Projeto de Banco de Dados

Tipos de Banco de Dados

Banco de dados Relacional

Um banco de dados relacional consiste em uma coleção de tabelas, que podem ser relacionadas através de seus atributos, ou seja uma linha de uma tabela pode estar sendo relacionada com uma outra linha em uma outra tabela.

Banco de dados rede

Enquanto no modelo relacional os dados e os relacionamentos entre dados são representados por uma coleção de tabelas, modelo de rede representa os dados por coleções de registros e os relacionamentos entre dados são representados por ligações.
Ou seja um banco de dados de rede consiste em uma coleção de registros que sào conectados uns aos outros por meio de ligações. Cada registro é uma coleção de campos (atributos), cada um desses campos contendo apenas um valor de dado. Uma ligação é uma associação entre precisamente dos registros.

Cliente

Nome	Rua	Cidade
Paulo	Oliveira	Campinas

Conta

Número	Saldo
100-01	100,00

Banco de dados Hierárquico

Assim como no modelo de Redes o modelo Hierárquico trabalho com os dados e relacionamentos como uma coleção de registros relacionados por ligações. A única diferença entre os dois é que o modelo Hierárquico os registros são organizados como coleções de árvores em vez de grafos arbitrários.

Cliente

Nome	Rua	Cidade
Paulo	Oliveira	Campinas

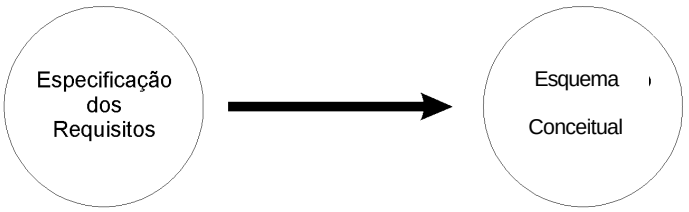
Conta

Número	Saldo
100-01	100,00

Fases do Projeto de Base de Dados

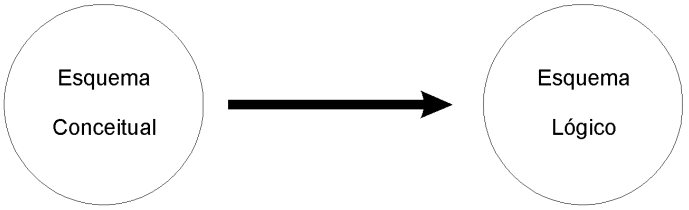
O Projeto de Base de Dados pode ser decomposto em:

Projeto Conceitual



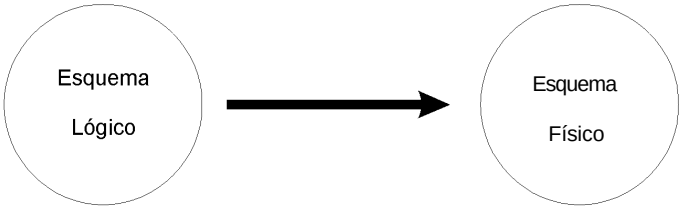
- Independe do DBMS escolhido
- Modelo Conceitual: Linguagem usada para descrever esquemas conceituais

Projeto Lógico



- Modelo lógico: Linguagem usada para especificar esquemas lógicos
- Pertencem a três classes: Relacional, Redes e Hierárquico

Projeto Físico



- Esquema físico: É a descrição da Implementação da base de dados em memória secundária. Descreve estruturas de armazenamento e métodos de acesso.
- Tem forte ligação com o DBMS específico.

Resumindo

- Projeto Conceitual: Não tem dependência com a classe do GBD a ser escolhido.
- Projeto Lógico: Tem dependência com a classe, mas não com o GBD específico.
- Projeto Físico: Total dependência do GBD específico.

Conclusões

Uma das vantagens em se trabalhar com projeto conceitual está na possibilidade de se adiar a escolha do GBD (mesmo a sua classe). O projetista deve concentrar o maior esforço nesta fase do projeto pois, a passagem para as outras fases é mais ou menos automática.

Outra vantagem está na possibildade de usuários não especialistas em bancos de dados darem diretamente a sua contribuição no projeto conceitual cuja maior exigência é a capaciade de abstração. A aproximação com o usuário final melhora bastante a qualidade do projeto.

Projeto Conceitual

O Projeto Conceitual produz um esquema conceitual a partir de "requisitos" de um mundo real.

- Projeto conceitual usa modelo de dados para descrever a realidade.
- Um modelo de dados se ampara em um conjunto de blocos de construção primitivas.

Abstração

Processo que consiste em mostrar as características e propriedades essenciais de um conjunto de objetos, ou esconder as características não essenciais.

Quando pensamos no objeto "bicicleta" de uma forma abstrata, normalmente "esquecemos" seus detalhes e as particularidades que as diferem entre si.

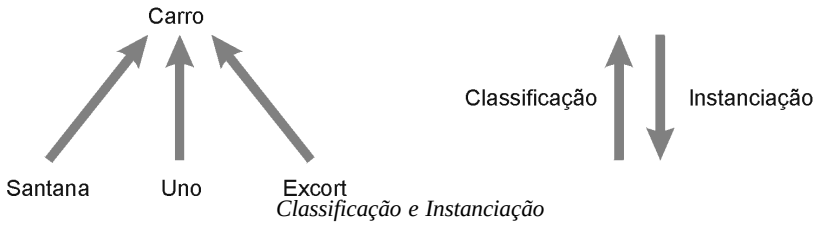
Abstrações em Projetos Conceituais

Existem 3 Tipos:

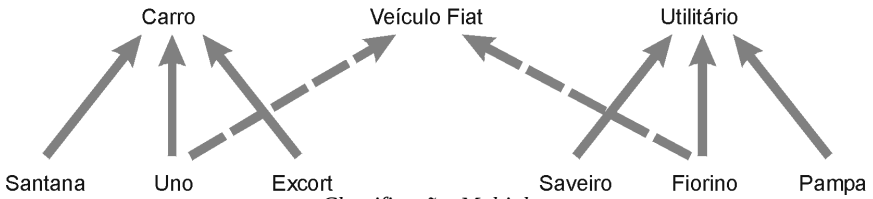
- Classificação
- Agregação
- Generalização

Classificação

Usada para reunir objetos do mundo real com propiedades comuns, formando (ou definindo) classes.



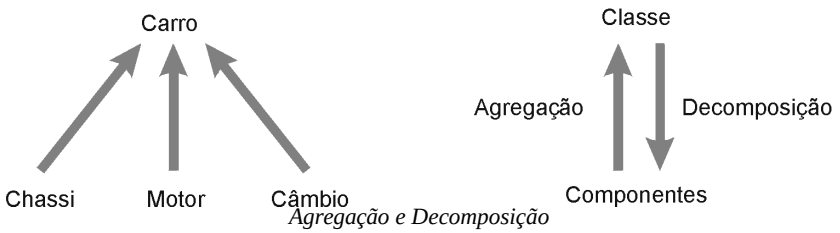
Classificação e Instanciação



Classificações Multiplas

Agregação

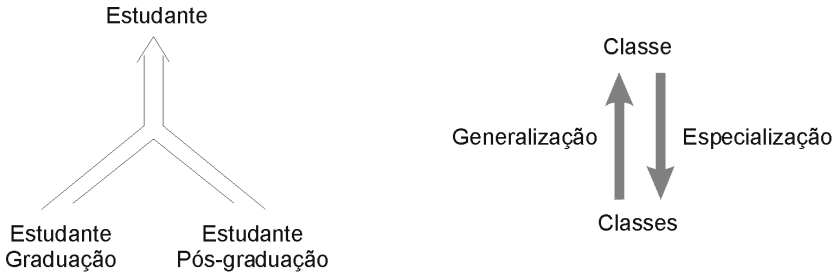
Usada para definir uma nova classe a partir de um conjunto de classes que representam suas partes componentes.



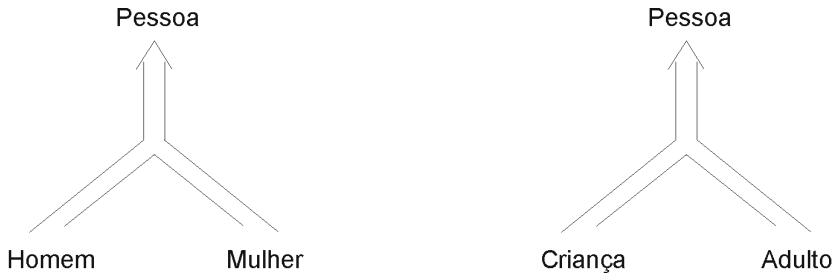
Agregação e Decomposição

Generalização

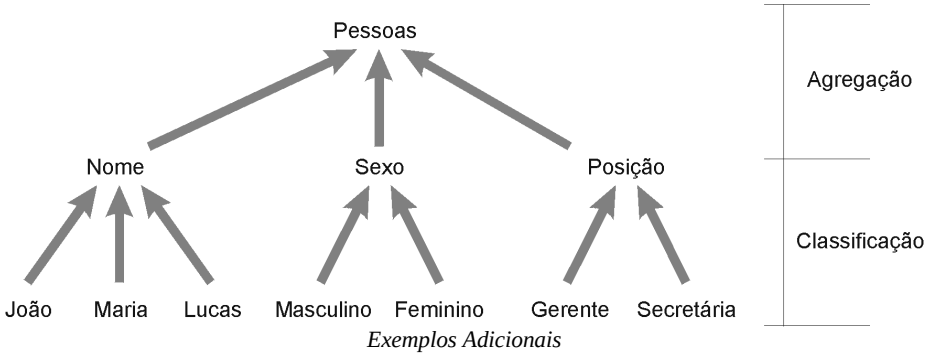
Usada para definir uma classe mais genérica a partir de duas ou mais classes.



Generalização e Especialização



Exemplos de Generalização



Exemplos Adicionais

Cobertura da Generalização

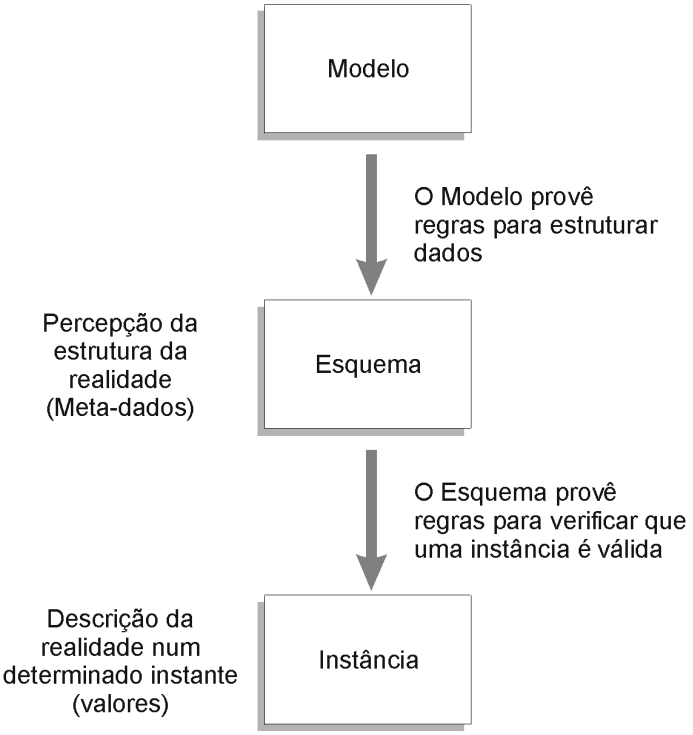
- Total / Exclusiva
- Total / Não Exclusiva

Modelos de Dados

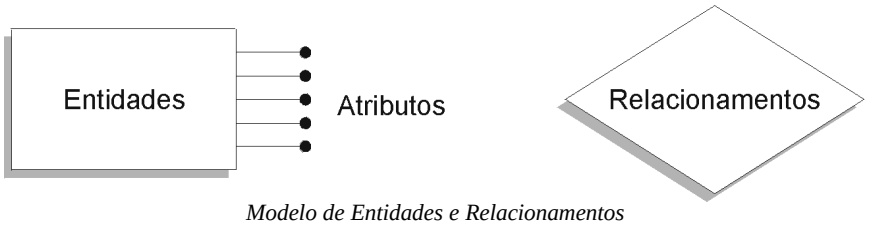
Conceitos: Modelo e Esquema

Um **modelo de dados** é uma coleção de conceitos usados para para descrever uma dada realidade. Estes conceitos são construidos com base nos mecanismos de abstração e são descritos através de representações gráficas e lingüísticas.

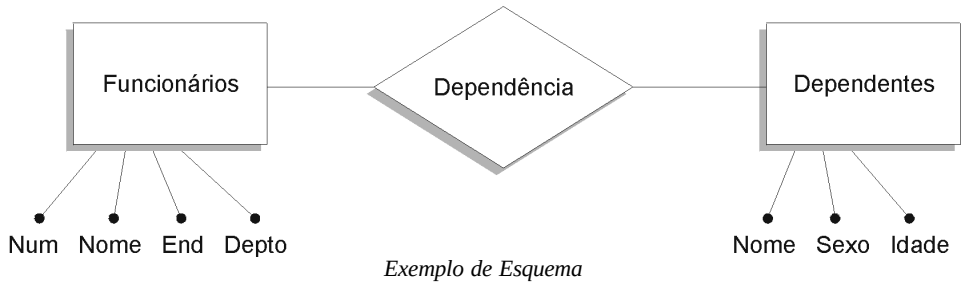
Um **esquema** é uma representação de uma porção específica da realidade usando-se um particular modelo de dados.



Para exemplificar vamos utilizar o modelo de entidades e relacionamentos (M.E.R.) o qual veremos com maior detalhamento mais adiante)



Modelo de Entidades e Relacionamentos



Exemplo de Esquema

Funcionários

Num	Nome	End	Depto
512	José	R. das Flores	Compras
6 03	João	R. das Rosas	Vendas
704	Carlos	R. das Acácias	Estoque
708	Pedro	R. das Camélias	Vendas
710	Maria	R. Azaléia	R/H

Dependentes

Nome	Sexo	Idade
Huguinho	M	12
Zezinho	M	13
Luizinho	M	12
Xuxa	F	20

Exemplos de Instância

Há dois tipos de modelos de dados:

- **Mod. Concietuais**: são ferramentas que representam a realidade num alto nível de abstração.
- **Mod. Lógicos**: suportam descrições de dados que podem ser processadas (por um computador). Incluem os modelos relacional, hierárquico e rede.

