

Módulo II

Mapeamento OO x Relacional

Prof. Ismael H F Santos

Ementa

- Modulo II – Mapeamento OO x Relacional
 - Mapeamento OO x Relacional

POO-Java

Mapeamento
OO-Relacional



April 05

Prof. Ismael H. F. Santos - ismael@tecgraf.puc-rio.br

3

Introdução

- Relevância do mapeamento de objetos para o modelo relacional:
 - A tecnologia OO como forma usual de desenvolver sistemas de software.
 - Sem dúvida os SGBDR dominam o mercado comercial.
- Princípios teóricos bastante diferentes.
- A tecnologia OO:
 - Objetos são abstrações de comportamento.
 - objetos: dados + funções.
- Tecnologia relacional:
 - lida com o armazenamento de dados tabulares.

April 05

Prof. Ismael H. F. Santos - ismael@tecgraf.puc-rio.br

4

Introdução

- Uma das primeiras atividades do projeto detalhado é o desenvolvimento do banco de dados a ser utilizado, se este não existir.
- Atividade: **projeto do banco de dados**.
 - Construção do esquema do banco de dados
 - Criação de índices
 - Armazenamento físico dos dados
 - Definição de visões sobre os dados armazenados.
 - Atribuição de direitos de acesso
 - Políticas de backup dos dados

Introdução

- Escopo: apenas os aspectos de mapeamento de informações entre as tecnologias de OO e relacional.
 - mapeamento do modelo de classes para o modelo relacional.
- Nota: ferramentas CASE e mapeamento automático; engenharia reversa.
- Nem sempre uma ferramenta CASE está disponível.
- Mesmo na existência de uma ferramenta, é importante um conhecimento básico dos procedimentos do mapeamento.

Mapeamento Objeto-Relacional

- DDL

- Entidades
 - Tabelas: objetos
 - *Schema*: classe
 - Linha: instância
 - Coluna: propriedade
- Relacionamentos
 - Chaves: referências

Mapeamento Objeto-Relacional

- DML

- *Query, SP, Trigger*: métodos
- Dataset: coleção de objetos
- Cursor: objeto iterador

Conceitos do modelo de dados relacional

- Fundamentado no conceito de *Relação*.
- Cada coluna de uma relação pode conter apenas *valores atômicos*.
- Uma *chave primária*: colunas cujos valores podem ser utilizados para identificar unicamente cada linha de uma relação.
- Associações entre linhas: valores de uma coluna fazem referência a valores de uma outra coluna (*chave estrangeira*).

April 05

Prof. Ismael H. F. Santos - ismael@tecgraf.puc-rio.br

9

Conceitos do modelo de dados relacional

- Uma chave estrangeira também pode conter *valores nulos*.
 - representado pela constante **NULL**.
- O NULL é normalmente é usado para indicar que um valor não se aplica, ou é desconhecido, ou não existe.



April 05

Prof. Ismael H. F. Santos - ismael@tecgraf.puc-rio.br

10

Conceitos do modelo de dados relacional

Departamento			
<u>id</u>	sigla	nome	idGerente
13	RH	Recursos Humanos	5
14	INF	Informática	2
15	RF	Recursos Financeiros	6

Alocação		
<u>id</u>	idProjeto	idEmpregado
100	1	1
101	1	2
102	2	1
103	3	5
104	4	2

Projeto		
<u>id</u>	nome	verba
1	PNADO	R\$ 7.000
2	BMMO	R\$ 3.000
3	SGILM	R\$ 6.000
4	ACME	R\$ 8.000

April 05

Prof. Ismael H. F. Santos - ismael@tecgraf.puc-rio.br

11

Conceitos do modelo de dados relacional

Empregado						
<u>id</u>	matrícula	CPF	nome	endereço	CEP	<u>idDepartamento</u>
1	10223	038488847-89	Carlos	Rua 24 de Maio,40	22740-002	13
2	10490	024488847-67	Marcelo	Rua do Bispo, 1000	22733-000	13
3	10377	NULL	Adelci	Av. Rio Branco, 09	NULL	NULL
4	11057	0345868378-20	Roberto	Av. Apicás, 50	NULL	14
5	10922	NULL	Aline	R. Uruguaiana, 50	NULL	14
6	11345	0254647888-67	Marcelo	NULL	NULL	15

April 05

Prof. Ismael H. F. Santos - ismael@tecgraf.puc-rio.br

12

Objetos Transientes x Persistentes

- Os objetos de um sistema podem ser classificados em persistentes e transientes.
- **Objetos transientes**: existem somente na memória principal.
 - Objetos de controle e objetos de fronteira.
- **Objetos persistentes**: têm uma existência que perdura durante várias execuções do sistema.
 - Precisam ser armazenados quando uma execução termina, e restaurados quando uma outra execução é iniciada.
 - Tipicamente objetos de entidade.

