLE** Caixa_Postal (Codigo NUMERIC NOT NULL, Localidade CHAR(45))

- O código acima está **enfatizando que** o atributo **Codigo** da tabela **Caixa Postal**, **NÃO ACEITA** valores nulos, ou seja, o campo Código deverá possuir **SEMPRE**, qualquer valor diferente de vazio.

**CREATE TABLE** Caixa_Postal (Codigo NUMERIC NOT NULL, Localidade CHAR(45) DEFAULT 'Ilhéus').

# Restrições (constraints)

- Restrições são regras a que os valores de uma ou mais colunas devem obedecer. Por exemplo, o conteúdo da coluna Sexo só poderá conter os valores “F” ou “M”, a coluna Idade não poderá conter valores negativos, o salário não poderá ser inferior ao salário mínimo(R$ 510,00).
- A utilização de restrições é a única garantia que temos de que os dados existentes nas colunas estão de acordo com as regras especificadas no projeto do Banco de Dados

# Restrições (constraints)

- **Existem alguns tipos distintos de restrições que se podem aplicar a colunas:**

- Constraint **NOT NULL**
- Constraint **CHECK**
- Constraint **UNIQUE**
- Constraint **PRIMARY KEY**
- Constraint **REFERENCES**

# Constraint CHECK

- A constraint **CHECK** permite realizar a validação dos dados inseridos na coluna, através da especificação de uma condição. São admitidos apenas os dados cujo resultado da avaliação da condição seja verdadeiro.

- No slide a seguir veremos alguns exemplos utilizando a constraint **CHECK**

# Constraint CHECK

- **Exemplos com constraints**

```
CREATE TABLE Dados_Pessoais
(
Codigo NUMERIC NOT NULL,
Nome CHAR(60) CHECK(Nome NOT LIKE '%Regilan%'),
Idade INTEGER NOT NULL CHECK(Idade >= 0 AND Idade <=
150),
Sexo CHAR CHECK (SEXO IN('M', 'F')),
Tempo_Servico INTEGER CHECK(Tempo_Servico >= 0)
)
```

# Constraint UNIQUE

- A constraint **UNIQUE** indica que os valores dessa coluna não podem se repetir.
- Em uma tabela podem existir tantas colunas UNIQUE quantas forem necessárias.

# Constraint UNIQUE

- **Veja o exemplo:**

```
CREATE TABLE Dados_Pessoais
(
Codigo NUMERIC NOT NULL,
Nome CHAR(60) UNIQUE,
CPF CHAR(15) UNIQUE,
Tempo_Servico INTEGER CHECK(Tempo_Servico
>= 0)
)
```

# Comando DROP TABLE

- O comando **DROP TABLE** permite remover uma determinada tabela de um Banco de Dados, e conseqüentemente, todos os dados existentes nela.
- **A sintaxe do comando é:**

**DROP TABLE** nome_da_tabela

**DROP TABLE** Dados_Pessoais

# Obrigado pela atenção

## Sigo à disposição pelo e-mail:

marcio.lemos@senairs.org.br