Obs: projeto de base de dados não é a única aplicação de modelos conceituais. Eles podem ser excelentes ferramentas para gestão em empresas.

Por recomendação do comitê ANSI/SPARC (metade dos anos 70) todo sistema de base de dados deveria ser organizado de acordo com 3 níveis de abstração de dados:

- **Externo**: também chamado de visão. Descreve o ponto de vista de grupos específicos de usuários sobre a porção da base de dados que é interessante preservar para aquele grupo particular.
- **Conceitual**: representação de alto nível, independente da máquina, sobre toda a base de dados. Também chamada de "*Enterprise Scheme*".
- **Interno**: descrição da implementação física da base de dados. Dependente da máquina.

M. E. R. - O Modelo de Entidades e Relacionamentos

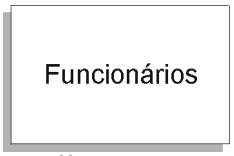
É o mais difundido modelo de dados para projeto conceitual de base de dados. Foi introduzido por Peter Chen (1976) e posteriormente recebeu extensões.

Elementos básicos do modelo: Entidades e Relacionamentos

Entidades

Representam classes de objetos do mundo real.

Exemplos: FUNCIONÁRIOS, ALUNOS, PROFESSORES, CIDADES, etc.

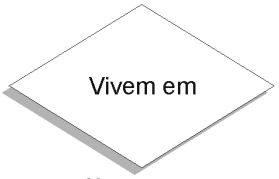


Representação gráfica da entidade Funcionários

Relacionamentos

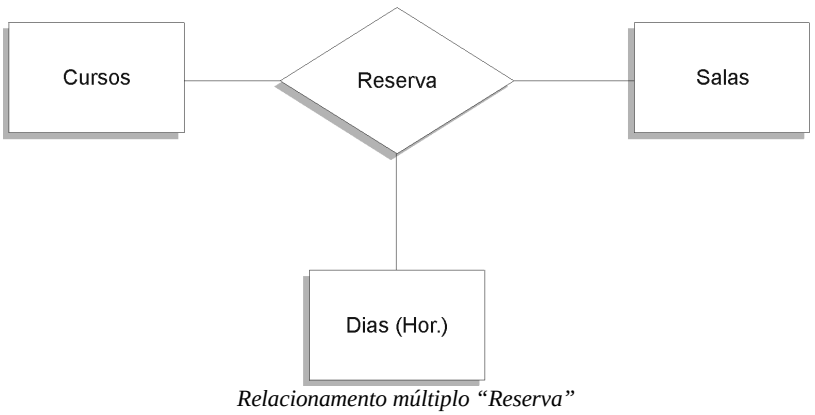
Representam agregações de duas ou mais entidades.

Exemplos: "Nascidos em" entre "Funcionários" e "Cidades" e "Vivem em" também entre "Funcionários" e "Cidades".



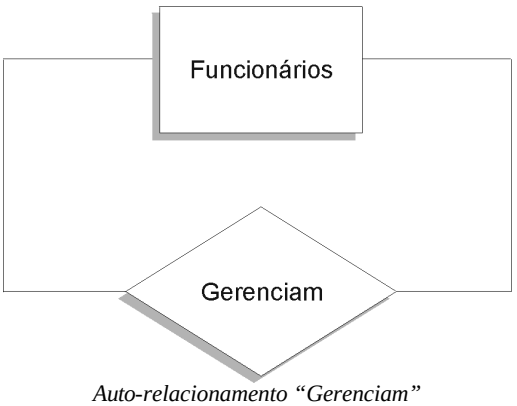
Representação gráfica da entidade "Vivem em"

O relacionamento pode conectar mais que duas entidades simultaneamente . Neste caso, é chamado "**relacionamento múltiplo**".



Relacionamento múltiplo "Reserva"

Um relacionamento pode conectar entidades de um mesmo conjunto. Neste caso temos o "**auto-relacionamento**".



Auto-relacionamento "Gerenciam"

Vamos considerar o esquema dos relacionamentos entre "Funcionários" e "Cidades". Para este esquema poderíamos ter a seguinte instância:

- FUNCIONÁRIOS = { f1, f2, f3, f4 }
- CIDADES = { c1, c2, c3 }
- VIVEM_EM = { <f1,c1>, <f2,c1>, <f3,c2>, <f4,c3> }
- NASCIDOS_EM = { <f1,c1>, <f2,c1>, <f3,c2>, <f4,c2> }

Cardinalidades podem ser expressas através de valores mínimos e máximos. Por exemplo:

- MIN_CARD (FUNCIONÁRIOS, VIVEM_EM) = 1
- MAX_CARD (FUNCIONÁRIOS, VIVEM_EM) = 1
- MIN_CARD (CIDADES, VIVEM_EM) = 0
- MAX_CARD (CIDADES, VIVEM_EM) = N

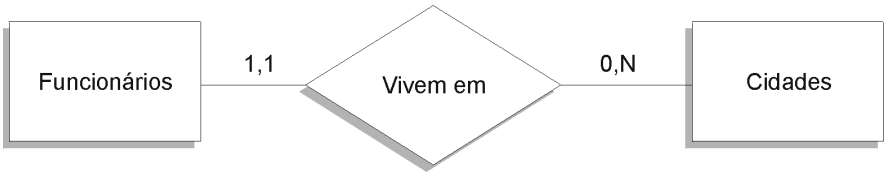
Nós declaramos através desta representação (língüistica) que o relacionamento "Vivem em" é "vários para um" entre "Funcionários" e "Cidades", através de "Vivem em".

A participação de "Funcionários" é obrigatória no relacionamento, enquanto a de "Cidades" é opcional.

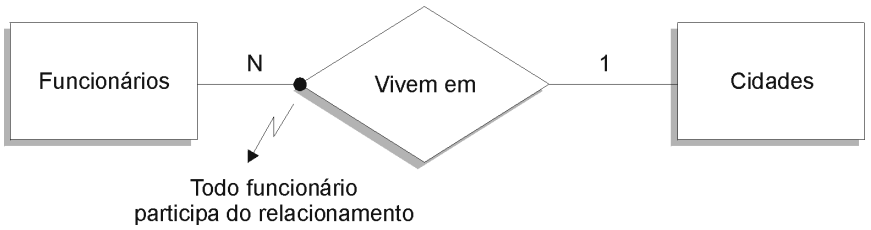
Outra forma de declarar as cardinalidades acima seria:

- CARD (FUNCIONÁRIOS, VIVEM_EM) = (1,1)
- CARD (CIDADES, VIVEM_EM) = (0,n)

Representação gráfica correspondente à declaração lingüística acima:

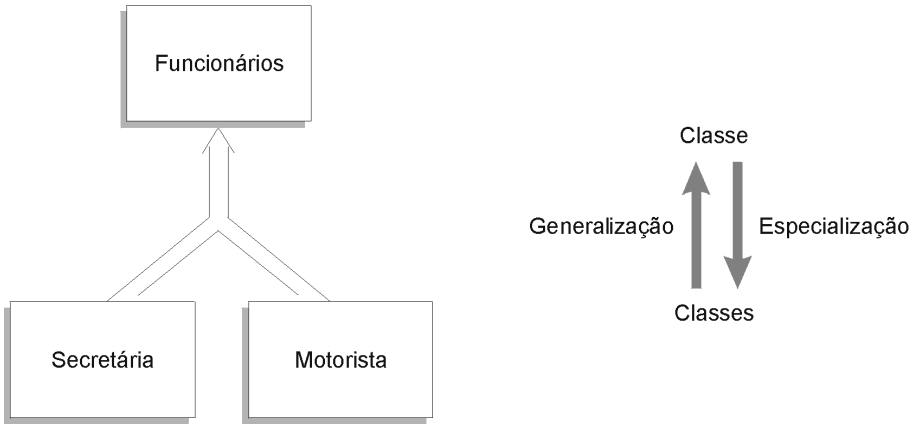


Outra representação para o exemplo acima é mostrada através do esquema abaixo. Esta será a representação que normalmente seguiremos.

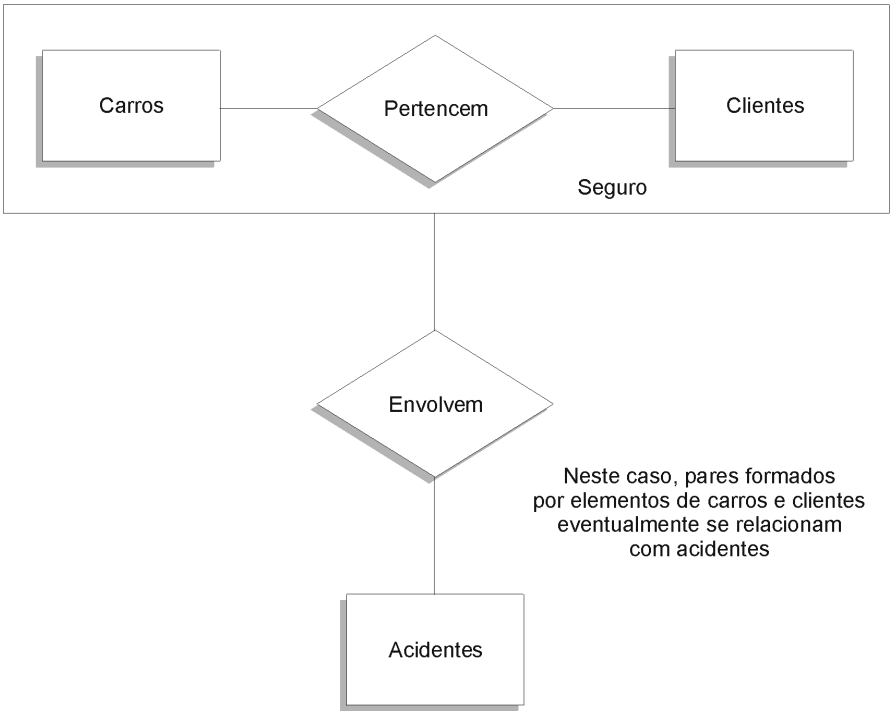


Existem casos práticos em que um conjunto de entidades representa elementos do mundo real que se subdividem em categorias. Esta subdivisão pode ser representada pelo particionamento do conjunto de entidades o que representa uma abstração de generalização (ou especialização).

Exemplo:



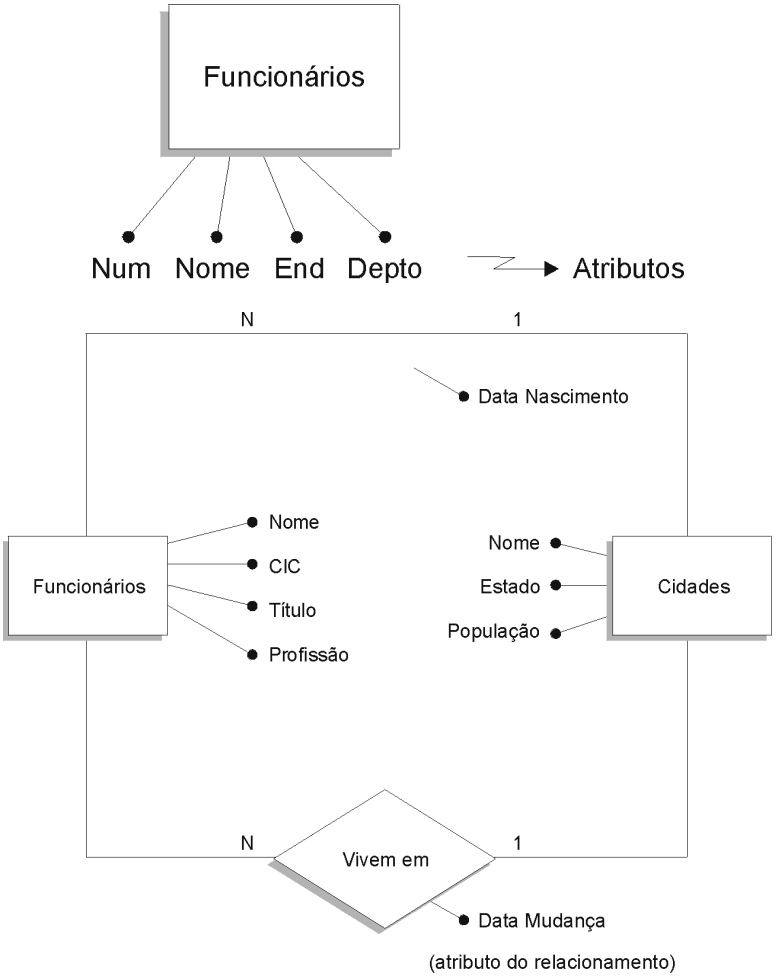
Extensões do Modelo: Agregação



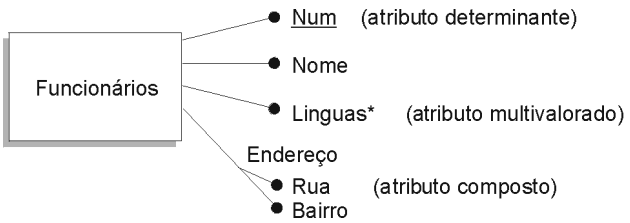
Atributos

Representam propriedades elementares de entidades ou relacionamentos.

