

**Curso: Engenharia de Software**

**Disciplina: Database Design**

Prof. André Santos – [profAndre.Santos@fiap.com.br](mailto:profAndre.Santos@fiap.com.br)

# **Introdução**

## **Conceitos gerais sobre Bancos de Dados**

---

Este material de apoio é apenas um guia de estudo e não substitui a leitura da referência bibliográfica e a consulta de anotações de sala de aula.

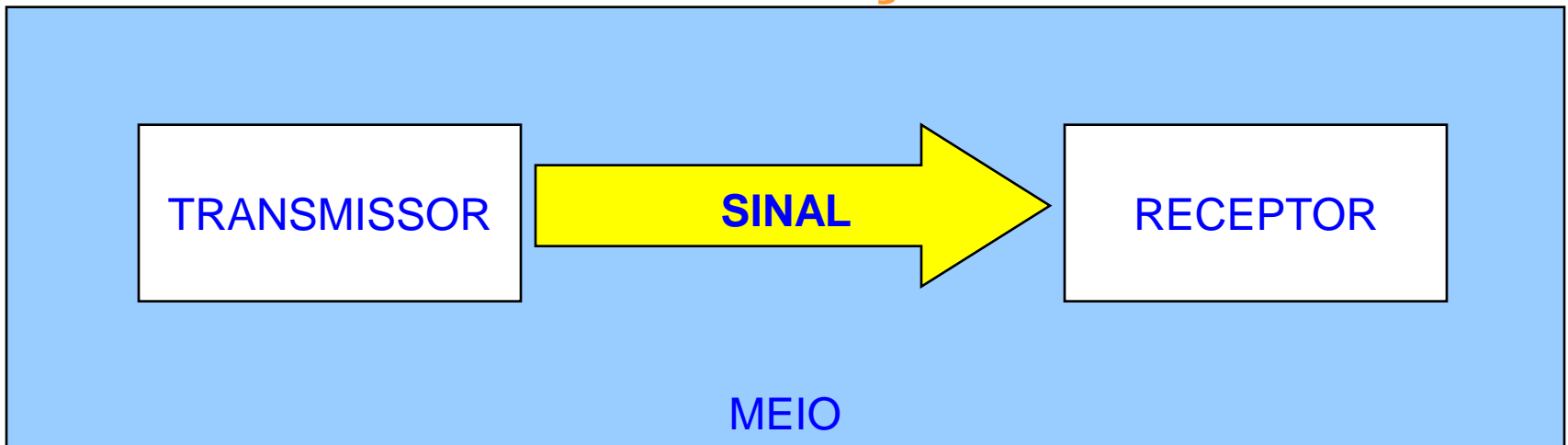
## Introdução – conceitos básicos

### Teoria da Informação

*Definição formal:*

- Informação se compõe de sinais que podem ser transmitidos.

### INFORMAÇÃO



Nota: O Dr. Claude E. Shannon (1916 – 2001) é considerado o criador da “Teoria da Informação”, através de artigo publicado em 1948.

## Introdução: conceitos básicos

- Exemplos :
  - Numa conversa, há duas pessoas que assumem alternadamente os papéis de transmissor e receptor, o sinal está na forma verbal codificado através da linguagem, além disso deve haver um meio para transmitir as ondas sonoras, que é o ar.
  - Uma escultura transmite informação visual, a imagem é composta por ondas luminosas através do espaço até um observador.
  - A dor é um tipo de informação, cujo transmissor é qualquer agente que sensibilize os nervos que são o meio por onde trafegam os sinais nervosos até o cérebro que os interpreta.

## Introdução – conceitos básicos

- Os sinais que compõem a informação também podem ser **armazenados** em alguma forma de **memória**: em livros, em fitas cassete, discos, em gravuras e fotografias, na mente humana, etc.

## Introdução – conceitos básicos

Distinção entre “informação” e “dados”, na Informática:

- **Informação** compõe-se de fatos apresentados de uma forma que possam ser compreendidos, que tenham significado para alguém
  - portanto, “informação” está mais relacionada a “conhecimento”.
- **Dado** é uma representação (registro) de uma informação.
  - Esta representação pode ser registrada fisicamente (armazenada): em papel, num disco de computador, etc.

## Introdução – conceitos básicos

- As **informações** devem ser traduzidas (codificadas) na forma de **dados** para que possam ser legíveis pelos computadores. Em geral, os dados se constituem de números, símbolos, caracteres, sinais, etc.
- Os dados devem ser organizados pelos sistemas de **computador** para que possam ser compreendidos pelas **pessoas**, isto é, para que tenham significado e representem, realmente, alguma **informação**.
- Por definição, os computadores processam “**dados**”, não “**informação**”.

## Introdução: conceitos básicos

- No senso comum, os termos "**informação**" e "**dados**" geralmente são usados como sinônimos — mesmo por profissionais de informática — e não precisamos, na maioria das vezes, ser tão rigorosos com a diferença de significado exposta anteriormente.
- Mas é importante termos em mente que, tecnicamente, pode-se fazer essa distinção entre os dois vocábulos.



## Introdução – conceitos básicos

Conforme a abordagem tecnológica, principalmente no que se refere a organização dos registros, temos bancos de dados do tipo:

- **Hierárquicos** – os registros (um registro pode ser denominado como “nó”) são ligados um ao outro numa estrutura hierárquica: um registro “pai” pode ter vários “filhos”, um “filho” somente um “pai”. Há o conceito de “nó raiz”.
- **Rede** – semelhante ao modelo hierárquico, porém um “nó filho” pode ter vários “pais” e não há o conceito de “nó raiz”.
- **Relacional** – baseado em relações bidimensionais (tabelas).
- **Orientado a Objetos** – classes e objetos (um objeto encapsula dados e procedimentos).
- **Objeto-Relacional** – relacional com suporte a objetos (estruturas de dados) complexos.

