



## **Apostila Fundamentos de Banco de Dados**

Este documento é propriedade intelectual do Núcleo de Educação a distância da NRsystem e distribuído sob os seguintes termos:

1. As apostilas publicadas pelo do Núcleo de Educação a distância da NRsystem podem ser reproduzidas e distribuídas no todo ou em parte, em qualquer meio físico ou eletrônico, desde que os termos desta licença sejam obedecidos, e que esta licença ou referência a ela seja exibida na reprodução.
2. Qualquer publicação na forma impressa deve obrigatoriamente citar, nas páginas externas, sua origem e atribuições de direito autoral (o Núcleo de Educação a distância da NRsystem e seu(s) autor(es)).
3. Todas as traduções e trabalhos derivados ou agregados incorporando qualquer informação contida neste documento devem ser regidas por estas mesmas normas de distribuição e direitos autorais. Ou seja, não é permitido produzir um trabalho derivado desta obra e impor restrições à sua distribuição. O Núcleo de Educação a distância da NRsystem deve obrigatoriamente ser notificado ([nrsystem@nrsystem.br](mailto:nrsystem@nrsystem.br)) de tais trabalhos com vista ao aperfeiçoamento e incorporação de melhorias aos originais.

Adicionalmente, devem ser observadas as seguintes restrições:

- A versão modificada deve ser identificada como tal
- O responsável pelas modificações deve ser identificado e as modificações datadas
- Reconhecimento da fonte original do documento
- A localização do documento original deve ser citada
- Versões modificadas não contam com o endosso dos autores originais a menos que autorização para tal seja fornecida por escrito.

A licença de uso e redistribuição deste material é oferecida sem nenhuma garantia de qualquer tipo, expressa ou implícita, quanto a sua adequação a qualquer finalidade. O Núcleo de Educação a distância da NRsystem não assume qualquer responsabilidade sobre o uso das informações contidas neste material.

The logo for NRSYSTEM, featuring the word "NRSYSTEM" in a bold, white, sans-serif font. The letters are slightly shadowed, giving them a three-dimensional appearance as if they are floating or attached to a surface. The background is a dark blue gradient with several semi-transparent, glowing squares of various sizes and shades of blue and white, similar to the image at the top of the page.

**NRSYSTEM**

## SUMÁRIO

INTRODUÇÃO.....	3
1. MODELOS DE DADOS .....	3
1.1 Modelo Entidade-Relacionamento .....	3
2. HISTÓRICO EVOLUTIVO DOS BANCO DE DADOS .....	4
2.1 Tipos de bancos de dados .....	4
3. ABSTRAÇÃO DE DADOS .....	5
4. DATABASE MANAGEMENT SYSTEM .....	6
5. PROPRIEDADES IMPLÍCITAS E PERFÍS DE USUÁRIOS DE BANCO DE DADOS ....	7
6. MODELO ENTIDADE RELACIONAMENTO (MER) .....	8
6.1 Tipos de Atributos mais usados.....	9
6.2 Cardinalidade de Relacionamentos.....	9
7. MER ESTENDIDO (EXPANDIDO) .....	10
8. O QUE É NORMALIZAÇÃO EM BANCO DE DADOS? .....	12
9. LINGUAGEM DE BANCO DE DADOS .....	16
9.1 Data Definition Language.....	18
9.2 Data Manipulation Language .....	20
9.3 Data Control Language .....	20
10. ESTRUTURAS EXISTENTES EM BANCOS DE DADOS .....	21
SUGESTÕES DE LEITURA.....	22

## INTRODUÇÃO

Muitos autores definem banco de dados de forma diferente, porém em todas elas tem-se uma ideia de coleção ou conjunto de dados armazenados que servem ou são usados em algumas situações específicas. Algumas considerações:

- Um arranjo aleatório de dados não pode ser considerado um banco de dados.
- Banco de dados são conjuntos de dados relacionados e acessíveis. Dados são fatos conhecidos e representam informações sobre um domínio específico.
- Os Bancos de dados (BD) desempenham um papel crítico em muitas áreas onde computadores são utilizados, principalmente nas áreas administrativa e comercial.
- A utilização de banco de dados permite o compartilhamento e manutenção de dados consistentes por diversas aplicações.

### 1. MODELOS DE DADOS

Os modelos de dados são classificados em três grupos: modelos de dados físicos; modelos lógicos baseados em objetos e de visão. Um modelo de dados nos permite descrever um projeto de banco de dados apoiando-se em ferramentas conceituais que podem ser usadas para descrever os dados, suas relações, semântica e restrições dos mesmos.

Embora existam vários modelos de dados diferentes, por exemplo: Modelo Relacional, Modelo Entidade-Relacionamento, Modelo de dados baseado em objeto, Modelo de dados semiestruturado, esta apostila tem como objetivo abordar o Modelo Entidade-Relacionamento, este é o modelo mais utilizado atualmente.

#### 1.1 Modelo Entidade-Relacionamento

Baseado na percepção do mundo real “minimundo”. Mudanças no minimundo provocam alterações na base de dados. O modelo Entidade-Relacionamento consiste na coleção de objetos básicos, chamados *entidades*<sup>1</sup> e nos *relacionamentos* existentes entre esses objetos.

---

<sup>1</sup> Uma entidade (entity) é um objeto que existe no mundo real e é distinguível dos outros objetos.

