# Fundamentos de Banco de Dados

## Introdução à Banco de Dados

### Cenário de dados

Dados são diferente de informação.

Dados devem ser persistidos e confiáveis.

Os dados eram mais numéricos e textuais.

### O que são Banco de dados?

Banco de dados não são SGBDs

Formalmente, quando **Dados relacionados** consistem em uma **base de dados.**

**Definição geral:** Podemos considerar uma coleção de palavras, que dentre elas há relacionamentos entre dados, constituindo então um baco de dados.

### SGBDs

Foram criados no contexto de reduzir os custos de operação para gerenciamento de dados.

* Modelo baseado em sistemas de arquivos
* Modelo em rede
* Modelo relacional: construindo com base na teoria dos conjuntos.
* Modelo Hierárquico

### Definição

**Construção**

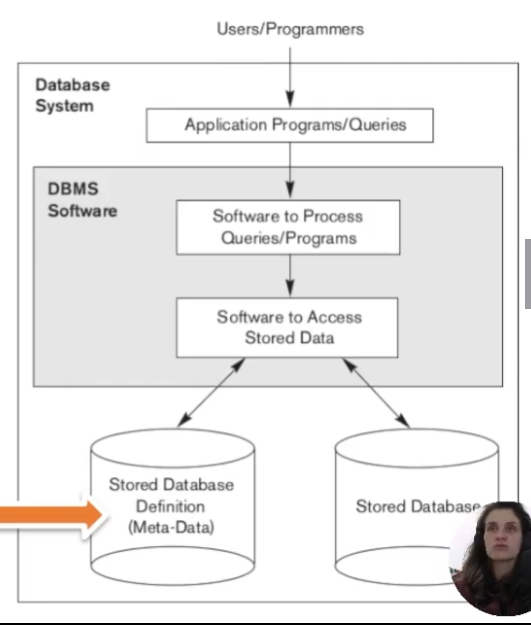
* Inserção de dados

**Manipulação**

* Recuperação

**Compartilhamento**

* Simultaneidade
* Acessos



**Metadados:** informações que fornecem uma descrição concisa dos dados contidos no BD

Por padrão o BD realiza o bloqueio e a liberação das tabelas.

Primeiros SGBDs do mercado

Evolução do modelo relacional - 80´s

* Feedback positivo
* Sistemas Distribuídos
* Desenvolvimento dos sistemas

### Modelo Hierárquico

IMS – Information Managements

Registros: links e dados

Organizado em formato de árvores

### Modelo em Rede

Links – ponteiros entre nós

Criado em 1964

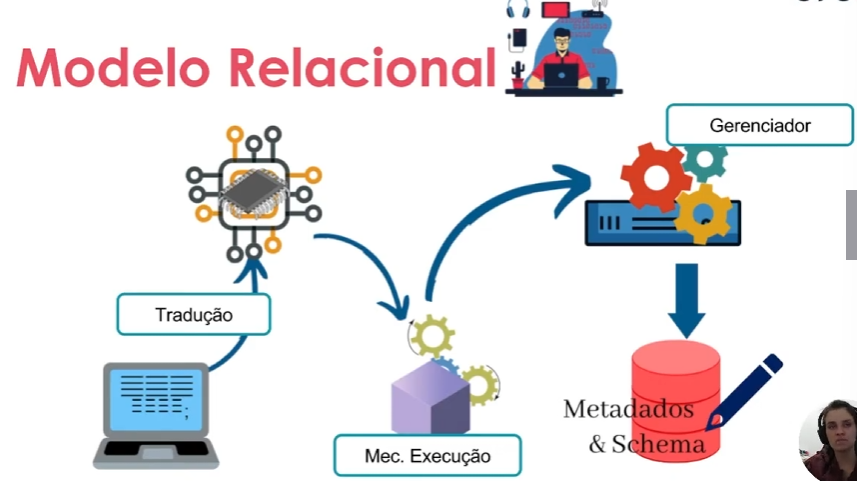
É necessário ter conhecimento de estrutura física do BD

### Modelo Relacional

* Álgebra relacional
* Relações
* TAD para armazenamento
* Transparência

Definição das tabelas e constrains para dados

Comandos traduzidos pelo processador LDD



Características:

* Altera e extrai informações
* Duráveis
* Transações (agrupa para executar)

### Storage & Buffer

* Gerenciador de armazenamento (movimento)
* Gerenciador de buffer (troca)

É possível ter vários SGBDs integrados

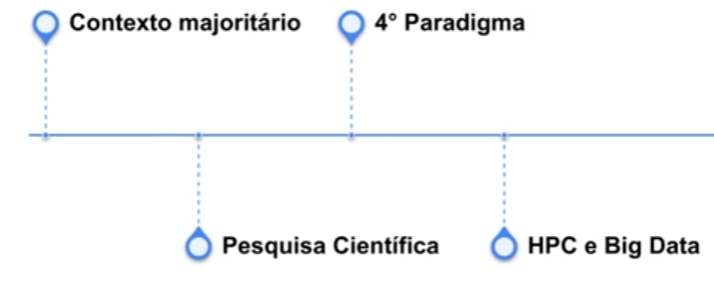
### SGBDs utilizados pelo mercado

1. Oracle (mais usados nas grandes corporações)
2. **MySQL** (solução de código aberto, usado em aplicações Web por ser leve)
3. Microsoft SQLServer
4. **PostgreSQL**
5. mongoDB (orientados a documentos, armazenando dados em blocos)
6. redis
7. ElasticSearch (relatórios kibana, fortado json)
8. Microsoft Acess (Não é um SGBD, mas acaba sendo utilizado pela simplicidade e custo baixo)
9. MariaDB: se originou do mySQL, tem consulta mais otimizado
10. Cassandra: noSQL, gratuito e openSource, usando para gerenciamento em larga escala

Características que popularizam um banco de dados:

* Popularidade
* Tempo de mercado
* Documentação
* Robustez
* Confiabilidade
* Segurança
* Multiplataforma

### A era dos dados e o futuro da modelagem



**Papel central – Sistemas Corporativos (será um modelo relacional)**

* Consultas simples

### Ambiente de pesquisa

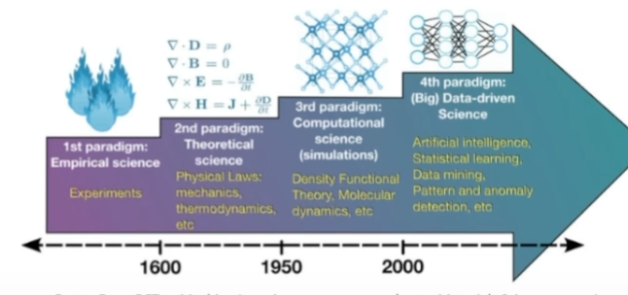
* N de tarefas computacionais
* Quantidade de dados
* Heterogeneidade
* Computação paralela e distribuída

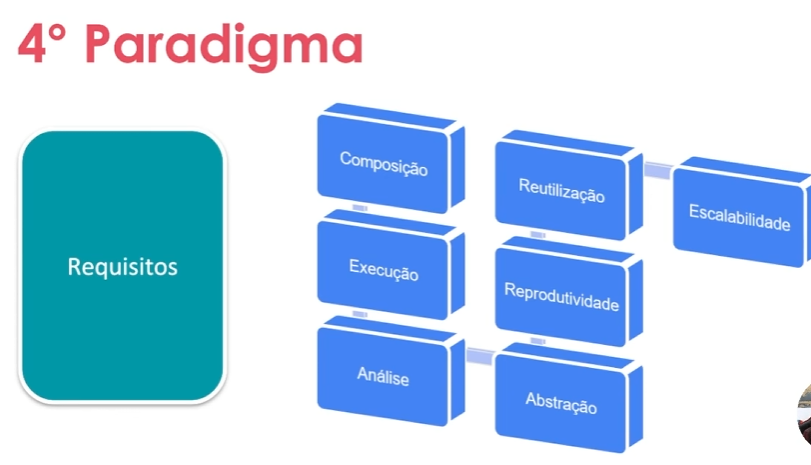
### 4 Paradigma

Instrumentos e simulações que geram grande volume de dados

**Novo modelo**: base na análise e exploração de dados (e-Ciência)

**Modelo anterior:** empírico, teórico e computacional.







### High Performance Computing

* Nós de processamento
* Sistema de arquivos paralelos – **sem persistência**
* Modelos: MPI, OpemMP, OpenCL
* Acesso: HFF5 e NetCDF

### Big Data

* Process e Storage: nós de processamento
* Sistema de arquivos paralelos – **persistente**
* Modelos: MapReduce, Spark, SGBDs paralelos

O Big Data os dados são persistentes, o HPC apenas processa.

### Novas tendências

* Carreira Tech em Data
* Data-driven
* Novos modelos de SGBDs

### Novos perfis:

* **Engenheiro de dados:** Desenho/construção; sustentação das soluções de dados. Extração de dados de fontes heterogêneas, disponibilizar os dados para serem consumidos pelos analistas e cientistas.
* **Analista de dados:** Criação de dashboards, apresentação visual dos dados. Busca entender o comportamento do negócio a partir dos dados. Realiza o diagnóstico, identifica possíveis motivos para comportamentos e verifica métricas.
* **Cientista de dados:** Modelagem, reconhecimento de padrões / predição. Busca responder perguntas atreladas ao contexto do negócio. Buscando insights através de técnicas de modelagem.

### Data-driven

* Análise de dados
* Entenda os dados
* Tome uma decisão

### Modelos NoSQL

* Documentos: mongoDB
  + Baseado em JSON
  + Baixa curva de aprendizado
  + Escalabilidade horizontal
  + Multi-plataforma
  + Transações ACID para multi-documento
  + Consultas: Suporta JS
* Wide-columns: Cassandra
  + Orientado: Facebook
  + Open-source: 2008
  + Performático
  + Descentralizado
  + Consultas: CQL
* Key-Value: redis
  + 2009, escrito em C
  + Compatível com outras linguagens
  + Performático
  + Support: strings, lists, maps, sets, JSON, graphs.
* Grafos: neo4j
  + Orientado à Grafos
  + Escrito em Java em 2007
  + TAD (Tipo abstrato de dado)
  + Cypher: query para grafos
  + Data science
  + Compatível: Python, NodeJS, GO, .NET e Java
* Orientado à Objetos: db4 objects
  + 2008 – open-source
  + Escrito em .NET e Java
  + Cross-plataform

## Sistemas de Gerenciamento de Banco de Dados

**Abordagem tradicional:** usando programação para controlar os dados. Maior complexidade de gerenciamento. Outras desvantagens são: **redundância de dados, esforço repetitivo, inconsistência de dados.**

**Abordagem BDs:** Estrutura centralizada de dados, os dados estão integrados. Possibilidade acesso a operações concorrentes. Sendo as principais características:

* **Abstração:** isolamento entre programa e dados. Determinar estrutura de dados;
* **Isolamento:**
* **Múltiplas visões:** visões distintas dos múltiplos dados
* **Autodescrição**: possui uma descrição das estruturas e suas regras, um esquema de banco de dados (DB schema) é a estrutura bem definida dos dados, isso **envolve os metadados e schemas**. **Os bancos NoSQL não possuem Schema.**
* **Compartilhamento**: compartilhamento facilitador, controle de concorrência;
* **Transação multiusuário:** levam os estados do banco de um estado para outro de maneira consistente.

O Administrador de banco de dados (DBA) é o responsável por gerenciar os Metadados e Schemas de um banco de dados.

### Natureza auto descritiva

Dentro dos metadados existem catálogos que descrevem a estrutura e regras dos dados, sempre representados por uma tabela de relações, tabela de dados, tabela de regras entre outras que sejam necessárias para descrever a estrutura do banco de dados.

### Isolamento

Nos SGBDs os dados são separados do programa da aplicação. Garantindo consistência nos dados além de não interferir na aplicação, essas características são chamadas de **Program-data Independence.**

Independência do programa e dados, garante transparência nas operações do SGBDs e na aplicação.

O Isolamento vem do uso dos catálogos.

