Resolução dos exercícios da lista BD05

Pedro Losco Takecian
Monitor

João Eduardo Ferreira
Professor

5 de abril de 2006

1. O que é união compatível? Por que as operações UNION, INTERSECTION e DIFFERENCE são operações que necessitam que as relações sejam união compatível?

   "União compatível"é uma condição sobre os tipos de tuplas de duas relações. Duas relações $R(A_1, A_2, \ldots, A_n)$ e $S(B_1, B_2, \ldots, B_n)$ são consideradas "união compatível"se:

   - possuírem o mesmo grau n;
   - para $1 \leq i \leq n$, temos $dom(A_i) = dom(B_i)$.

   Assim, as tuplas das duas relações devem possuir o mesmo número de atributos e, para cada par de atributos correspondentes, devemos ter o mesmo domínio.

   As operações binárias UNION, INTERSECTION e DIFFERENCE são operações conhecidas da teoria dos conjuntos e foram definidas sobre dois conjuntos que possuem o mesmo tipo de elemento. Assim, estas operações somente fazem sentido se as duas relações tiverem o mesmo tipo de tupla e, para isto, a condição acima deve ser satisfeita.

2. Discuta algumas das consultas nas quais seja necessário renomear atributos a fim de especificar consultas não ambíguas.

   Tomemos como exemplo as seguintes relações (simplificadas):

EMPREGADO			
codEmp	nome	endereço	cidade

PROJETO		
codProj	nome	cdEmp

   Supondo que exista um relacionamento "trabalha_em"com cardinalidade 1:n (cada projeto tem apenas um empregado), podemos querer achar o nome do projeto no qual determinado empregado de nome 'Eduardo Lima' trabalha. Temos então:

   $\text{EMP\_PROJ} \leftarrow \text{EMPREGADO} \bowtie_{\text{codEmp=cdEmp}} \text{PROJETO}$

   $\text{EMP} \leftarrow \Pi_{\text{NOME}}(\sigma_{\text{NOME='Eduardo Lima'}}(\text{EMP\_PROJ}))$

   Percebe-se que EMP_PROJ possui dois campos "nome", o que inviabiliza a seleção feita pelo nome 'Eduardo Lima'. Isto ocorre pois nada nos garante que a busca será feita no atributo de EMPREGADO e não no atributo de PROJETO. Além desta confusão, neste mesmo exemplo, vemos outra operação que torna-se impossível devido à ambigüidade de nomes: a projeção. Nela, não é possível saber se queremos obter o nome do empregado ou o nome do projeto como resultado da operação.

3. Discuta os vários tipos de operações JOIN. Resuma em forma de tabela que contenha o nome da operação, o propósito da operação e a notação.

   A operação JOIN, de maneira geral, é utilizada para combinar tuplas relacionadas de duas relações em uma única tupla, nos permitindo, assim, processar relacionamentos entre relações. No JOIN, somente as combinações de tuplas que satisfazem a *condição de junção* (quando houver uma) aparecem na relação resultante.

   Uma operação JOIN com uma condição de junção de caráter geral é chamada de THETA JOIN. Nela, tuplas cujos valores de atributos de junção sejam nulos não aparecem no resultado. Devido a este fato, as informações das relações participantes podem não ser preservadas em sua totalidade.

   Quando o único operador de junção utilizado é o "=", este JOIN é chamado de EQUIJOIN. No resultado de um EQUIJOIN, sempre temos um ou mais pares de atributos que possuem valores idênticos em cada tupla, devido à condição de igualdade especificada. Notando a redundância existente nestes pares, criou-se o NATURAL JOIN, que retira os atributos supérfluos da relação resultante (desde que estes tenham o mesmo nome nas duas relações).

Caso não exista nenhuma condição de junção especificada, temos outro tipo de join, o CROSS JOIN. Ele retorna as combinações entre todas as tuplas das duas relações, assim como ocorre com um produto cartesiano.

Existe ainda, como extensão da operação JOIN, o OUTER JOIN. Ele surgiu da necessidade de se preservar todas as tuplas, mesmo que não produzam uma combinação com as tuplas da outra relação. Assim, as tuplas que não têm correspondentes são mantidas no resultado e os atributos ausentes têm seus valores preenchidos com "nulls". Conforme esperado, as tuplas das relações componentes com a mesma chave são representadas apenas uma vez no resultado, refletindo os valores de atributos das relações de origem. Dependendo das relações cujas informações se quer manter, pode-se aplicar um RIGHT OUTER JOIN (direita da operação), um LEFT OUTER JOIN (esquerda) ou ainda, um FULL OUTER JOIN (ambas).