## 2. HISTÓRICO EVOLUTIVO DOS BANCO DE DADOS

- ✓ Até 1960: Sistema de Arquivos (Pascal, C, etc.)
- ✓ Final de 1960 : Modelo Hierárquico - Exemplo: IMS (IBM)
- ✓ 1970 e início de 1980: Modelo de Redes Exemplo: IDMS, DMS-II (Unisys)
- ✓ Meados de 1980: Modelo Relacional (Codd) Exemplo: DB-2, SQL-DS (IBM), Oracle, Ingres,...
- ✓ Final de 1980: Modelo Orientado a Objetos e Objeto-Relacional Exemplo: Orion, Informix, Jasmine, Oracle...

### 2.1 Tipos de bancos de dados

**Banco de Dados não Relacional:** Modo regular, os arquivos são escritos de forma sequencial, o acesso geralmente é mais lento em comparação ao banco de dados Relacional. Estão em uso desde os anos 60, entretanto, novos tipos de bancos de dados não relacionais estão sendo desenvolvidos (NoSQL). A principal vantagem de um tipo de dados não relacional é que, diferente da abordagem tradicional, eles lidam bem com dados desestruturados (e-mails, dados multimídia e dados provenientes de mídias sociais).

**Banco de Dados Relacional:** Os dados são organizados em tabelas permitindo o relacionamento entre as mesmas. Uma relação trata-se de associação entre duas ou mais entidades. Em comparação ao Modelo Não Relacional, podemos citar como principais vantagens: padrão adotado mundialmente, maior velocidade de acesso aos dados e menor espaço de armazenamento.

**Exemplo de utilização:** Uma Faculdade/Universidade possui um ou mais Campus, o(s) Campus, possuem um ou mais cursos, os cursos por sua vez, são formados por Turmas com vários alunos. A figura 1 ilustra o BD relacional que atende esta necessidade.

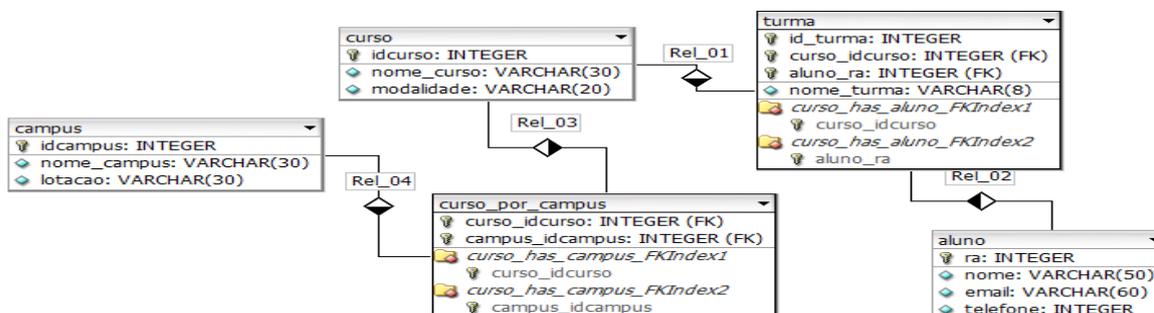


Figura 1 - Exemplo de banco de dados relacional

### 3. ABSTRAÇÃO DE DADOS

Em virtude do grande número de usuários de BD que não são treinados em computação, faz-se necessário simplificar sua estrutura para melhor interação entre usuários e sistema. O grande objetivo de um sistema de BD é oferecer uma visão “abstrata” dos dados aos usuários.

O conceito de abstração está associado à característica de se observar somente os aspectos de interesse, sem se preocupar com maiores detalhes envolvidos. No contexto de abstração de dados, um BD pode ser visto sem se considerar a forma como os dados estão armazenados fisicamente.

A figura 2 exemplifica os três níveis gerais de abstração de um banco de dados: Lógico, Físico e Visão.

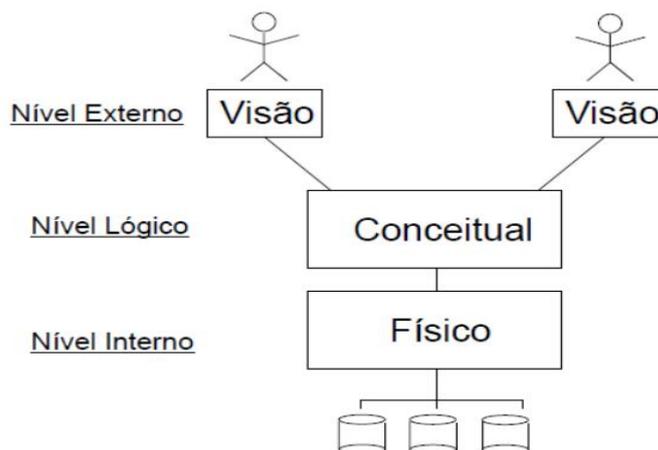


Figura 2 - arquitetura de um banco de dados em três níveis

**Visão** - Nível de abstração mais alto (considerada a visão do grupo de usuários) descreve apenas parte do banco de dados, muitos usuários não precisam de todas as informações sobre o banco de dados.

**Nível Lógico** (Visão Conceitual) - nível de abstração intermediário, descreve quais dados estão armazenados e que relação existe entre eles (descreve o banco de dados inteiro).

**Nível Físico** (Visão Interna) - nível de abstração mais baixo, visão do responsável pela manutenção e desenvolvimento do SGBD. Neste nível existe a preocupação de como os dados serão armazenados.