### Autodescrição

Manter a estrutura do banco de dados que o define

### Compartilhamento de dados e processamento de transações multiusuários

Um banco de dados pode ser acesso por múltiplos acessos, ou seja, diferentes grupos de pessoas acessam o mesmo banco. Sendo necessário manter manutenção e integração do banco constantemente. Para isso é usando os **sistemas de Controle de concorrência.**

**Exemplo 1:** reserva de acentos para uma viagem.

Uma abordagem para garantir O Controle é usar o **online transação processing (OLTP).** Consiste em agrupar e registras todas as transações do banco e executá-las, sendo operações de **Insert, Update, Delete**.

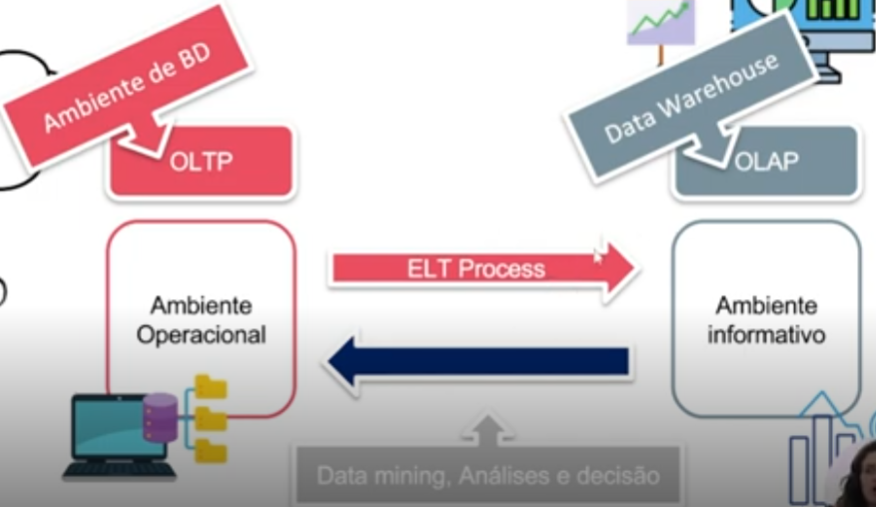
Promover Perfomance para executar uma operação ponto a ponto de maneira segura

OLTP:

* App multiusuário
* Gerenciador de transações concorrentes.
* Execução sem interferência (Atomicidade, executa tudo ou não executa nada)

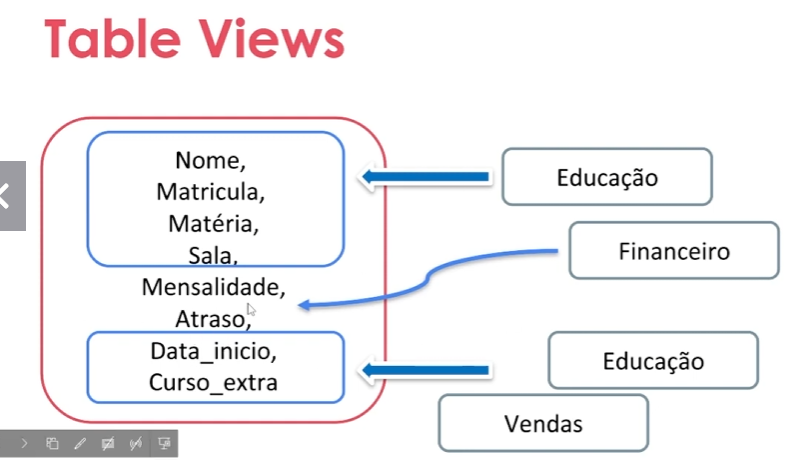
Contexto do OLTP:

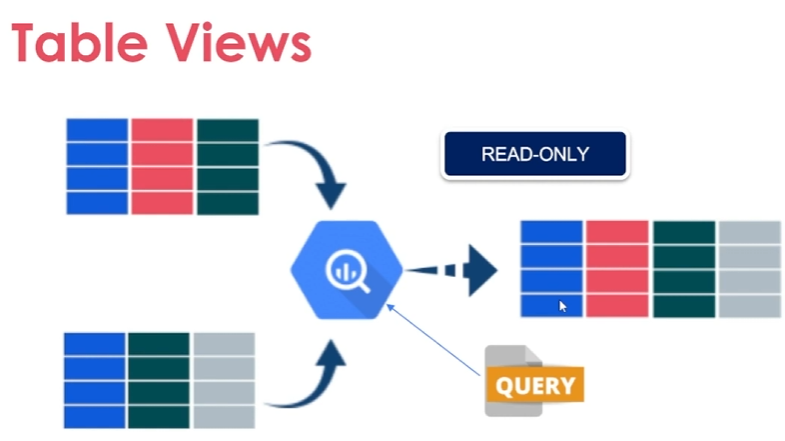
* Orientado a transação (Transaction-driven)
* Contexto operacional
* Processamento de dados



### Múltiplas Visões

Diversas pessoas de áreas distintas estão interessadas em ter uma visão sobre os dados. A visão do financeiro é diferente do setor educacional. Ou seja, existem perspectivas diferentes do mesmo contexto.





Uma view irá refletir em uma consulta que o usuário está esperando

### Atores

BDs simples: uma ou poucas pessoas acessam

Big Organizations: uma grande quantidade de pessoas acessa (ordem de 10k).

**Designer:** irá estrutura o DB, não precisa ser perfeito, mas performático.

* Identificar dados e requisito
* Representação e estrutura
* Fase preliminar

**Administrador (DBA)**

* Gerencia recursos: base de dados, SGBD, softwares adicionais.
* Orquestração
* Autorização de acesso

**Usuários Finais**

* Acessos via Querying
* Categorizados.
* Subcategorias:
  + Casuais
  + Ingênuos (consultas encapsuladas dentro de uma API): exemplo de uso de softwares, aplicações, ferramentas.
  + Sofisticados
  + Standalone: possui um BD pessoal.

Os SGBDs visam implementar diversas facilidades para os usuários do SGBD.

**Engenheiros de softwares:** responsável por desenvolver API que permite as conexões encapsuladas entre banco e aplicação;

### Workers em background

Fora do contexto de BD

* **Designer dos sistemas de SGBD:** Implementação dos módulos e interfaces do SGBD como um pacote de software
* **Implementação dos sistemas de SGBD:** Implementação dos módulos e interfaces do SGBD como um pacote de software
* **Pessoal de operação e manutenção:** Responsável pelo ambiente de hardware e software para SGBD
* **Desenvolvedores de ferramentas**: Ferramentas opcionais para diversos fins, como: performance, modelagem, análise, segurança.

### Vantagens de Utilizar a Abordagem de SGBDs

* **Controle de redundância:** evitar arquivos em duplicidade, evitando desperdício, redundância e inconsistência e updates desnecessários.
* **Restrição de acesso**: permissões especificas: read-only, update, restricted view.
* **Prover persistência:** evitar problema de Impedance Mismatch Problem
* **Prover estrutura:** Caching, Buffering, Indexação.
* **Backup e Recovery:**

Representação de Relações complexas.

Variedade de dados interconectados

Integridade de dados

* Regra de Domínio
* Integridade Referencial
* Asserções
* Gatilhos
* Dependência Funcionais

Inferência | Ações com regras | Triggers (Gatilhos que ações serão executadas)

### Ganhos em utilizar SGBDs

* **Padronização:** Tipos de dados, Display, Relatórios;
* **Redução de tempo no desenvolvimento da aplicação**
* **Flexibilidade**: As modificações no banco não afetam a aplicação.
* **Disponibilidade de informações atualizadas**: Update imediato
* **Economia com escalabilidade**: custo operacional e gerenciamento mais otimizado.

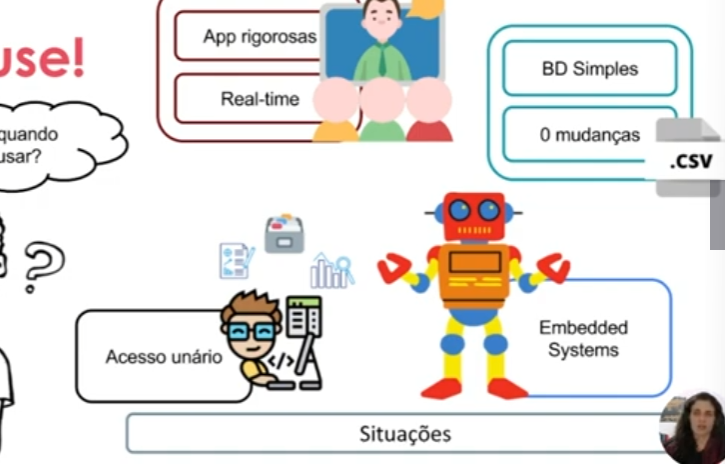
### Quando não usar SGBDs

**Relação Custo-benefício e Custo de Overhead.** Quando a aplicação não requisita o custo mínimo de um SGBD.

**Custos**

* Investimento Inicial
* Generalidade na definição e processamento
* Segurança, controle de concorrência, recovery, funções de integridade.

**Cenários que não precisam**



## Modelagem de Dados para Banco de Dados

Por que modelar? Compreensão do sistem

* Representação
* Modelo
* Referência

**Modelagem:** possui foco na descrição e relacionamento dos elementos que compõe a representação do contexto (mini-mundo).

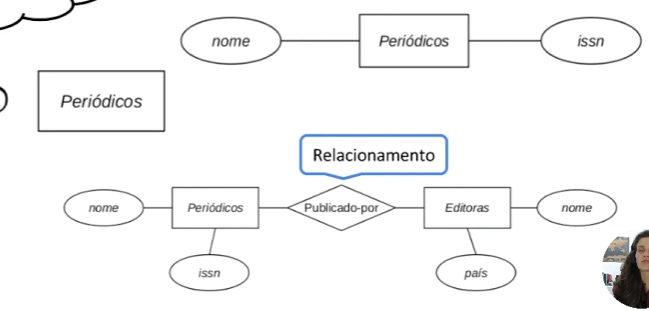
* Delimitando o contexto dos dados
* Requisitos para criação do modelo
* Definindo estrutura relacional
* Implementando – criando o DB.

**Esquema: facilita a compreensão do contexto dos dados**

Modelo de alto nível:

* Entidade-Relacionamento
* UML (Unified Modeling Language): utiliza o paradigma de OO.

Exemplo de modelo entidades-relacionamento:



Linguagem declarativa baseada na teoria dos conjuntos: CREATE, DELETE, FROM, WHERE.

Exemplo 1:

# Criando um banco

CREATE DATABASE firstexample;

CREATE TABLE periodicos(

Id integer,

Nome varchar(120),

Issn integer,

PRIMARY KEY (id)

);

# Criando a tabela de editora

CREATE TABLE editora(

Id integer,

Nome\_editora varchar(120),

Pais integer,

PRIMARY KEY(id)

);

# Criando a tabela periódicos

CREATE TABLE periódicos(

Id integer,

Nome varchar(120),

Issn integer,

PRIMARY KEY(id),

FOREIGN KEY (id) REFERENCES editora(id)

);

### Comandos para mySQL

**CREATE DATABASE** NomeBancoDeDados;

**USE** NomeBancoDeDados;

**SHOW** databases;

**CREATE TABLE** tabela(

nome tipo propriedades,

propriedades extras # (chave primaria e estrangeira)

)

**Exemplo:**

**create** **table** periodicos(

id **int** **auto\_increment** **primary** **key**, # o auto\_increment considera a **tentativa de incremento** e não os valores

nome\_periodico **VARCHAR**(20),

issn **INT** **unique**,

id\_editora **INT**

);

**Drop database** nomeBancoDeDados;

**Show tables**; # precisa estar no database

**Alter table** NomeTabela PropriedadeAlterada;

**Exemplo:**

**alter** **table** periodicos **add** **constraint** fk\_editora\_periodico **FOREIGN** **key** (id\_editora) **references** editora(id);

**Inser into** NomeTabela (ParChaves) **values** (ParValores);

**Exemplo:**

**insert** **into** editora (nome\_editora, pais) **values** ('IEEE', 'EUA'), ('DataScienceMagazine', 'EUA');

**select** ConjuntoPesquisa **from** NomeTabela;

**exemplo:**

**select** \* **from** periodicos;

## Arquitetura de Banco de Dados

### Modelos

Capacidade de abstração

Abstração = essencial

**Modelo de dados conceitual:** visão de alto nível, modelo Entidade-Relacionamento, generalização, especialização

* Entidade
* Atributos
* Relacionamento

**Modelo representacional:** meio caminho entre o conceitual e físico

* Constrains
* Linguagens
* Operacionais

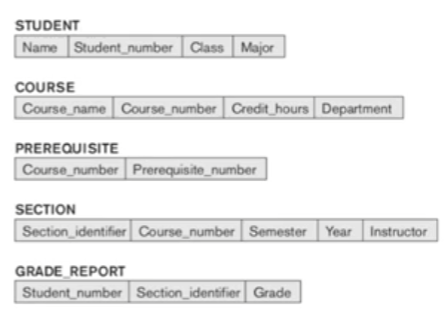
**Modelo de dados Físico:** especialista, visão de baixo nível, nível de hardware

* Índices, Hases
* Armazenamento

Todos são modelos de dados auto descritivos

### Esquemas

Descrição, Diagrama. Somente a estrutura do essencial



### Instâncias

Os dados mudam, por isso existem Snapshot (fotos de como os dados estavam).