Exemplos:

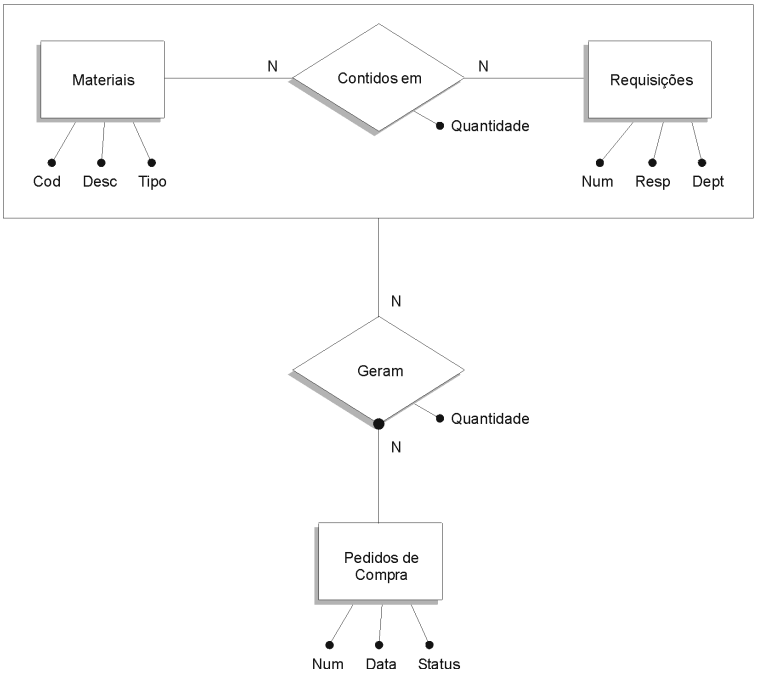


Tipos de Atributos

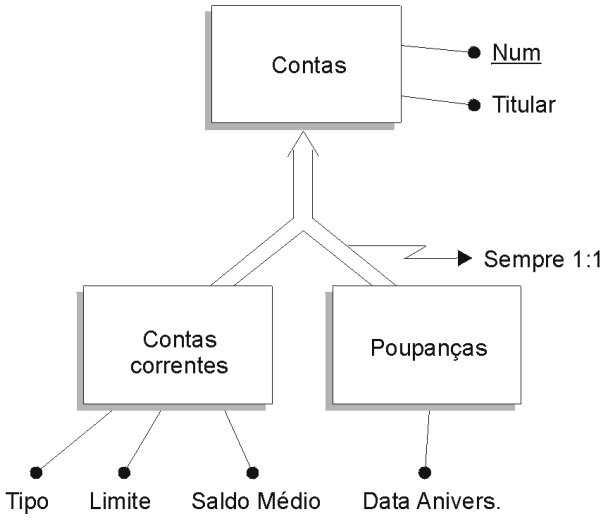


Obs: os atributos determinantes determinam univocamente um objeto dentro de um conjunto de entidades.

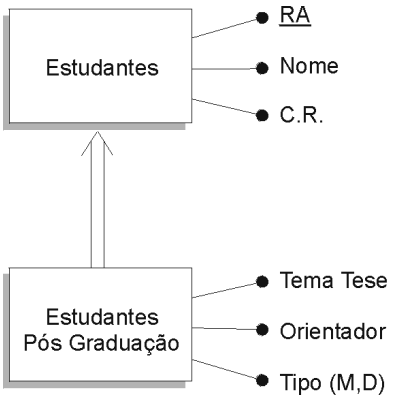
Exemplos adicionais:



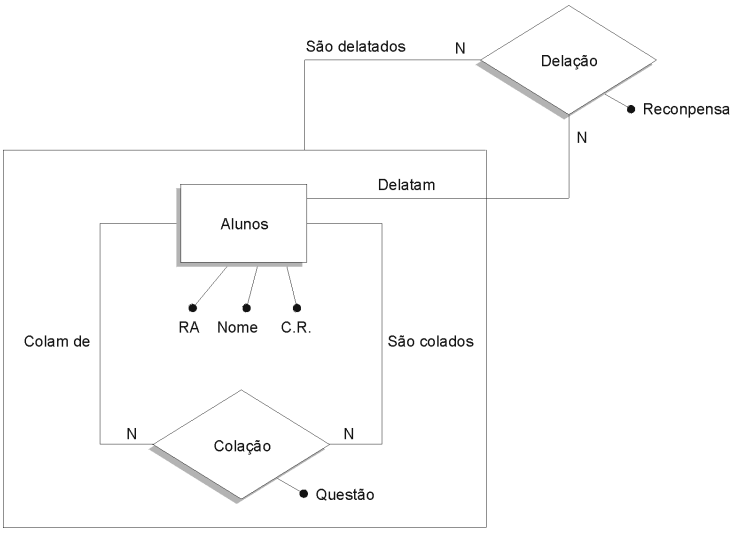
Exemplo 1



Exemplo 2



Exemplo 3

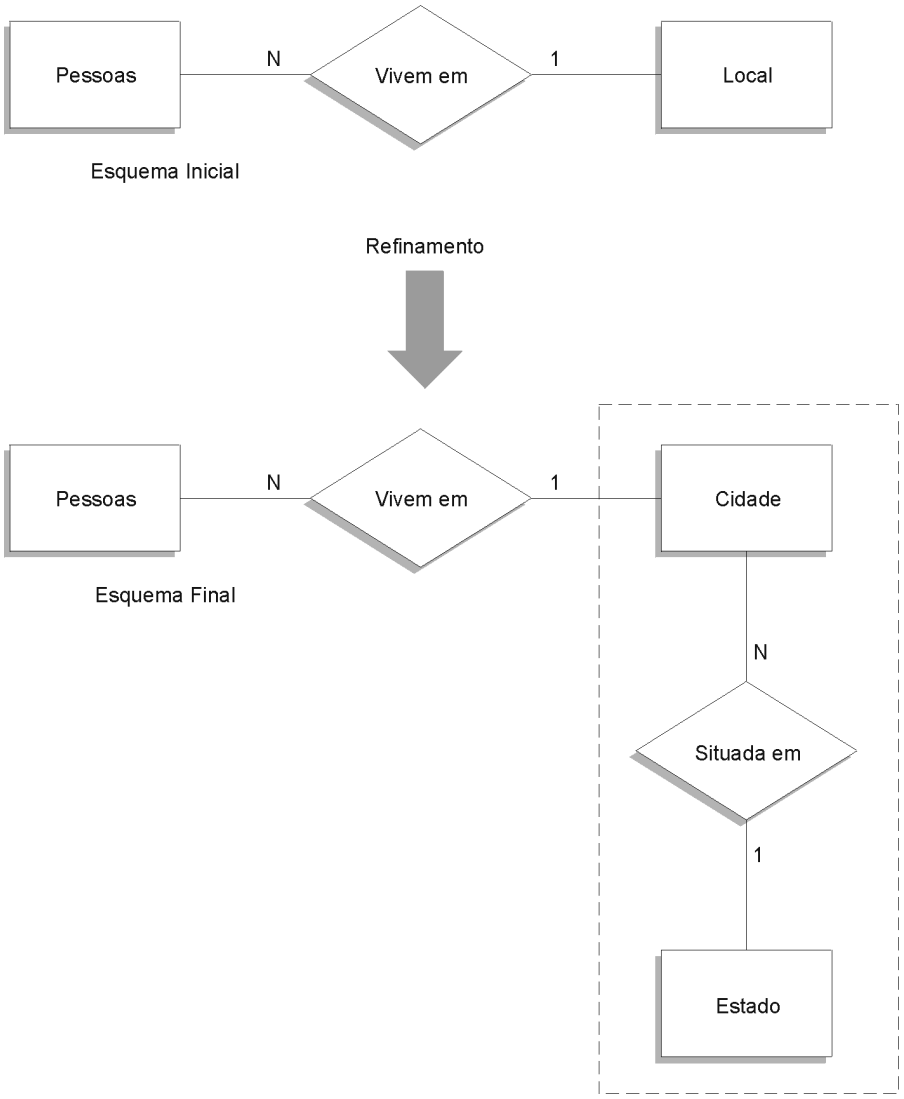


Exemplo 4

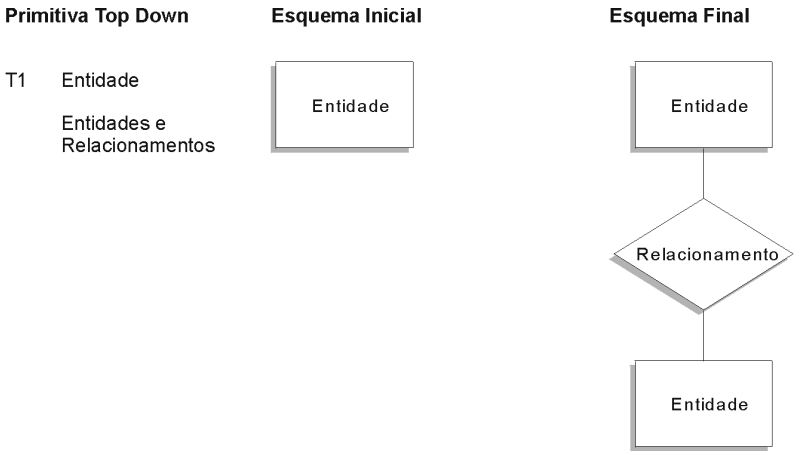
O relacionamento “Colação” relaciona pares de alunos. Cada par envolve dois alunos: um com o status de “Colador” e outro com o status de “Colado”. O relacionamento “Delação” envolve um aluno na condição de “Delator” e um par de alunos envolvidos na cola (Colador e Colado).

Transformações e Estratégias de Projeto

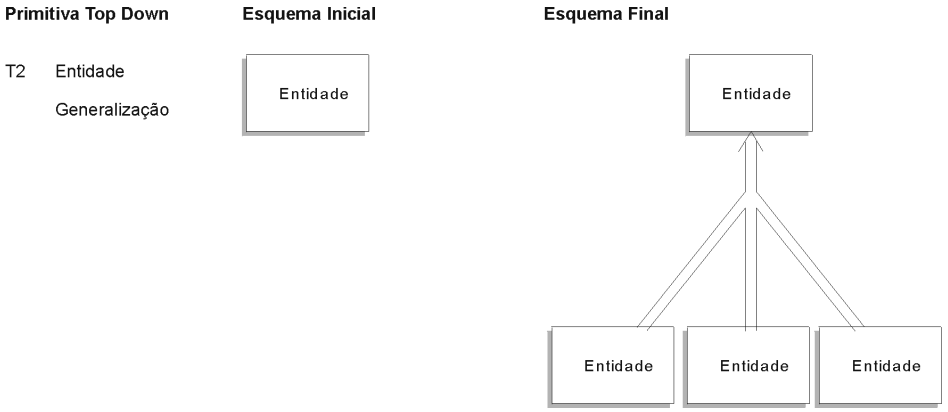
- Primitivas: top down e bottom up
- Estratégias: top down, bottom up e mista
- Metodolgias: guiam o projeto através de estratégias primitivas