Embora os detalhes referentes à forma como dados estejam armazenados e mantidos não interessem a maioria dos usuários, a disponibilidade e eficiência para manipulação destes dados são fundamentais para sua utilização. A figura 3 mostra o processo de abstração de dados.

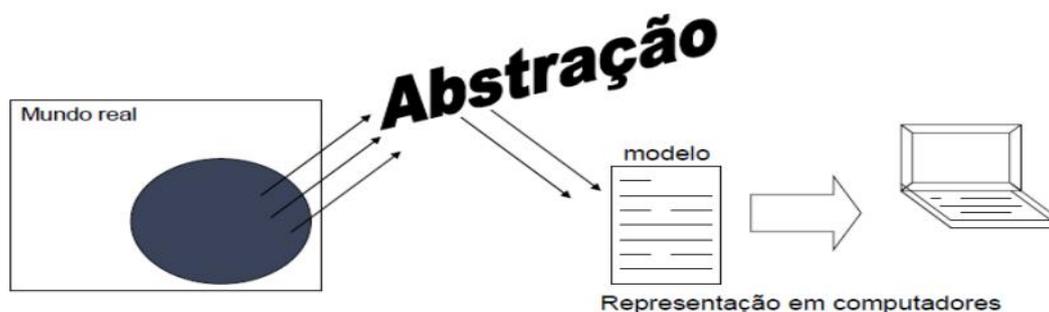


Figura 3 - Abstração de dados

#### 4. DATABASE MANAGEMENT SYSTEM

O sistema de gerenciamento de banco de dados (SGBD) não deve ser confundido com o próprio banco de dados; um SGBD é constituído por um conjunto de dados associados a um conjunto de programas para acesso a esses dados. A função de gravar uma informação, alterá-la ou até mesmo recuperá-la é do banco de dados, cabe ao sistema de gerenciamento permitir ou não o acesso ao banco de dados.

O sistema de gerenciamento pode não trazer grandes benefícios a bancos de dados pequenos, simples e de pouco acesso, porém, ele é vital para bancos de dados com grande volume de informações e com acessos simultâneos por vários usuários.

## 5. PROPRIEDADES IMPLÍCITAS E PERFÍS DE USUÁRIOS DE BANCO DE DADOS

- ✓ Um banco de dados é uma coleção logicamente coerente de dados com algum significado inerente;
- ✓ Um banco de dados é projetado e construído com dados para um propósito específico;
- ✓ Ele possui um grupo de usuários e algumas aplicações pré-concebidas, as quais esses usuários estão interessados;
- ✓ Um banco de dados representa algum aspecto do mundo real e a alteração neste mundo real tem que ser refletida no banco de dados.

Em um pequeno banco de dados de uso pessoal, uma única pessoa vai definir, construir e manipular o BD. Por outro lado, em um grande banco de dados com muitos (ou milhões) de usuários e com restrições de acesso, podemos identificar diferentes perfis de pessoas que interagem com o banco de dados:

- Administrador do Banco de Dados (DBA)
- Projetista do Banco de Dados
- Analista de Sistemas
- Programador de Aplicações
- Usuário (final)

<b>Analista de Sistemas</b> Determina os requisitos dos usuários e desenvolve especificações que atendam a estes requisitos.	<b>Usuário (final)</b> Um banco de dados existe para a utilização do usuário final, onde normalmente seu trabalho requer consultas e atualizações. A maioria dos usuários utilizam programas voltados ao desempenho profissional, utilizando-os no seu dia-a-dia.
<b>Programadores</b> Implementam as especificações na forma de programas elaborando toda a documentação.	
<b>Administrador de Dados (DBA)</b> É o supervisor do banco de dados, responsável pela autorização de acesso ao banco, monitoramento e coordenação de uso. Está envolvido com os aspectos físicos do banco de dados (estruturas de armazenamento, métodos de acesso, etc.).	<b>Projetista do banco</b> Responsável pela identificação dos dados e elaboração de estruturas apropriadas para armazená-los. Compreende os requisitos necessários ao grupo de usuários do banco de dados antes de sua implementação.

Quadro 1 - Perfis de usuários de um banco de dados

## 6. MODELO ENTIDADE RELACIONAMENTO (MER)

O MER (Modelo-Entidade-Relacionamento) é um modelo de dados conceitual de alto-nível, ou seja, seus conceitos foram projetados para serem compreensíveis aos usuários, descartando detalhes de como os dados são armazenados. Este modelo foi desenvolvido a fim de facilitar o projeto de banco de dados permitindo a especificação de um esquema que representa a estrutura lógica global do Banco de Dados.

Atualmente, o MER é usado principalmente durante o processo de projeto da base de dados.

A representação gráfica do MER é exibida na figura 4



Figura 4 - Elementos gráficos do MER

**Entidade** - Identifica o objeto de interesse do sistema e tem "vida" própria, ou seja, a representação abstrata de um objeto do mundo real sobre o qual desejamos guardar informações. Uma Entidade é algo da realidade sendo modelada e deve ser identificada de modo único.

**Relacionamento** - Associação entre entidades ou com a mesma entidade (auto relacionamento).

Exemplos de Entidades em um BD: Clientes, Fornecedores, Alunos, Funcionários, Departamentos, etc.