O esquema é equivalente ao estado inicial de um banco de dados.

Quando o estado é modificado, surge um snapshot

**Metadados**: informações sobre a estrutura dos dados

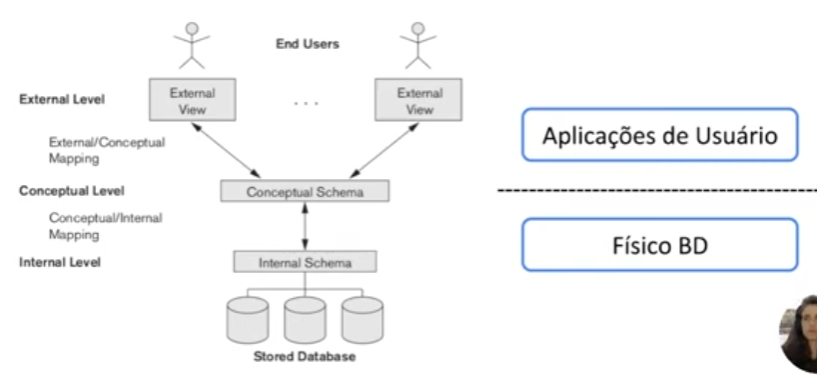
* Descrição esquema
* Construtores
* Constrains

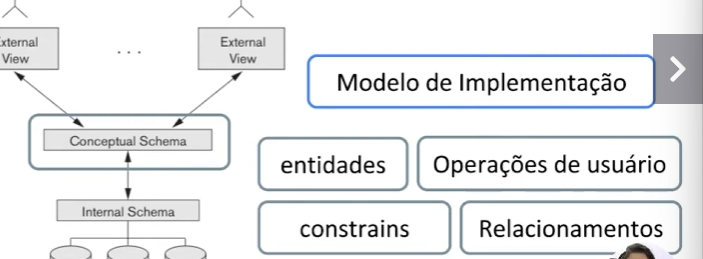
### Arquitetura de Three-Schema

Não é explicitamente suportado pelos SGBDs

* Catálogo
* Isolamento data/programa
* Views

Uma modificação no schema não influência nos demais schemas.





### Linguagens para SGBD

Lingaugens e interfaces – ligadas ao usuário – **Data Definition Language (DDL)**

Existem SGBDS com separação física entre as linguagens, mas são raros, a divisão ocorre em: Storage Definition Language (SDL) e View Definition Language (VDL)

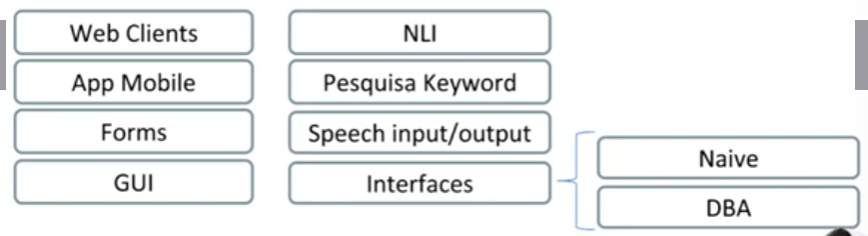
### DDL – Linguagem de definição

Baseada na Linguagem DML

* **DML** – separação em operação de baixo nível, procedural (se preocupa no como), e alto nível, não procedural (se preocupa em fazer).

### Interfaces

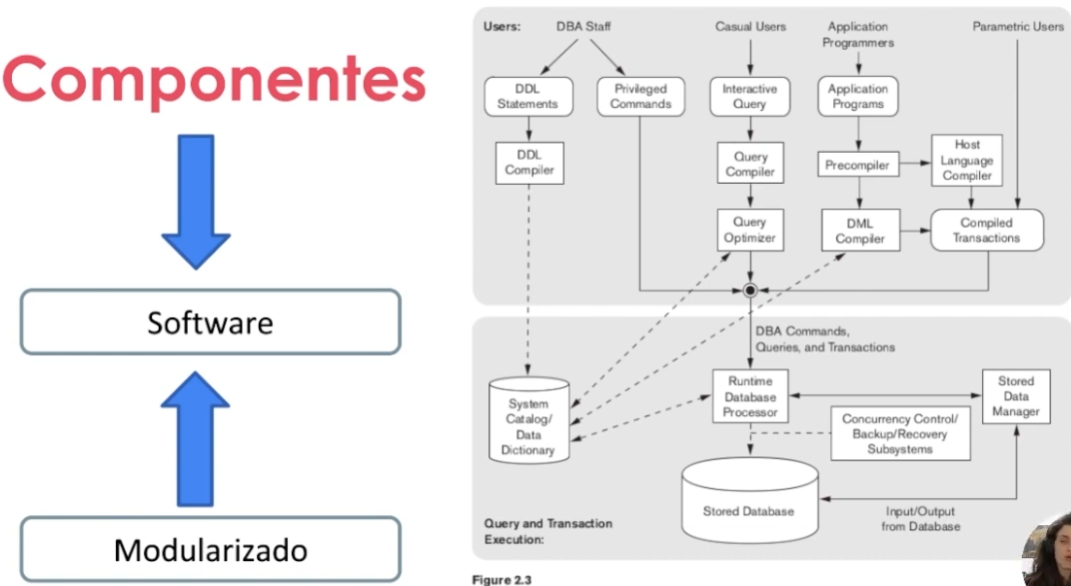
Como os usuários irão acessar os dados



Naive – Operações repetitivas

DBA – comandos de alto nível de privilégio

### Ambientes e utilitários de SGBD



### Utilitários

* Loading
* Backup
* Reorganização do storage
* Monitoramento

Ferramentas e aplicações

* **Data dictionary system:** armazena informações de decisão de design, padrões de utilização, descrição de aplicações
* **Softwares de comunicação:** terminal, workstations, pcs
* **DB designer**

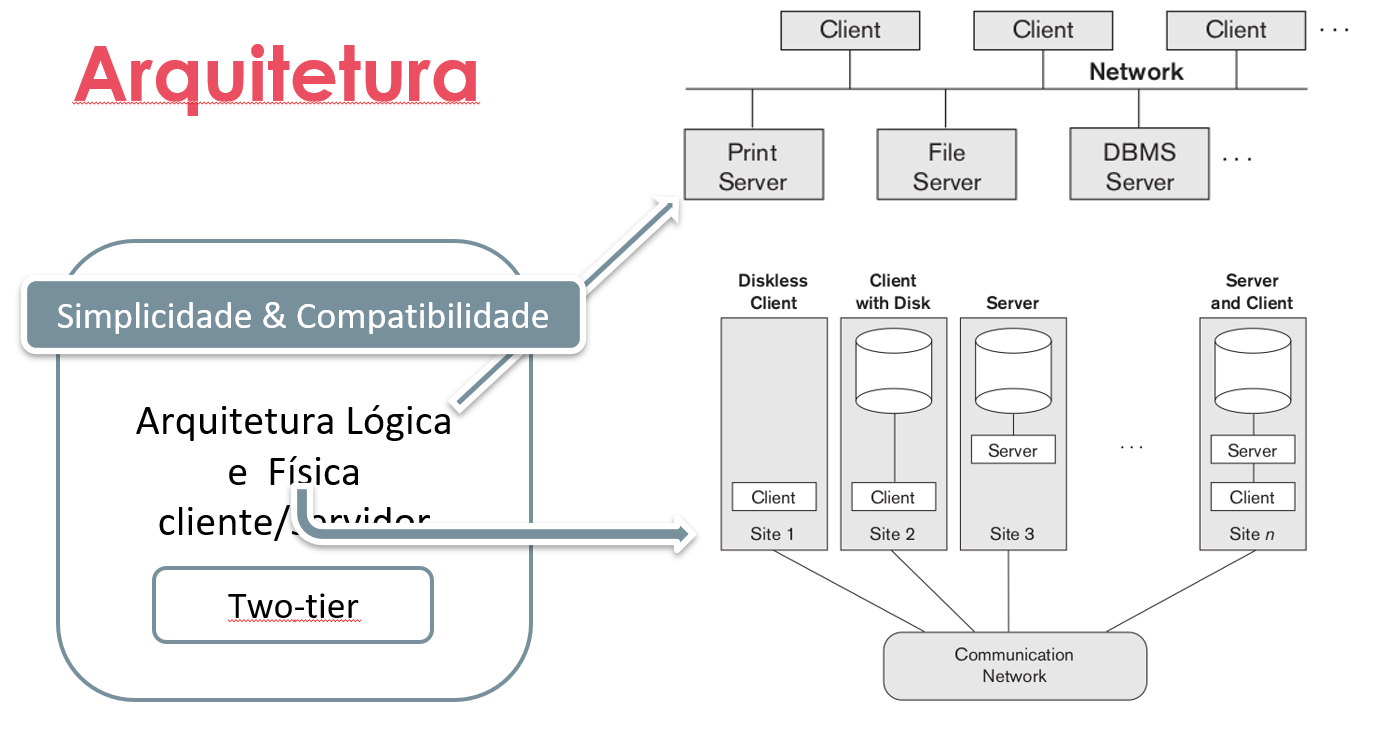
**Arquitetura Cliente Servidor**

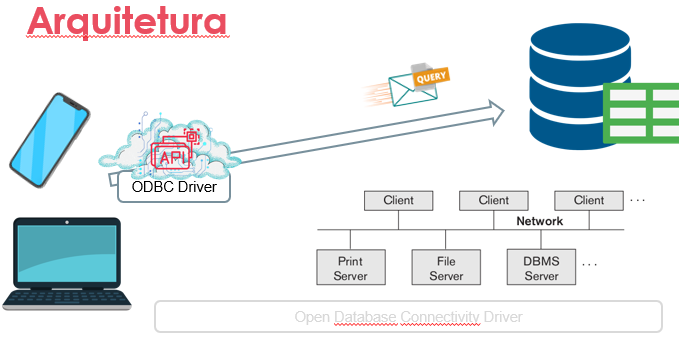
Um SGBD pode ter arquitetura centralizada ou distribuída.

