Relações Derivadas

A SQL-92 permite o uso de uma expressão de subconsulta na cláusula *from*. A relação resultante deve receber um nome e os atributos precisam ser rebatizados.

"Gerar uma subconsulta com uma relação de nomes de todas as agências e suas médias de saldos em contas correspondentes":

```
(select        nome_agência, avg(saldo)
 from          depositante
 group by      nome_agência)
as resultado   (nome_agência, saldo_médio)
```

"Encontre a média dos saldos das agências em que a média dos saldos em conta é maior que $ 1.200":

```
select   nome_agência, saldo_médio
from     (select nome_agência, avg(saldo)
          from depositante
          group by nome_agência)
         as resultado (nome_agência, saldo_médio)
where    saldo-médio > 1200
```

Visões

Definimos uma visão em SQL usando o comando *create view*. Para definir a visão, precisamos dar-lhe um nome e definir a consulta que prorcessará essa visão. A forma do comando *create view* é:

```
create view v as <expressão da consulta>
```

Como exemplo, consideramos uma visão composta dos nomes de agências e nomes de clientes que tenham uma conta ou um empréstimo na agência:

```
create view   todos_clientes as
              (select     nome_agência, nome_cliente
               from       depositante, conta
               where      depositante.nro_conta =
                          conta.nro_conta)
              union
              (select     nome_agência, nome_cliente
               from       devedor, empréstimo
               where      devedor.nro_empréstimo =
                          empréstimo.nro_empréstimo)
```

Os nomes dos atributos de visão devem ser especificados explicitamente, conforme segue:

```
create view   empréstimo_total_agência
              (nome_agência, empréstimo_total) as
   select     nome_agência, sum(saldo)
   from       empréstimo
   group by   nome_agência
```

A visão anterior cede para cada agência a soma dos totais de todos os empréstimos da agência. Uma vez que a expressão soma(total) não possui um nome, o nome do atributo é especificado explicitamente na definição da visão. Os nomes de visão podem aparecer em qualquer lugar onde o nome de uma relação aparece. Usando

a visão todos_clientes, podemos encontrar todos os clientes da agência Lages, escrevendo:

```
select   nome_cliente
from     todos_clientes
where    nome_agência = "Lages"
```

As definições de visões ficam no banco de dados até que um comando *drop view* nome_visão seja executado.

Composição de Relações

Junções tomam duas relaçòes e têm como resultado uma outra relação.

As operações de junção de exemplo usarão as relações empréstimo e devedor, ilustradas abaixo:

nome_agência	*nro_empréstimo*	*total*
Lages	L-170	3000
Blumenau	L-230	4000
Tubarão	L-260	1700

empréstimo

nome-cliente	*nro_empréstimo*
João	L-170
Paulo	L-230
Cláudio	L-155

devedor

Inicialmente, um exemplo de junções internas:

```
empréstimo inner join devedor on empréstimo.nro_empréstimo =
                                 devedor.nro_empréstimo
```

Resultando:

nome_agência	*nro_empréstimo*	*total*	*nome_cliente*	*nro_empréstimo*
Lages	L-170	3000	João	L-170
Blumenau	L-230	4000	Paulo	L-230

O atributo nro_empréstimo aparece duas vezes na tabela resultante, sendo uma correspondente à relação empréstimo e outra à relação devedor. Para rebatizar o resultado da relação de uma junção e os atributos, usamos a cláusula as:

```
empréstimo inner join devedor on empréstimo.nro_empréstimo =
                                 devedor.nro_empréstimo as
        lb (agência, nro_empréstimo, total, cliente, num_emp)
```

Um exemplo do uso da operação de junção *left outer join*:

```
empréstimo left outer join devedor on empréstimo.nro_empréstimo =
                                      devedor.nro_empréstimo
```

O *left outer join* é processado como segue. Primeiro, o resultado da junção interna é processado conforme mostrado anteriormente. Então, para toda tupla T da relação empréstimo do lado esquerdo que não apresente correspondência com nenhuma tupla da relação devedor do lado direito da junção interna, uma tupla R é adicionada ao resultado da junção, da maneira como será descrita. Os atributos da tabela R que são derivados da relação do lado esquerdo são preenchidos pelos valores da tupla T, e os restantes são preenchidos com valores nulos. A relação resultante é mostrada abaixo:

nome_agência	*nro_empréstimo*	*total*	*nome_cliente*	*nro_empréstimo*
Lages	L-170	3000	João	L-170
Blumenau	L-230	4000	Paulo	L-230
Tubarão	L-260	1700	nulo	nulo

Um exemplo de uso da operação *natural join*:
`empréstimo` *natural inner join* `devedor`

Essa expressão processa a junção natural de duas relações. O único atributo de nome comum às duas relações `empréstimo` e `devedor` é `nro_empréstimo`. O resultado da expressão é mostrado abaixo:

nome_agência	*nro_empréstimo*	*total*	*nome_cliente*
Lages	L-170	3000	João
Blumenau	L-230	4000	Paulo

O atributo `nro_empréstimo` aparece somente uma vez no resultado da junção *natural*, embora apareça duas vezes na junção com a condição *on*.