**Atributo** - Informações que desejamos guardar sobre a instância de entidade<sup>2</sup> (características que possibilitam diferenciar as entidades). Exemplo: Nome do aluno, Código da turma, Endereço do fornecedor, Sexo do funcionário, etc.

---

<sup>2</sup> Instância ou ocorrência de Entidade - São os elementos da entidade. Exemplo: Aluno nº 10, Funcionário João, etc.

## 6.1 Tipos de Atributos mais usados

**Simple (monovalorado):** Tem valor único, como por exemplo, o número da rua.

**Composto:** Pode ser referenciado de duas formas, hora no todo, hora em partes. Ex.: Endereço, composto por rua, numero, cidade, CEP.

**Multivalorado:** Podem existir instâncias em que um atributo possua um conjunto de valores para uma única entidade. Por exemplo, o atributo telefone ou e-mail podem possuir mais de um valor atribuído.

**Determinante:** Define o campo que será a chave primária na tabela. Identificador único.

**OBS.** Existem ainda os atributos: nulo e derivado.

## 6.2 Cardinalidade de Relacionamentos

Cardinalidade representa a frequência com que existe o relacionamento, em outras palavras, consiste no número máximo de relacionamento entre as entidades.

### TIPOS DE RELACIONAMENTO:

Relacionamento 1 para 1 (1:1)

Relacionamento 1 para muitos (1:N)

Relacionamento muitos para muitos (N:M)

Exemplos de relacionamentos do tipo 1:N e N:M são mostrados na figura 5.

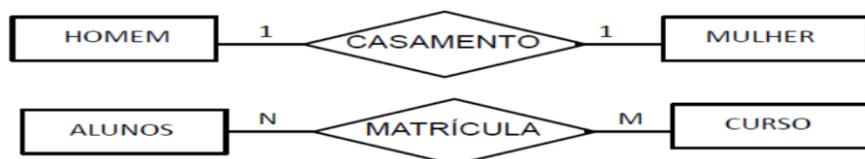


Figura 5 – Exemplos de relacionamentos

## 7. MER ESTENDIDO (EXPANDIDO)

Características:

- Introduce semântica adicional ao MER
- Utilizado na modelagem de aplicações mais complexas, tais como CAD/CAM, BD gráficos, BD geográficos.

Conceitos:

- Subclasse, superclasse, hierarquia, herança.
- Generalização, especialização.
- Agregação.

A figura 6 representa o conceito de MER Estendido

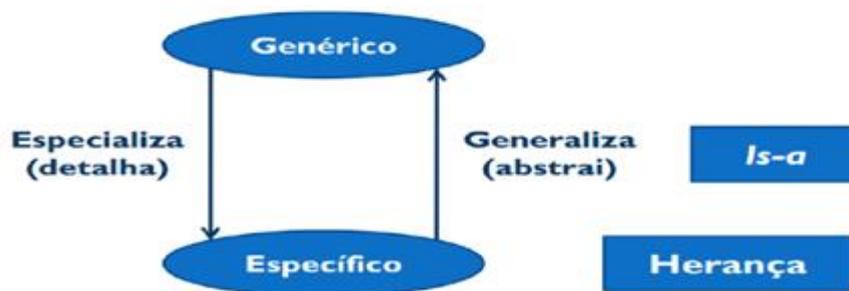


Figura 6 - MER Estendido (Generalização/especialização)

No MER Estendido, os conceitos superclasse (supertipo), subclasse (subtipo), herança, generalização, especialização estão intimamente relacionados.

### Subclasse/ Superclasse

- Subclasse (subtipo):
  - Subconjunto de entidades
  - Resulta do agrupamento de entidades em subgrupos de um tipo-entidade
- Outro exemplo:
  - Superclasse (supertipo): tipo-entidade Empregado.
  - Subclasses (subtipos): Secretário, Engenheiro, Técnico.

### Generalização / Especialização

No **MER** Estendido, a entidade generalista (superclasse) recebe os atributos que são comuns para todas as subclasses (especialista), os atributos da superclasse são herdados pelas subclasses. O exemplo apresentado na figura 7 ilustra o conceito de herança de atributos.