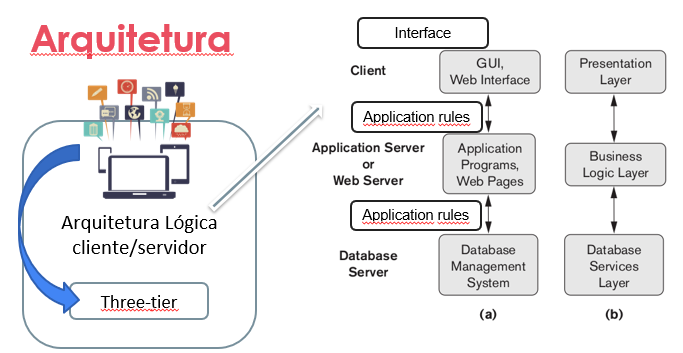
Arquitetura Centralizada:



Arquitetura distribuída:







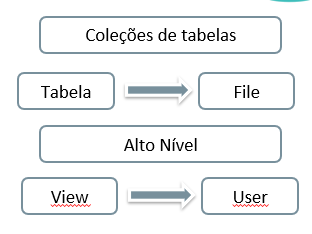
### Classificação de SGBDs

Critérios de classificação:

* **Modelos de dados:** Relacional, não relacional
* **Número de usuários:** quantidade de usuários
* **Números de sites: fontes heterogêneas:** big data, replicação, DB federado
* **Custos:** solução pessoal, AWS, Azure
* **Tipos de caminho de acesso:** como lidar com a estrutura de arquivos.

Performance - Online transacion processing (OLTP)

Classificação – relacional

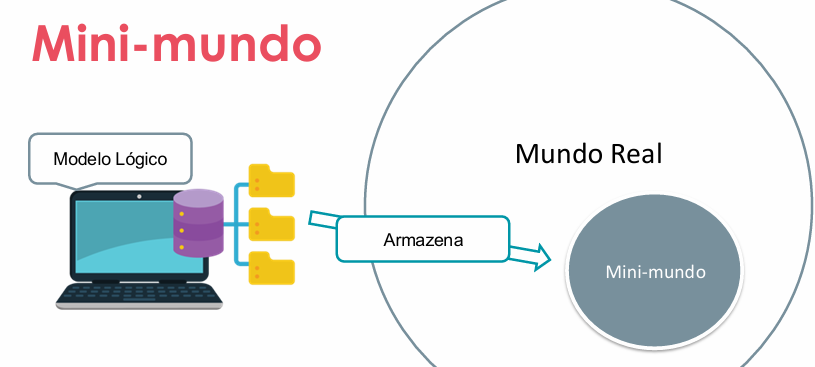


**Desafio textual**

* **Dados e banco de dados**
  + Dados são uma forma de representa fatos, conceitos e objetos do mundo real, podendo serem processador para gerar informação.
  + Banco de dados é uma coleção organizada de dados que estão relacionadas, com o objetivo de facilitar o acesso e uso.
* **SGBD, Sistema de banco de dados e Catálogo de BD**
  + SGBD são sistemas de gerenciamento de banco de dados, são ferramentas essenciais para o controle e funcionamento dos bancos de dados, sendo responsáveis por definir, criar, manipular e gerir diferentes aspectos relacionados aos bancos de dados.
  + Sistema de banco de dados é um conjunto formado pelo banco de dados, SGBD, usuários, aplicações e os procedimentos utilizados para acessar e gerenciar dados.
  + Catálogo de BD é uma estrutura presente em um SGBD que armazena metadados, usada para definir a estrutura de um banco de dados, como tabelas, colunas, tipos de dados, restrições, índice e permissões.
* **Independência program/data, user view**
  + Independência program/data é uma das características de sistemas de banco de dados, onde uma mudança na aplicação não irá afetar o sistema de banco de dados e vice-versa.
  + User view é outra característica do SGBDs que permite que diferentes usuários possam ter visões diferentes de um mesmo conjunto de dados, ou seja, pessoas de diferentes setores conseguem utilizar um mesmo conjunto de dados para objetivos diferentes.
* **DBA, transações canned, metadados e aplicação de processamento de transação**
  + DBA se trata do administrador de banco de dados, um profissional responsável por gerir o banco de dados a nível de implementação e manutenção.
  + Transações canned: são transações pré-definidas e padronizadas, geralmente incorporadas em aplicações.
  + Metadados são os dados que descrevem os dados, ou seja, é uma estrutura que descreve como os dados estão representados, já aplicação de processamento de transação é um tipo de aplicação que executa operações sobre o banco de dados de forma controlada e consistente.

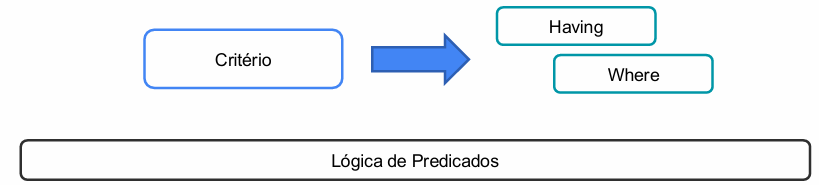
# Modelo de Entidade Relacional com Banco de Dados

**Mundo fechado:** dentro do contexto a ser modelado, eu tenho um mundo fechado

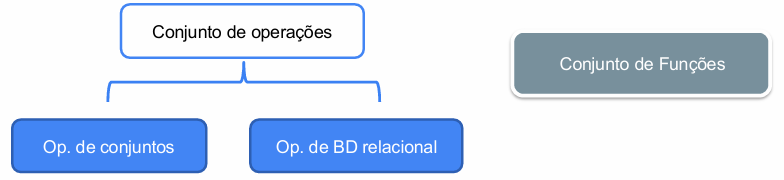


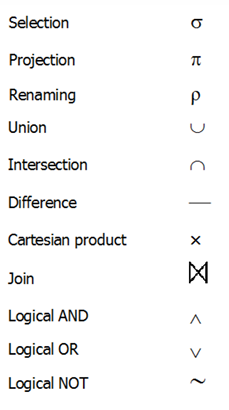
## Álgebra relacional

O **predicado** é a parte da oração que contém o verbo e que traz informações sobre o sujeito



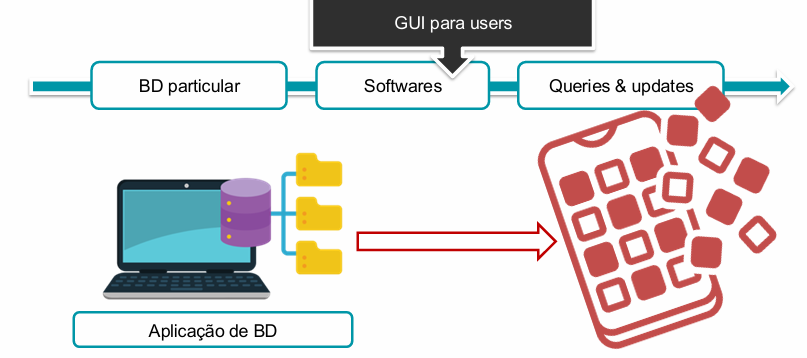
Linguagem formal para consulta/extração de dado.





## Processos

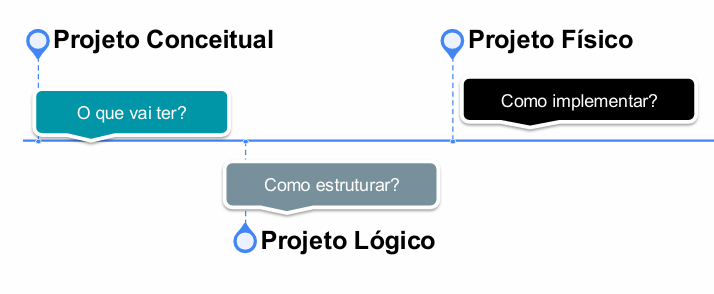
* Projeto Conceitual
* Projeto Lógico
* Projeto Físico
* Validação
* Produção
* Manutenção



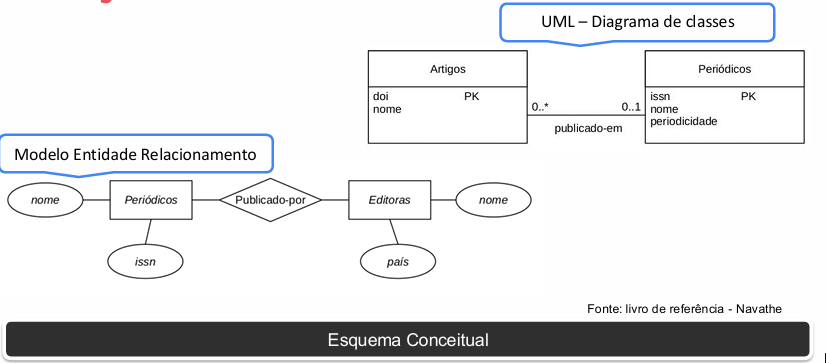
**Desenvolvimento**

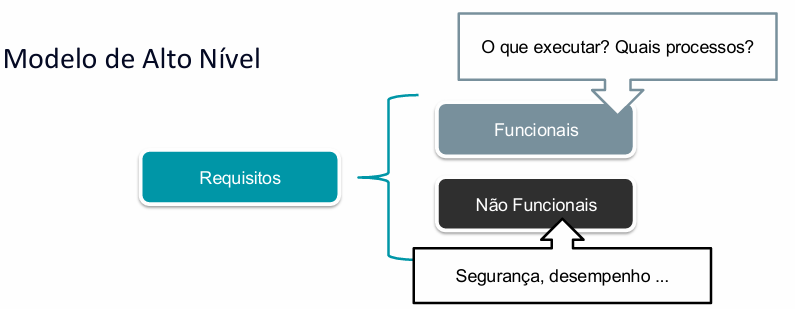
* Entender o contexto e requisitos
* Perfil (o que eu quero representar?)
* Implementação
* Modelo
* Arquitetura
* Funcionalidades

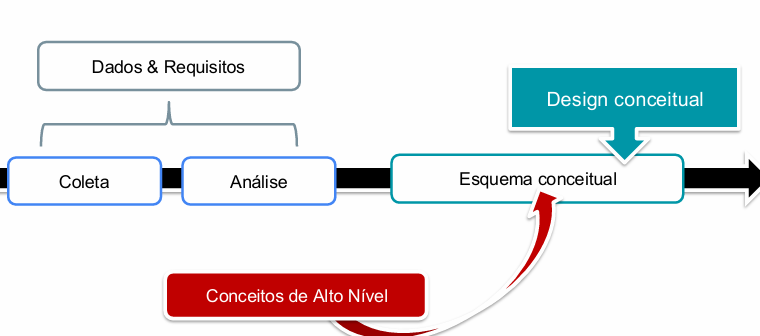
**Modelagem**

****

## Projeto Conceitual



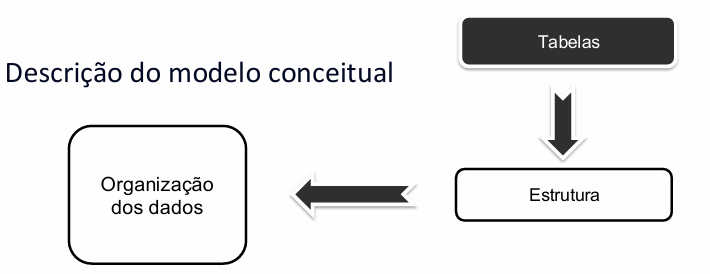


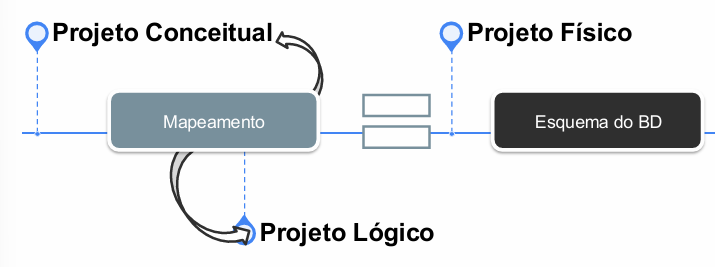


**Como criar o modelo?**

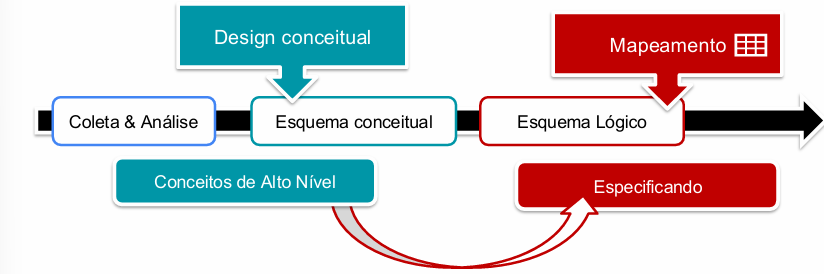
* **Linguagem de modelagem de dados:** gráficas, texto

## Projeto Lógico

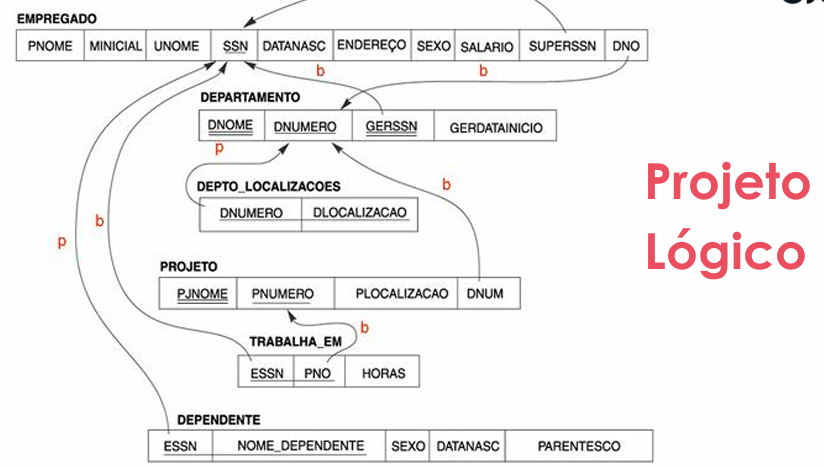




* Criação do Esquema Lógico – apesar de ser independente do projeto física, ainda possui dependência das características do modelo do SGBD
* Instalação e Configuração do SGBD
* Criação do Esquema do BD



O projeto lógico já se preocupa com características, como primary key.