Primitivas top down

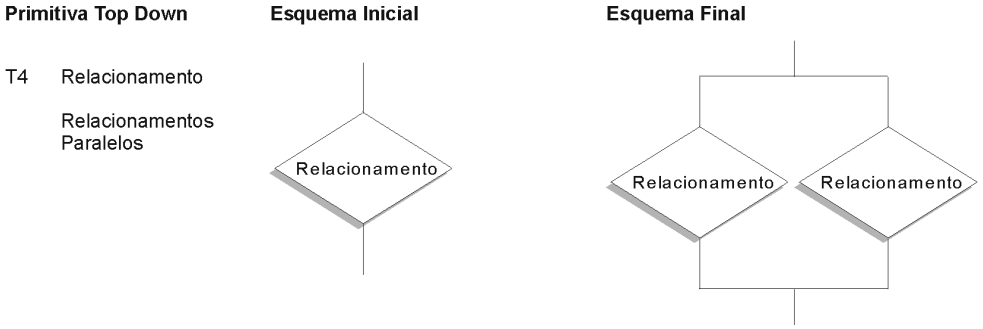


Primitiva Top Down T1

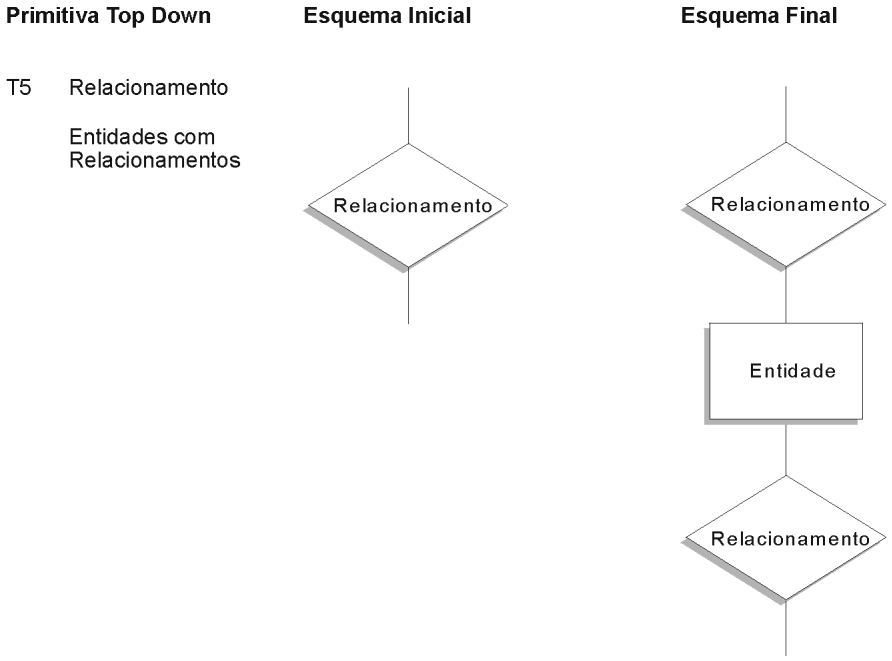


Primitiva Top Down T2

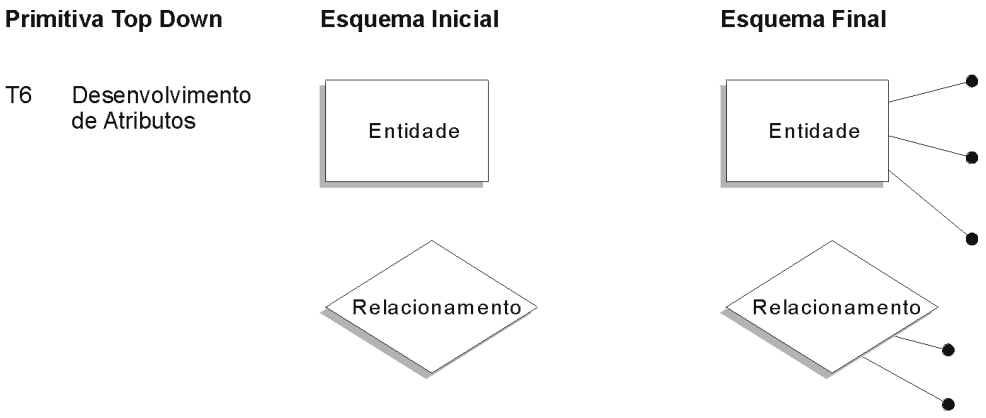
Primitiva Top Down T3



Primitiva Top Down T4

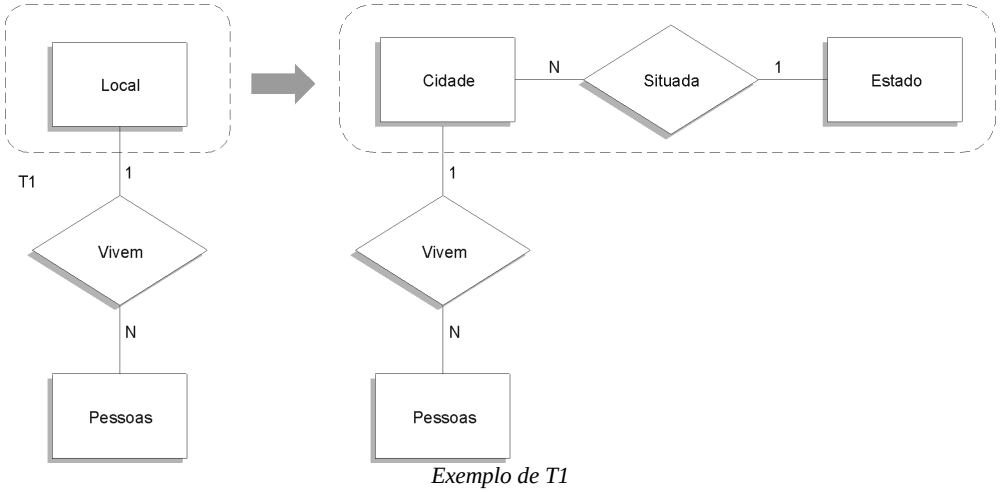


Primitiva Top Down T5

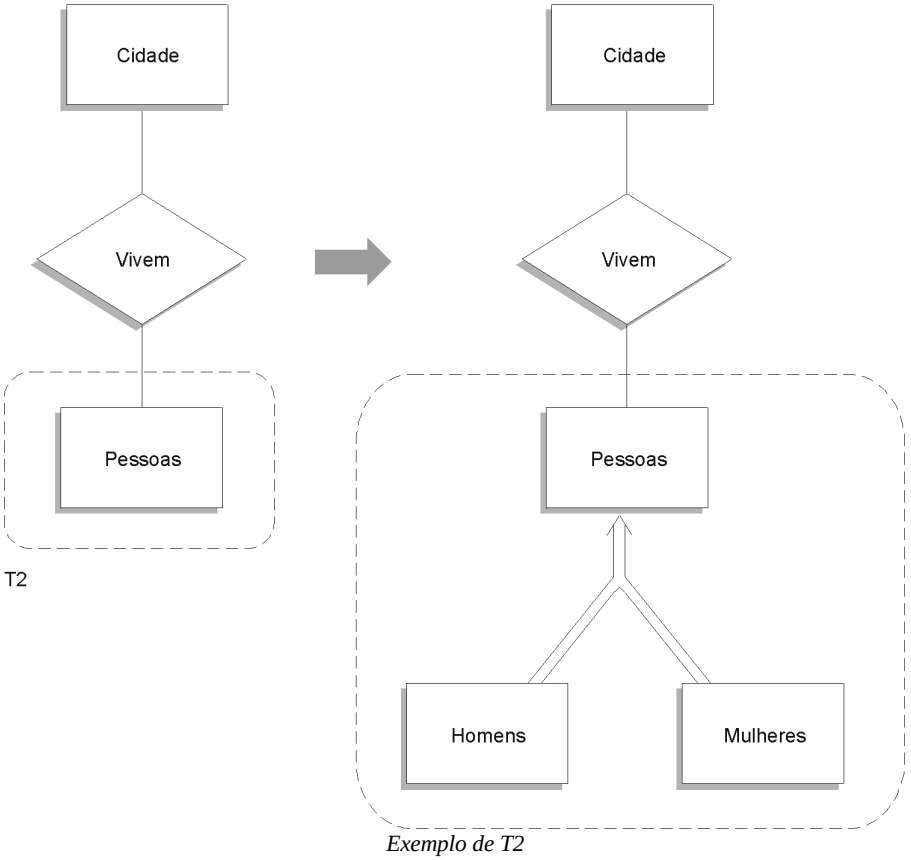


Primitiva Top Down T6

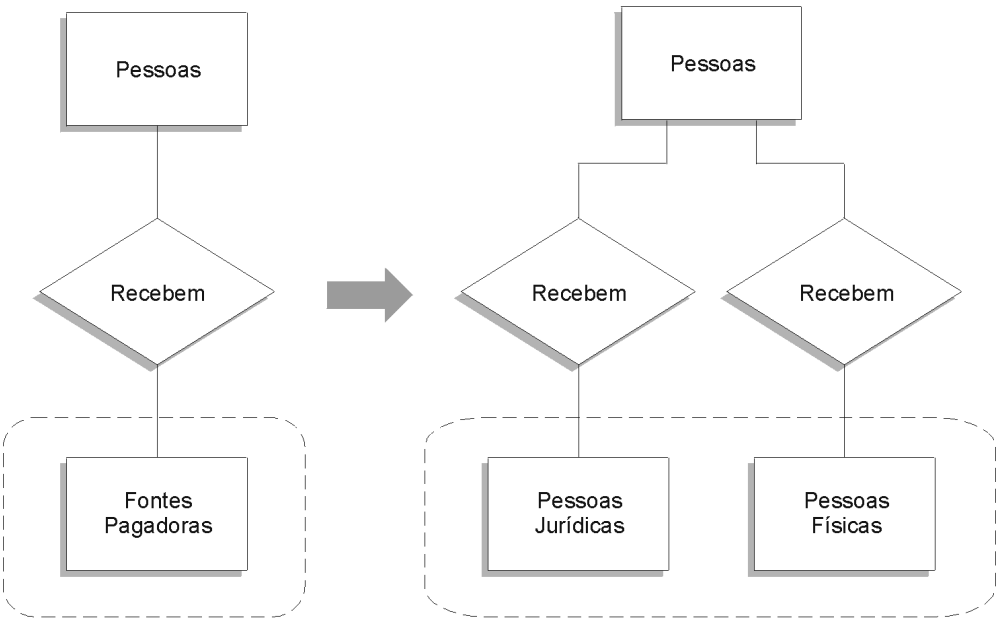
Exemplos de primitivas *top down:*



Exemplo de T1

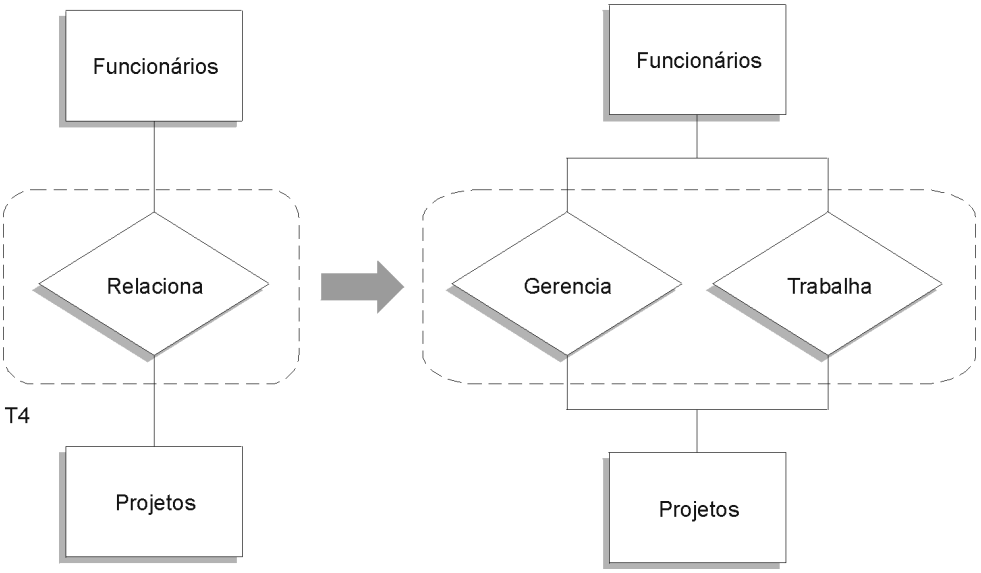


Exemplo de T2



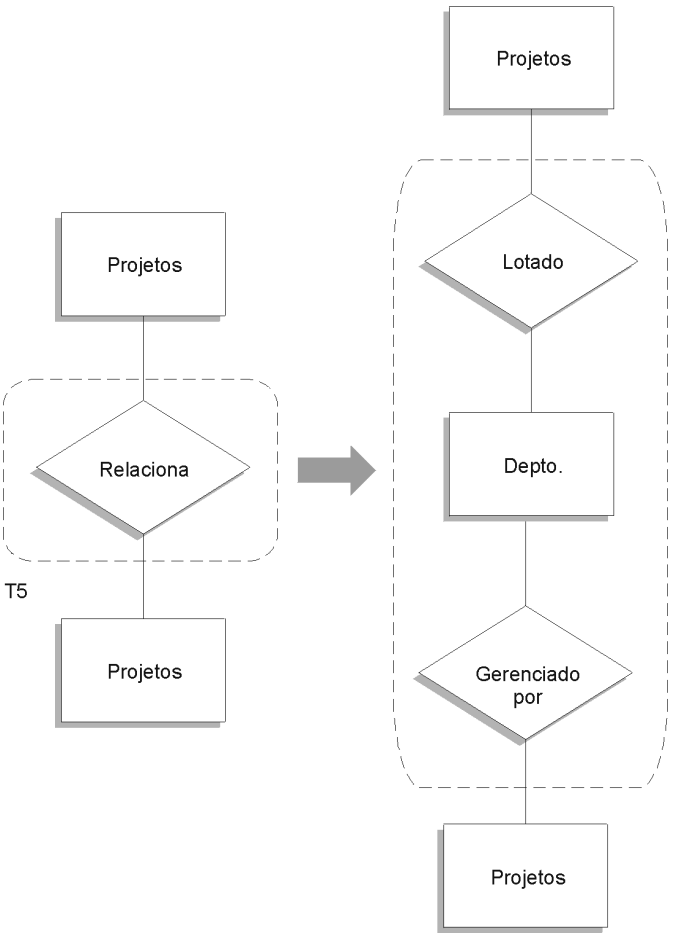
T3

Exemplo de T3

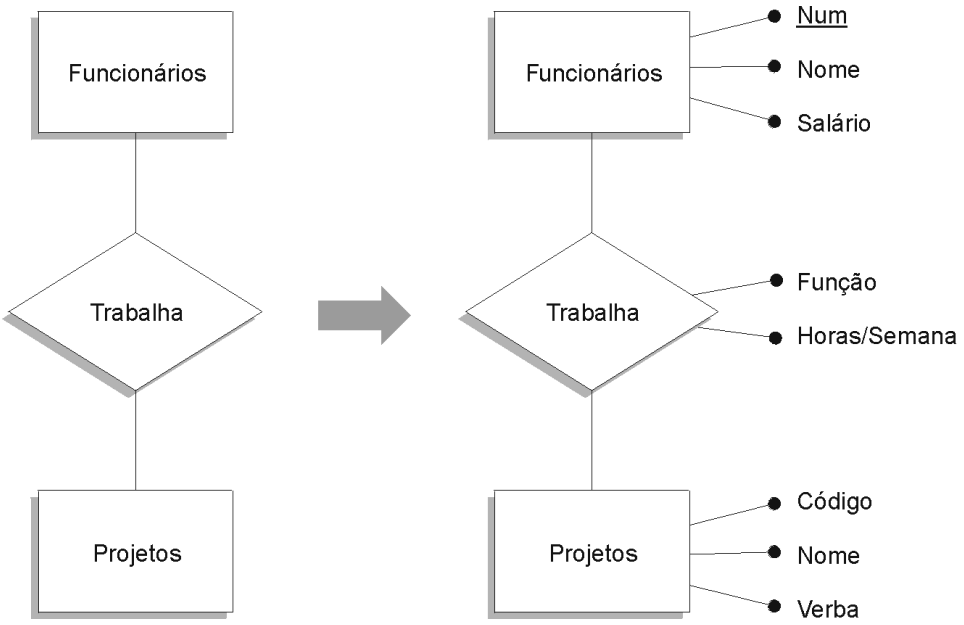


T4

Exemplo de T4

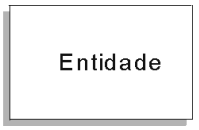


Exemplo de T5

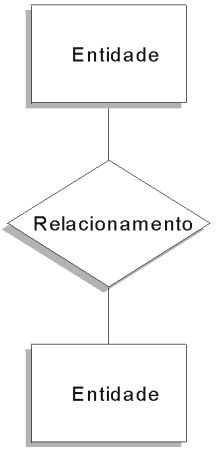


Exemplo de T6

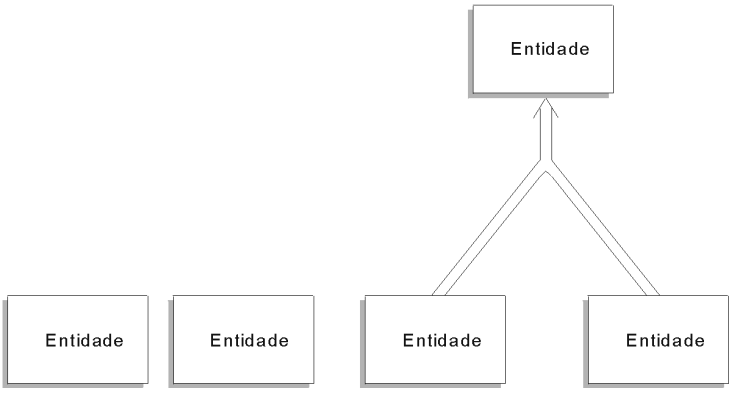
Primitivas bottom up

Primitiva Button Up	Esquema Inicial	Esquema Final
B1 Geração da Entidade		

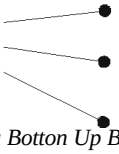
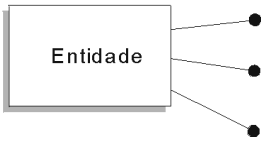
Primitiva Botton Up B1