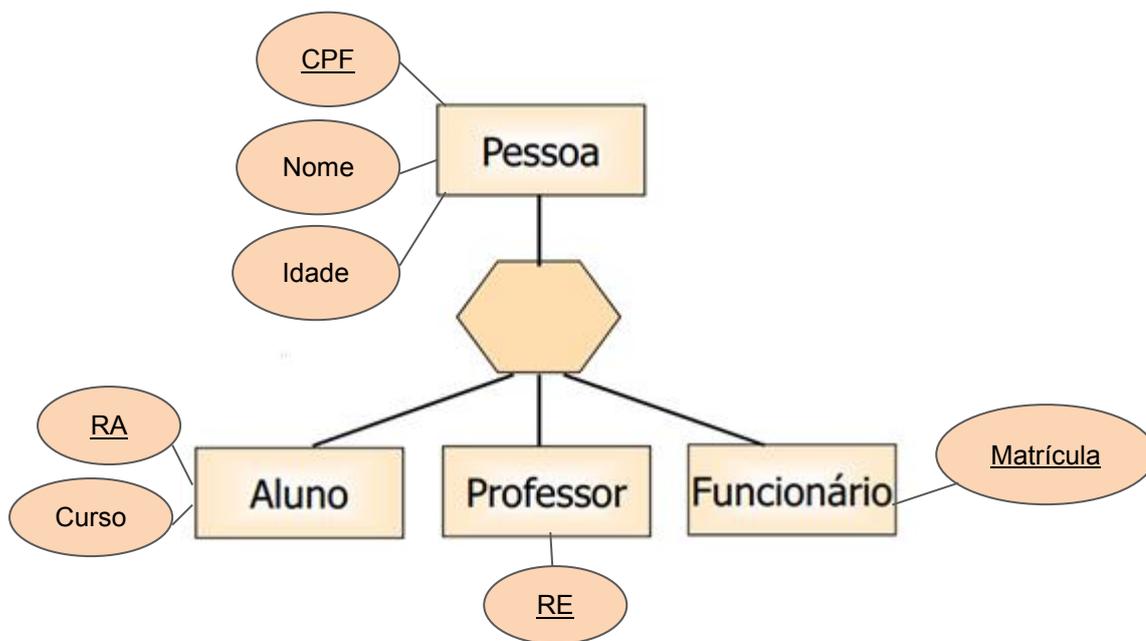


Figura 7 – Herança dos atributos da entidade generalista pelas entidades especialistas

### Agregação

Uma entidade do tipo agregação, pode englobar:

- Dois tipos entidades e um tipo relacionamento;
- Dois tipos-relacionamentos e um tipo entidade.

Exemplo: O tipo-entidade *Cursos* é composto dos tipos-entidade *Turmas* e *Alunos* e do tipo-relacionamento *Tem*.

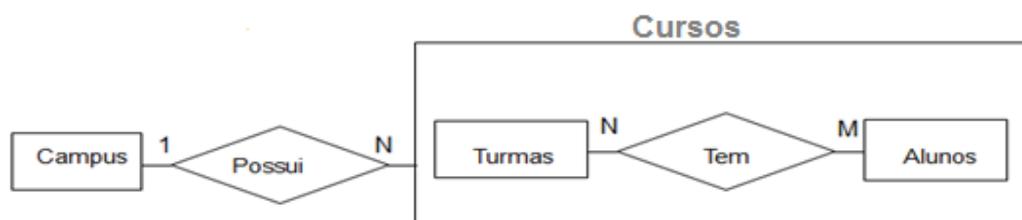


Figura 8 - Exemplo de agregação

## 8. O QUE É NORMALIZAÇÃO EM BANCO DE DADOS?

Normalização é o processo onde se aplica regras a todas as entidades (tabelas) do banco de dados a fim de evitar falhas no projeto, como por exemplo, inconsistência e redundância de dados.

Passos a serem aplicados para consolidação da 1FN:

- Identificação da chave primária da tabela;
- Identificação da coluna que contem dados repetidos e removê-las;
- Criação de uma nova tabela com chave primária para armazenamento do dado repetido;
- Criar uma relação entre a tabela principal e a tabela secundária.

### 1FN – PRIMEIRA FORMA NORMAL

Uma entidade estará na 1FN, se e somente se:

- Todos os seus atributos forem atômicos;
- Não existam conjuntos de atributos repetidos descrevendo a mesma característica.

O quadro 2 apresenta um exemplo de tabela que não atende a 1FN:

idCliente	nomeCliente	Endereco	Telefones
1	Carrefour	Av. Pedro Américo, 23 09110-560 Vila Homero Thon	4979-8801 4458-2089
2	ABR	Rua Nove, 2 22499-000 Leblon	2222-1100 2321-2299
3	Pentol	Rua Conti, 120 23454-231 Ipanema	2333-1188

Quadro 2 - Exemplo de tabela Clientes (não normalizada)

**Análise:** Todos os clientes possuem Rua, CEP e Bairro e os dados estão sendo armazenados na mesma coluna (Endereço), isto fere o princípio de valores atômicos. Podemos também identificar que em alguns registros, o campo Telefones está armazenando mais de 1 número de telefone por cliente.

No exemplo anterior (quadro 2) existem atributos multivalorados para endereços e telefones, neste caso, para que a tabela atenda a primeira forma normal é necessário que haja um “desmembramento”.

O quadro 3 mostra a tabela Clientes após normalização de acordo com a 1FN (1ª forma normal)

<b>idCliente</b>	<b>nomeCliente</b>	<b>Rua</b>	<b>Cep</b>	<b>Bairro</b>
1	Carrefour	Av. Pedro Américo, 23	09110-560	Vila Homero Thon
2	ABR	Rua Nove, 2	22499-000	Leblon
3	Pentol	Rua Conti, 120	23454-231	Ipanema

**Quadro 3 - Tabela Clientes normalizada**

Após o desmembramento, uma segunda tabela foi gerada (TelefoneClientes). Como todos os campos devem ser dependentes de uma chave, neste caso, foi criada a coluna **idClienteTelefone** como chave primária.

Importante notar que a coluna **idCliente** na tabela TelefonosClientes cumpre o papel de chave estrangeira, isso nos permite relacionar o número de telefone ao respectivo cliente, conforme exibido no quadro 4.

<b>idClienteTelefone</b>	<b>idCliente</b>	<b>Telefone</b>
1	1	4978-8801
2	1	4458-2089
3	2	2222-1100
4	2	2321-2299
5	3	2333-1188

**Quadro 4 - TelefonosClientes**

## 2FN – SEGUNDA FORMA NORMAL

Uma entidade está na 2FN, se e somente se, estiver na 1FN e todos seus atributos (colunas) não chaves, dependam unicamente da chave primária.