## Projeto Físico

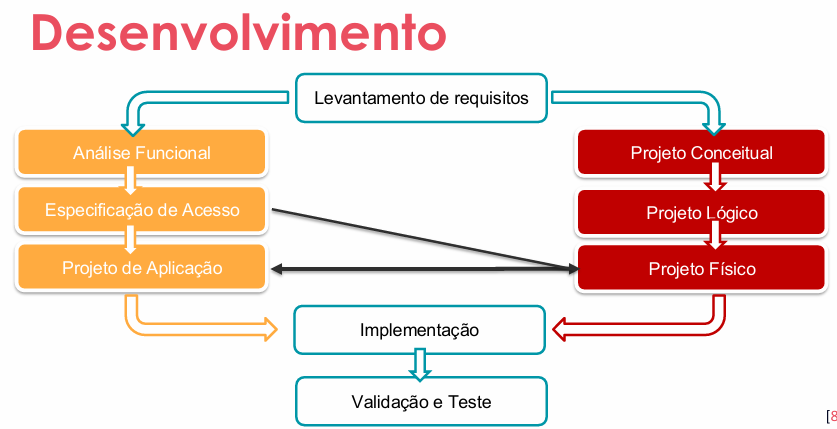
Descrever o modelo conceitual

Diretamente ligado ao SGBD

Parâmetros Físicos:

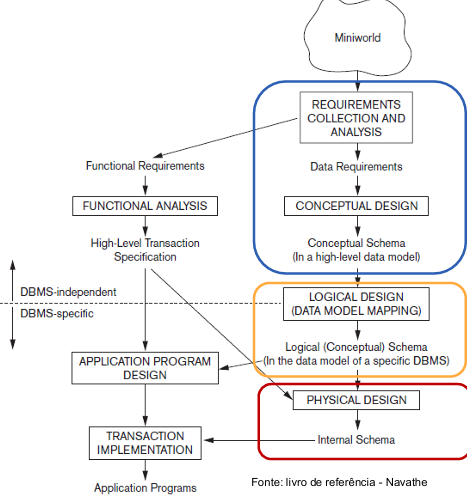
* Estruturas e índices
* Organização e caminhos de Arquivos
* Segurança, performance

