SISTEMAS DE BANCO DE DADOS 1

AULA 8

Controle de Consistência, Dependência Funcional e Normalização

Vandor Roberto Vilardi Rissoli

APRESENTAÇÃO

- Controle de Consistência
- Dependência Funcional
- Normalização
- Referências

Controle de Consistência

No Modelo Relacional, cada *esquema de uma relação* possui um número de atributos (grau da relação), enquanto que o *esquema de um BD relacional* consiste de uma série de esquemas de relações.

A avaliação de um mapeamento proposto por um projetista, ou resultante de um ME-R, deve respeitar algumas características que o conceitue como um mapeamento eficiente.

A indicação de que um esquema relacional está eficiente é realizada quando as relações produzidas representam os dados de forma consistente e sem redundância.

Controle de Consistência

No intuito de orientar o projetista na definição e avaliação de um esquema relacional é aplicado um conjunto de critérios informais que o auxilie, sendo alguns destes critérios apresentados a seguir:

A) **Semântica dos atributos de uma relação**

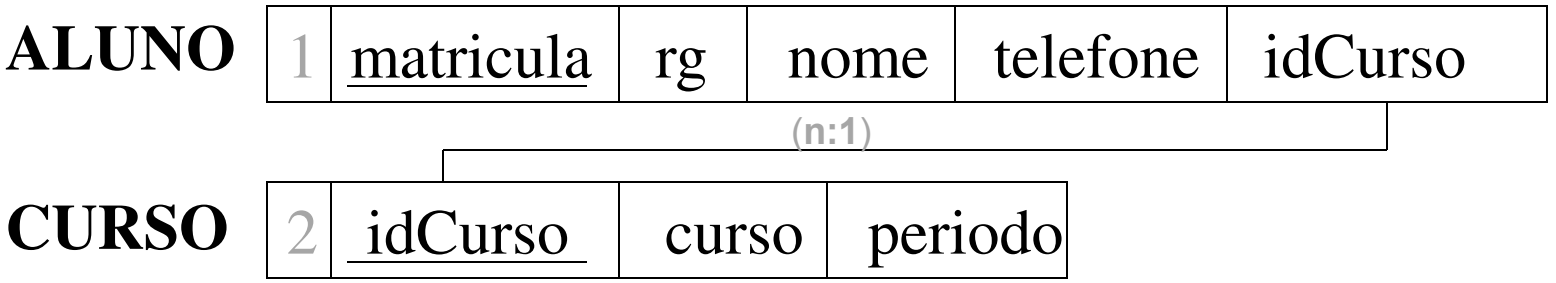
Sempre que um conjunto de atributos é agrupado em uma relação, um determinado significado é associado a eles. Este significado (ou semântica) identificam como devem ser interpretados os atributos pertencentes a cada tupla da relação.

Por exemplo a relação ALUNO_CURSO representada a seguir:

ALUNO_CURSO (nome,rg,matricula,telefone,curso,periodo,idCurso)

Controle de Consistência

O esquema anterior (ALUNO_CURSO), apesar de representar claramente o seu significado (os alunos que fazem um curso), não é aconselhado, pois possui atributos de duas entidades. A representação mais correta, por meio de esquemas seria:



Controle de Consistência

B) **Redundância de valores**

Sempre deve-se evitar a definição de dados em mais que um local (esquema) no BD, ou seja, a duplicação (ou redundância) das dados no BD.

Como exemplo, considere a relação ALUNO_CURSO contendo os alunos e os seus respectivos cursos. Se mais de um aluno estudar em um mesmo curso, os valores dos atributos do curso deverão aparecer duplicados para cada tupla de aluno.

Com isso podem ocorrer alguns problemas nas operações de atualização, tais como:

Controle de Consistência

Inserção: para colocar um novo aluno, deve-se preencher os atributos do curso que ele faz, ou colocar atributos nulos, quando ele não fizer nenhum curso (pesquisador). Para possuir um curso sem alunos, torna-se necessário colocar valores nulos para os atributos do aluno, o que significa que a chave da tupla será nula, violando a restrição de integridade de chave.