Persistência em Java

- Alternativas de Persistência em Java
 - Serialização
 - Bases de Dados Relacionais
 - JDBC – Java Database Connectivity
 - ANSI SQLJ
 - Bases de Objetos
 - PSE – Persistent Storage Engines
 - ODBMS – Bases de Dados de Objetos
 - Componentes Transacionais Distribuídos
 - EJB – Enterprise JavaBeans

Mapeamento de objetos para o modelo relacional

- Utilização de um SGBDR: necessidade do mapeamento dos valores de atributos de objetos persistentes para tabelas.
- É a partir do modelo de classes que o mapeamento de objetos para o modelo relacional é realizado.
 - Semelhante ao de mapeamento do MER.
 - Diferenças em virtude de o modelo de classes possuir mais recursos de representação que o MER.

Mapeamento de objetos para o modelo relacional

- **Importante:** o MER e o modelo de classes não são equivalentes.
 - Esses modelos são freqüentemente confundidos.
 - O MER é um modelo de dados, enquanto que o modelo de classes modela objetos (dados e comportamento).
- **Notação (simplificada):**
 - Cada relação é representada através do seu nome e dos nomes de suas colunas entre parênteses.
 - Chaves primárias são sublinhadas
 - Chaves estrangeiras são tracejadas.

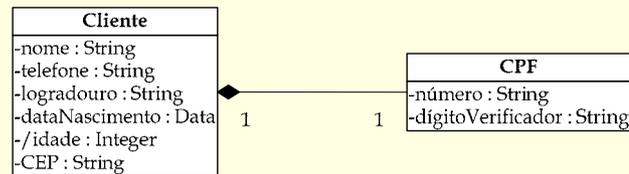
Mapeamento de objetos para o modelo relacional

- Exemplos a seguir utilizam sempre uma **coluna de implementação** como chave primária de cada relação.
- Uma coluna de implementação é um identificador sem significado no domínio de negócio.
- Essa abordagem é utilizada para:
 - manter uma padronização nos exemplos
 - e por ser uma das melhores maneiras de associar identificadores a objetos mapeados para tabelas.

Mapeamento: Classes e seus atributos

- Classes são mapeadas para relações.
 - Caso mais simples: mapear cada classe como uma relação, e cada atributo como uma coluna.
 - No entanto, pode não haver correspondência unívoca entre classes e relações..
- Para atributos o que vale de forma geral é que um atributo será mapeado para uma ou mais colunas.
- Nem todos os atributos de uma classe são persistentes (atributos derivados).

Mapeamento: Classes e seus atributos

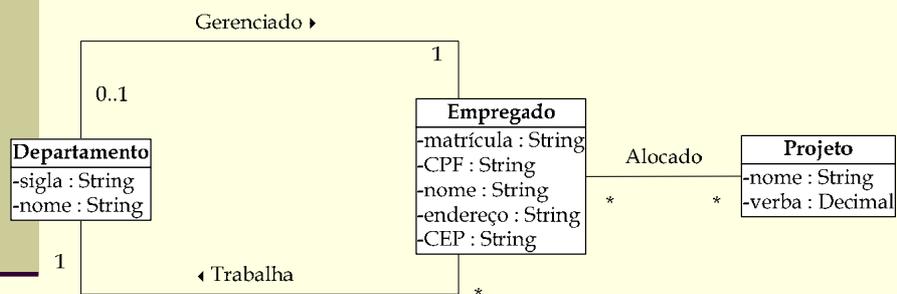


```
Cliente(id, CPF, nome, telefone, logradouro, dataNascimento, idCEP)
CEP(id, número, sufixo)
Cliente(id, nome, telefone, logradouro, dataNascimento, CPF, CEP)
```

Mapeamento: Associações

- O procedimento utiliza o conceito de **chave estrangeira**.
- Há três casos, cada um correspondente a um tipo de **conectividade**.
- Nos exemplos a seguir, considere, sem perda de generalidade, que:
 - há uma associação entre objetos de duas classes, Ca e Cb.
 - estas duas classes foram mapeadas para duas relações separadas, Ta e Tb.