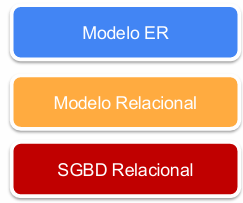
## Fase de Desenvolvimento



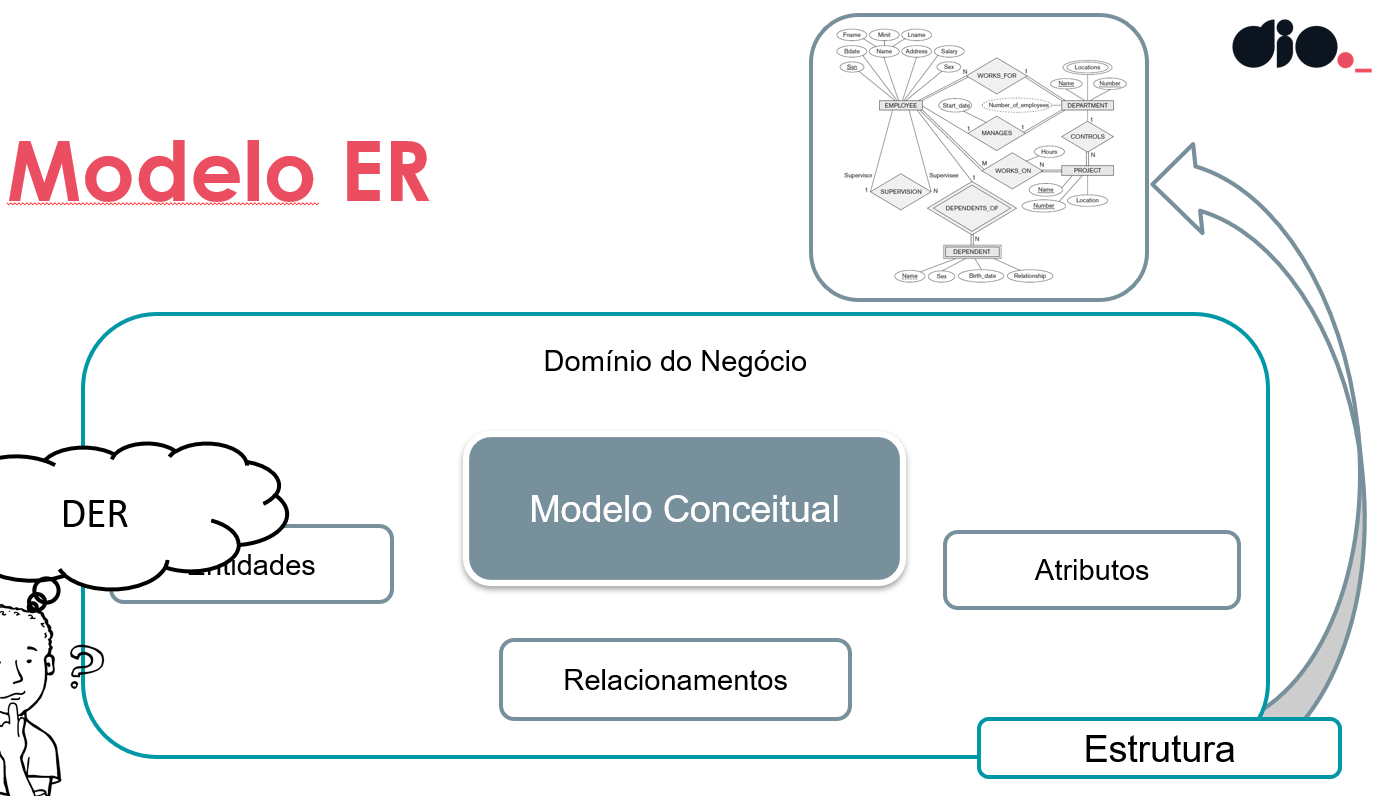
Uso dos dados: **OLTP e OLAP**

Modelo híbrido: **HTAP**





## Modelo ER: Tipos de Entidades, Chaves e Atributos



**Diagrama ER**

Foco no esquema

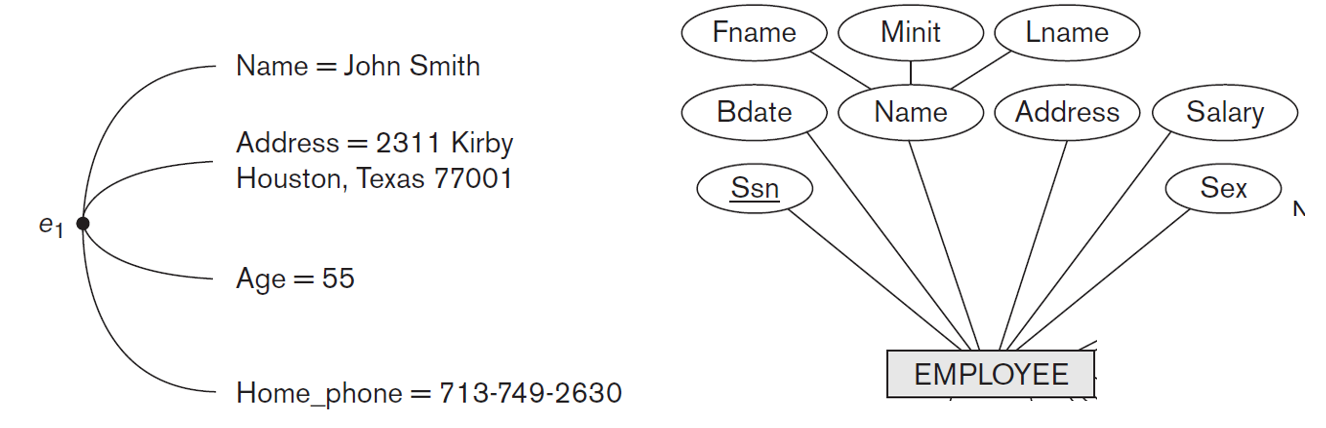
**Diagramas**

* Raras modificações
* Facilidade de manipular
* Esquema ER do banco de dados

**Entidades**

* Componente básico
* Existência independente
* Atributos

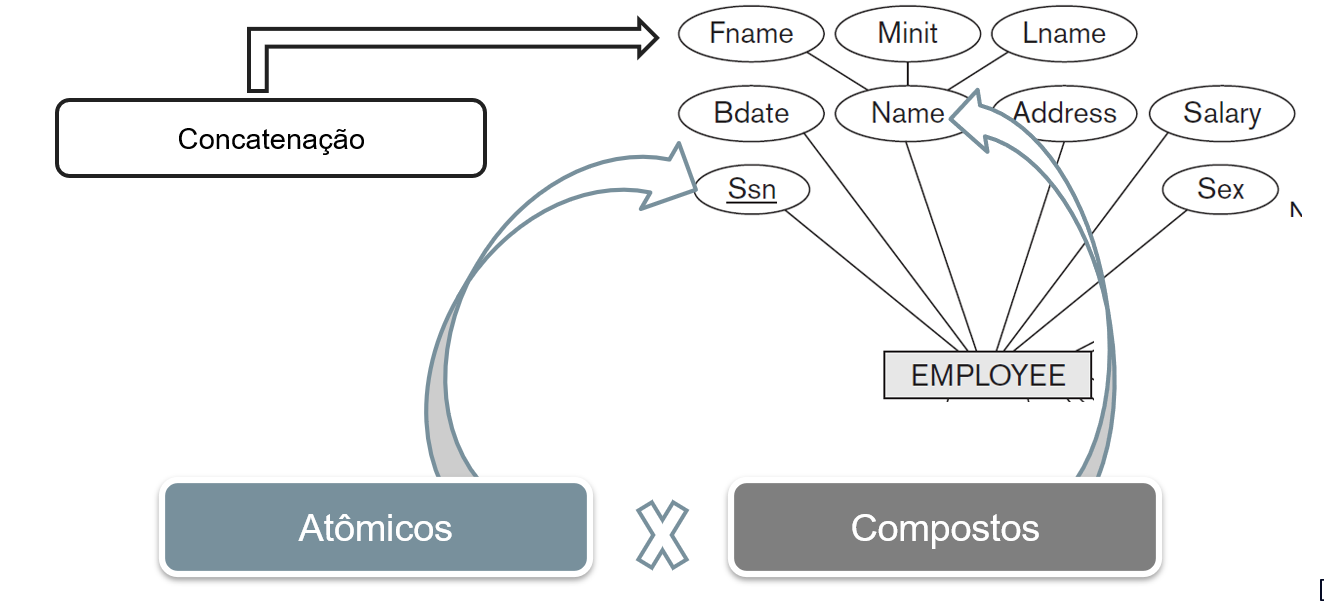
Entidades <> Objetos

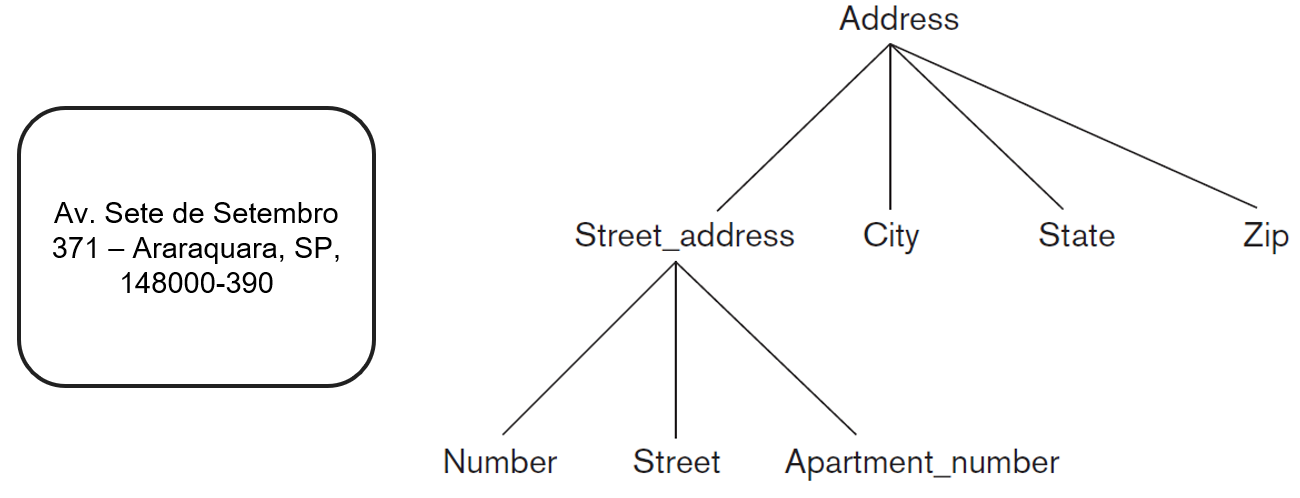


**Atributos – propriedades de uma entidade**

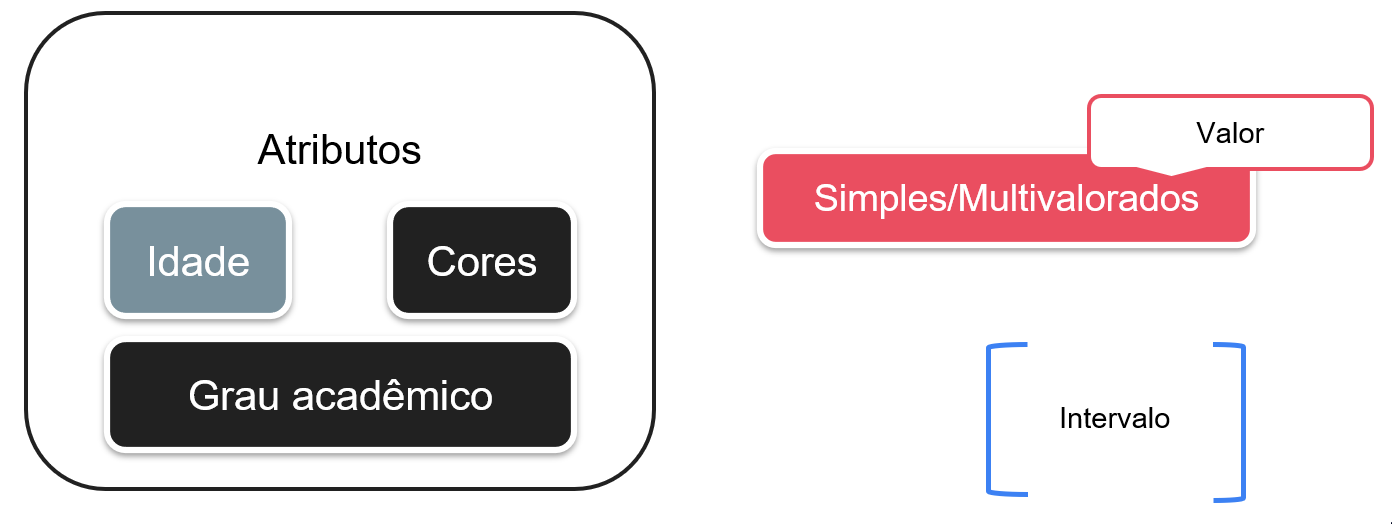
* Características/Descrição das entidades
  + Atômicos x composto
  + Atributos multivalorados
  + Armazenados e derivados
  + Atributos Nulos
  + Atributos Complexos
* Atributos relacionados as instâncias