Remoção: quando for removido o último aluno de um determinado curso, todos os dados referentes a este curso serão perdidos.

Atualização: a alteração do atributo de período de um curso deverá ser feita em todas as tuplas da relação, onde um aluno faça este curso, caso contrário o banco de dados se tornará inconsistente.

Controle de Consistência

C) **Valores nulos em tuplas**

Um atributo que possua valor nulo permite diferentes interpretações, tais como:

1- atributo não se aplica para esta tupla

2- valor do atributo para esta tupla é desconhecido

3- valor é conhecido mas encontra-se ausente (não inserido)

Sempre procure evitar atributos que possam ser nulos. Quando for inevitável, verifique se estas tuplas acontecem em pequeno número na relação.

Controle de Consistência

Uma das importâncias fundamentais na definição de esquemas relacionais é manter a consistência dos dados em todo o banco de dados.

O controle de consistência dos dados pode ser exercido em três níveis: pelo gerenciador (SGBD), pelo aplicativo, ou pela própria construção de um sistema. De uma forma geral o controle de consistência obtido pela própria construção do sistema é mais eficiente, pois normalmente não implica em perda de desempenho do SGBD durante a sua execução.

O controle por meio da própria construção do sistema é obtido, no Modelo Relacional, construindo-se as relações segundo regras que garantem a manutenção de certas propriedades. As relações que atendem a determinadas regras diz-se estarem em uma determinada "Forma Normal".

Controle de Consistência

O processo de Normalização pode ser considerado como a aplicação de uma série de regras, que constituem as Formas Normais.

Este processo irá provocar a decomposição de esquemas insatisfatórios, de algumas relações, em novas relações que tenham cuidados com as situações já apresentadas.

A Normalização também permite ao projetista controlar o quanto de consistência é garantida pela maneira de construção do sistema (estrutura), e quanto deve ser de responsabilidade dos aplicativos e/ou SGBD.

Porém deve-se analisar até que ponto a Normalização deve chegar, pois Normalizar demais pode diminuir a eficiência dos aplicativos, e de menos abri a possibilidade das inconsistências.

Dependência Funcional

Baseia-se no reconhecimento que os valores de alguns atributos contribuem na identificação de outros. Uma dependência funcional significa que ao se conhecer o valor de um atributo, então pode-se sempre determinar o valor de outro. A notação usada na teoria relacional é:

A → B onde se lê que "A **determina funcionalmente** B", ou que "B **depende de** A".

Exemplo: conhecendo a matrícula de um aluno, então posso determinar o seu nome, ou conhecendo o código de um produto, então posso determinar o seu preço.

Dependência Funcional

Uma dependência multivalorada significa que ao se conhecer o valor de um atributo, então pode-se sempre determinar os valores de um conjunto de outros atributos. A notação usada na teoria relacional é:

A →→ B onde se lê "A **determina muitos** B's"

Exemplo: conhecendo um curso posso determinar a lista de seus alunos.

Uma das maneiras de controlar a consistência é por meio das *Dependências Funcionais* existentes entre os atributos armazenados.

Normalização

As Formas Normais, processo chamado de Normalização, são identificadas por meio de números, onde serão usadas a primeira, segunda e terceira forma normal no decorrer da disciplina.

No Modelo Relacional a normalização principal é a primeira, pois é considerada parte da própria definição de uma relação.

Primeira - (1 FN)

Uma relação se encontra na Primeira Forma Normal se todos os domínios de atributos possuem apenas valores atômicos (simples e indivisíveis). Com isso a relação é construída sem atributos compostos e multivalorados em suas tuplas.