## Formas de armazenamento (processamento de dados)

Unidades de registro de informação:

- **Bit** – dígito binário – é a menor unidade de dados que o computador pode tratar.
- **Byte** – conjunto de 8 bits – normalmente equivalente a um “caractere” (letra, algarismo, sinal ou símbolo).

OBS.: Na codificação ASCII (256 caracteres), um caractere equivale a um byte.

Já na tabela Unicode, a codificação pode ser feita com 16 bits (65.536 caracteres) — na realidade, podem ser utilizados 8 bits, 16 bits ou 32 bits (mais de 4 bilhões de códigos de caracteres possíveis).

## Formas de armazenamento (processamento de dados)

Unidades de registro de informação:

- **Campo** – conjunto de caracteres (bytes), formando um item de informação (**atributo**) referente a um ser, ou a uma coisa.
  - Geralmente um campo é identificado por um nome.
  - Exemplos:
    - Nome
    - Data de Nascimento
    - CPF
    - Peso Líquido

## Formas de armazenamento (processamento de dados)

- **Campos** fornecem características (**atributos**) sobre seres ou coisas. Podem ter diversas finalidades, dentre as quais podemos citar:
  - **Identificadores** - realizam a função de distinguir.
    - Código, Nº do RG, CPF, Nome
  - **Descritores** - têm a função de descrever.
    - Tamanho, Peso, Cor, Forma, Idade
  - **Endereçamento** (ou localização) - determinam a localização de algo em uma região.
    - Endereço, Caixa Postal, Departamento, Prateleira
  - **Temporais** - determinam quando ocorreu um evento.
    - Data de Nascimento, Hora de Fabricação
  - **Quantificadores** - indicam uma quantidade ou valor.
    - Preço de Venda, Quantidade Produzida, Número de Dependentes
  - **Classificadores** - indicam o tipo, categoria ou classe.
    - Cargo, Função, Tipo de Transação, Classe do Cliente
  - **Condicionais** - indicam o estado (condição atual) de alguma coisa.
    - Estado Civil, Ativo, Pendente, Realizado, Com Defeito

## Formas de armazenamento (processamento de dados)

Unidades de registro de informação:

- **Registro** – é o conjunto de campos que se referem ao mesmo ser ou coisa.
  - Normalmente um registro possui um campo (ou conjunto de campos) identificador, cujo valor permite localizar (distinguir) o registro entre os demais.
  - Exemplos:
    - Registro de Cliente
    - Ficha de Paciente
    - Registro de Venda

## Formas de armazenamento (processamento de dados)

Unidades de registro de informação:

- **Arquivo** – (arquivo lógico) um conjunto de registros referentes a seres ou coisas semelhantes (do mesmo tipo).
  - Normalmente, os registros de um arquivo possuem uma mesma estrutura, isto é, os mesmos campos aparecem em todos os registros (nas mesmas posições e tamanhos).
- Exemplos:
  - Cadastro de Clientes
  - Registros de Venda
  - Arquivo de Produtos
  - Tabela de Veículos

## Formas de armazenamento (processamento de dados)

Armazenamento de informação:

- **Banco de Dados** (BD) – conjunto de arquivos relacionados entre si, organizados de forma útil (acesso aos dados).
  - Em geral, um BD forma a base de um sistema de informação a respeito de uma atividade ou assunto.
  - Em inglês: database (DB).
- Exemplo:
  - Um banco de dados de Vendas, formado pelos arquivos de Clientes, Produtos, Registros de Vendas, Vendedores.

## Introdução – conceitos básicos

**SGBD** – sistema gerenciador de banco de dados: software que permite a criação e manutenção de bases de dados (armazenamento, organização e gerenciamento de dados).

- Em inglês: DBMS - Database Management System.
- Um SGBD normalmente possui recursos que possibilitam a confecção de aplicações.



## Introdução – conceitos básicos

Operações fundamentais de um SGBD:

- **Inclusão** de registros.
- **Recuperação** (acesso e leitura) dos dados.
- **Atualização** (alteração) de registros e campos.
- **Exclusão** de registros.

Notas:

- No desenvolvimento de aplicações, é comum a utilização do termo “**CRUD**” (*create, read, update, delete*), para programas de manutenção cadastral básica.
- Na linguagem SQL, as respectivas instruções fundamentais (para implementar um CRUD) são:  
**INSERT, SELECT, UPDATE e DELETE.**

# Múltiplos de bytes (medidas de memória)

Tradicionalmente, para unidades de medida de dados binários, temos:

1 Byte (B) = 8 bits (geralmente equivalendo a 1 caractere)

1 Kilobyte (KB) = 1024 B =  $2^{10}$  bytes

1 Megabyte (MB) = 1024 KB =  $2^{20}$  = 1.048.576 bytes

1 Gigabyte (GB) = 1024 MB =  $2^{30}$  = 1.073.741.824 bytes

1 Terabyte (TB) = 1024 GB =  $2^{40}$  = 1.099.511.627.776 bytes

1 Petabyte (PB) = 1024 TB =  $2^{50}$  = 1.125.899.906.842.624 bytes

1 Exabyte (EB) = 1024 PB =  $2^{60}$  = 1.152.921.504.606.846.976 bytes

1 Zetabyte (ZB) = 1024 EB =  $2^{70}$  = 1.180.591.620.717.411.303.424 bytes

1 Yottabyte (YB) = 1024 ZB =  $2^{80}$  = 1.208.925.819.614.629.174.706.176 bytes

K [ou “k”, minúsculo] (kilo) → mil; M (mega) → milhão; G (giga) → bilhão; T (tera) → trilhão; P (peta) → quatrilhão; E (exa) → quintilhão; Z (zeta) → sextilhão; Y (yotta) → septilhão.

Nota: Não confundir múltiplos de “byte” (“B”, maiúsculo) com os de “bit” (“b”, minúsculo).