Mapeamento: Associações



April 05

Prof. Ismael H. F. Santos - ismael@tecgraf.puc-rio.br

21

Mapeamento: Associações 1:1

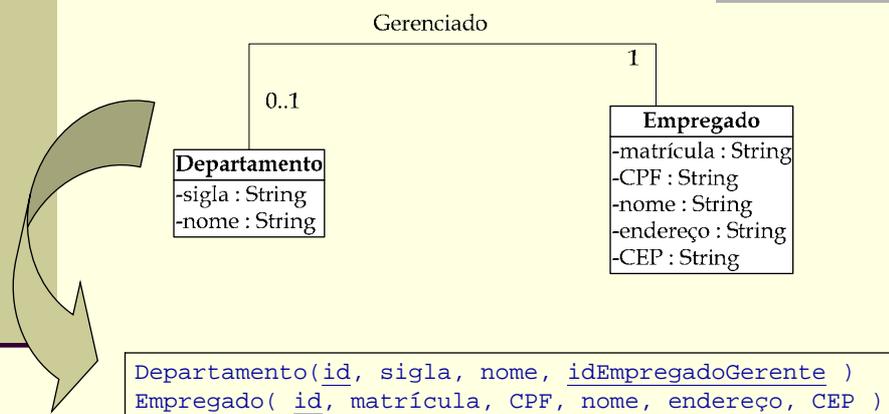
- Deve-se adicionar uma chave estrangeira em uma das duas relações para referenciar a chave primária da outra relação.
- Escolha da relação na qual a chave estrangeira deve ser adicionada com base na **participação**.
- Há três possibilidades:
 - Obrigatória em ambos os extremos.
 - Opcional em ambos os extremos.
 - Obrigatória em um extremo e opcional no outro extremo.

April 05

Prof. Ismael H. F. Santos - ismael@tecgraf.puc-rio.br

22

Mapeamento: Associações 1-1



April 05

Prof. Ismael H. F. Santos - ismael@tecgraf.puc-rio.br

23

Mapeamento: Associações 1-muitos

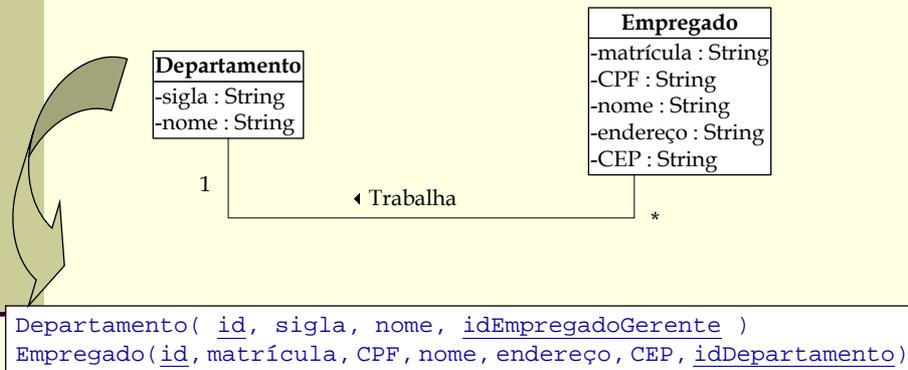
- Seja Ca a classe na qual cada objeto se associa com muitos objetos da classe Cb.
- Sejam Ta e Tb as relações resultantes do mapeamento de Ca e Cb, respectivamente.
- Neste caso, deve-se adicionar uma chave estrangeira em Ta para referenciar a chave primária de Tb.

April 05

Prof. Ismael H. F. Santos - ismael@tecgraf.puc-rio.br

24

Mapeamento: Associações 1-muitos



April 05

Prof. Ismael H. F. Santos - ismael@tecgraf.puc-rio.br

25

Mapeamento: Associações muitos-muitos

- Seja Ca a classe na qual cada objeto se associa com muitos objetos da classe Cb.
- Sejam Ta e Tb as relações resultantes do mapeamento de Ca e Cb, respectivamente.
- Uma **relação de associação** deve ser criada.
 - Uma relação de associação serve para representar a associação muitos para muitos entre duas ou mais relações.