Tipos de Junções e Condições

Cada uma das variantes das operações de junção em SQL-92 consiste em um tipo de junção e em uma condição de junção. As condições de junção definem quais tuplas das duas relações apresentam correspondência e quais atributos são apresentados no resultado de uma junção. O tipo de junção define como as tuplas em cada relação que não possuem nenhuma correspondência (baseado na condição de junção) com as tuplas da outra relação devem ser tratadas. Abaixo, os tipos de junções e condições de junção:

Tipos de junção	Condições de junção
inner join	*natural*
left outer join	*on* <predicado>
right outer join	*using* (A_1, A_2, ..., A_n)
full outer join	

O uso de uma condição de junção é obrigatório para junções externas, mas opcional para junções internas. A condição de junção *natural* tem a ordem dos atributos no resultado da junção na seguinte maneira: primeiro, os atributos comuns a ambas as relações, depois os atributos para os quais não há correspondência aos da relação do lado esquerdo e, por último, os atributos sem correspondência aos da relação do lado direito.

O tipo de junção *right outer join* é simétrico a *left outer join*. As tuplas da relação do lado direito que não correspondem a nenhuma tupla da relação do lado esquerdo são preenchidos com nulos e adicionados ao resultado da junção externa direita.

A expressão seguinte é um exemplo da combinação dos tipos de junção *natural* e *right outer join*:
`empréstimo` *natural right outer join* `devedor`

Resultado:

nome_agência	*nro_empréstimo*	*total*	*nome_cliente*
Lages	L-170	3000	João
Blumenau	L-230	4000	Paulo
nulo	L-155	nulo	Cláudio

A condição de junção *using* $(A_1, A_2, \ldots, A_n)$ é similar a *natural*, exceto pelo fato de que seus atributos de junção são os atributos $(A_1, A_2, \ldots, A_n)$ em vez de todos os atributos comuns a ambas as relações. Os atributos $(A_1, A_2, \ldots, A_n)$ devem ser somente os atributos comuns a ambas as relações e eles aparecem apenas uma vez no resultado da junção.

O tipo *full outer join* é uma combinação dos tipos de junções externas à esquerda e à direita. Depois que o resultado de uma junção interna é processado, as tuplas da relação do lado esquerdo que não correspondem a nenhuma das tuplas do lado direito são preenchidas com valores nulos e, depois, adicionadas ao resultado. Similarmente, as tuplas da relação do lado direito que não coincidem com nenhuma das tuplas da relação do lado esquerdo são também preenchidos com nulos e adicionadas ao resultado. Por exemplo, o resultado da expressão:

```
empréstimo full outer join devedor using (nro_empréstimo)
```

nome_agência	*nro_empréstimo*	*total*	*nome-cliente*
Lages	L-170	3000	João
Blumenau	L-230	4000	Paulo
Tubarão	L-260	1700	nulo
nulo	L-155	nulo	Cláudio

Como outro exemplo de uso da operação de junção externa, a consulta "encontre todos os clientes que tenham uma conta, mas nenhum empréstimo no banco" pode ser escrita:

```
select   d-CN
from     (depositante left outer join devedor
           on depositante.nome_cliente = devedor.nome_cliente)
         as db1 (d-CN, nro_conta, b-CN, nro_empréstimo)
where    b-CN is null
```

Similarmente, a consulta "encontre todos os clientes que tenham uma conta ou um empréstimo (mas não os dois) no banco" pode ser escrita usando junção externa *natural total*, como:

```
select   nome_cliente
from     (depositante natural full outer join devedor)
where    nro_conta is null or nro_empréstimo is null
```

A SQL-92 também oferece dois outros tipos de junções, denominados *cross join* e *union join*. O primeiro é equivalente a uma junção interna sem uma condição de junção; o segundo é equivalente a uma junção externa total na condição de "falso" – em que a junção interna é vazia.