Universidade Federal da Fronteira Sul - UFFS
Campus Chapecó
Ciência da Computação
Banco de Dados I
Prof.: Denio Duarte

Notas de Aula - Modelo Relacional - Cap. 03 - Elmasri 6ª Edição

O modelo relacional representa um banco de dados (BD) como um conjunto de relações (também conhecido por tabelas). Cada relação se assemelha a uma tabela de valores. Se enxergarmos uma relação como uma tabela temos uma linha de uma tabela representando um fato que corresponde a uma instância (ou uma ocorrência) de uma entidade (ou relacionamento).

Uma instância ou ocorrência de uma relação ou tabela Cliente poderia ser:
045 Fulano R. Joinville, 20
Na representação tabela seria:

Codigo	Nome	Ender
045	Fulano	R. Joinville, 20

Assim, no modelo relacional temos:
Os *atributos* que representam a menor característica de um objeto que é chamado de *relação* ou *tabela*.
Uma ocorrência (um fato) de uma relação ou tabela é chamada de tupla ou linha.

Domínios, Atributos, Tuplas e Relações

• Um domínio $\mathcal{D}$ é um conjunto de valores atômicos.
Ex.: um *int* é um domínio de inteiros.
CPF pode ter como domínio uma sequência de 11 dígitos de 0 à 9 (nnnnnnnnnnn)
• Um esquema de relação $\mathcal{R}$, indicado por $\mathcal{R}(A_1, A_2, \ldots, A_n)$, é composto pelo nome da relação $\mathcal{R}$ e um conjunto de atributos $A_1, A_2, \ldots, A_n$. Cada atributo A_i $(0 < i \leq n)$ é o nome de uma papel desempenhado por algum domínio $\mathcal{D}$ em $\mathcal{R}$. $\mathcal{D}$ é chamado de domínio de A_i e é indicado como $dom(A_i)$.
• O grau (ou aridade) de $\mathcal{R}$ é o número de atributos de $\mathcal{R}$, denota-se $grau(\mathcal{R})$ ou $arity(\mathcal{R})$
• A cardinalidade de uma relação $\mathcal{R}$ é o número de tuplas de $\mathcal{R}$, denotado por $card(\mathcal{R})$
Ex.: $Aluno(matric, cpf, nome, ender)$ ou $Aluno(matric : string, cpf : long\ int, nome : string, ender : string)$ representa o esquema da relação (ou tabela) *Aluno* que possui 4 atributos: *matric*, *cpf*, *nome* e *ender*. A primeira representação, que será utilizada neste curso, não indica o domínio de cada atributo, já a segunda indica.
$grau(Aluno) = 4$ ou $arity(Aluno) = 4$
• Uma relação (ou instância da relação) r do esquema de relação $\mathcal{R}(A_1, A_2, \ldots, A_n)$, indicado por $r(\mathcal{R})$, e o conjunto de n-tuplas $r = \{t_1, t_2, \ldots, t_n\}$. Cada n-tupla t é uma lista ordenada de n valores ($grau(\mathcal{R})$) $t =< v_1, v_2, \ldots, v_n >$ onde cada valor v_i $(0 < i \leq n)$ é um elemento de $dom(A_i)$ ou um valor *nulo* (*null*) - valor especial que indica valor inexistente/vazio. O i-ésimo valor de t, que corresponde ao atributo A_i, e denotado por $t[A_i]$ (ou $t[i]$ se utilizarmos a notação sem nome ou posicional).

Ex.: $t_2 =< 046, 66677733344, Ciclano, R.\text{Itajaí} >$, $t_2[nome] = Ciclano$, $t_2[1] = 046$
A tabela a seguir representa uma instância da relação *Aluno* cuja cardinalidade é 3 ($card(Aluno) = 3$), ou seja, possui 3 tuplas (ou linhas).

Aluno

matric	cpf	nome	ender
045	11122233344	Fulano	R. Joinville, 20
046	66677733344	Ciclano	R. Itajaí, 120
047	77788833344	Beltrano	R. Gaspar, 220

Formalmente

A instância da relação $r(\mathcal{R})$ é uma relação matemática de grau n nos domínios $dom(A_1)$, $dom(A_2)$, ..., $dom(A_n)$ que é um subconjunto do produto cartesiano dos domínios que definem $\mathcal{R}$:

$$r(\mathcal{R}) \subseteq (dom(A_1) \times dom(A_2) \times \ldots \times dom(A_n))$$

Produto cartesiano: todas as possíveis combinações dos domínios.
• O produto cartesiano acima indica todas as combinações possíveis do $r(\mathcal{R})$.
• O estado da relação corrente reflete apenas as tuplas válidas (subconjunto do produto cartesiano dos domínios) que representam um estado particular do mundo real de $\mathcal{R}$
• É possível que várias atributos tenham o mesmo domínio. Os nomes dos atributos, então, indicam diferentes papéis ou interpretações do domínio. Ex.: o atributo *idade* e *num_amigos* pertencem ao domínio dos inteiros positivos mas o papel do primeiro é indicar a idade de uma determinada entidade e o segundo o número de amigos que uma determinada entidade tem.
• As tuplas em uma relação não tem ordem. Porém, fisicamente, uma tabela tem ordem pois as tuplas (linhas) estão armazenadas no disco.