Exemplo:

CURSO(curso, periodo, idCurso,(nome, matricula)) – não 1FN

Normalização

Para levar uma relação a atender as exigências da primeira forma normal (normalizar) tem-se que realizar a aplicação de algumas "**regras**" para as seguintes situações:

1- Cada um dos atributos compostos (ou não atômicos) devem ser "divididos" em seus componentes

2- Para os atributos multivalorados tem-se duas alternativas:

A) Pequena quantidade de valores possíveis, substitui-se o atributo multivalorado por um conjunto de atributos de mesmo domínio, cada um mono valorado representando uma ocorrência do valor;

B) Quantidade de possíveis valores desconhecida, grande ou muito variada, retira-se da relação o atributo multivalorado e cria-se uma nova relação que tem o mesmo conjunto de atributos chave, mais o atributo multivalorado também como chave, mas tomado como mono valorado.

Normalização

A 1 FN é uma das maneiras de controlar a consistência por meio da própria estrutura do sistema, sendo também fundamental para a conceituação do sistema.

Segunda - (2 FN)

Uma relação esta na Segunda Forma Normal quando ela está na 1 FN e todos os atributos que não participam da chave primária são dependentes diretos de toda a chave primária.

Para se normalizar uma relação para a 2 FN as "**regras**" seguintes devem ser seguidas.

Normalização

A normalização para um relação estar na 2 FN deve:

1- Verificar os grupos de atributos que dependem da mesma parte da chave primária.

2- Retirar da relação todos os atributos de um desses grupos.

3- Criar uma nova relação contendo esse grupo como atributo não chave, e os atributos que determinam esse grupo como chave primária.

4- Repetir o procedimento para cada grupo, até que a relação toda somente contenha atributos que dependam da chave primária.

Normalização

Esta normalização evita a inconsistência devido a duplicidade de informações, além da perda de dados em operações de remoção ou de alterações na relação.

Note que a normalização de relações é realizada, na grande maioria das vezes, decompondo-se uma relação em duas ou mais relações.

Exemplo: não está na 2 FN

EMPR_PROJ (matrEmpr, codProj, horas, nomeEmpr, nomeProj, localProj)

Normalização

continuação do exemplo...

EMPR_PROJ (matrEmpr, codProj, horas, nomeEmpr, nomeProj, localProj)

- Dependências funcionais parciais em relação à chave primária:
  matrEmpr → nomeEmpr
  codProj → nomeProj, localProj
- Dependências funcionais totais em relação à chave primária:
  (matrEmpr, codProj) → horas

Esquema que atende a 2 FN:

EMPR (matrEmpr, nomeEmpr)

PROJ(codProj, nomeProj, localProj)

EMPR_PROJ(matrEmpr,codProj, horas)

Normalização

Terceira - (3 FN)

Uma tabela está na Terceira Forma Normal se estiver na 2 FN e não possuir dependências transitivas (ou indireta). Uma depen-dência transitiva ocorre quando A → B e B → C, então A → C.

Em outras palavras, deve-se evitar que qualquer atributo não chave seja dependente funcional de outro atributo não chave.

Exemplo: não está na 3 FN

EMPR_DEPTO (matrEmpr, nomeEmpr, dataNasc, codDep, nomeDep)

Normalização

continuação do exemplo...

EMPR_DEPTO (matrEmpr, nomeEmpr, dataNasc, codDep, nomeDep)

Dependências transitivas (indiretas):

matrEmpr → codDep → nomeDep

Esquema que atende a 3 FN:

EMPR (matrEmpr, nomeEmpr, dataNasc, endEmpr, codDep)

DEPTO (codDep, nomeDep)

Restrição **EMPR_FK** de chave estrangeira (codDep) que Referencia DEPTO(codDep)

Normalização

Da mesma forma que a 2 FN, a normalização para a 3 FN evita a inconsistência devido a duplicidade de informações, além da perda de dados em operações de remoção ou de alterações na relação.

Quarta - (4 FN)

A Quarta Forma Normal (4 FN) faz uso de dependências funcionais multivaloradas.