**Atômicos x composto**

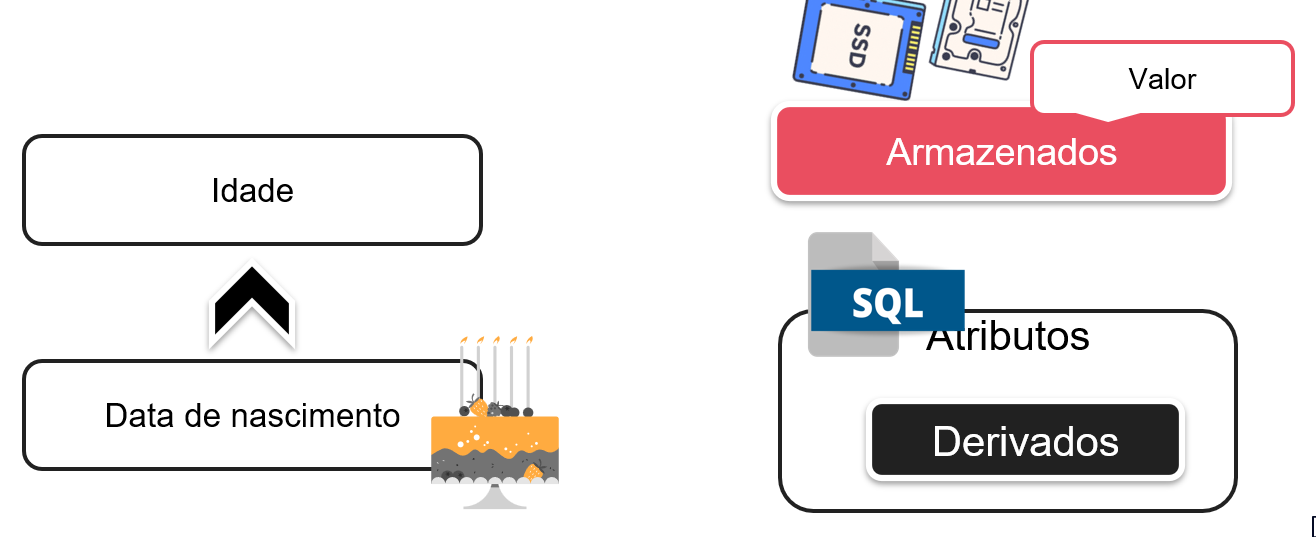
****

****

**Atributos multivalorados**

****

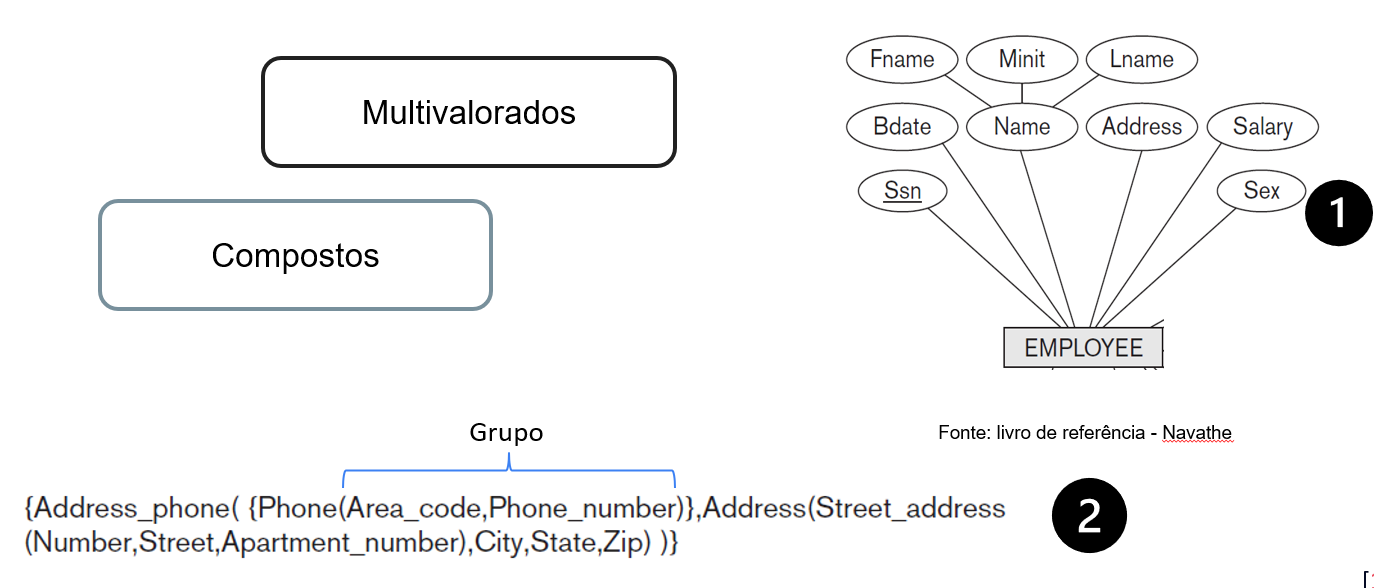
**Armazenados e derivados**

****

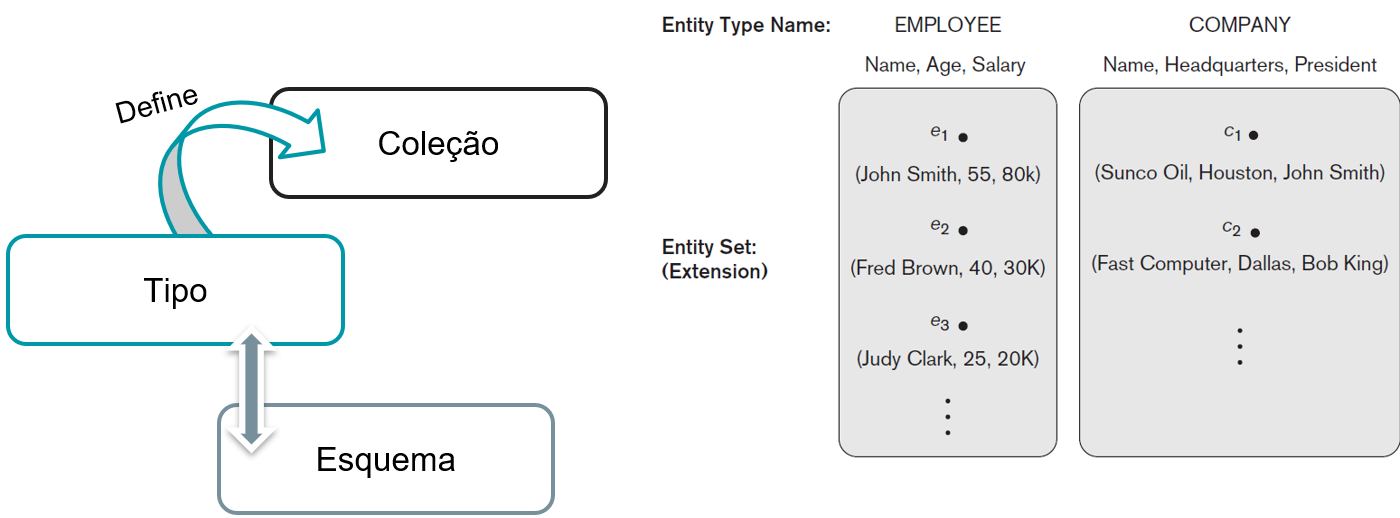
**Atributos Nulos**

* Valores faltando
* Grau acadêmico

**Atributos Complexos**

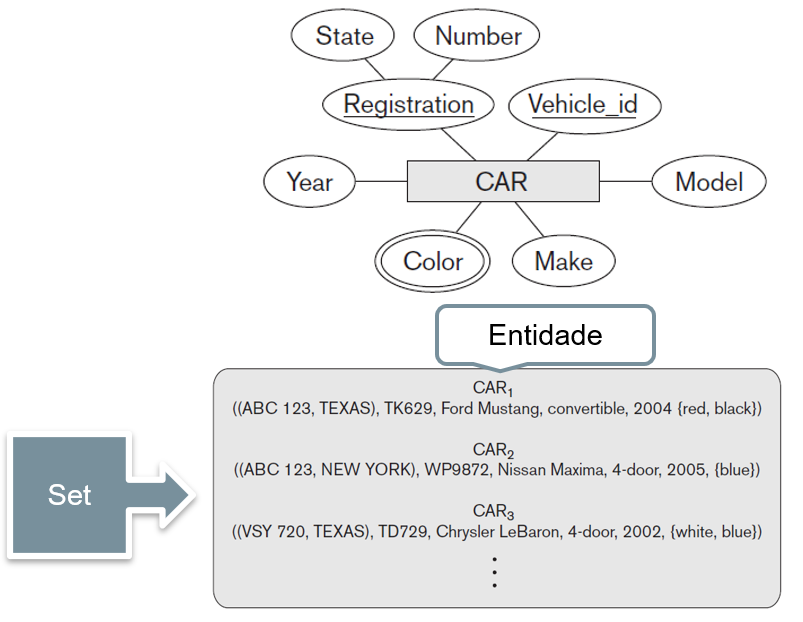


**Entidades**



**Elementos DER**

* Entidades – conjunto de entidade é uma instância (set).
* Atributos – multivalorados com elipse dupla
* Chaves – identificação, sublinhado



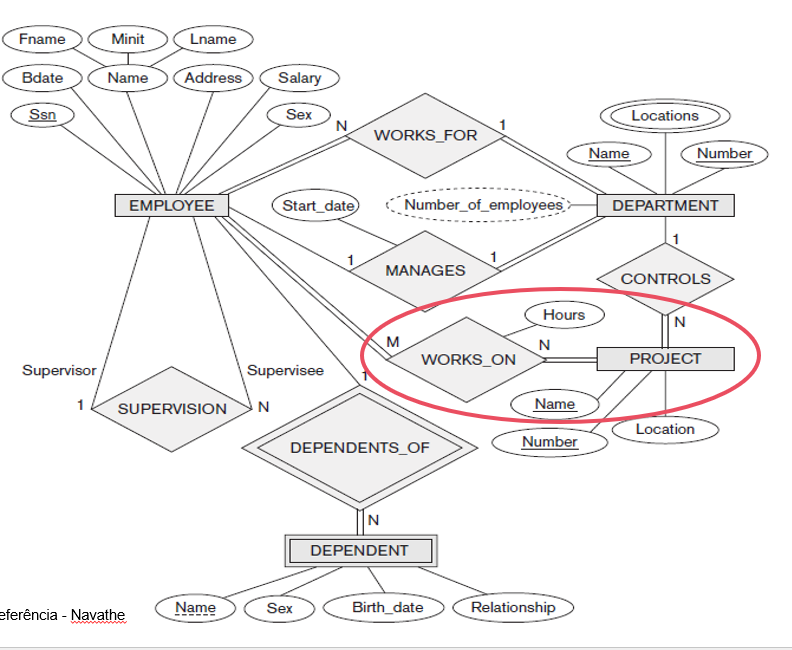
### Entidade Fraca

* Chave não obrigatória
* Dependência
* Exclusão cascata

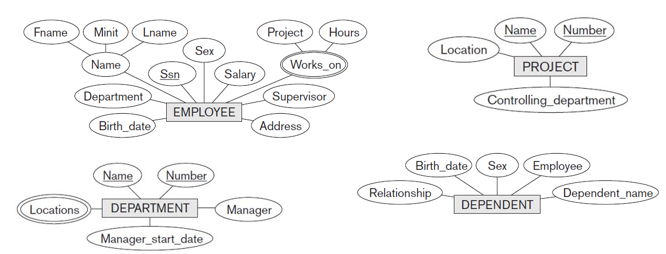
Diagrama

Descrição gerada automaticamente

### Exemplo de modelo conceitual

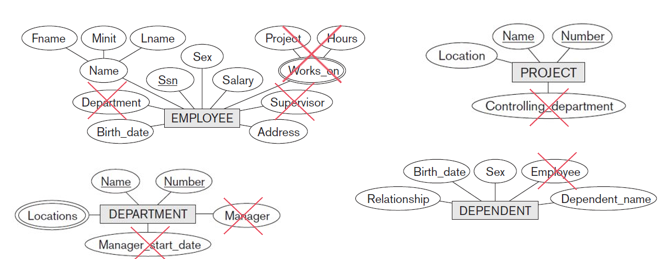


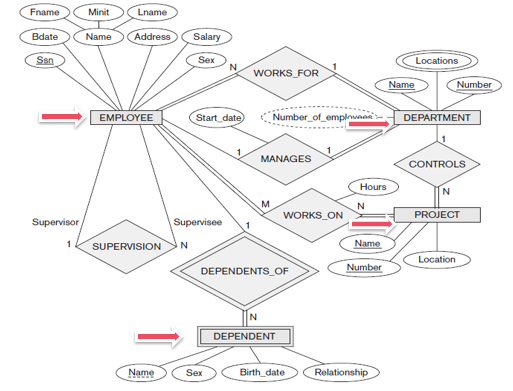
#### Refinando o exemplo



**Relacionamentos**

* Empregado x Departamento: **Gerencia**
* Empregado x Departamento: **Trabalha para**
* Departamento x Projeto: **Controla**
* Empregado x Empregado: **Supervisiona**
* Empregado x Projeto: **Trabalha em**
* Empregado x Dependente: **Dependentes de**

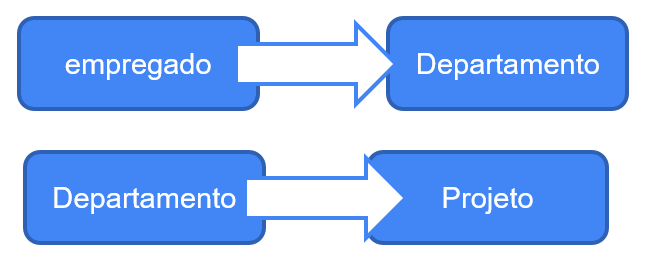


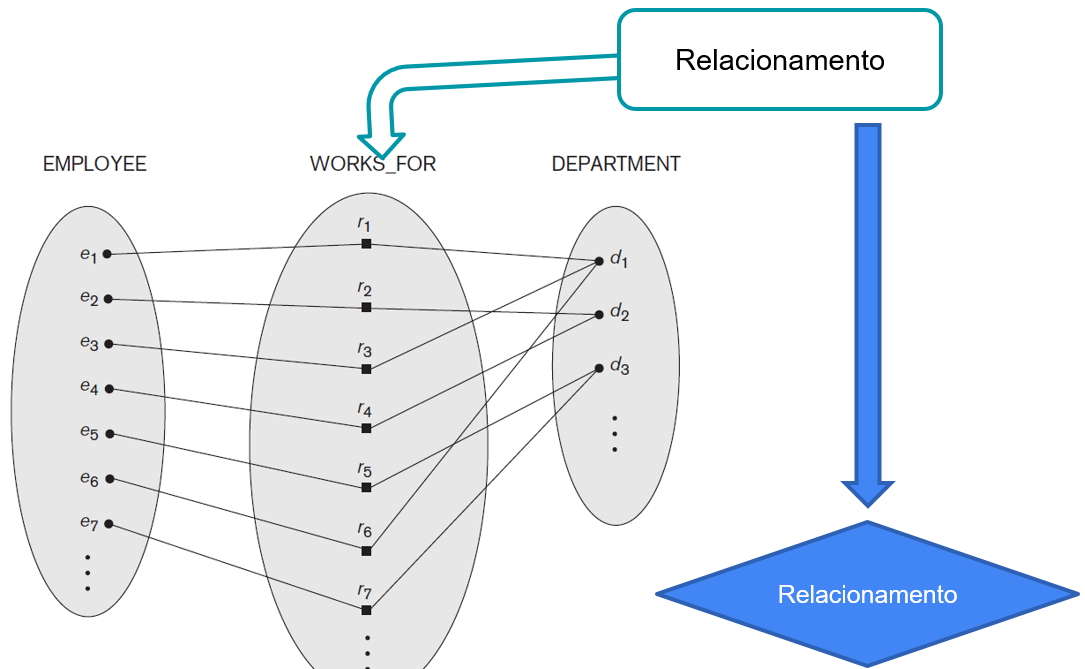


### Relacionamentos, Papéis e Constraints estruturais

**Relações implícitas**

* Gerente do departamento
* Controle do projeto
* Supervisor





**Classificado**

* Grau
* Auto relacional ou não
* Cardinalidade

Papel em cada entidade trás significado ao relacionamento

**Auto relacional**

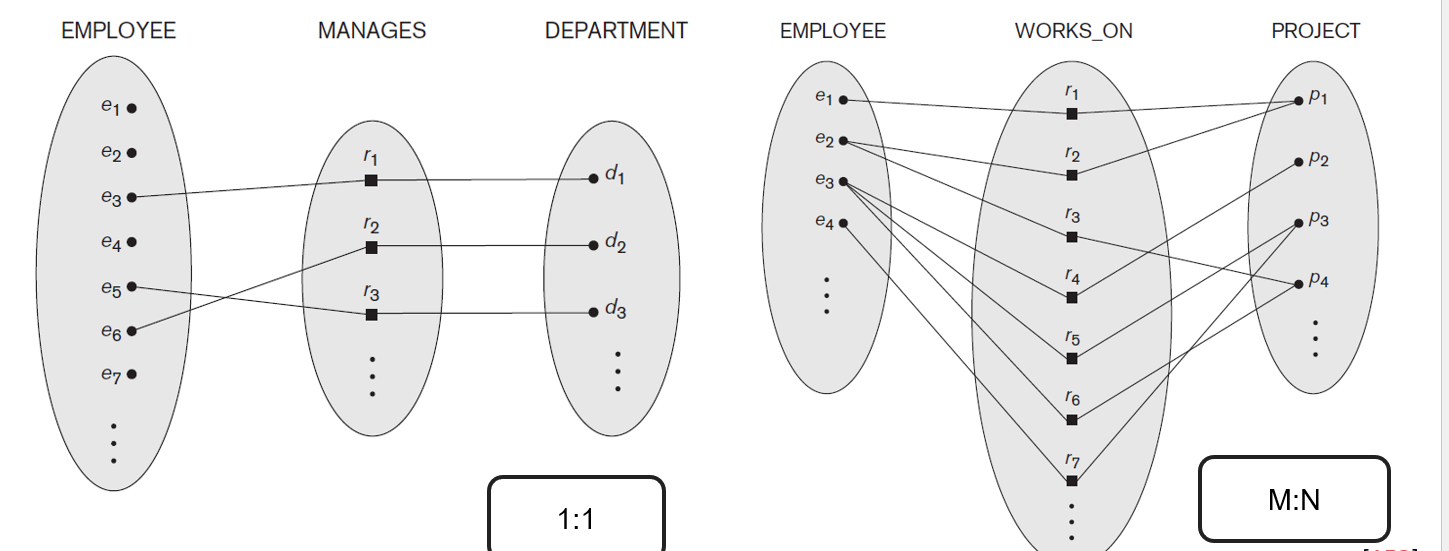


### Constraints de Relacionamento Binários

#### Cardinalidade

Corresponde ao número máximo de instância que participam de um determinado relacionamento.