Notação

- Um esquema de uma relação $\mathcal{R}$ de grau n é indicado por $\mathcal{R}(A_1, A_2, \ldots, A_n)$
- As letras maiúsculas Q, R e S indicam nomes de relação.
- As letras minúsculas q, r e s indicam estados de relação.
- As letras t, u, v indicam tuplas.
- Quando for utilizado o nome da relação, por exemplo $ALUNO$, também indicará o conjunto atual de tuplas nessa relação (ou seja, o *estado de relação atual*, enquanto $ALUNO$ ($Nome$, CPF, ...) se refere apenas ao esquema de relação.
- Um atributo A pode ser qualificado como o nome da relação R a qual pertence utilizando a notação $R.A$. Isso pois duas relação diferentes podem ter nome de atributos iguais.
- Uma n-tupla t em uma relação r(R) é indicado por $t =< v_1, v_2, \ldots, v_n >$, onde v_i é o valor correspondente ao atributo A_i. Os valores componentes de uma tupla podem ser representados da seguinte maneira:
  - Tanto $t[A_i]$ quanto $t.A_i$ (e às vezes $t[i]$) se referem ao valor v_i em t para o atributo A_i.
  - Tanto $t[A_u, A_w, \ldots, A_z]$ quanto $t.(A_u, A_w, \ldots, A_z)$, onde $A_u, A_w, \ldots, A_z$ é uma lista de atributos de R, se referem à subtupla de valores $< v_u, v_w, \ldots, v_z >$ de t correspondentes aos atributos especificados na lista.
  - Ex.: dada a tupla t =<045,11122233344,'Fulano','R. Joinville, 20'> da relação $CLIENTE$ (apresentada anteriormente), temos $t[Nome]$ =<'Fulano'> e $t[Codigo, Ender]$ =<045,'R. Joinville, 20'>

Fundamentos do Modelo Relacional

Um banco de dados relacional é um modelo de banco de dados em que os objetos da aplicação são representados por relações ou tabelas. Nesse modelo, uma tabela T_1 pode ter um atributo especial (um ponteiro) que indica uma outra tabela T_2 em que T_1 faz referência. Por exemplo, podemos ter uma tabela que armazene dados de um objeto carro. Caso queiramos saber os dados do proprietário do carro (por exemplo, o cliente do nosso estacionamento), ao invés, de acrescentarmos mais atributos na tabela carro com os dados do proprietário, criamos uma nova tabela que armazene os dados dos proprietários e criamos um *ponteiro* em carro apontando para o proprietário.
Exemplo em que não segue o modelo relacional.

Carro

placa	modelo	ano	nomep	cpf	ender
M222	Fusca	1977	Sr. Fusca	123222	R. Joinville, 20
C445	Chevete	1981	Ciclano	33344	R. Itajaí, 120
X777	Maverick	1978	Beltrano	67544	R. Gaspar, 220

Perceba que o nome do proprietário está armazenado na tabela *Carro*. Isso parece uma boa ideia mas pode causar problemas futuros, por exemplo, se um proprietário tiver mais de um carro, os dados deste proprietário terão que ser repetidos:

Carro

placa	modelo	ano	nomep	cpf	ender
M222	Fusca	1977	Sr. Fusca	123222	R. Joinville, 20
C445	Chevete	1981	Ciclano	33344	R. Itajaí, 120
X777	Maverick	1978	Beltrano	67544	R. Gaspar, 220
T222	Corcel	1972	Ciclano	33344	R. Itajaí, 120

Agora o proprietário <Ciclano, 33344, R. Itajaí, 120> aparece duas vezes na tabela *Carro*. O modelo relacional foi proposto, entre outros, para evitar esta redundância. Assim, no banco, cada objeto tem a sua própria tabela e caso seja necessário ter uma referência entre as tabelas, uma delas é escolhida para ter o *atributo ponteiro*. A tabela que possuirá o *atributo ponteiro* é aquela que depende dos dados da outra. No nosso exemplo, se pensarmos em existência, um proprietário tem que existir para um carro existir , assim carro depende do proprietário então a tabela *Carro* receberá este *atributo ponteiro*.