Exemplo: Mbyte (MB) ≠ Mbit (Mb) → 1 Mbit = 1.000.000 bits →  $\div 8$  = 125.000 bytes.

# Introdução – conceitos básicos

Comparação de terminologias aplicáveis:

| <b>Processamento de Dados</b>  | <b>SGBD Relacionais</b>  | <b>Teoria do Modelo Relacional</b> | <b>Modelo de Dados (E-R)</b>          | <b>Orientação a Objetos</b>     |
|--------------------------------|--------------------------|------------------------------------|---------------------------------------|---------------------------------|
| Campo                          | Coluna                   | Atributo                           | Atributo                              | Propriedade, Atributo           |
| Registro                       | Linha                    | Tupla                              | Ocorrência ou instância (“Entidade”?) | Objeto (instância)              |
| Arquivo (arquivo lógico)       | Tabela                   | Relação                            | Entidade (ou Tipo Entidade)           | Classe                          |
| Banco de Dados                 | Banco de Dados (esquema) | Base de Dados (esquema)            | Modelo de Dados (esquema)             | Repositório / Modelo de Classes |
| Campo Chave                    | Chave Primária           | Chave Primária (Chave Candidata)   | Atributo(s) Identificador(es)         | Identidade do Objeto            |
| (chave externa, ponteiro ?)    | Chave Estrangeira        | Chave Estrangeira                  | Relacionamento                        | Relacionamento                  |
| Programa, Rotina, Procedimento | Restrição de Integridade | Restrição de Integridade, Domínio  | Regra de Integridade                  | Método, Operação                |

# Introdução – conceitos básicos

Comparação de terminologias aplicáveis (inglês):

| Data Processing             | Relational DBMS      | Relational Model Theory      | Data Model (E-R)                 | Object-Oriented          |
|-----------------------------|----------------------|------------------------------|----------------------------------|--------------------------|
| Field                       | Column               | Attribute                    | Attribute                        | Property, Attribute      |
| Record                      | Row                  | Tuple                        | Occurrence, instance ("Entity"?) | Object (instance)        |
| File (logical file)         | Table                | Relation                     | Entity (Entity Type)             | Class                    |
| Database (Data Bank)        | Database, Schema     | Database, Schema             | Data Model (schema)              | Repository / Class Model |
| Key Field                   | Primary Key (PK)     | Primary Key (Candidate Key)  | Identifier Attribute(s)          | Objeto Identifier (OID)  |
| (external key, pointer ?)   | Foreign Key (FK)     | Foreign Key                  | Relationship                     | Relationship             |
| Program, Routine, Procedure | Integrity Constraint | Integrity Constraint, Domain | Integrity Rule                   | Method, Operation        |

# Introdução – conceitos básicos

Linguagens computacionais utilizadas em cada paradigma:

- **Processamento de dados** (tradicional)
  - Linguagens de programação de 3ª e 4ª geração (geralmente “procedurais”)
- **SGBD Relacionais**
  - SQL e linguagens “procedurais” embutidas
  - Interfaces gráficas baseadas em QBE (Query by Example)
- **Teoria do Modelo Relacional**
  - Álgebra Relacional e Cálculo Relacional
- **Modelagem de Dados** (Entidade-Relacionamento)
  - Notações gráficas (conforme padrões) e descrição semântica dos relacionamentos
- **Orientação a Objetos** (OO)
  - Linguagens de programação orientadas a objeto

## Introdução – conceitos básicos

Formas mais comuns de **acesso** em arquivos:

- **Sequencial** – os registros são processados (pesquisa, leitura, gravação) sempre na seqüência em que foram armazenados, partindo do início do arquivo.
- **Indexado** – o acesso aos registros é feito através de estruturas auxiliares (os índices), normalmente através algoritmo de busca em “árvore B” (B-Tree). Nos índices basicamente constam valores de campos específicos do arquivo de dados (chave de acesso), ordenados, e os respectivos endereços físicos dos registros.
- **Hash** – utiliza algoritmos para cálculo das posições físicas dos registros, a partir dos respectivos campos-chave.

## Introdução – conceitos básicos

Em contraposição ao acesso sequencial, os tipo de acesso “indexado” e “hash” são classificados como acesso “direto” (também são utilizados os termos “randômico” ou “aleatório”).

- A adequação (vantagens e desvantagens) de cada forma de acesso vai depender de fatores como:
  - Meios físicos de armazenamento: fitas magnéticas, discos, etc.
  - Tipo de processamento: em “batch” (lote) ou em tempo real.
  - Tamanho da base de dados versus capacidade disponível para armazenamento (memória de massa).
  - Requisitos de desempenho (tempo de resposta) do sistema.

## Introdução – conceitos básicos

- “**Arquivos texto**” são recursos comuns em vários sistemas de processamento de dados.
- Não há um padrão único. Via de regra, a estrutura dos arquivos é determinada pelos programas aplicativos que tratam essas bases de dados.
- Há também padrões definidos. Exemplos:
  - **CNAB** (arquivos bancários)
  - **XML** (Extensible Markup Language)



## Introdução – conceitos básicos

- Em geral, para organização da estrutura desses arquivos texto, são estabelecidos “marcadores” (caracteres especiais) como:
  - **Separador de Campos** – por exemplo: vírgula, ponto-e-vírgula ou tabulação (em arquivos tipo **CSV**).
  - **Separador de Registros** – geralmente caracteres de quebra de linha:
    - **CR** – carriage return (caractere ASCII **13**), equivalente ao código da tecla “Enter”.
    - **LF** – line feed (caractere ASCII **10**).
    - Nota: padrões nos sistemas operacionais:
      - Microsoft (Windows / DOS) = CR + LF
      - Unix / Linux = LF
      - Apple (Mac OS) = CR
  - **Final de Arquivo** – conhecido como **EOF** (end of file).