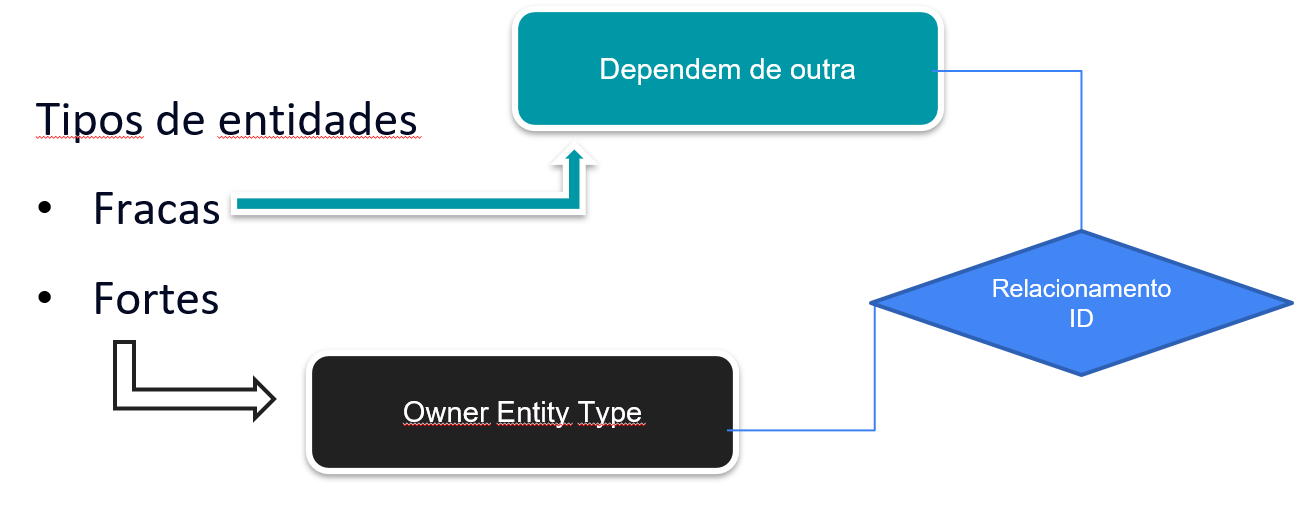
1:1 1:N N:1 1:3 M:N



#### Participação

* Relação de dependência
* Minimum Cardinality Constrain
* Classificação: total e parcial

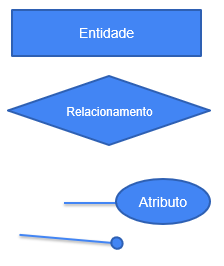
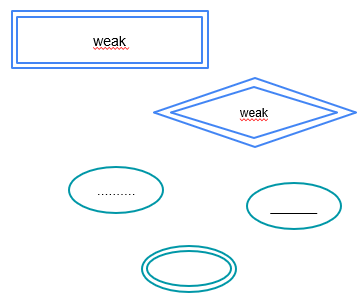
### Tipos de Entidades Fracas



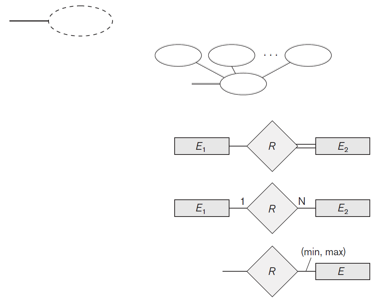
### Refinando o Design ER para Company

#### Notações

* **Entidades:** classes / objetos
* **Relacionamentos:** agregações
* **Atributos:** propriedades elementares
* **Weak:** Entidades, relacionamentos
* **Chave parcial:** linha pontilhada
* **Chave principal:** linha continua
* **Dependência de existência:** ||

* Atributo derivado
* Atributo composto
* Participação total
* Cardinalidade
* Constraint estrutural



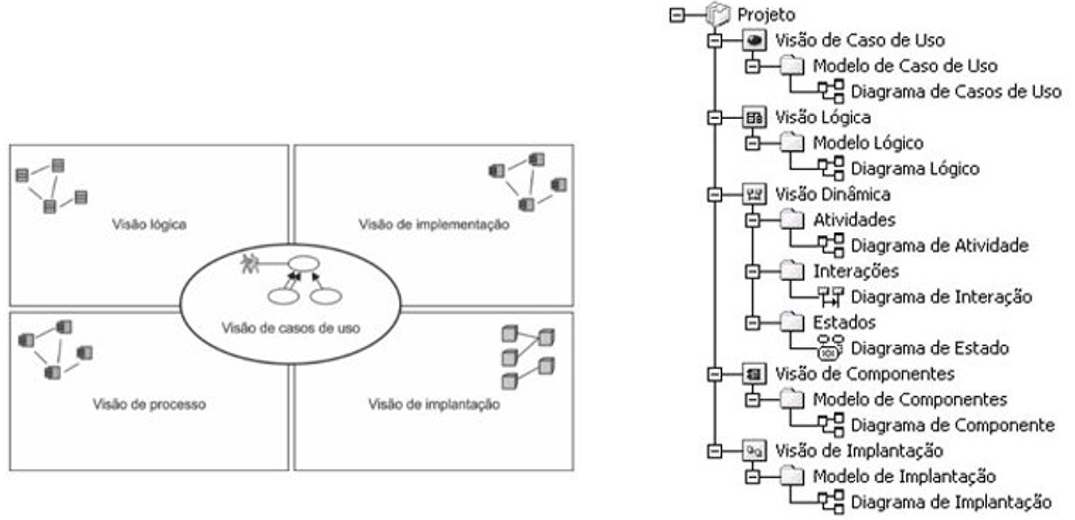
#### Construtores do Esquema

Convenção – Nomeando (Dando significado)

* Entidade e Relacionamento
* Atributo
* Função

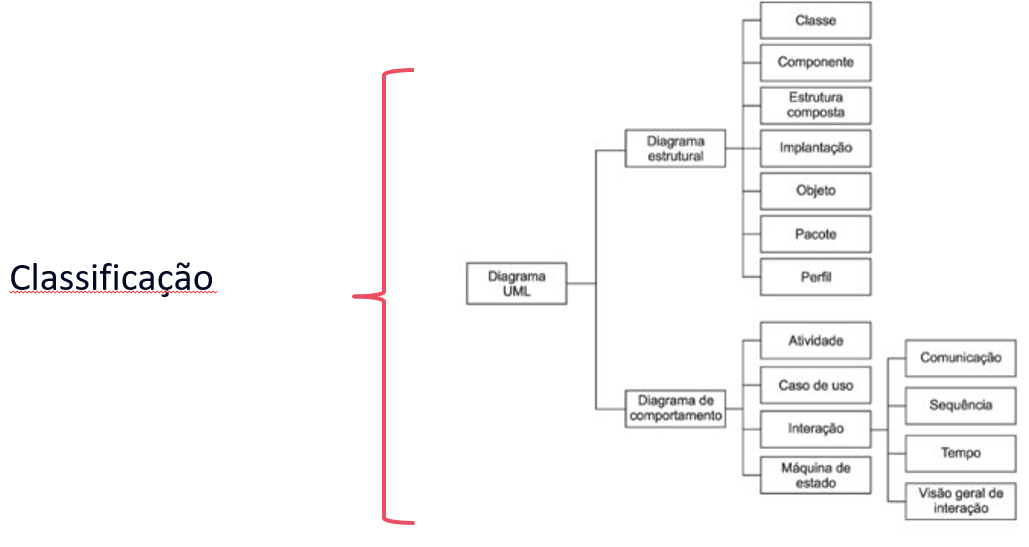
### Alternativas – UML

* Linguagem para desenvolvimento de software
* Trabalha com visões: interpretada e construída
  + Orientação a objeto
  + Liberdade para o desenvolvedor
  + Compreensão facilitada



Principais diagramas:

* Caso de uso
* Classe
* Interação



**Foco: Diagrama de Classes**

* Classe (entidade)
* Atributos
* Operações
* Associações
* Atributo de Link
* Associação binária
* Associação reflexiva
* Multiplicidade \*
* Relacionamentos
  + Associação e
  + Agregação
* Direcional: Atenção ao direcionamento da UML
* Weak
  + qualified association
  + Discriminator

#### Relacionamento N-ário

* Ternário, ..., N-ário (Trade-off na tomada de decisão
* Perspectiva diferente do binário
* Obs: instância refletem a visão (s, j,p)

## Modelo de Relacionamento de Entidade Aprimorado

* Evolução do modelo ER – 1970
* EER: novos conceitos semânticos
* Desenvolvidos fora da área de BDs
* Diagrama ERR

### Modelo ER extendido

* Generalização e Especialização
* Categoria ou Union Type
* Herança: Atributo e relacionamento



Herança: Princípio próprio à programação orientada a objetos (POO) que permite criar uma classe a partir de uma já existente

* Reutilização de código
* Agregar atributos e métodos
* Especialização de classes

Especialização: subclasses

Conjunto de subclasses: tipos de entidade

Diferenciação: propriedade, atributos

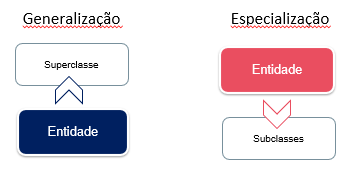
Instâncias das superclasses e subclasses

Mesma representação de mundo real

Papel especializado

Generalização

* Inverso da especialização
* Propriedades comuns as entidades



Constraint:

* Predicated-defined (condição)
* Attribute-defined Specialization (Especialização definida por atributo / Atributo definidor da especialização).
* User-defined
* Disjointness constraint (d) – conjuntos disjuntos
* Overlapping (o)
* Completeness constraint – total (linha dupla), parcial (linha simples)

Hierarquia e rede de especialização

Union Type (u)

* Coleção de objetos de diferentes tipos de entidades
* Representação: subclasse

Decisão sobre design de BD e UML

Generalização/Especialização : pode ser destrinchado em subclasses

Union Type/Categorização: classes diferentes que possuem uma relação

Guideline

* Sub/superclasses: acurácia x desordem
* Merge de subclasse em uma superclasse: s/relacionamentos e poucos atributos
* Union Types: evitados por padrão
* Escolhas guiadas pelo mini-mundo

Representação por UML

* Foco na modelagem de dados
* Diagrama de classes
* Classes concretas
* Classes base e classes (nós) folha

Terminologias

Knowledge representation (KR)

Ontologia – Domínio

Similaridades

* Processo de abstração para identificação de componentes
* Prove recursos para modelagem de dados

KR

* Modelagem semântica – mais abrangente: regras, conhecimento espacial e temporal
* Reasoning mechanisms
* Mistura de instância e esquema

Conceitos abstratos

* Classificação e Instanciação: atribuição de objetos/entidades a classes/tipo de entidade
* Identificação: processo abstrato de identificação única
* Especialização e Generalização: Refinamento utilizando sub e superclasses
* Agregação e Associação: abstração para composição de objetos complexos

Ontologia

Alimenta o BD com informações sobre o domínio, suas propriedades e relações

Descrição:

* Thesaurus - relacionamentos x conceitos
* Taxonomy – relacionamento de conceitos com estruturas
* Esquema detalhado – descrição de conceitos
* Lógica teórica – lógica matemática para definição de conceitos

## Projeto de Banco de dados

**Contexto:** Levantamento de Requisitos

**Projeto Conceitual:** Modelo ER

**Projeto Lógico:** Modelo Relacional

### Exemplo 1: Ordem de serviço

# Explorando a Linguagem de Consulta a Banco de dados SQL

# Técnicas Avançadas em Banco de Dados com MySQL

# Transações & Gerenciamento de Banco de Dados com no MySQL