Passos a serem aplicados para consolidação da 2FN:

- Identificar colunas que não são funcionalmente dependentes da chave primária da tabela;
- Remover a coluna da tabela e criar uma nova tabela com esses dados.

O exemplo apresentado no quadro 5 exibe um exemplo de tabela para controle de pedidos que não atende a 2FN.

idPedido	dataPedido	codProduto	nomeProduto	qtde	valorUnitario	valorTotal
1	22/11/2012	1234	HD sata 320GB Samsung	2	190	380
2	22/11/2012	1240	Placa Mãe Asus P8h61-LGA 1155 s/r Microatx	3	235	705
3	22/11/2012	1242	Mini mouse óptico USB Bright	5	20	100

Quadro 5 - Tabela de Controle de Pedidos não normalizada

**Análise:** Neste caso, temos uma tabela que armazena dados de vendas de produtos. Para aplicar a 2FN, devemos “separar” os dados dos produtos, pois estes não dependem da coluna chave, neste caso, devemos criar uma entidade que contenham somente dados dos produtos.

O quadro 6 exibe a tabela Controle de Pedidos após sua normalização.

idPedido	dataPedido	codProduto	qtde	valorUnitario	valorTotal
1	22/11/2012	1234	2	190	380
2	22/11/2012	1240	3	235	705
3	22/11/2012	1242	5	20	100

Quadro 6 - Tabela Controle de Pedidos normalizada

A normalização da tabela Controle de Pedidos gerou uma segunda tabela (Produtos) conforme quadro 7. Para podermos relacionar a tabela Produtos com a tabela Controle de pedidos, a chave primária da tabela Produtos (codProduto) deve ser chave estrangeira na tabela Controle de Pedidos.

codProduto	nomeProduto
1234	HD sata 320GB Samsung
1240	Placa Mãe Asus P8h61-LGA 1155 s/r Microatx
1242	Mini mouse óptico USB Bright

Quadro 7 - Tabela Produtos

### 3FN – TERCEIRA FORMA NORMAL

Uma entidade está na 3FN, se e somente se, estiver na 2FN e todos os atributos (colunas) não chave, forem mutuamente independentes não havendo dependência funcional entre elas; todas dependem única e exclusivamente da chave primária de forma irredutível.

Passos a serem aplicados para consolidação da 3FN:

- Identificar as colunas que são funcionalmente dependentes das outras colunas não chave;
- Remover essas colunas

O quadro 8 apresenta a tabela Controle de Pedidos (estudada anteriormente), ela não atende a 3FN, pois existem campos com dependência funcional.

idPedido	dataPedido	codProduto	qtde	valorUnitario	valorTotal
1	22/11/2012	1234	2	190	380
2	22/11/2012	1240	3	235	705
3	22/11/2012	1242	5	20	100

Quando 8 - Tabela Controle de Pedidos

**Análise:** temos duas colunas indicando os valores do produto, estes campos possuem dependência funcional, neste caso, devemos remover a coluna valorUnitario e acrescentá-la a tabela Produtos.

## 9. LINGUAGEM DE BANCO DE DADOS

Um sistema de banco de dados (BD) deve proporcionar dois tipos de linguagens: uma específica para as estruturas do BD e outra para expressar consultas e atualizações nas estruturas. A linguagem mais utilizada em banco de dados é a SQL (Strutured Query Language)

### Tipos de Dados SQL

Abaixo segue uma relação como os tipos de dados básicos do SQL Server, sendo que, os tipos que estiverem marcados com \* somente funcionam a partir do SQL Server 2000

- TINYINT: Valores numéricos inteiros variando de 0 até 256.
- SMALLINT: Valores numéricos inteiros variando de -32.768 até 32.767.
- INT: Valores numéricos inteiros variando de -2.147.483.648 até 2.147.483.647.
- \*BIGINT: Valores numéricos inteiros variando de -92.23.372.036.854.775.808 até 9.223.372.036.854.775.807.
- BIT: Somente pode assumir os valores 0 ou 1. Utilizado para armazenar valores lógicos.
- DECIMAL (I,D) e NUMERIC(I,D): Armazenam valores numéricos inteiros com casas decimais utilizando precisão. I deve ser substituído pela quantidade de dígitos total do número e D deve ser substituído pela quantidade de dígitos da parte decimal (após a vírgula). DECIMAL e NUMERIC possuem a mesma funcionalidade, porém DECIMAL faz parte do padrão ANSI e NUMERIC é mantido por compatibilidade. Por exemplo, DECIMAL (8,2) armazena valores numéricos decimais variando de - 999999,99 até 999999,99.
- SMALLMONEY: Valores numéricos decimais variando de -214.748,3648 até 214.748,3647.
- MONEY: Valores numéricos decimais variando de -922.337.203.685.477,5808 até 922.337.203.685.477,5807.
- REAL: Valores numéricos aproximados com precisão de ponto flutuante, indo de - 3.40E + 38 até 3.40E + 38.
- FLOAT: Valores numéricos aproximados com precisão de ponto flutuante, indo de -1.79E + 308 até 1.79E + 308.
- SMALLDATETIME: Armazena hora e data variando de 1 de janeiro de 1900 até 6 de junho de 2079. A precisão de hora é armazenada até os segundos.
- DATETIME: Armazena hora e data variando de 1 de janeiro de 1753 até 31 de Dezembro de 9999. A precisão de hora é armazenada até os centésimos de segundos.
- CHAR(N): Armazena N caracteres fixos (até 8.000) no formato não Unicode. Se a quantidade de caracteres armazenada no campo for menor que o tamanho total especificado em N, o resto do campo é preenchido com espaços em branco.
- VARCHAR(N): Armazena N caracteres (até 8.000) no formato não Unicode. Se a quantidade de caracteres armazenada no campo for menor que o tamanho total