# Introdução – conceitos básicos

- Com relação a tamanho de registros e campos:
  - **Tamanho Fixo** – não necessitam de separadores de campo (a posição determina o campo no registro).
  - **Tamanho Variável** – necessitam reconhecer separadores de campos (e exceções), mas oferecem melhor aproveitamento de espaço físico.

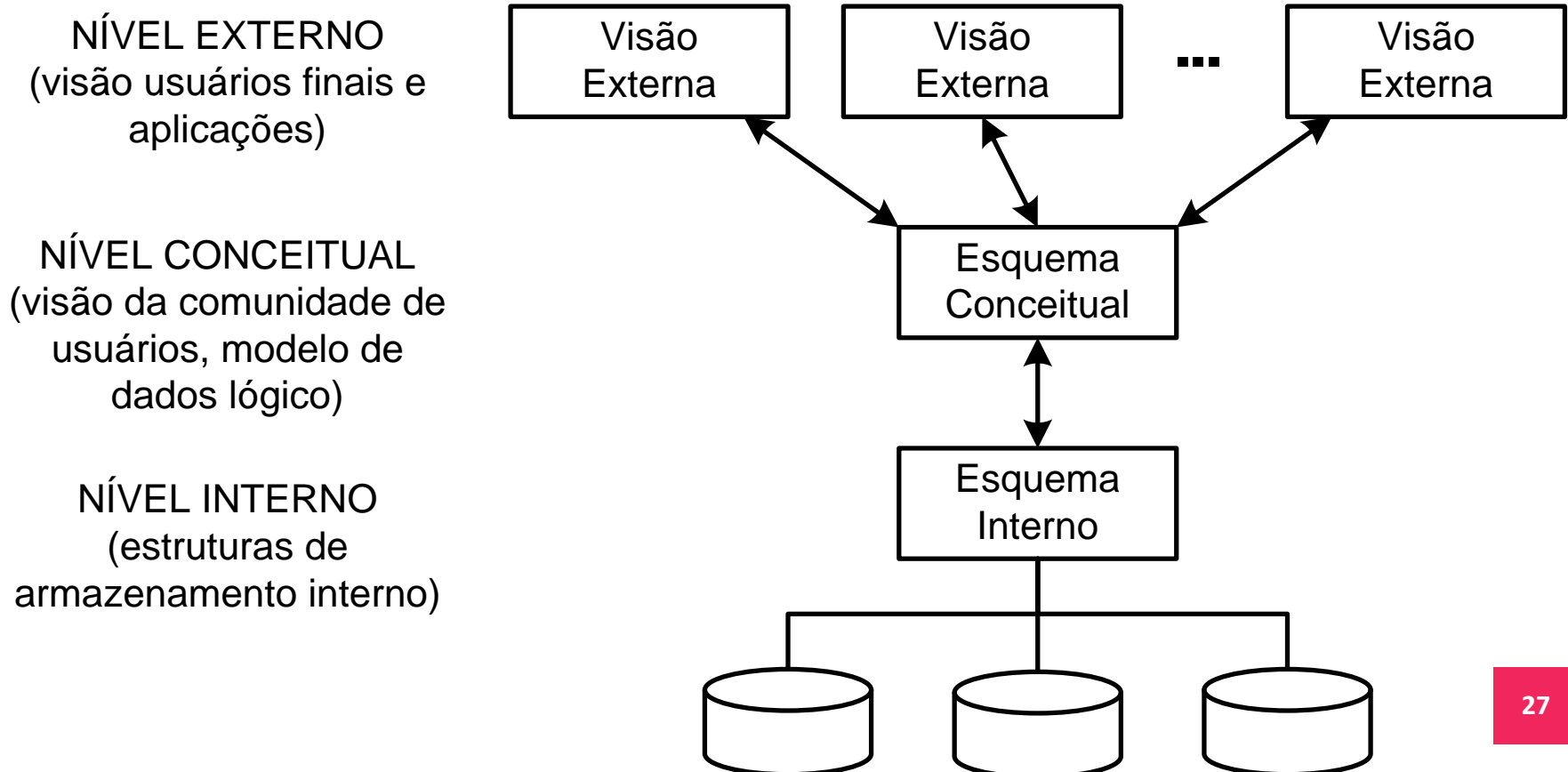
## Notas:

- Na codificação **ASCII** (256 caracteres), um caractere equivale a um byte.
- Já na tabela **Unicode** (65536 caracteres ou mais), a codificação muitas vezes é feita com 16 bits (2 bytes) — na realidade, pode-se utilizar:
  - 1 byte (8 bits = 256 combinações).
  - 2 bytes (16 bits = 65536 combinações), padrão “UTF-16”.
  - 4 bytes (32 bits = mais de 4 bilhões de combinações), padrão “UTF-32”.
  - Obs.: No padrão “**UTF-8**”, muito recomendado, a codificação é de tamanho variável, de 1 a 4 bytes.
- Para compatibilidade, a codificação Unicode engloba o padrão ASCII “original” (7 bits = 128 combinações, com códigos de 0 até 127).

# Arquitetura ANSI/SPARC em 3 níveis para BD

Em 1975, o grupo de estudo ANSI/SPARC (*American National Standards Institute / Standards Planning And Requirements Committee*), propôs um modelo de abstração para a arquitetura de banco de dados, baseado em 3 níveis.

A maioria dos sistemas de BD atuais atendem a esses conceitos.