Considere a relação Depto_Projeto_Pecas a seguir, onde:

Depto →→ Proj

e

Depto →→ Peca

Depto_Projeto_Pecas

Depto	Proj	Peça
D1	J1	P1
D1	J1	P2
D1	J2	P1
D1	J2	P2
D2	J3	P2
D2	J3	P4
D2	J4	P2
D2	J4	P4
D2	J5	P2
D2	J5	P4
D3	J2	P5
D3	J2	P6

Normalização

Observe que para adicionar uma peça para um departamento deve-se criar uma nova linha (tupla). Remover uma peça ou um projeto de uma linha pode resultar em perda de informação. A atualização de uma peça ou de um projeto pode requerer que muitas linhas sejam atualizadas.

A solução é "quebrar" esta relação em duas relações, uma com (Depto, Proj) e outra com (Depto, Peca).

→ Existem outras formas normais, porém a quarta, a quinta e as outras não são baseadas nas dependências funcionais.

Normalização

Realizando a Normalização:

- A normalização é aplicada em uma relação por vez;
- Uma relação vai sendo dividida criando outras relações;
- Inicia-se das formas normais menos rígidas (1 FN→2 FN→...) até chegar em uma normalização considerada satisfatória.

A decisão de normalizar ou não uma relação é um compromisso entre garantir a eliminação de inconsistências no banco de dados e a eficiência de acesso. A normalização para formas apoiadas em dependências funcionais evita inconsistências, usando para isso a própria construção do banco de dados.

→ Se a consistência não for um fator fundamental, pode-se abrir mão da normalização.

Tipos de Dados

Para iniciar a definição de um Banco de Dados é necessário conhecer os tipos de dados que o banco pode manipular.

De forma geral serão manipulados os tipos de dados:

* numéricos (-45 | 0 | 25.57)
* literais (José Roberto | porcelana | A)
* datas (dia, mês e ano | 23 / 12 / 01) - hora

A sintaxe que identifica estes tipos em ORACLE são:

* **number**(*n*,*d*) – numéricos (**int** e outros)
* **varchar2**(*n*) – caracteres variáveis até 4000 símbolos
* **date** – de 1/1/4712 A.C. até 31/12/9999 D.C.

→ *n* corresponde ao comprimento ou tamanho

→ *d* corresponde a quantidade de dígitos decimais

Dados Obrigatórios

O armazenamento de dados em uma relação, normalmente, necessita de alguns dados obrigatórios, enquanto que outros dados podem ser informados somente para algumas situações.

Exemplo: suponha o cadastro de um aluno na relação abaixo.

ALUNO(matricula, nome, nascimento, rg, cpf, telefone)

Para a efetivação da matrícula os dados coletados de um aluno deverão atender aos atributos especificados na relação acima, porém existiram situações em que um aluno ainda não fez o seu documento de CPF e também não possua telefone. Isso seria motivo para não realizar a matrícula do mesmo?

Dados Obrigatórios

A obrigatoriedade de um atributo é identificado pela expressão NOT NULL que deve estar vinculada a todos os atributos que são obrigatórios em uma relação.

No exemplo anterior todas as informações seriam obrigatórias, com exceção do telefone e dependendo das normas de tal escola o CPF.

Descrição da Relação

Conhecendo os tipos de dados para cada atributo da relação (tabela), e algumas das restrições que as envolvem no momento de sua criação, é possível elaborar a descrição da relação que será implementada pela Linguagem SQL.

Descrição da Relação

Exemplo: a descrição da relação a ser criada no BD por meio de uma instrução DDL seria:

```
CREATE TABLE ALUNO (
    matricula          number(8)        NOT NULL,
    nome               varchar2(30)     NOT NULL,
    dtNascimento       date             NOT NULL,
    rg                  varchar(10)     NOT NULL,
    cpf                number(11),
    telefone           number(12)    ) ;
```

Os dados a serem armazenados nesta relação teriam que atender as restrições informadas de tipo de dado, tamanho e obrigatoriedade.

Note também a pontuação necessária para essa instrução.

Descrição da Relação

Na relação apresentada como exemplo, também é identificada a restrição de **integridade referencial**, onde a chave primária simples consiste do atributo **matricula**.