- especificado em N, o resto do campo não é preenchido.
- TEXT: Armazena caracteres (até 2.147.483.647) no formato não Unicode. Se a quantidade de caracteres armazenada no campo for menor que 2.147.483.647, o resto do campo não é preenchido. Procure não utilizar este tipo de dado diretamente, pois existem funções específicas para trabalhar com este tipo de dado.
  - NCHAR(N): Armazena N caracteres fixos (até 4.000) no formato Unicode. Se a quantidade de caracteres armazenada no campo for menor que o tamanho total especificado em N, o resto do campo é preenchido com espaços em branco.
  - NVARCHAR(N): Armazena N caracteres (até 4.000) no formato Unicode. Se a quantidade de caracteres armazenada no campo for menor que o tamanho total especificado em N, o resto do campo não é preenchido.
  - NTEXT: Armazena caracteres (até 1.073.741.823) no formato Unicode. Se a quantidade de caracteres armazenada no campo for menor que 1.073.741.823, o resto do campo não é preenchido. Procure não utilizar este tipo de dado diretamente, pois existem funções específicas para trabalhar com este tipo de dado.

### Structured Query Language

O SQL foi desenvolvido originalmente no início dos anos 70 nos laboratórios da IBM em San Jose, dentro do projeto de um sistema de Banco de Dados, o System R, que tinha por objetivo demonstrar a viabilidade da implementação do modelo relacional proposto por Edgar Frank Codd matemático britânico;

Atualmente, o SQL é o padrão adotado em banco de dados mundialmente, trata-se de uma linguagem declarativa, através dela especificamos como o “resultado” será apresentado, mas não definimos o caminho para chegar até ele. Os recursos disponibilizados pelo SQL são agrupados em 5 funcionalidades:



Figura 9- Funcionalidade da Linguagem SQL

## 9.1 Data Definition Language

Linguagem de Definição de Dados, por meio da DDL podemos definir estruturas do banco tais como: tabelas, visões, sequenciais e outras estruturas.

- O resultado no uso da DDL constitui em um arquivo especial chamado de dicionário ou diretório de dados.
- Um dicionário de dados é um arquivo de metadados<sup>3</sup>



Figura 10 – Esquema funcional de metadados

### Comandos da DDL:

CREATE (criação de estrutura), ALTER (alterar estrutura) e DROP (permite remover ou excluir uma estrutura).

**Exemplo 1:** Sintaxe para criação de banco de dados

```
CREATE DATABASE nome_banco;
```

**Exemplo 2:** Sintaxe para criação de tabela (Clientes) com chave primária simples

```
CREATE TABLE Clientes(
codigo INT NOT NULL AUTO_INCREMENT,
nome VARCHAR(50) NOT NULL,
rua VARCHAR(60) NOT NULL,
cep VARCHAR(15) NOT NULL,
bairro VARCHAR(60) NOT NULL,
PRIMARY KEY(Codigo)
);
```

<sup>3</sup> Metadados: dados a respeito de dados (informações adicionais). Em um sistema de Banco de Dados, esse arquivo ou diretório é consultado antes que o dado real seja manipulado

## CONSTRAINTS

Em SQL, o conceito de constraints consiste em limitar o tipo de dados que serão introduzidos em uma tabela (restrições).

### Tipos comuns de restrições:

**NOT NULL:** define que um campo da tabela é obrigatório (deve receber um valor na inserção de um registro);

**UNIQUE:** Garante que todos os valores numa coluna são diferentes;

**AUTO\_INCREMENT:** define que o sistema deverá gerar sequência incremental automaticamente;

**PRIMARY KEY:** define um campo ou conjunto de campos para identificar de forma única cada registro.

**OBS.** Quando um campo é definido como chave primária, seu valor não pode se repetir em registros diferentes.

### Exemplo 3: Criação de tabela com chave composta

```
CREATE TABLE Contas(  
Numero INT (6) NOT NULL,  
Saldo DECIMAL (6,2) NOT NULL,  
Agencia_Numero INT (5) NOT NULL,  
PRIMARY KEY(Numero, Agencia_Numero)  
);
```

### Exemplo 4: Criação de tabela com chave composta

```
CREATE TABLE Contas(  
Numero INT (6) NOT NULL,  
Saldo DECIMAL(6,2) NOT NULL,  
Agencia_Numero INT(5) NOT NULL,  
CONSTRAINT pk_Contas PRIMARY KEY(Numero,Agencia_Numero)  
);
```

## 9.2 Data Manipulation Language

Linguagem de manipulação de dados Principais comandos:

SELECT, INSERT, UPDATE, DELETE

Alguns autores definem que o comando SELECT faz parte de uma subdivisão chamada DQL (Data Query Language – Linguagem de Consulta de Dados).