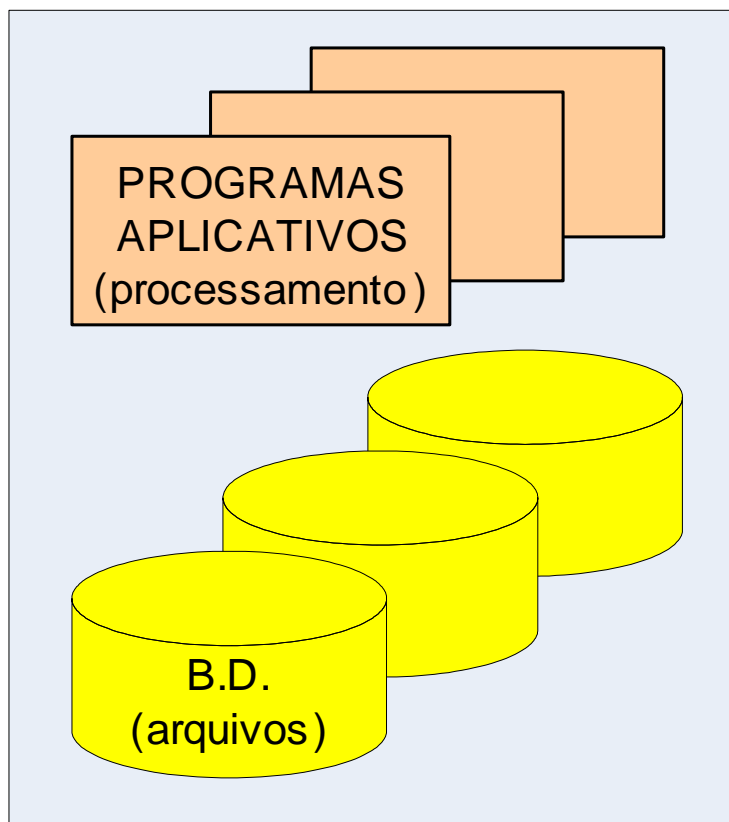


# Arquiteturas e ambientes

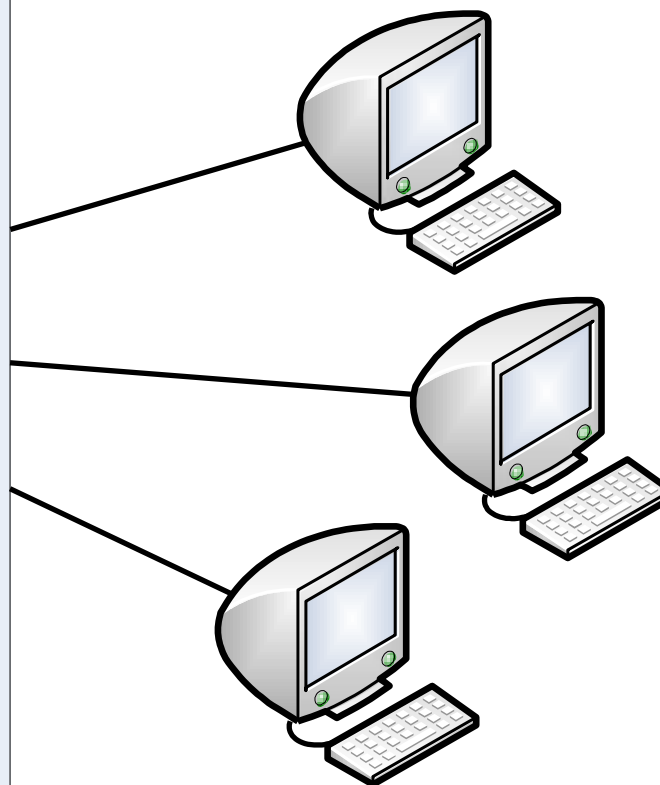
## Ambiente Mainframe

### MAINFRAME

(computador de grande porte)



TERMINAIS “BURROS”  
(monitor de vídeo, teclado e interface de comunicação)



## Arquiteturas e ambientes

# Ambiente Mainframe

- Vantagens:
  - Altíssimo poder de processamento, confiabilidade e segurança.
- Desvantagens:
  - **Alto custo** (inclusive de infra-estrutura).
  - Pouca flexibilidade.