Esta restrição será descrita no final da relação, onde a sintaxe, na forma geral, seria:

CONSTRAINT <identificador> Primary Key (<atributo>)

Para o exemplo anterior essa restrição em SQL seria:

CONSTRAINT Aluno_**PK** Primary Key (matricula)

Supondo que a restrição desta relação fosse uma chave primária composta por **matricula** e **dtNascimento** se teria:

CONSTRAINT Aluno_**PK** Primary Key (matricula, nascimento)

Descrição da Relação

Uma restrição de integridade referencial implanta a chave primária que permitirá um relacionamento com outras relações de forma consistente, por meio de uma chave estrangeira (ou secundária).

Este tipo de chave também deve ser identificada de forma clara na descrição da relação que será criada. Observe a sintaxe correta a ser efetuada na relação que possui o atributo que será descrito como uma chave estrangeira.

Forma Geral

CONSTRAINT <identificador> Foreign Key (<atributo dessa relação>) References <outra relação>(<atributo>)

→ O uso de vírgulas também é necessário para separar uma atributo do outro, quando as chaves primárias envolvidas forem compostas.

Descrição da Relação

Continuando o exemplo anterior, suponha a existência da relação CURSO que possui relacionamento com a relação ALUNO com cardinalidade identificada de que um aluno pode fazer só um curso e que um curso pode possui vários alunos (ALUNO **n : 1** CURSO). A representação dos esquemas seriam:

ALUNO(matricula, nome,nascimento,rg, cpf,telefone, idCurso)

CURSO(idCurso, nome, periodo)

Estas relações possuem chaves primárias, além de uma chave estrangeira que implementa o seu relacionamento.

Observe a descrição das relações e a identificação coerente das restrições descritas a seguir.

Descrição da Relação

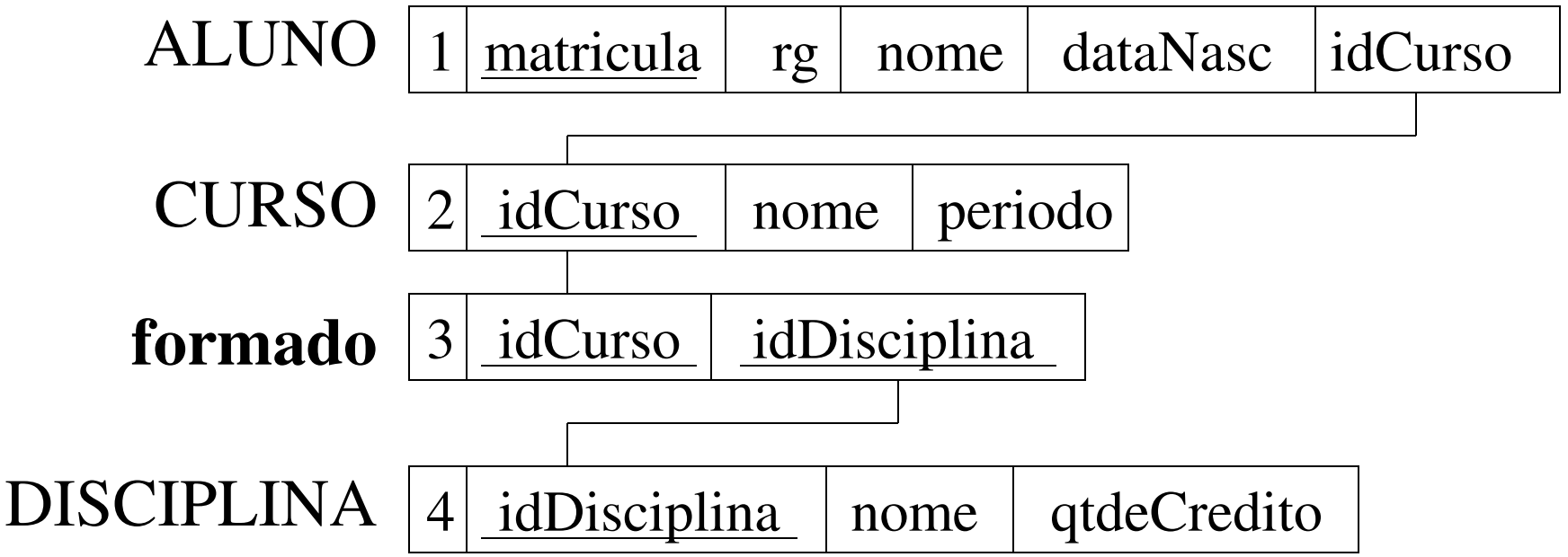
Descrição do exemplo ALUNO x CURSO em SQL:

```
CURSO (
        idCurso         number(2)           NOT NULL,
        nome            varchar2(20)        NOT NULL,
        período         varchar2(15)        NOT NULL,
CONSTRAINT Curso_PK  Primary Key (idCurso));

ALUNO (
        matricula       number(8)           NOT NULL,
        nome            varchar2(30)        NOT NULL,
        dtNascimento    date                NOT NULL,
        rg              varchar(10)         NOT NULL,
        cpf             number(11),
        telefone        number(12),
        idCurso         number(2),
CONSTRAINT Aluno_PK  Primary Key (matricula),
CONSTRAINT Aluno_Curso_FK Foreign Key (idCurso)
                References Curso(idCurso) );
```