Primitiva Button Up	Esquema Inicial	Esquema Final
B2 Geração do Relacionamento		

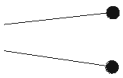
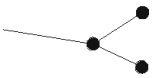
Primitiva Botton Up B2

Primitiva Button Up	Esquema Inicial	Esquema Final
B3 Generalização		

Primitiva Botton Up B3

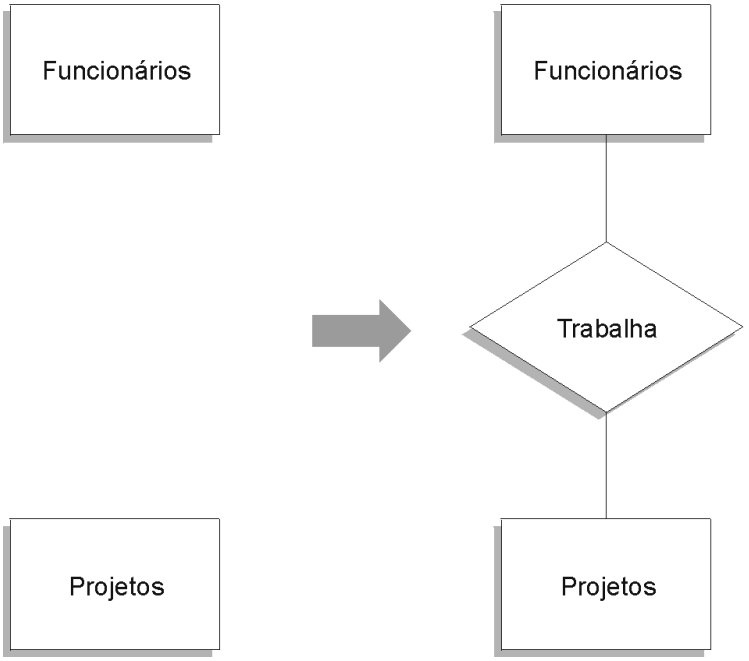
Primitiva Button Up		Esquema Inicial	Esquema Final
B4	Geração de Entidade a partir do atributo		

Primitiva Botton Up B4

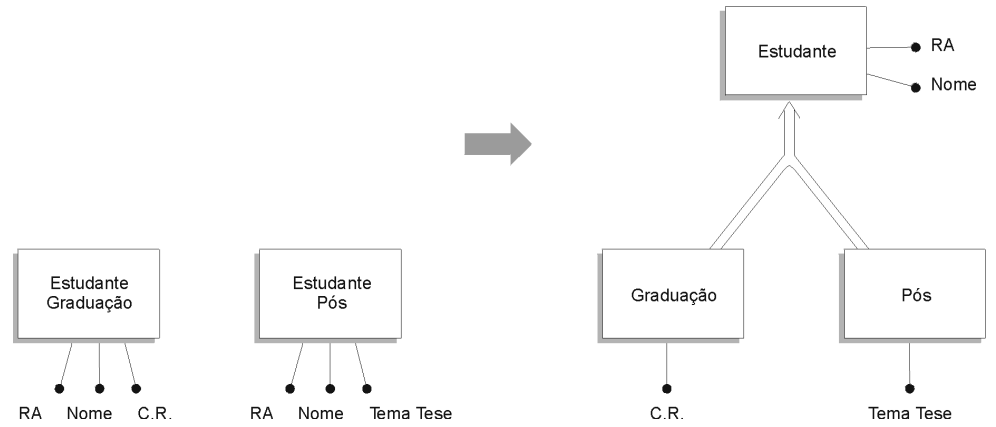
Primitiva Button Up		Esquema Inicial	Esquema Final
B5	Geração de atributo composto		

Primitiva Botton Up B5

Exemplo de primitivas *bottom up:*

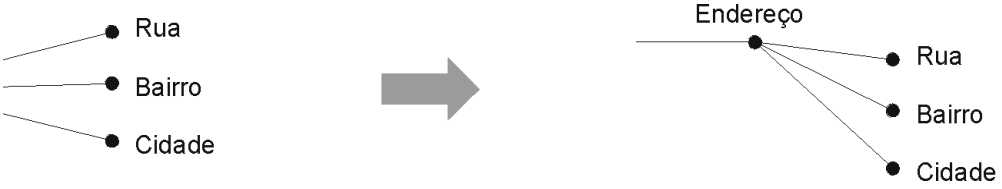


Exemplo de B2



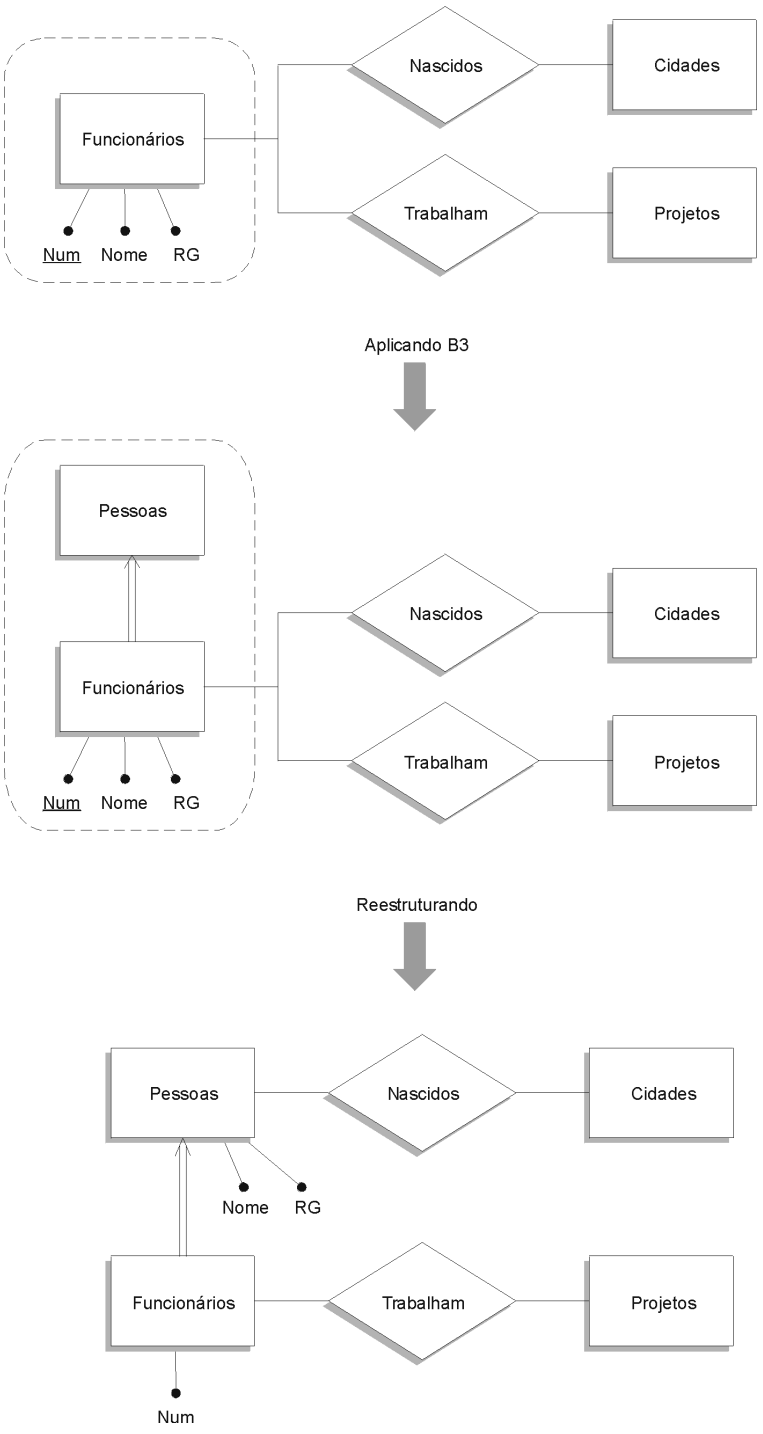
Exemplo de B3

Exemplo de B4



Exemplo de B5

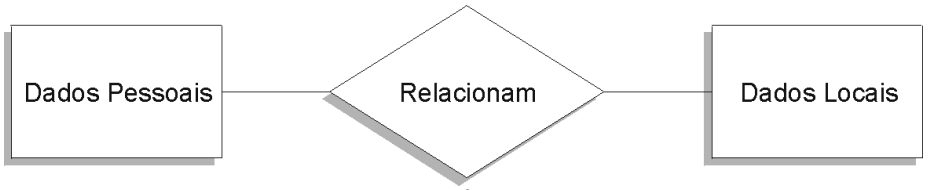
Necessidade de reestruturação



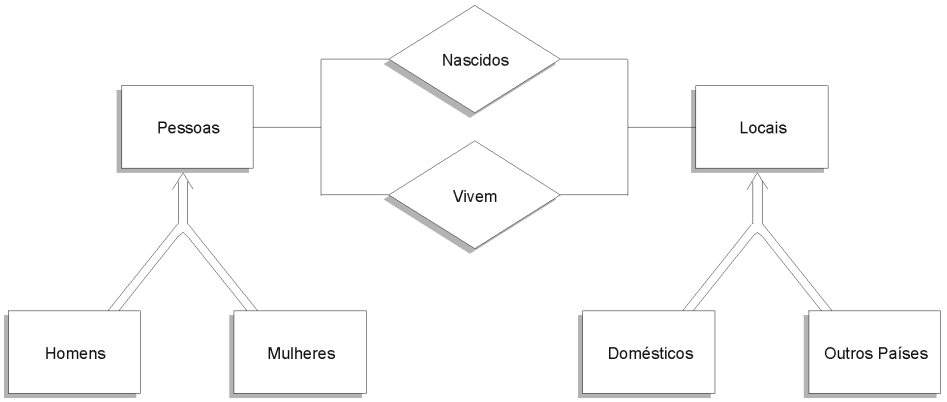
Estratégia top down

Exemplo de um projeto estritamente *top down*

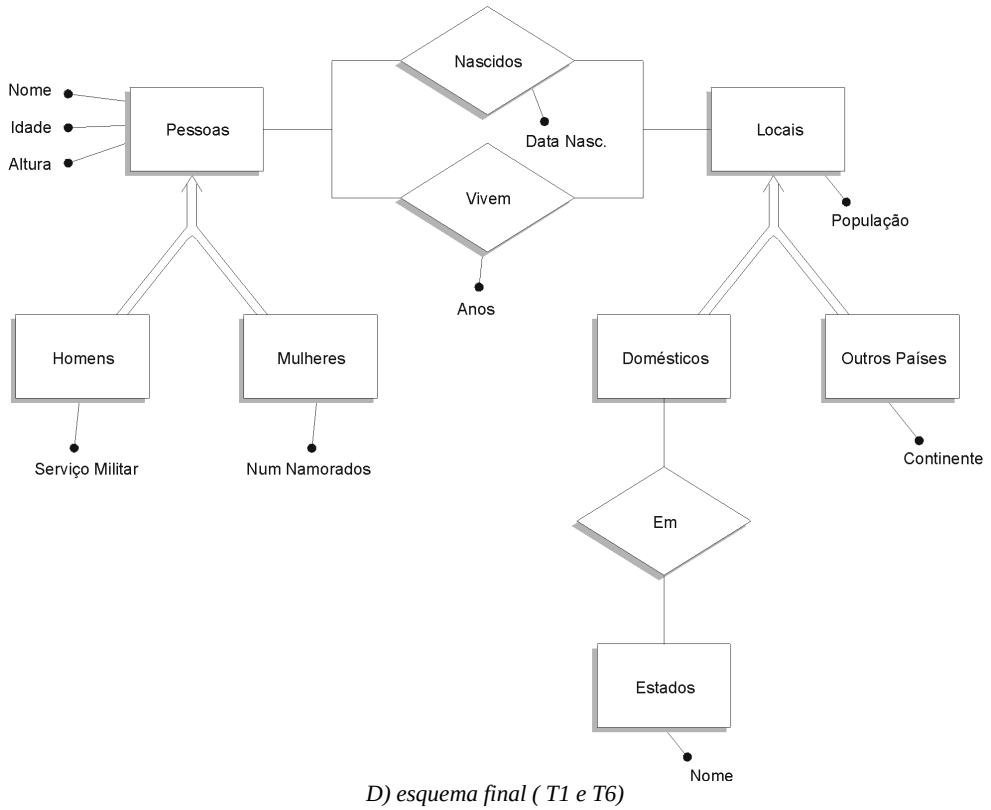
A) primeiro refinamento



B) segundo refinamento (T1)



C) terceiro refinamento (T2 e T6)



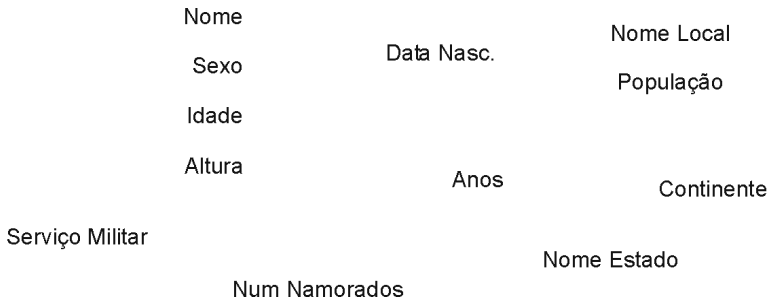
D) esquema final (T1 e T6)

Vantagens do top down: o projetista pode através de refinamentos independentes analisar um conceito a cada instante.