**Exemplo 5:** Sintaxe do comando INSERT

```
INSERT INTO Clientes(nome, rua, cep, bairro)
VALUES ('João da Silva', 'Rua 7, 32', '09000-110', 'Vila Guilherme');
```

**OBS.** Neste exemplo, não existe a necessidade de declarar o campo código, pois o sistema irá gerar o código automaticamente (na tabela Clientes este campo é do tipo auto\_increment). Os valores devem ser declarados entre aspas simples toda vez que o dado utilizado for do tipo varchar.

**Exemplo 6:** Sintaxe do comando SELECT

```
SELECT codigo, nome FROM Clientes;
```

## 9.3 Data Control Language

Linguagem de Controle de Dados. Principais comandos: GRANT, REVOKE e DENY.

**GRANT:** habilita acesso a dados e operações (atribui permissão);

**REVOKE:** remove a permissão de acesso a dados e operações;

**DENY:** nega permissão a um usuário ou grupo para realizar operação em um objeto ou recurso.

**Exemplo 7:** Sintaxe do comando GRANT:

```
GRANT SELECT, UPDATE ON nome_tabela TO usuario, outro_usuario;
```

**Exemplo 8:** Sintaxe do comando REVOKE:

```
REVOKE SELECT, UPDATE ON nome_tabela TO usuario, outro_usuario;
```

## 10. ESTRUTURAS EXISTENTES EM BANCOS DE DADOS

**Visões (view):** Consultas SQL previamente programadas disponíveis para rápido acesso, não sevem para armazenar dados, sua função é armazenar critérios de seleção de dados, permitem dados atualizados sempre que as tabelas em questão sofrem alteração.

**Exemplo 9:** Sintaxe do comando WIEW

`CREATE VIEW DadosClientes AS`

**Índices:** Estruturas que gerenciam a ordenação de valores dos campos informados para melhorar a performance de processamento do banco de dados sobre estes campos e seus respectivos registros.

**Exemplo 10:** Sintaxe para criação de índice

`CREATE INDEX NomeIndex ON NomeRelaçãoR(Atributo);`

**Exemplo 11:** indexar a relação do empregado com base no departamento.

`CREATE INDEX EmpDeplIndex ON Empregado(Ndep);`

### DIAGRAMA SIMPLIFICADO DA ARQUITETURA DO SISTEMA DE BANCO DE DADOS

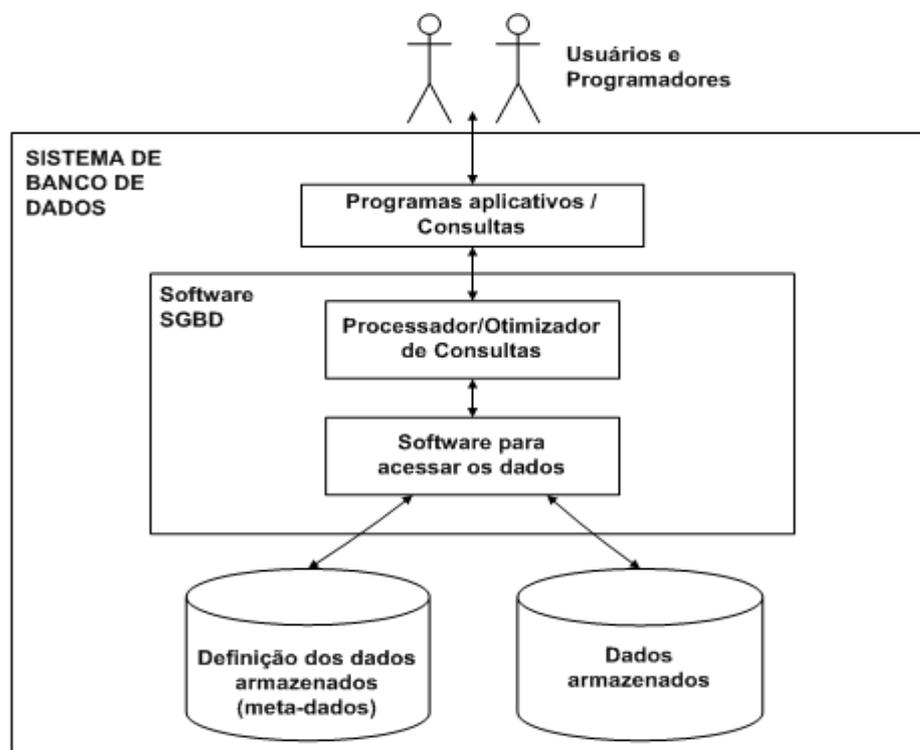


Figura 11 – arquitetura de um Sistema de Banco de Dados

## GLOSSÁRIO

**Sistemas de BD** - Sistema de software composto pelos programas de aplicação, pelo SGBD e pelo BD.

**SGBD** - Software para gerenciamento (controle de acesso) que utiliza linguagens de dados para manter o BD.

**Bancos de Dados** - Conjuntos de dados relacionados e acessíveis. Dados são fatos conhecidos e representam informações sobre um domínio específico.

## SUGESTÕES DE LEITURA

COSTA R.F. **SQL Básico** – Disponível em: [www.nrsystem.com.br/SQL.zip](http://www.nrsystem.com.br/SQL.zip)

DATE, C.J. **Introdução a Sistemas de Banco de Dados**, Rio de Janeiro, Editora Campus.

HEUSER, C. A. **Projeto de banco de Dados**, Editora Sagra Luzzatto.

KORTH, H. F. e Silberschatz, A. **Sistemas de Banco de Dados**, São Paulo.