## Arquiteturas e ambientes

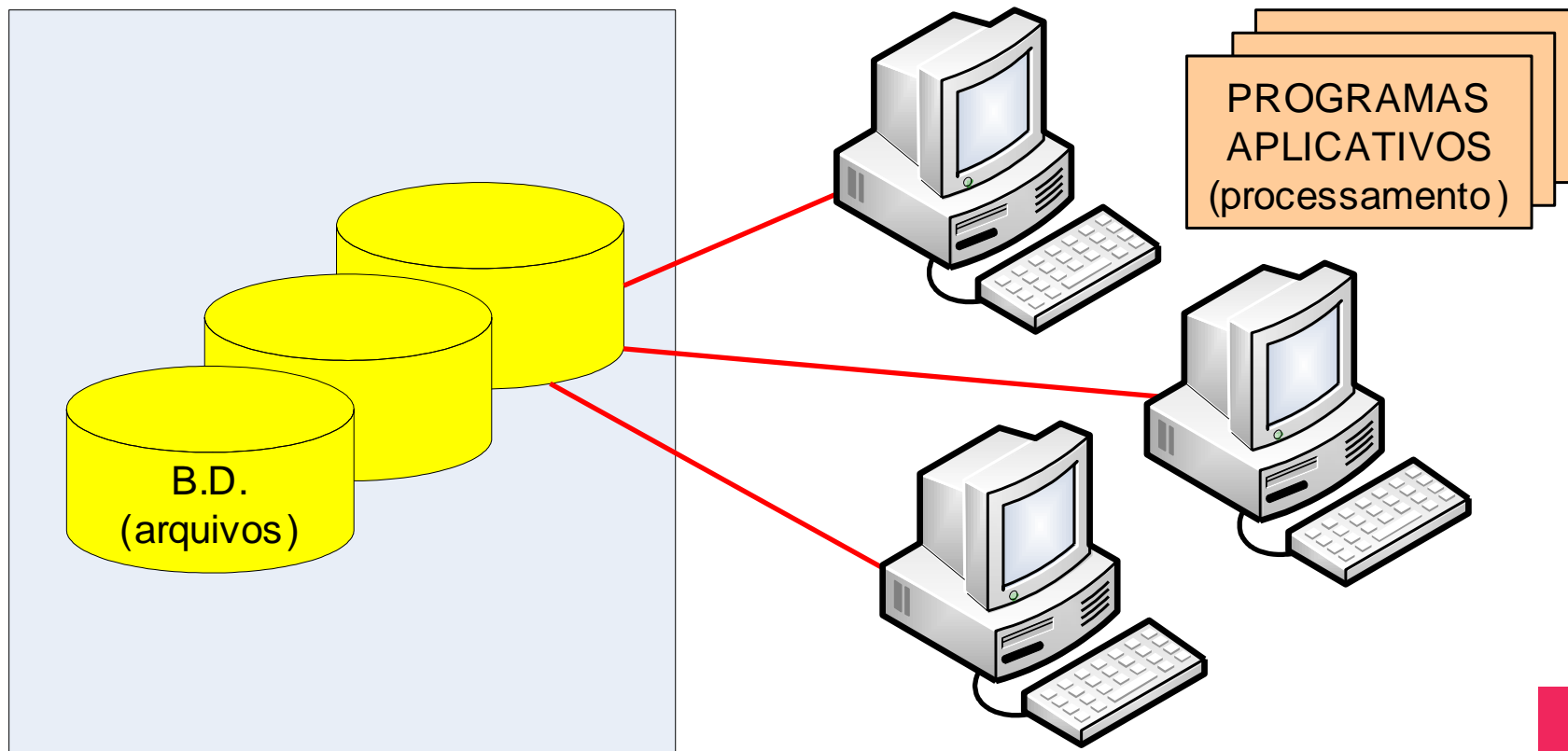
# Servidor de Arquivos em Rede Local

SERVIDOR DE ARQUIVOS

na Rede Local

(basicamente armazenamento)

MICROCOMPUTADORES

(estações de trabalho com  
processamento dos aplicativos)

## Arquiteturas e ambientes

# Servidor de Arquivos em Rede Local

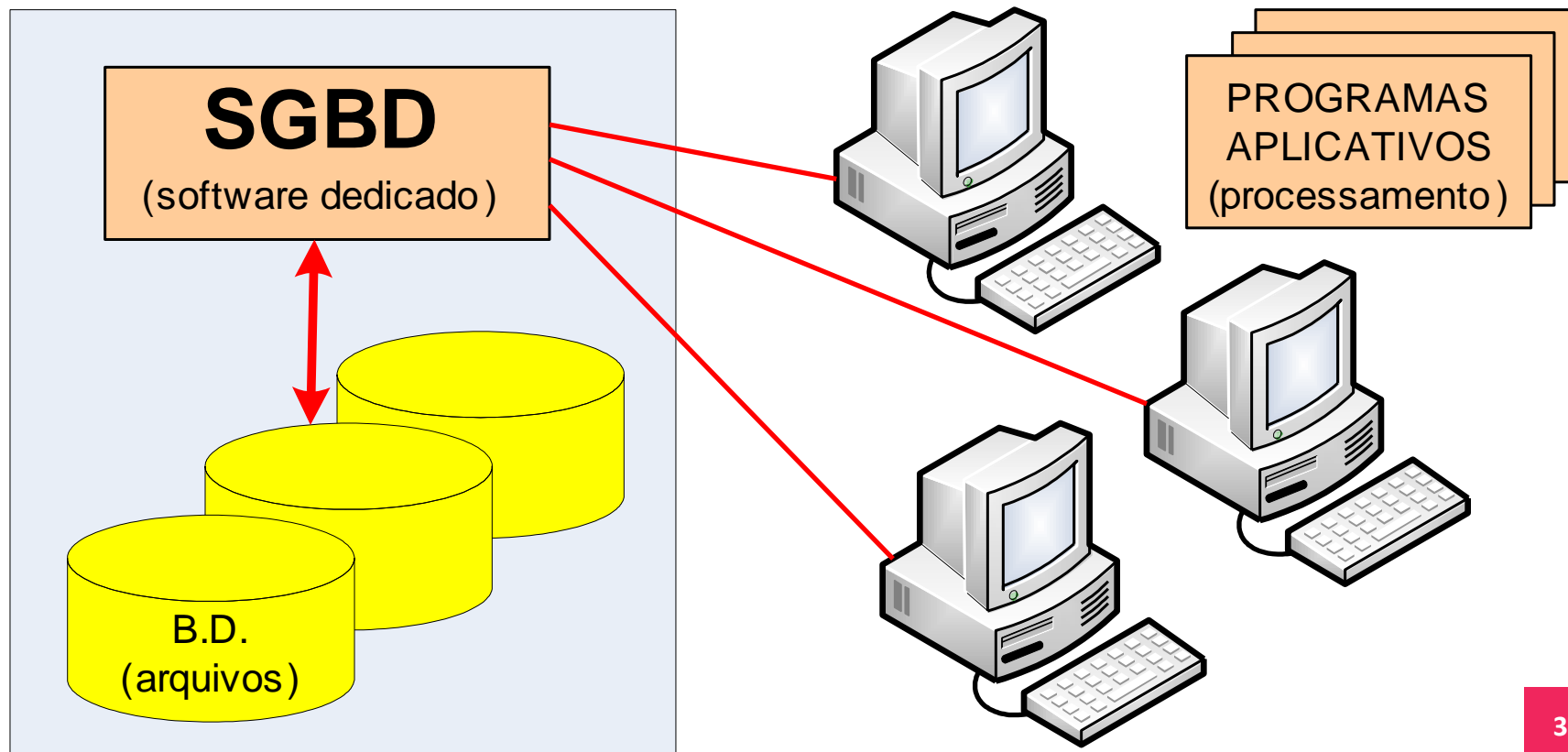
- Vantagens:
  - Baixo custo.
  - Não necessita de infra-estrutura tão cara.
  - Flexibilidade.
  - Facilidade no desenvolvimento de aplicações.
- Desvantagens:
  - Alto tráfego de rede (problemas de desempenho).
  - Baixa confiabilidade e pouca tolerância a falhas.
  - Integridade de dados não garantida pelo servidor: dependente das aplicações do “lado cliente”.