Carro

placa	modelo	ano	prop
M222	Fusca	1977	123222
C445	Chevete	1981	33344
X777	Maverick	1978	67544
T222	Corcel	1972	33344

Proprietario

nomep	cpf	ender
Sr. Fusca	123222	R. Joinville, 20
Ciclano	33344	R. Itajaí, 120
Beltrano	67544	R. Gaspar, 220

Perceba, agora, que a tabela *Carro* possui um novo atributo *prop* que indica o ponteiro para encontrar os dados do proprietário do carro na tabela *Proprietario*. O atributo *prop* é a representação do atributo *cpf* da tabela *Proprietario*. Assim, para encontrarmos os dados dos proprietários de cada carro, basta procurarmos uma ocorrência em *Proprietario* cujo o *cpf* seja igual a *prop*.

A próxima seção apresenta alguns conceitos importantes do modelo relacional bem como o exemplifica nesta seção (chamado de chave estrangeira).

Restrições

Restrições são derivadas das regras do minimundo que o banco de dados representa. Representam alguma restrição para o armazenamento e manutenção dos dados. O banco de dados que respeita as restrições impostas em um determinado estado é dito que tal banco está em um *estado válido*. Principais tipos de restrições:

- Restrições implícitas (modeladas no próprio modelo de dados).
- Restrições explícitas (expressas no próprio esquema da relação).
- Regras de negócio (restrições implementadas pela aplicação).
- Dependências funcionais (restrições de dependência de dados).

Restrições baseadas em esquema: restrições de domínio (domain constraints), restrições de chaves (key constraints), restrições sobre valores *nulos* (null constraint), restrições de integridade de entidade (integrity entity constraint) e restrições de integridade referencial (referential integrity constraint).

- Domínio: dentro de cada tupla, o valor de cada atributo A deve ser um valor indivisível de $dom(A)$.
- Valores *nulo*: indica se um valor $NULL$ pode ser permitido para um atributo ou não. Geralmente, os atributos opcionais de uma relação aceita valores *nulos*. É importante lembrar que o valor *nulo* é um valor vazio, ou seja, não pode ser representado. O valor *nulo* é importante para os atributos que as vezes não possuem valor para uma determinada tupla. Por exemplo, se em uma tabela que armazena dados dos clientes de uma empresa, o projetista do banco de dados acrescentar o atributo que represente o número da carteira nacional de habilitação (CNH), esse atributo terá que aceitar o valor *nulo* pois nem todas as pessoas têm CNH.
- Chave (key): uma relação é um conjunto de tuplas e, neste contexto, as tuplas são distintas entre si. Isso significa que duas tuplas não podem ter a mesma combinação de valores para todos os seus atributos. Normalmente, existe um subconjunto de atributos de um esquema de relação R com a propriedade de que duas tuplas em qualquer estado r de R não deverão ter a mesma combinação de valores. Suponha que SSa é um subconjunto de atributos da relação R que respeitam a restrição de chave, assim, não existirão duas tupas t e u tal que $t[SSa] = u[SSa]$, ou seja, para qualquer estado r de R e dadas duas tuplas quaisquer t e u, a propriedade $t[SSa] \neq u[SSa]$ deve ser respeitada. O subconjunto SSa é chamado de superchave (superkey) de R. Por definição, todos os atributos de uma relação R compõem a superchave de R. Porém, dado o conjunto de atributos de R chamado de $ABCD$, se encontramos um subconjunto de atributos ABC tal que para qualquer t e u a propriedade $t[ABC] \neq u[ABC]$, ABC é uma superchave menor que $ABCD$. Esse raciocínio pode ser continuado até encontrarmos um subconjunto onde a propriedade acima não é mais respeitada. Neste caso, o último subconjunto (que pode ser unitário) que respeitou a propriedade de distinção é chamado de *chave* (key) ou *superchave mínima* (minimal superkey). Em uma relação pode haver várias *chaves* porém apenas uma delas deve ser escolhida para representar unicamente uma tupla, esta chave será chamada de *chave primária* (primary key) e as outras de *chaves candidatas* (candidate key). Observe que uma *chave* deve valer para qualquer estado em qualquer momento de uma relação R. Por exemplo, o atributo $Nome$ da relação *Proprietario* pode se configurar com *chave* em um determinado estado da relação porém, pelo conhecimento anterior, é sabido que em

um momento futuro, este atributo poderá ter valores repetidos pois dois clientes diferentes podem ter o mesmo nome. Por outro lado, utilizando a mesma relação, o atributo cpf pode ser utilizado como chave pois não existem duas pessoas diferentes com o mesmo CPF garantindo a unicidade das tuplas.