Exercício de Fixação

1) De acordo com o Diagrama de Esquemas a seguir faça as descrições das relações que serão implementadas neste banco de dados, descrevendo corretamente as restrições de **integridade de entidade, referencial** e colocando como comentário ("--") os valores para a **integridade semântica**, quando ela existir sobre algum atributo.



Exercício de Fixação

2) A Farmácia Ai-Ai fez a aquisição de alguns equipamentos de informática para conseguir gerenciar melhor os dados que são necessários ao seu funcionamento. Para o desenvolvimento de um banco de dados você foi contratado, e a especificação do problema foi sintetizada a seguir. Os produtos comercializados nesta farmácia são adquiridos de fornecedores únicos para cada produto, apesar de serem diversos os fornecedores (ou fabricantes). Cada fabricante é identificado por meio de seu CGC, nome, endereço completo e único da sede do distribuidor e telefones de distribuição. Os produtos comercializados possuem código de controle, nome comercial, tipo de embalagem, quantidade e preço unitário. A farmácia comercializa dois tipos de produtos: os medicamentos que apresentam uma fórmula e uma tarja e os itens de perfumaria que tem tipo e fragrância.

Exercício de Fixação

... continuação do exercício.

A venda destes produtos também deverá ser acompanhada pelo banco de dados, onde cada venda deverá armazenar a data de compra, número da nota fiscal, nome do cliente, quantidade, preço total e o imposto a ser recolhido.

Somente para os medicamentos é necessário a apresentação da receita médica que terá o CRM do médico, a identificação do medicamento e a data da emissão da receita.

Baseado na especificação anterior elabore o projeto de banco de dados a partir do ME-R até a implementação física dessa solução para farmácia Ai-Ai. De acordo com o projeto elaborado realize o mapeamento deixando os esquemas dentro da **terceira forma normal** (3FN), respeitando todas as principais restrições que devem ser observadas para o funcionamento adequado do Banco de Dados Relacional.

Referência de Criação e Apoio ao Estudo

Material para Consulta e Apoio ao Conteúdo

- ELMASRI, R. e NAVATHE, S. B., Fundamentals of Database Systems, Addison-Wesley, 3rd edition, 2000
  - Capítulo 14
- SILBERSCHATZ, A. & KORTH, H. F., Sistemas de Banco de Dados - livro
  - Capítulo 7
- HEUSER, C. A., Projeto de Banco Dados - livro
  - Capítulo 6
- Universidade Católica de Brasília – site
  - http://cae.ucb.br/conteudo/programar
    (escolha no menu superior a opção **Banco de Dados** seguida de **Banco de Dados I**)