## Arquiteturas e ambientes

**Arquitetura Cliente/Servidor (2 camadas)**

SERVIDOR DE  
BANCO DE DADOS  
na Rede Local

MICROCOMPUTADORES  
(estações de trabalho com  
processamento dos aplicativos)





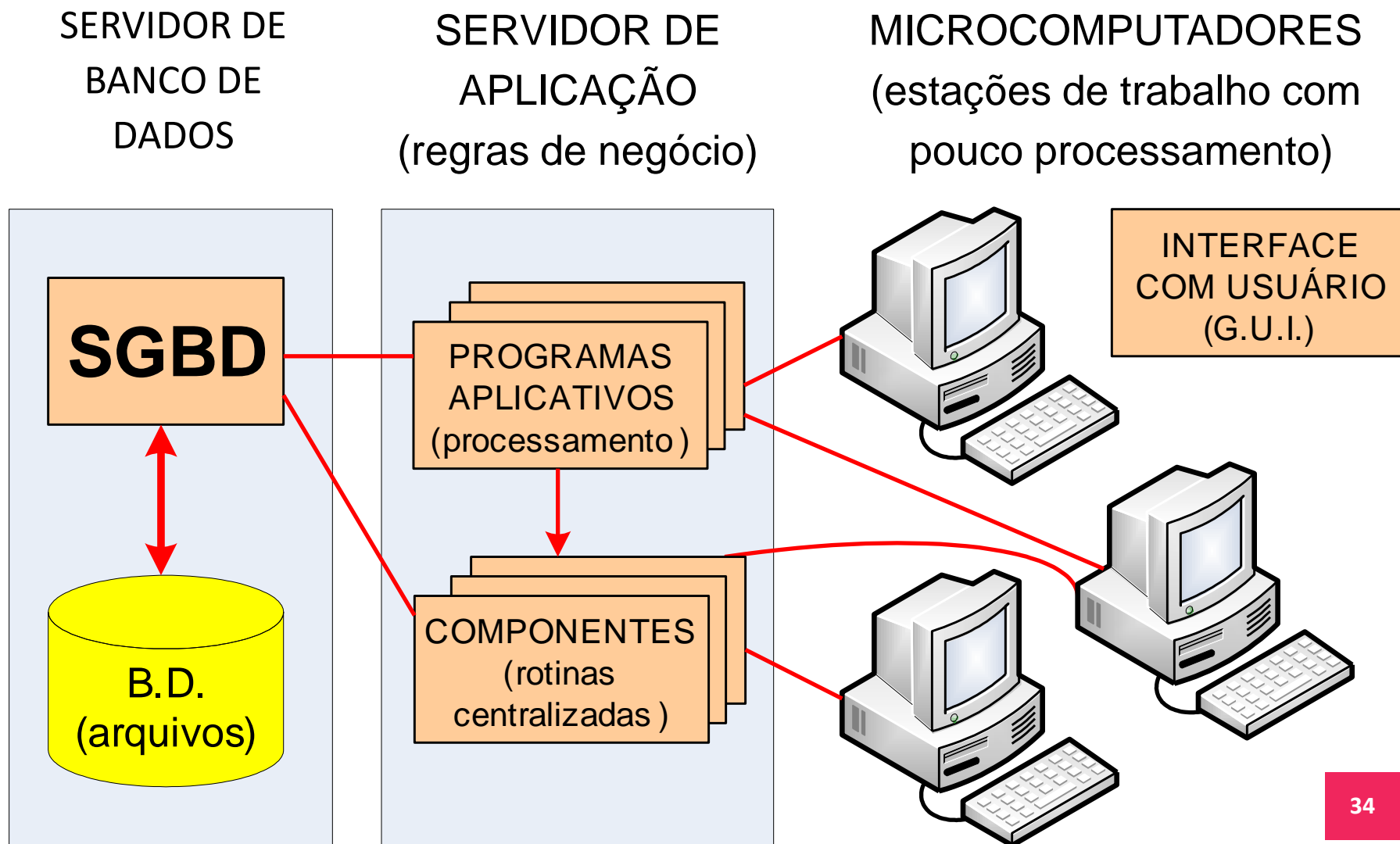
## Arquiteturas e ambientes

# Arquitetura Cliente/Servidor (2 camadas)

- 2 camadas (*two tier*).
- Vantagens:
  - Redução do tráfego de rede (otimização dos recursos e melhor desempenho).
  - Grande confiabilidade e tolerância a falhas.
  - Integridade de dados garantida pelo servidor.
- Desvantagens:
  - Custo um pouco mais alto, se comparado com o ambiente de aplicações com servidor de arquivos:
    - Investimento na máquina servidora
    - Licenças de software (SGBD)
  - Desenvolvimento de sistemas aplicativos mais sofisticados: necessita de mão-de-obra mais especializada.

## Arquiteturas e ambientes

## Arquitetura Cliente/Servidor (3 camadas)



## Arquiteturas e ambientes

# Arquitetura Cliente/Servidor (3 camadas)

- Características – 3 camadas (*three tier*):
  - Camada de interface com o usuário (**apresentação**)
  - Camada de aplicação (**regras de negócio**)
  - Camada de dados (**persistência**) – SGBD.
- Vantagens:
  - Em comparação com a arquitetura em 2 camadas, oferece uma melhor organização (e melhor gerenciamento) através da centralização das regras de negócio.
- Desvantagens:
  - Custo mais alto:
    - Investimento em mais máquinas servidoras
    - Licenças de software (SGBD) e do(s) softwares que dão suporte ao servidor de aplicação.
  - OBS.:** Porém, teoricamente, o **custo pode ser mais baixo**, caso o parque de estações de trabalho seja constituído por microcomputadores de menor valor (*thin clients*) — já que essa arquitetura não exige grande capacidade de processamento nas máquinas-cliente dos usuários.
  - Desenvolvimento de sistemas aplicativos mais sofisticados: necessita de mão-de-obra bem mais especializada.