April 05

Prof. Ismael H. F. Santos - ismael@tecgraf.puc-rio.br

26

Mapeamento: Associações muitos-muitos

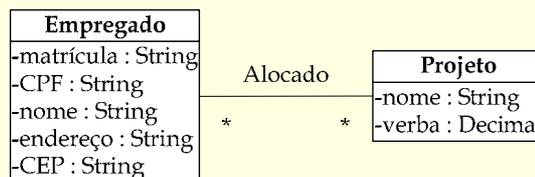
- Equivalente à aplicação do mapeamento *um para muitos* duas vezes, considerando-se os pares (Ta, Tassoc) e (Tb, Tassoc).
- Alternativas para definir a chave primária de Tassoc.
 - definir uma **chave primária composta**.
 - criar uma coluna de implementação que sirva como chave primária simples da relação de associação.

April 05

Prof. Ismael H. F. Santos - ismael@tecgraf.puc-rio.br

27

Mapeamento: Associações muitos-muitos



```
Departamento(id, sigla, nome, idEmpregadoGerente)
Empregado(id, matricula, CPF, nome, endereco, CEP, idDepartamento)
Alocacao(idProjeto, idEmpregado, nome, verba)
Projeto(id, nome, verba)
```

```
Departamento(id, sigla, nome, idEmpregadoGerente)
Empregado(id, matricula, CPF, nome, endereco, CEP, idDepartamento)
Alocacao(id, idProjeto, idEmpregado, nome, verba)
Projeto(id, nome, verba)
```

April 05

Prof. Ismael H. F. Santos - ismael@tecgraf.puc-rio.br

28

Mapeamento: Agregações

- Forma especial de associação → *mesmo* procedimento para realizar o mapeamento de associações pode ser utilizado.
- No entanto, a diferença semântica influi na forma como o SGBDR deve agir quando um registro da relação correspondente ao *todo* deve ser excluído ou atualizado.
 - Remoção ou atualização em cascata.
 - Pode ser implementado como *gatilhos* e *procedimentos armazenados*.

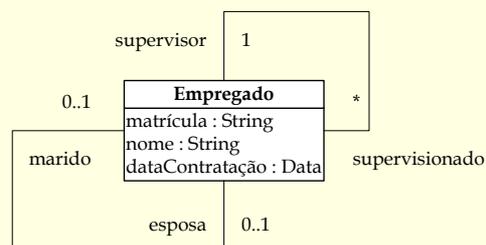
Mapeamento: Agregações

- O padrão de acesso em agregações (composições) também é diferente do encontrado nas associações.
 - Quando um objeto *todo* deve ser restaurado, é natural restaurar também os objetos *parte*.
 - Em associações, isso nem sempre é o caso.
 - Definição de *índices* adequados é importante para acesso eficiente aos objetos *parte*.

Mapeamento: Associações Reflexivas

- Forma especial de associação → *mesmo* procedimento para realizar o mapeamento de associações pode ser utilizado.
- Em particular, em uma associação reflexiva de conectividade *muitos para muitos*, uma relação de associação deve ser criada.

Mapeamento: Associações Reflexivas



```
Empregado(id, matricula, nome, dataContratacao, idCônjuge, idSupervisor)
```

Mapeamento: Associações n-árias

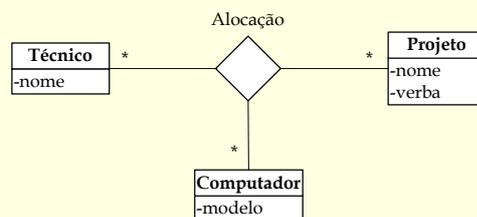
- Associações n-árias ($n \geq 3$): procedimento semelhante ao utilizado para associações binárias de conectividade *muitos para muitos*.
 - Uma relação para representar a associação é criada.
 - São adicionadas nesta relação chaves estrangeiras.
 - Se a associação n-ária possuir uma classe associativa, os atributos desta são mapeados como colunas da relação de associação.