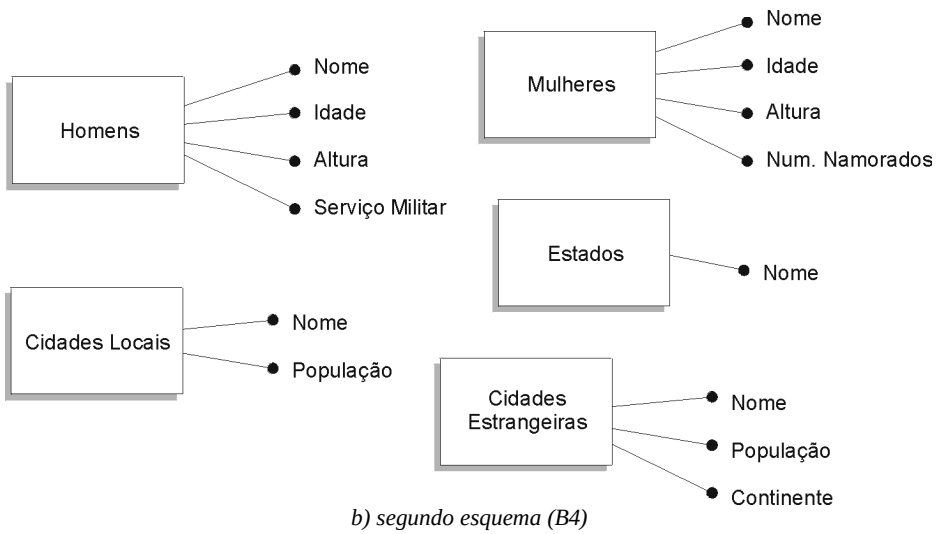
Desvantagem: nem sempre o projetista tem em mente a visão "*high-level*".

Estratégia *bottom up*

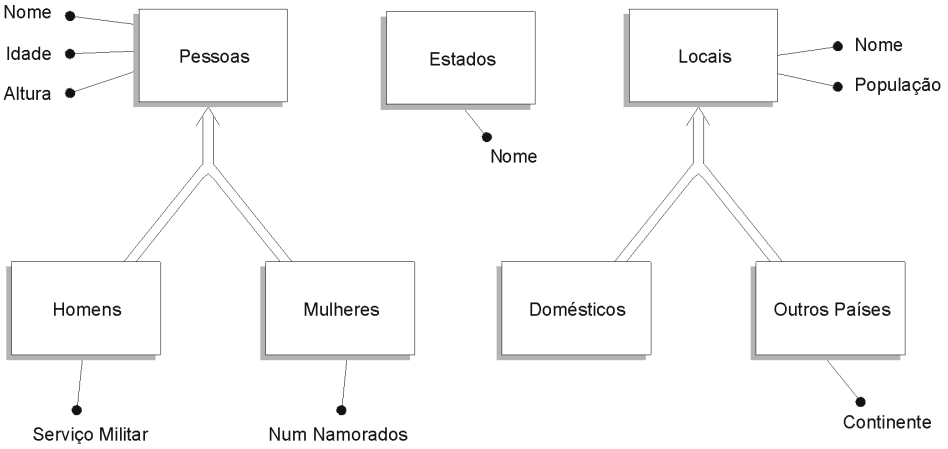
Exemplo de um projeto estritamente *bottom up:*



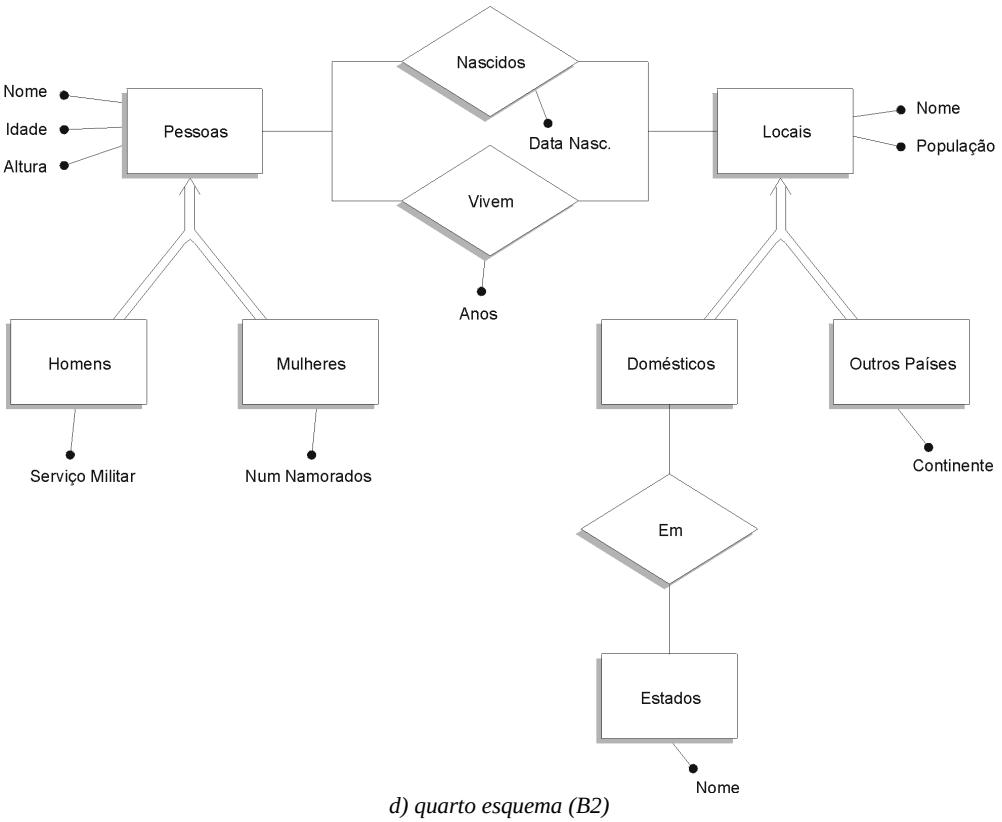
A) primeiro esquema



b) segundo esquema (B4)



C) terceiro esquema (B3)



d) quarto esquema (B2)

Comparações entre as metodologias bottom up e top down

Vantagens do top down

É conveniente em organizações altamente estruturadas onde o gerente tem uma visão completa do domínio da aplicação em alto nível de abstração.

Vantagens do bottom up

É conveniente em organizações não muito estruturadas onde é fácil discutir detalhes e depois agregá-las.

Dicas

- Dica 1: tentar conduzir uma sessão de projeto de forma top down na sua essência e excepcionalmente usar primitivas bottom up (para, por exemplo, quando o projetista esqueceu algum conceito no nível mais alto do refinamento).
- Dica 2: mesmo que o projetista tenha lançado mão de conceitos bottom up, tentar fazer a documentação como se ele fosse top down.

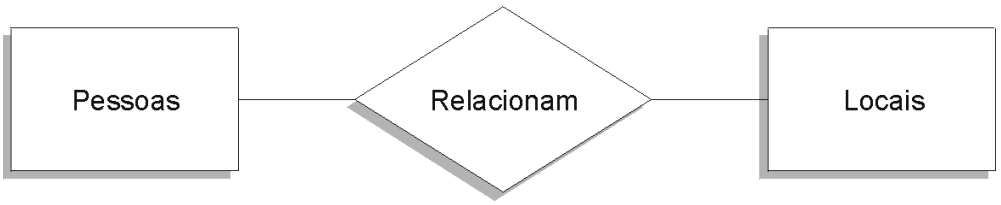
Estratégia mista

Envolve conceitos *top down e bottom up.*

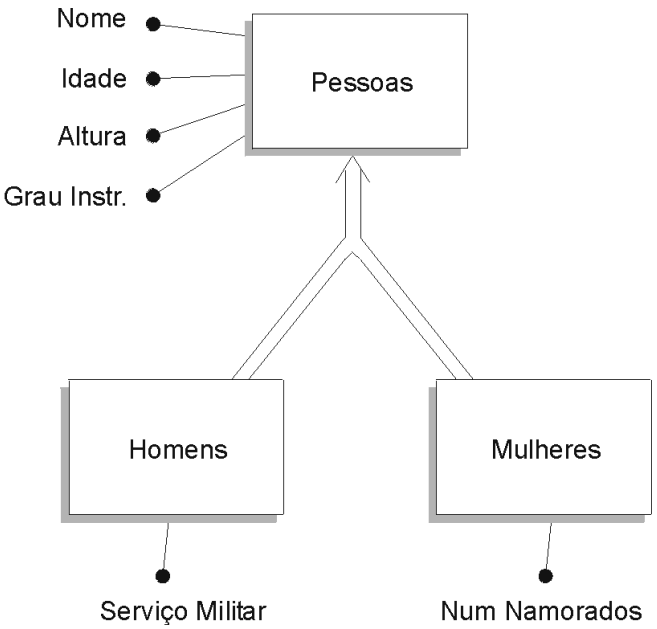
A estratégia mista é baseada no particionamento controlado dos requisitos.

O projetista produz um "*frame*" (ou esqueleto) para posterior integração . O "*frame*" contem os conceitos mais importantes da aplicação e os "*links*" entre as partições.

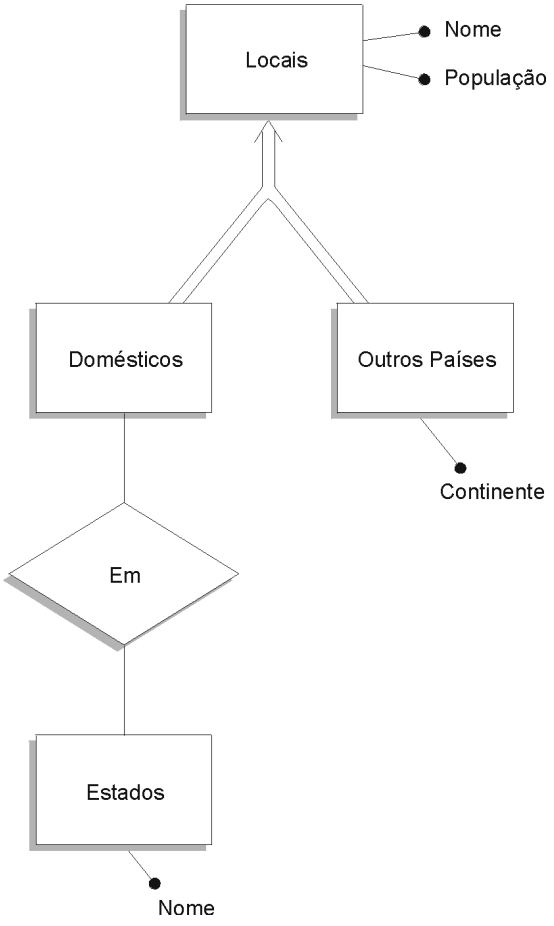
Estratégia mista - exemplo



A) esquema esqueleto



B) esquema pessoas

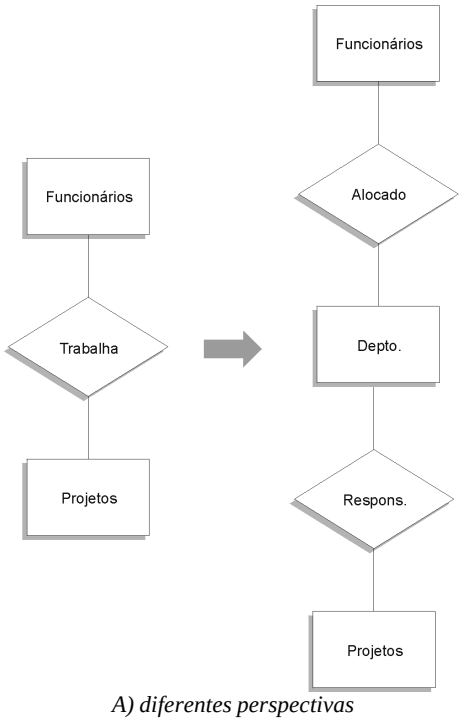


C) esquema local

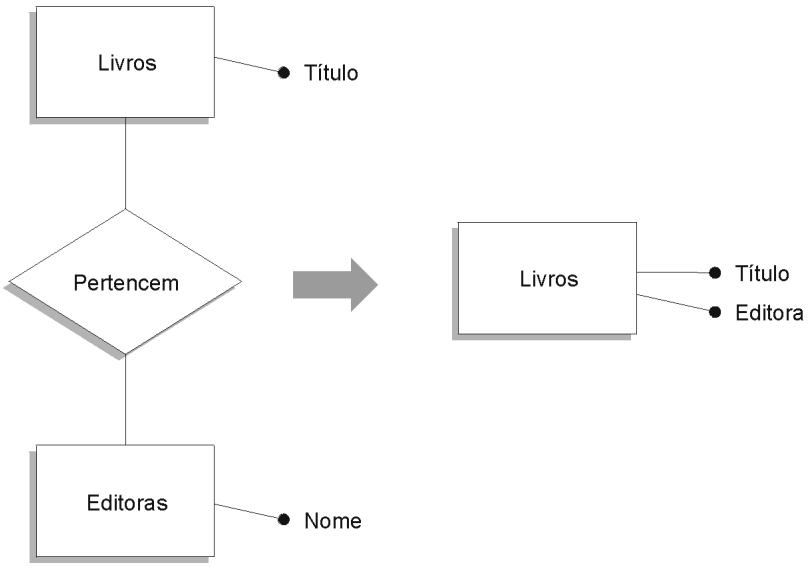
Integração de visões

É necessária quando a aplicação foi "quebrada" em partes e necessita de integração para gerar o esquema final a partir dos diferentes "*views*" . Também é necesária quando as visões partiram de diferentes projetistas ou a partir de bases de dados diferentes.