## Arquiteturas e ambientes

# Arquitetura em “n” camadas (*multi-tier*)

- Variações, normalmente com divisão (especialização) da camada de aplicação:
  - Camada de interface com o usuário (**apresentação**).
  - **Servidor de apresentação** – exemplo: servidor *Web*.
  - Camada de aplicação (**regras de negócio**).
  - Camada de dados (**persistência**) – SGBD.
- Exemplo: Arquitetura “Java Enterprise” (J2EE), com um servidor de aplicação Java (“EJB container”).
- Obs.: Não confundir diferentes conceitos para o termo “camada”:
  - **Tier** (camada “física”), mais relacionada à infraestrutura, separação dos componentes servidores (de *hardware* e de *software*) do sistema como um todo, com processos distintos.
  - **Layer** (camada “lógica”), corresponde à organização da aplicação (dentro do *software*, ou seja, do código-fonte desenvolvido), como seus componentes são agrupados (conforme suas finalidades básicas) e como devem interagir (por exemplo, seguindo o padrão “MVC” – *Model-View-Controller*).

## Principais características de um SGBD Relacional

- Segurança de acesso
- Integridade de dados
- Controle de transações
- Acesso multi-usuário
- Tolerância a falhas
- Suporte a uma linguagem de consulta e manipulação de dados
- Independência física e lógica de dados
- Dicionário de dados (“metadados”)

## Perfis profissionais

Profissionais de T.I. que trabalham com bancos de dados:

- **DBA** (Database Administrator) – administrador de banco de dados.
- **AD** (ou DA – Data Administrator) – administrador de dados.
- **Desenvolvedores em geral:**
  - Programadores
  - Analista de sistemas
  - Arquitetos
- **Analista de BI** (Business Intelligence).

## Perfis profissionais: DBA

**DBA** (Database Administrator) – administrador de banco de dados

- É o profissional responsável por manter e gerenciar servidores de banco de dados.
- É especializado em um ou mais SGBD's — além de conhecimentos em sistemas operacionais, equipamentos servidores (hardware), redes e protocolos, desenvolvimento de aplicações, arquitetura de sistemas, entre outros correlatos.
- Trabalha em conjunto com outros profissionais especializados. Por exemplo: administradores de rede, administradores de sistemas, desenvolvedores e analistas de sistemas, AD's, etc.
- Em geral, ter um ambiente com SGBD exige que se tenha um DBA (ao menos em tempo parcial).

## Perfis profissionais: DBA

Principais funções de um **DBA**:

- Instalação e configuração do SGBD.
- Definição das estruturas físicas de armazenamento (com as estratégias de acesso).
- Definição (e testes) das estratégias de backup e recuperação.
- Realização de backup's.
- Implementação do controle de acesso ao SGBD.
- Suporte à equipe de desenvolvedores (em questões técnicas que envolvam o SGBD) .
- Monitoração do desempenho e disponibilidade dos servidores de BD.
- Monitoração do crescimento das bases de dados.
- Projeto e gerenciamento de bases de dados distribuídas (sincronização, replicação, integração).
- Em algumas empresas, também assume funções de AD (DA).



## Perfis profissionais: AD

**AD** – administrador de dados (DA – data administrator)

- DBA  $\neq$  DA
- AD é o profissional que trabalha com atividades relacionadas a projeto de bases de dados de sistemas aplicativos:
  - Modelos de dados (normalmente apoiando o trabalho dos analistas e/ou validando e homologando o trabalho destes);
  - Padrões de nomenclatura para objetos do BD (tabelas, colunas, constraints, etc.);
  - Integração de sistemas a bases de dados corporativas;
  - Manutenção de modelos e dicionário de dados corporativos.
- Poucas empresas possuem “administradores de dados” (como função dedicada e formalizada na equipe).

## Perfis profissionais: Desenvolvedores


Muitas vezes, o profissional de desenvolvimento de sistemas é “multifuncional”, ou seja, desempenha mais de uma função.

- Porém, dependendo da empresa, os papéis podem ser separados dentro da equipe.
- No que diz respeito a “banco de dados”, sempre é desejável que os desenvolvedores tenham uma visão abrangente (metodologia, técnicas e de produtos/plataformas). Porém os conhecimentos mais pertinentes aos perfis de desenvolvimento são:
  - **Analistas de Sistemas**: modelagem de dados e SQL — em geral, é um perfil mais próximo ao AD.
  - **Arquitetos**: estrutura e recursos do SGBD, frameworks de persistência — em geral, é um perfil mais próximo ao DBA.
  - **Programadores**: SQL, linguagem procedural do SGBD, API's, recursos e técnicas de desenvolvimento do SGBD.

## Perfis profissionais: Analista de BI

**Analista de BI** – Semelhante à função de “analista de sistemas”, porém é um profissional especializado na área de B.I. (*Business Intelligence* — inteligência de negócios).

- Conhecimentos específicos:
- Indicadores de gestão de negócios.
- Análise de dados para informações gerenciais (estratégicas).
- Ambiente de DW (data warehouse) e data marts.
- Processos de ETL (extração, transformação e carga) de dados.
- Modelagem dimensional (de dados).
- Estatística (e softwares relacionados).
- Data Mining (“mineração de dados”).
- OLAP (Online Analytical Processing)
- SQL, ferramentas de consulta e geradores de relatórios.



Copyright © 2025 Prof. André Luís Pereira dos Santos

Todos direitos reservados. Reprodução ou divulgação total ou parcial deste documento é expressamente proibido sem o consentimento formal, por escrito, do Professor (autor).