Podemos resumir assim:

Nome	**Propósito**	**Notação (exemplo)**
THETA JOIN	junção com uma condição geral	$A \bowtie_{x \neq y} B$
EQUIJOIN	junção com uma condição de igualdade	$A \bowtie_{x=y} B$
NATURAL JOIN	EQUIJOIN com a retirada de atributos redundantes	$A * B$
CROSS JOIN	junção sem condição especificada (produto cartesiano)	$A \times B$
LEFT OUTER JOIN	junção com inclusão de todas as tuplas da relação da esquerda	$A \supset\bowtie_{x=y} B$
RIGHT OUTER JOIN	junção com inclusão de todas as tuplas da relação da direita	$A \bowtie\subset_{x=y} B$
FULL OUTER JOIN	junção com inclusão de todas as tuplas	$A \supset\bowtie\subset_{x=y} B$

4. Especifique as seguintes consultas sobre a base de dados mostrada abaixo (Figura 5.3 da apostila) usando operações relacionais discutidas neste capítulo.
Obs: Para simplificação das consultas, o atributo 'PNOME' da relação 'PROJETO' está renomeado aqui para 'PJNOME'

EMPREGADO									
pnome	mnome	snome	nss	datanasc	endereco	sexo	salario	nsssuper	ndep

PROJETO			
pjnome	pnúmero	plocalização	dnum

DEPARTAMENTO			
dnome	dnúmero	nssger	datinicger

LOCAIS_DEPTO	
dnúmero	dlocalização

TRABALHA_EM		
nssemp	pnro	horas

DEPENDENTE				
nssemp	nomedependente	sexo	dataaniv	relação

(a) Recuperar os nomes de empregados do departamento 5 que trabalham mais que 10 horas no projeto 'ProdutoX'.

EMPS_DEPT_5 $\leftarrow \sigma_{\text{NDEP=5}}$(EMPREGADO)
EMPS5_TRABALHA $\leftarrow$ EMPS_DEPT_5 $\bowtie_{\text{NSS=NSSEMP}}$ TRABALHA_EM
EMPS5_PROJETO $\leftarrow$ EMPS5_TRABALHA $\bowtie_{\text{PNRO=PNUMERO}}$ PROJETO
EMPS $\leftarrow \sigma_{\text{HORAS>10 AND PJNOME='ProdutoX'}}$(EMPS5_PROJETO)
RESULT $\leftarrow \Pi_{\text{PNOME,MNOME,SNOME}}$(EMPS)

(b) Listar os nomes dos empregados que tenham um dependente com o mesmo nome (PNOME).

EMPS_DEPENDS $\leftarrow$ EMPREGADO $\bowtie_{\text{NSS=NSSEMP}}$ DEPENDENTE
EMPS_MESMONOME $\leftarrow \sigma_{\text{PNOME=NOMEDEPENDENTE}}$(EMPS_DEPENDS)
RESULT $\leftarrow \Pi_{\text{PNOME,MNOME,SNOME}}$(EMPS_MESMONOME)

(c) Encontrar os nomes de empregados que são diretamente supervisionados por 'Franklin Wong'.

SUPER(SPNOME,SMNOME,SSNOME,SNSS) $\leftarrow \Pi_{\text{PNOME,MNOME,SNOME,NSS}}$(EMPREGADO)
EMPS_SUPER $\leftarrow$ EMPREGADO $\bowtie_{\text{NSSSUPER=SNSS}}$ SUPER
EMPS_FRANK $\leftarrow \sigma_{\text{SPNOME='Franklin' AND SSNOME='Wong'}}$(EMPS_SUPER)
RESULT $\leftarrow \Pi_{\text{PNOME,MNOME,SNOME}}$(EMPS_FRANK)

(d) Para cada projeto, listar o nome do projeto e o total de horas (de todos os empregados) gastos em cada projeto.

PROJ_TRABALHA ← PROJETO $\bowtie_{\text{PNUMERO=PNRO}}$ TRABALHA_EM
RESULT(PNOME,SOMAHORAS)← $_{\text{PJNOME}}\Im_{\text{SUM HORAS}}$(PROJ_TRABALHA)

(e) Recuperar os nomes dos empregados que trabalham em todos os projetos.