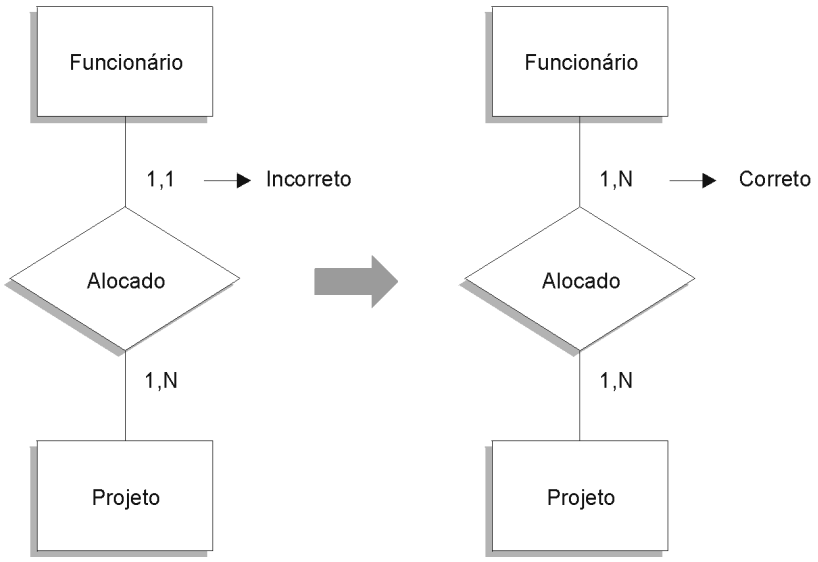
Problemas que influenciam a atividade de integração



A) diferentes perspectivas

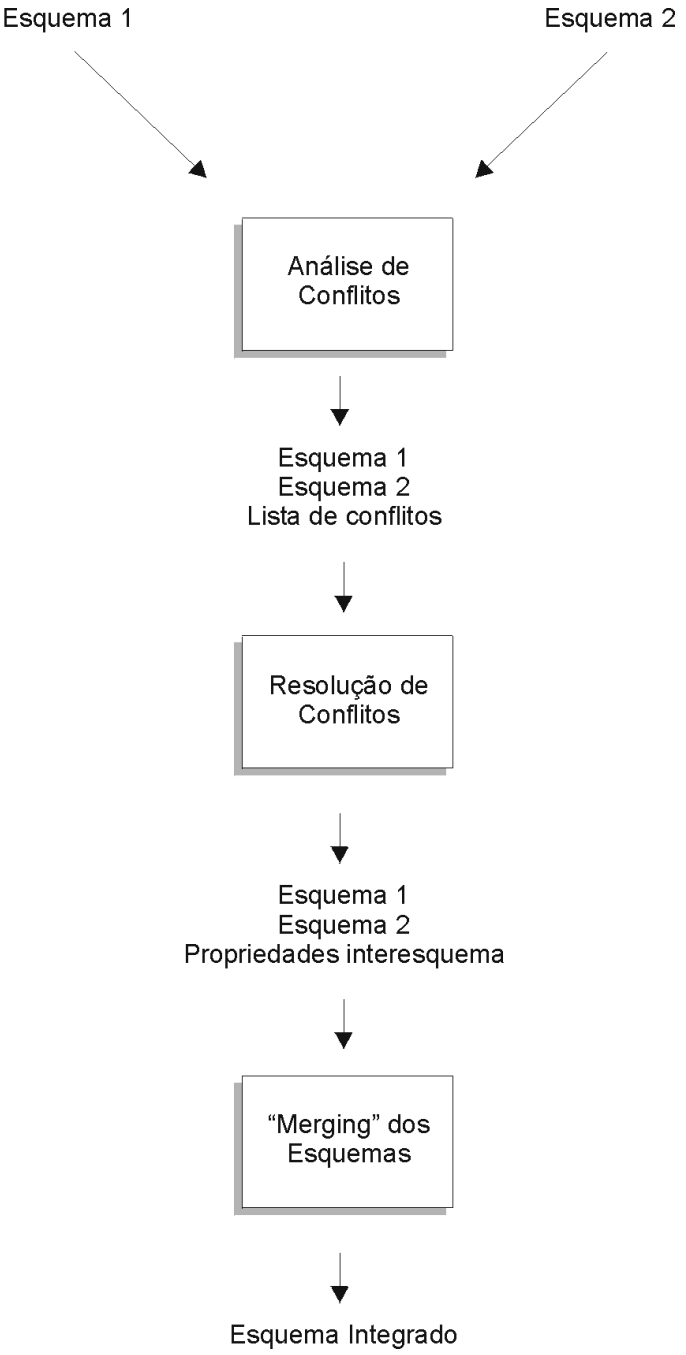


B) construções equivalentes

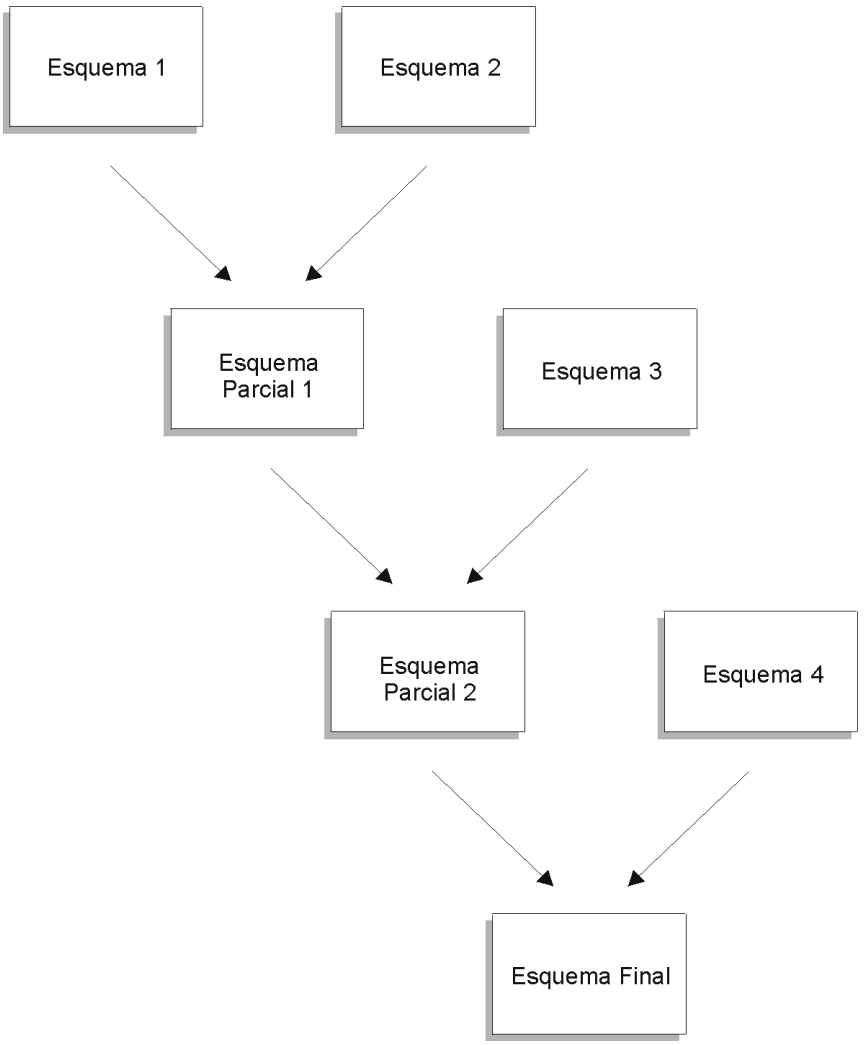


c) especificação incorreta

Procedimentos para integração das visões

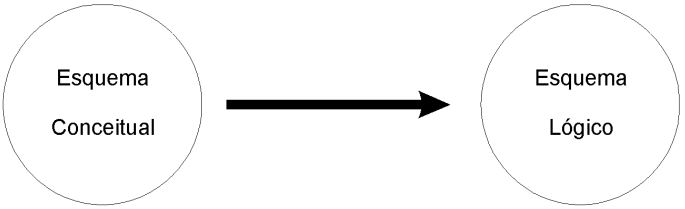


Quando mais que dois esquemas devem ser integrados a "dica" é fazê-lo dois a dois como mostra a figura abaixo.

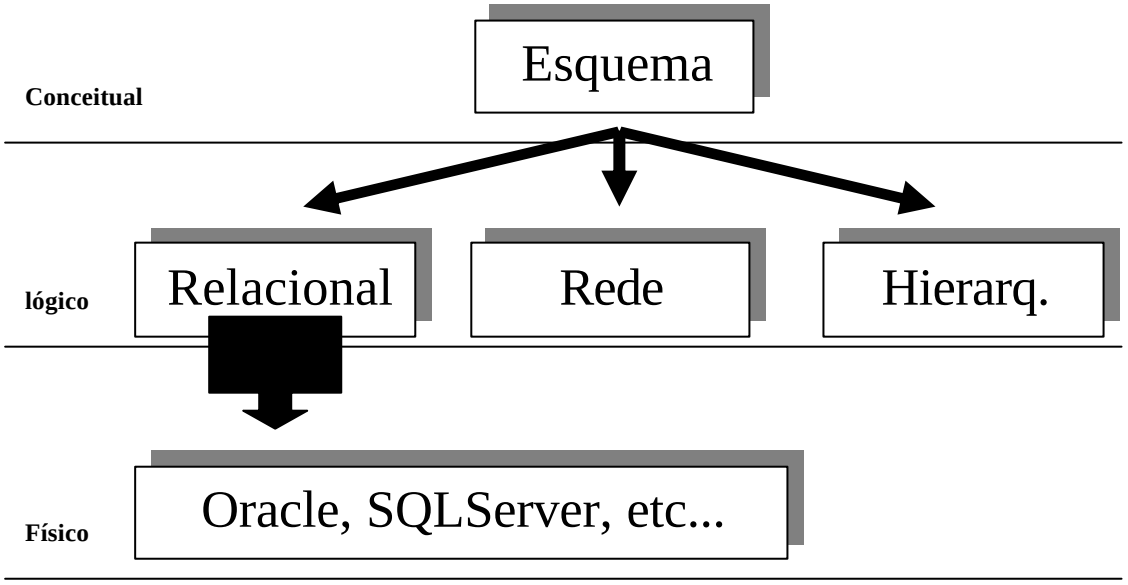

Esquema 1
Esquema 2
Esquema Parcial 1
Esquema 3
Esquema Parcial 2
Esquema 4
Esquema Final

Mapeamento para o Modelo Relacional

A obtenção de um modelo lógico a partir de um modelo conceitual pode ser feita pela aplicação de um conjunto de regras bem definidas. Essas regras basicamente atuarão em dois grupos distintos de elementos: as estruturas de relacionamento, agregação e especialização de um lado e as entidades e seus atributos de outro.



Mapeamento do Modelo Relacional

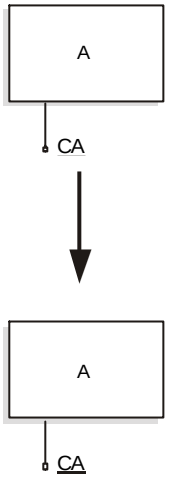


Essas regras são apresentadas mais abaixo :

Entidades

Entidades

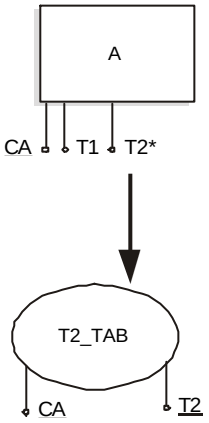
As entidades irão gerar sempre uma tabela no Modelo Relacional.



Entidades - Tabelas

Atributos Multivalorados

Os atributos Multivalorados criarão uma tabela auxiliar, que receberá os atributos determinantes (chave) da tabela principal e o próprio atributo se tornará um atributo determinante nessa nova tabela.

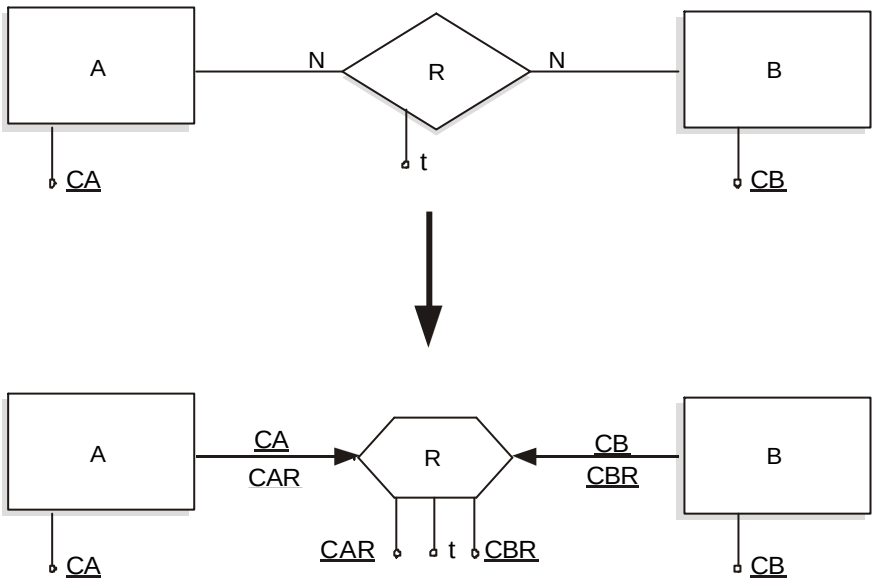


Atributos Multivalorados

Relacionamentos

Relação N:N

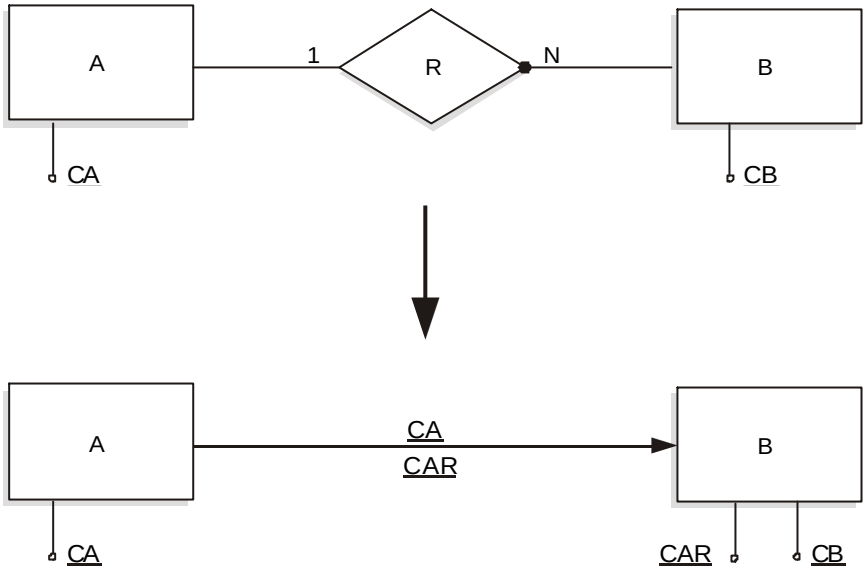
Nos relacionamentos N:N, cada entidade e o relacionamento virarão tabelas.