As restrições vistas anteriormente são aplicadas dentro de uma relação. A seguir, apresentaremos as restrições que ocorrem interrelação. Para tal, definiremos banco de dados relacional e esquema de banco de dados relacional.
Um esquema de banco de dados relacional S é um conjunto de esquemas de relação $S = \{R_1, R_2, \ldots, R_n\}$ e um conjunto de restrições de integridade (RI) - integrity constraints (IC). Um estado de banco de dados relacional DB de S é um conjunto de estados de relação $DB = \{r_1, r_2, \ldots, r_m\}$, tal que cada r_i é um estado de R_i e tal que os estados da relação r_i satisfaçam as restrições de integridade especificadas em RI. Um estado de um banco de dados relacional que obedece às RI é chamado de *estado válido* (estado consistente), caso contrário é chamado de *estado inválido* (estado inconsistente).
Integridade referencial: essa restrição é especificada entre duas relações e é utilizada para manter a consistência entre tuplas de duas relações. Informalmente, a restrição de integridade referêncial afirma que uma tupla t em uma relação R que referencia outra relação S precisa se referir a uma *tupla existente* em S.

A integridade referencial é implementada com o conceito de *chave estrangeira* (foreign key). Uma chave estrangeira é um restrição de integridade entre dois esquemas de relação R_1 e R_2. Um conjunto de atributos FK de R_1 (tal que $FK \subseteq R_1$) é uma chave estrangeira de R_1 que referencia R_2, se e somente se:

1. FK tem os mesmo domínio (ou domínios) dos atributos que FK referencia em R_2. É dito que FK referencia ou refere-se à relação R_2.

2. FK na tupla t_1 do estado corrente $r_1(R_1)$ ou

   (a) Casa com um conjunto de atributos PK (possivelmente a chave primária) de alguma tupla t' de um estado corrente $r_2(R_2)$ ou

   (b) É nulo.

No nosso exemplo de carro e proprietário, podemos retirar as seguintes informações quanto às restrições:

- *placa* e *cpf* são chaves primárias das tabelas (relações) *Carro* e *Proprietario*, respectivamente.

- O atributo *prop* é uma chave estrangeira em *Carro* que aponta para o atributo *cpf* de *Proprietario*. Dois pontos importantes: (i) o atributo *prop* só aceitará um valor pré-existente em *cpf* da tabela *Proprietario* e (ii) geralmente, o atributo referenciado deve ser uma chave na tabela referenciada, no nosso caso, *cpf* e chave primária de *Proprietario*.

Exemplo:
Dados esquemas de relação $Aluno(mat, nome, ender, codcurso)$ e $Curso(cod, nome, carga)$ sendo *mat* e *cod* as chaves primárias de *Aluno* e *Curso*, respectivamente, e *codcurso* a chave estrangeira em *Aluno* que referencia *cod* em *Curso*. Abaixo dois exemplos de estados do banco de dados dessas relações.

Instância válida

Aluno

mat	nome	ender	codcurso
0123	João da Silva	R. XII, 10	5
0124	Maria Ia	R. Xis, 110	1
0130	Julieta	R. Poá, 40	5
0142	Pedro	R. Pia, 320	5

Curso

cod	nome	carga
1	Corte e Costura	2900
5	Ballet Clássico	1000

Instância inválida

Aluno

mat	nome	ender	codcurso
0123	João da Silva	R. XII, 10	5
0124	Maria Ia	R. Xis, 110	1
0130	Julieta	R. Poá, 40	**15**
0142	Pedro	R. Pia, 320	5

Curso

cod	nome	carga
1	Corte e Costura	2900
5	Ballet Clássico	1000

O banco de dados do segundo exemplo possui uma instância inválida pois *codcurso* que é uma chave estrangeira de *Aluno* que aponta para *cod* em $Curso$ possui um valor inexistente, isto é, 15 não existe no atributo *cod* de $Curso$.

Notação utilizada em sala de aula para representar os esquemas de relação:

- Atributos obrigatórios: coloca-se apenas o nome sem sinais particulares
- Atributos opcionais: sublinha-se com uma linha tracejada.
- Chave primária: sublinha-se com uma linha contínua.
- Chave: coloca-se um asterisco após o nome do atributo.
- chave estrangeira: colocar, entre parênteses, a relação referenciada. Vamos considerar que o atributo referenciado é a chave primária da tabela colocada entre parênteses.

Acertando a nomeclatura das relações *Aluno* e $Curso$ conforme as regras acima:

$$Aluno(\underline{mat}, nome, email^*, \underline{ender}, codcurso(Curso))$$

$$Curso(\underline{cod}, nome, carga)$$

Comentários: os atributos *mat* e *cod* são chaves primárias de suas tabelas, o atributo *ender* é opcional (é o único os outros todos são obrigatórios), ou seja, aceita ficar vazio para uma determinada tupla, o atributo *codcurso* é uma chave estrangeira que aponta para *cod* da tabela *Curso*. Finalmente, o atributo *email* é uma chave que não foi escolhida para ser primária, mas não pode ter seus valores repetidos.