April 05

Prof. Ismael H. F. Santos - ismael@tecgraf.puc-rio.br

33

Mapeamento: Associações n-árias



```
Técnico( id, nome )  
Projeto( id, nome, verba )  
Computador( id, modelo )  
Alocação( id, idProjeto, idTécnico, idComputador )
```

April 05

Prof. Ismael H. F. Santos - ismael@tecgraf.puc-rio.br

34

Mapeamento: Classes Associativas

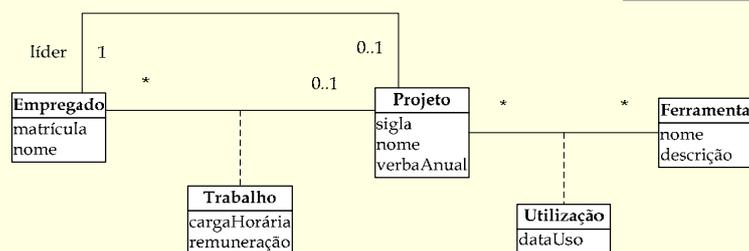
- Para cada um dos casos de mapeamento de associações, há uma variante onde uma classe associativa é utilizada.
- Mapeamento é feito através da criação de uma relação para representá-la.
 - Os atributos da classe associativa são mapeados para colunas dessa relação.
 - Essa relação deve conter chaves estrangeiras que referenciem as relações correspondentes às classes que participam da associação.

April 05

Prof. Ismael H. F. Santos - ismael@tecgraf.puc-rio.br

35

Mapeamento: Classes Associativas



```
Empregado(id, matrícula, nome)
Projeto(id, sigla, nome, verbaAnual, idEmpregadoLíder)
Ferramenta(id, nome, descrição)
Utilização(id, idFerramenta, idProjeto, dataUso )
Trabalho(id, idEmpregado, idProjeto, cargaHorária, remuneração)
```

April 05

Prof. Ismael H. F. Santos - ismael@tecgraf.puc-rio.br

36

Mapeamento: Generalizações

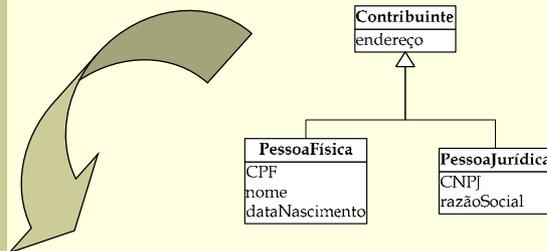
- Três formas alternativas de mapeamento:
 - Uma relação para cada classe da hierarquia
 - Uma relação para toda a hierarquia
 - Uma relação para cada classe concreta da hierarquia
- **Nenhuma** das alternativas de mapeamento de generalização é a melhor.
 - Cada uma delas possui vantagens e desvantagens.
 - Escolha de uma delas depende das do sistema sendo desenvolvido.
 - A equipe de desenvolvimento pode decidir implementar mais de uma alternativa.

April 05

Prof. Ismael H. F. Santos - ismael@tecgraf.puc-rio.br

37

Mapeamento: Generalizações



```
Contribuinte(id, endereço)
PessoaFísica(id, nome, dataNascimento, CPF, idContribuinte)
PessoaJurídica(id, CNPJ, razãoSocial, idContribuinte)
Pessoa(id, nome, endereço, dataNascimento, CPF, CNPJ, razãoSocial, tipo)
PessoaFísica(id, dataNascimento, nome, endereço, CPF)
PessoaJurídica(id, CNPJ, endereço, razãoSocial)
```

April 05

Prof. Ismael H. F. Santos - ismael@tecgraf.puc-rio.br

38

Mapeamento: Generalizações

- A 1ª alternativa (uma relação para cada classe da hierarquia) é a que melhor reflete o modelo OO.
 - classe é mapeada para uma relação
 - as colunas desta relação são correspondentes aos atributos específicos da classe.
- Desvantagem: desempenho da manipulação das relações.
 - Inserções e remoções e *junções*.

Mapeamento: Generalizações

- A 2ª alternativa de implementação é bastante simples, além de facilitar situações em que objetos mudam de classe.
- Desvantagem: alteração de esquema
 - Adição ou remoção de atributos.
 - tem o potencial de desperdiçar bastante espaço de armazenamento:
 - hierarquia com várias classes “irmãs”
 - objetos pertencem a uma, e somente uma, classe da hierarquia.