PROJETOS ← Π_{PNUMERO}(PROJETO)
EMPS_TRAB ← EMPREGADO $\bowtie_{\text{NSS=NSSEMP}}$ TRABALHA_EM
EMPS_TRABALHA(PNOME,MNOME,SNOME,NSS,PNUMERO) ← $\Pi_{\text{PNOME,MNOME,SNOME,NSS,PNRO}}$(EMPS_TRAB)
RESULT ← $\Pi_{\text{PNOME,MNOME,SNOME}}$(EMPS_TRABALHA ÷ PROJETOS)

(f) Recuperar os nomes dos empregados que não trabalham em quaisquer projetos.

TODOSEMPS ← $\Pi_{\text{PNOME,MNOME,SNOME,NSS}}$(EMPREGADO)
EMPS_PROJETO ← $\Pi_{\text{PNOME,MNOME,SNOME,NSS}}$(EMPREGADO $\bowtie_{\text{NSS=NSSEMP}}$ TRABALHA_EM)
EMPS_NENHUM ← TODOSEMPS − EMPS_PROJETO
RESULT ← $\Pi_{\text{PNOME,MNOME,SNOME}}$(EMPS_NENHUM)

(g) Para cada departamento, recuperar o nome do departamento e a média salarial dos empregados que trabalham no departamento.

DEPT_PROJ ← DEPARTAMENTO $\bowtie_{\text{DNUMERO=DNUM}}$ PROJETO
DEPT_PROJ_TRABALHA ← DEPT_PROJ $\bowtie_{\text{PNUMERO=PNRO}}$ TRABALHA_EM
DEPT_PROJ_TRAB_EMP ← DEPT_PROJ_TRABALHA $\bowtie_{\text{NSSEMP=NSS}}$ EMPREGADO
RESULT(DNOME,MEDIASALARIAL)← $_{\text{DNOME}}\Im_{\text{AVERAGE SALARIO}}$(DEPT_PROJ_TRAB_EMP)

(h) Recuperar a média salarial de todos os empregados femininos.

EMPS_FEM ← $\sigma_{\text{SEXO='F'}}$(EMPREGADO)
RESULT(MEDIA)← $\Im_{\text{AVERAGE SALARIO}}$(EMPS_FEM)

(i) Encontrar os nomes e endereços de empregados que trabalham em ao menos um projeto localizado em Houston mas cujo departamento não possua localização em Houston.

PROJS_HOUSTON ← $\sigma_{\text{PLOCALIZAÇÃO='Houston'}}$(PROJETO)
PROJS_HOUSTON_DEPT ← PROJS_HOUSTON $\bowtie_{\text{DNUM=DNUMERO}}$ LOCAIS_DEPTO
PROJS_MESMOLOCAL ← $\sigma_{\text{DLOCALIZAÇÃO=PLOCALIZAÇÃO}}$(PROJS_HOUSTON_DEPT)
PROJS ← PROJS_HOUSTON - PROJS_MESMOLOCAL
TRAB_PROJS ← TRABALHA_EM $\bowtie_{\text{PNRO=PNUMERO}}$ PROJS
EMPS ← EMPREGADO $\bowtie_{\text{NSS=NSSEMP}}$ TRAB_PROJS
RESULT ← $\Pi_{\text{PNOME,MNOME,SNOME,ENDEREÇO}}$(EMPS)

(j) Listar os sobrenomes dos gerentes de departamentos que não tenham dependentes.

GER(NSSEMP) ← Π_{NSSGER}(DEPARTAMENTO)
EMPSCOMDEPENDENTES ← Π_{NSSEMP}(DEPENDENTE)
GERSEMDEP ← GER − EMPSCOMDEPENDENTES
GERENTES ← EMPREGADO $\bowtie_{\text{NSS=NSSEMP}}$ GERSEMDEP
RESULT ← Π_{SNOME}(GERENTES)

(k) Generalize a consulta (i) acima para listar os nomes e endereços de empregados que trabalham em um projeto em alguma cidade, mas que o departamento não tenha nenhuma localização nessa cidade.

PROJ_DEPT ← PROJETO $\bowtie_{\text{DNUM=DNUMERO}}$ LOCAIS_DEPTO
PROJS_MESMOLOCAL ← $\sigma_{\text{DLOCALIZAÇÃO=PLOCALIZAÇÃO}}$(PROJ_DEPT)
PROJS_TOTAL ← Π_{PNUMERO}(PROJETO)
PROJS ← PROJS_TOTAL - PROJS_MESMOLOCAL
TRAB_PROJS ← TRABALHA_EM $\bowtie_{\text{PNRO=PNUMERO}}$ PROJS
EMPS ← EMPREGADO $\bowtie_{\text{NSS=NSSEMP}}$ TRAB_PROJS
RESULT ← $\Pi_{\text{PNOME,MNOME,SNOME,ENDEREÇO}}$(EMPS)