Relacionamento N:N

Relação 1:N ou N:1 com obrigatoriedade

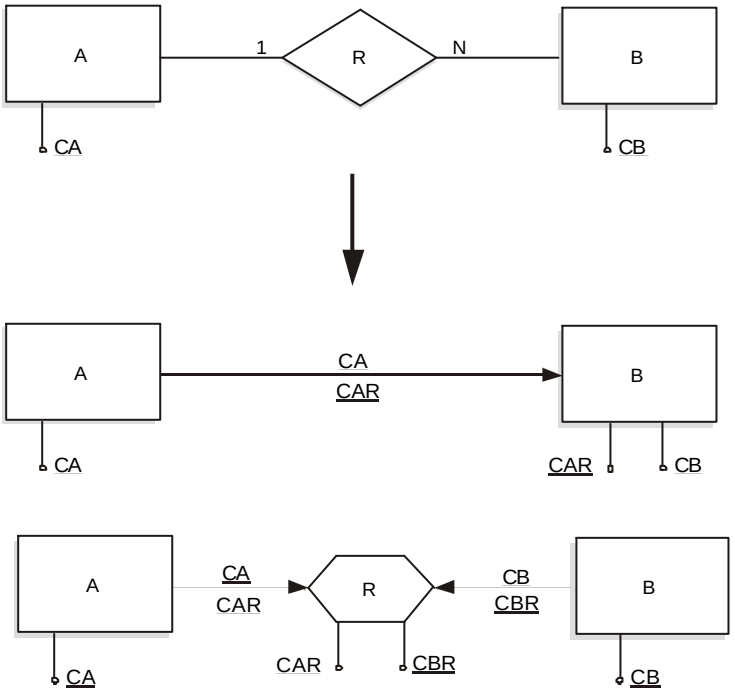
Para esses relacionamentos a entidade fraca, ou seja aquela entidade onde a obrigatoriedade se encontra, irá "vencer" e receberá o atributo determinante (chave).



Relacionamento 1:N com obrigatoriedade

Relação 1:N ou N:1 sem obrigatoriedade

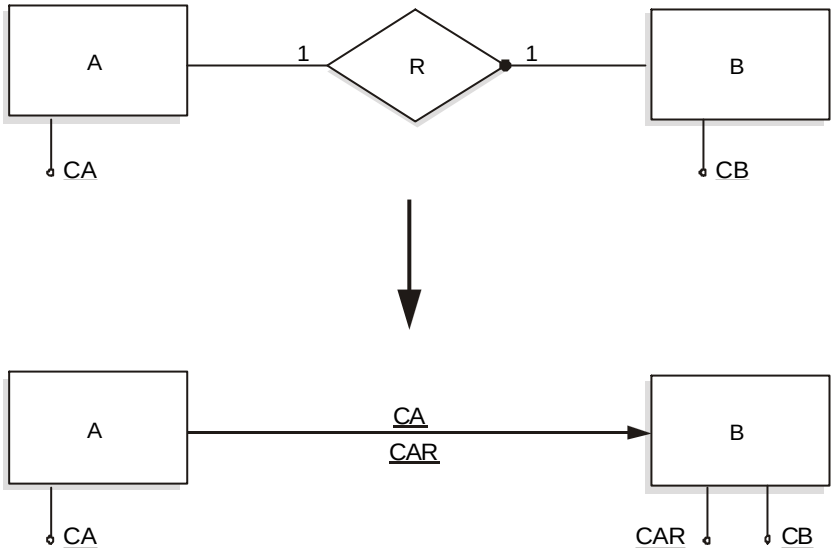
Para esses relacionamento podemos ter duas possibilidades, dependendo do modelamento que está sendo feito. Podemos ter a entidade recebendo o atributo determinante (chave) ou o relacionamento se tornando mais uma tabela.



Relacionamento 1:N sem obrigatoriedade

Relação 1:1 com obrigatoriedade

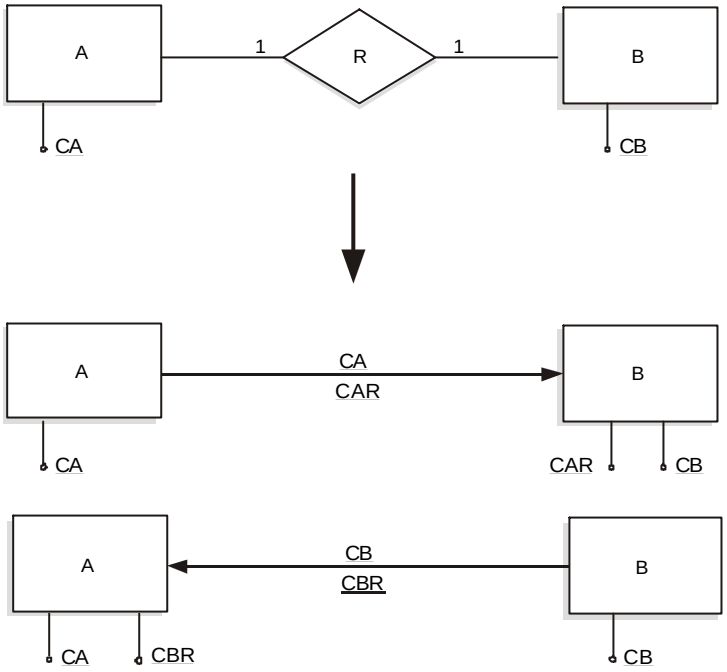
Assim como nos relacionamento 1:N com obrigatoriedade, nesses relacionamentos a entidade fraca, ou seja aquela entidade onde a obrigatoriedade se encontra, irá "vencer" e receberá o atributo determinante (chave).



Relacionamento 1:1 com obrigatoriedade

Relação 1:1 sem obrigatoriedade

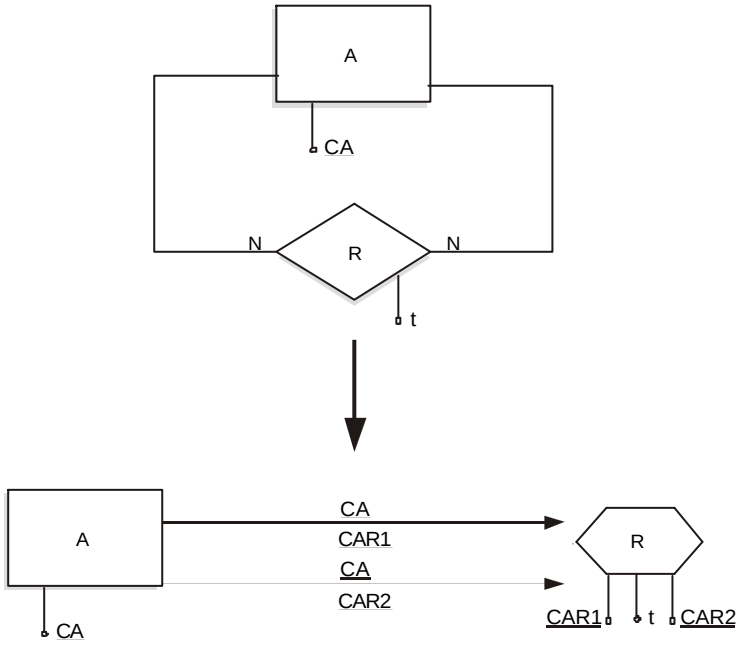
Nesses relacionamentos o próprio mundo que está sendo modelado deverá identificar qual das duas entidades deverão receber o atributo determinante (chave).



Relacionamento 1:1 sem obrigatoriedade

Autorelacionamento N:N

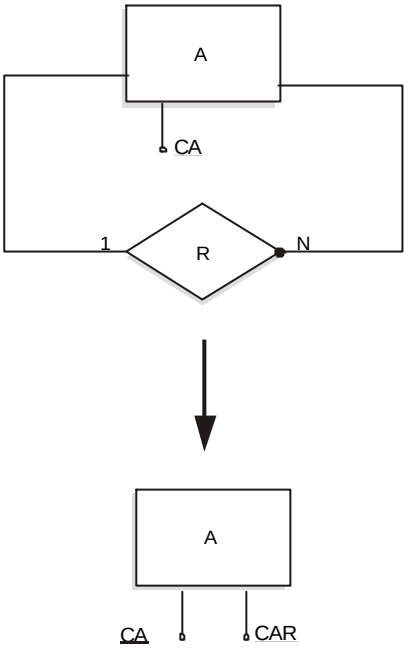
Nesses relacionamentos a entidade e o relacionamento gerarão duas tabelas, o relacionamento por sua vez receberá dois atributos determinantes da entidade.



Autorelacionamento N:N

Autorelacionamento 1:N ou N:1 com obrigatoriedade

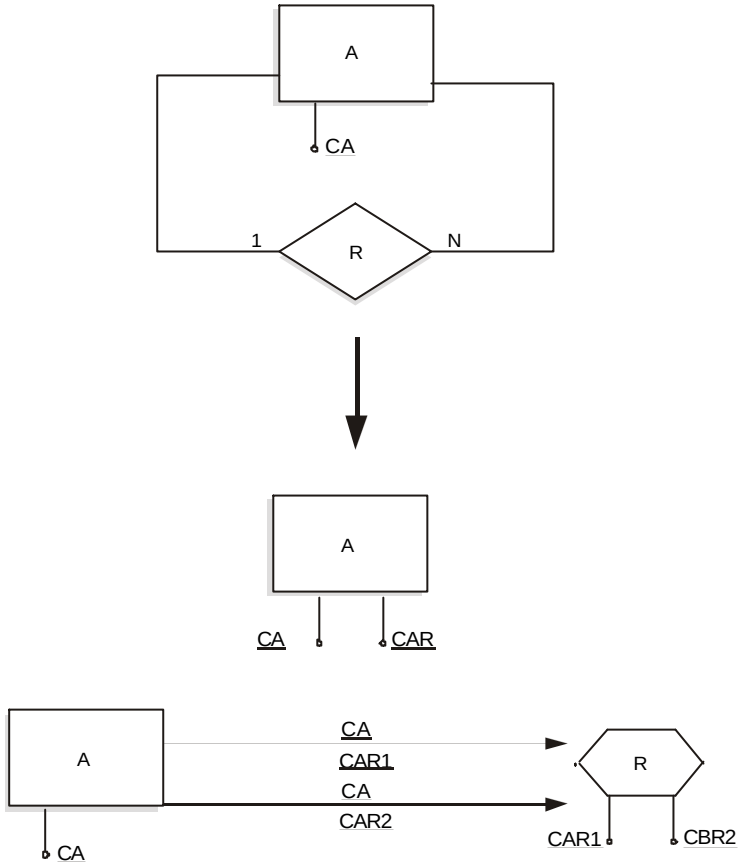
Nesses relacionamentos 1:N com obrigatoriedade, a entidade é fraca, e sendo assim irá receber o atributo determinante (chave).



Autorelacionamento 1:N ou N:1 com obrigatoriedade

Autorelacionamento 1:N ou N:1 sem obrigatoriedade

Para esses relacionamento podemos ter duas possibilidades, dependendo do modelamento que está sendo feito. Podemos ter a entidade recebendo o atributo determinante (chave) ou o relacionamento se tornando mais uma tabela.

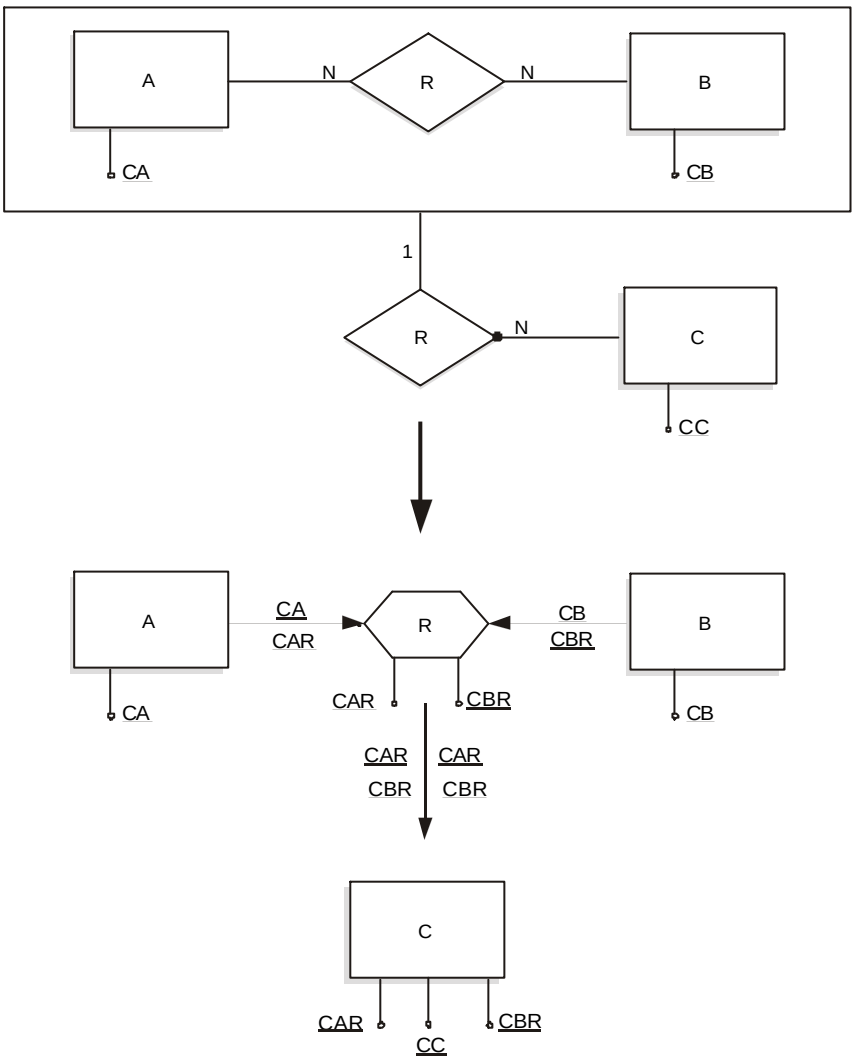


Autorelacionamento 1:N ou N:1 sem obrigatoriedade

Agregação

O relacionamento N:N é resolvido da forma já vista anteriormente. A agregação nada mais é do que o relacionamento entre relacionamentos, desta forma a relação com a entidade C vai acontecer conforme as regras mostradas anteriormente (considerando que o relacionamento A:B gere uma "entidade" agregada.

Agregação



Relacionamento Triplo

Nesses relacionamentos as regras de atribuição do atributo determinante vai depender do modelamento, além se seguir todas as regras determinadas anteriormente. O importante é notar que todas as entidades estão relacionadas ao mesmo tempo.

Relacionamento Triplo