Mapeamento: Generalizações

- A 3ª alternativa apresenta a vantagem de agrupar os objetos de uma classe em uma única relação.
- Desvantagem: quando uma classe é modificada, cada uma das relações correspondentes as suas subclasses deve ser modificada.
 - Todas as relações correspondentes a subclasses devem ser modificadas quando a definição da superclasse é modificada.

O modelo de dados objeto-relacional (MOR)

- É uma extensão do modelo relacional, onde são adicionadas características de OO.
- Um SGBDOR é um SGBD que armazena informações de acordo com o MOR.
 - pode processar dados relacionais e objetos.
- Padronizado em 1999 (SQL99)
- Permite a definição de estruturas de dados arbitrariamente complexas (classes).
 - Colunas podem conter valores de tipos de dados estruturados.
 - O mapeamento de objetos pode ser feito mais diretamente.

O modelo de dados objeto-relacional (MOR)

- Os SGBDOR são ideais para certas aplicações especiais, como CAD/CAM.
- No entanto, atualmente estes SGBDO não dão suporte completo ao padrão SQL99.
 - um maior suporte a este padrão tornará o mapeamento de objetos mais fácil.
- Fato: existe uma plataforma imensa de sistemas que usam o modelo relacional puro.
 - Mais que isso, existe uma grande resistência em substituir esses sistemas.
- Isso leva a crer que o mapeamento de objetos para o modelo relacional ainda irá durar por muitos anos.

Manutenção de objetos persistentes

- Outros aspectos relativos ao armazenamento de objetos devem ser definidos:
 - Materialização: restaurar um objeto a partir do banco de dados quando necessário.
 - Atualização: enviar modificações sobre um objeto para o banco de dados.
 - Remoção: remover um objeto do armazenamento persistente.
- Essas funcionalidades permitem que objetos perdurem e sejam modificados em diversas execuções do sistema.

Acesso direto ao banco de dados

- Solução simples: fazer com que cada objeto persistente possua comportamento que permita a sua restauração, atualização ou remoção do mecanismo persistente conforme necessário (SQL).
- Fácil implementação em 4GLs (controles *data aware*).
- Desvantagens:
 - Classes da lógica do negócio acopladas às classes relativas à interface e ao acesso ao BD.
 - Migração do sistema de um SGBD para outro.
 - Lógica da aplicação fica desprotegida de eventuais modificações na estrutura do BD.

April 05

Prof. Ismael H. F. Santos - ismael@tecgraf.puc-rio.br

45

Acesso direto ao banco de dados

- Desvantagens:
 - Cada programador deve ter conhecimento sobre SQL.
 - A **coesão** das classes diminui porque cada classe deve possuir responsabilidades relativas ao armazenamento e materialização de seus objetos.
- A dificuldade de manutenção e extensão do código fonte resultante praticamente proíbe a utilização desta abordagem para sistemas complexos.

April 05

Prof. Ismael H. F. Santos - ismael@tecgraf.puc-rio.br

46

A camada de persistência

- **Objetivo: isolar os objetos de mudanças no mecanismo de armazenamento.**
 - Se um SGBD diferente tiver que ser utilizado pelo sistema, por exemplo, somente a camada de persistência é modificada; os objetos de domínio permanecem intactos.
- **diminuição do acoplamento entre os objetos e a estrutura do BD torna:**
 - mais flexível (pode ser modificado para se adaptar a novos requisitos)
 - mais portátil (pode ser transportado para outras plataformas de HW ou SW).

A camada de persistência

- **Desvantagens:**
 - intermediação traz uma sobrecarga de processamento ao sistema, o que pode diminuir o seu desempenho.
 - pode aumentar a complexidade da realização de certas operações que seriam triviais com o uso direto de SQL.
- **Entretanto, as vantagens adquiridas pela utilização de uma camada de software, principalmente em sistemas complexos, geralmente compensam as desvantagens.**

A camada de persistência

- Algumas funcionalidades necessárias em uma camada de persistência
 - Implementação da camada de persistência
 - Persistência de objetos
 - Manipulação de coleções de objetos
 - Gerenciamento de transações
 - Mapeamento de nomes de atributos para nomes de colunas