[MAC0313] Introdução aos Sistemas de Bancos de Dados

Aula 18
Linguagem SQL (Parte 4)

Consultas Envolvendo Agrupamento/Agregação e Junções

17 de outubro de 2017

Profª Kelly Rosa Braghetto

(Adaptação dos slides do prof. Jeffrey Ullman, da *Stanford University*)

Agregações

- **SUM**, **AVG**, **COUNT**, **MIN** e **MAX** podem ser aplicados a uma coluna na cláusula SELECT para produzir a agregação da referida coluna.
- Além disso, **COUNT(*)** conta o número de tuplas.

Exemplo: Agregação

- A partir de

  Venda(nome_lanch, nome_refri, preço),

  encontre os preços médio e máximo da Fanfa:

```
SELECT AVG(preço), MAX(preço)
FROM Venda
WHERE nome_refri = 'Fanfa';
```

Agrupamento

- Depois de uma expressão SELECT-FROM-WHERE, podemos adicionar GROUP BY e uma lista de atributos.
- A relação resultante do SELECT-FROM-WHERE é agrupada de acordo com os valores de todos os referidos atributos e qualquer agregação é aplicada somente dentro de cada grupo.

Exemplo: agrupamento

- A partir de

  Venda(nome_lanch, nome_refri, preço), encontre o preço médio de cada refri:

```
SELECT nome_refri, AVG(preço) as média
FROM Venda
GROUP BY nome_refri;
```

nome_refri	média
Fanfa	2.33
...	...

Exemplo: agrupamento

- A partir de Venda(nome_lanch, nome_refri, preço) e Frequentador(nome_cliente, nome_lanch), encontre, para cada cliente, o preço médio da Fanfa nas lanchonetes que ele frequenta :

```
SELECT nome_cliente, AVG(preço)
FROM Frequentador, Venda
WHERE nome_refri = 'Fanfa' AND
Frequentador.nome_lanch =
                    Venda.nome_lanch
GROUP BY nome_cliente;
```

Computa todas as tuplas cliente-lanch-preço para Fanfa.

Depois, as agrupa pelo cliente.

Restrição no SELECT: listas com agregação

- Se um agrupamento é usado, então cada elemento da lista do SELECT precisa ser:
  1. Uma agregação, ou
  2. Um atributo da lista do GROUP BY.

Exemplo de consulta incorreta

- Alguém pode pensar que é possível encontrar a lanchonete que vende Fanfa mais barato usando:

```
SELECT nome_lanch, MIN(preço)
FROM Venda
WHERE nome_refri = 'Fanfa';
```

- Mas essa consulta **NÃO** é permitida em SQL.

Cláusulas HAVING

- HAVING <condição> pode aparecer depois da cláusula GROUP BY
- Se aparecer, a condição é aplicada sobre cada grupo. Grupos que não satisfazem a condição são eliminados da resposta da consulta

Exemplo: HAVING

- A partir de
  Venda(nome_lanch, nome_refri, preço) e
  Refrigerante(nome, fabricante),
  encontre o preço médio dos refris que são servidos em pelo menos 3 lanchonetes ou que são fabricados pela Cola-Coca.

Solução

```
SELECT nome_refri, AVG(preço)
FROM Venda
GROUP BY nome_refri
HAVING COUNT(nome_lanch) >= 3   OR
nome_refri  IN
      (SELECT nome
       FROM Refrigerantes
       WHERE fabricante =  'Cola-Coca');
```

Grupos de refri com pelo menos 3 lanchonetes não nulas e também grupos em que o fabricante é a Cola-Coca.

Refris fabricados pela Cola-Coca.

Requisitos para as condições do HAVING

- Vale qualquer coisa dentro de uma subconsulta
- Fora de subconsultas, o HAVING pode referenciar um elemento somente se ele for:
  1. Um atributo agrupador, ou
  2. Uma agregação

  (essa é a mesma condição usada para cláusulas SELECT com agregação)

Expressões de Junção (JOIN)

- SQL possui várias versões de junções
  - Mas é sempre possível obter o mesmo efeito delas por meio de uma consulta do tipo SELECT-FROM-WHERE
- A expressões JOIN podem ser usadas no lugar de relações em uma cláusula FROM.

Produto Cartesiano

- É o tipo de junção mais simples:

  SELECT * FROM **R CROSS JOIN S**;

- As relações envolvidas no produto também podem ser subconsultas parentizadas (isso vale para todos os tipos de JOIN)
- O produto cartesiano sozinho raramente é útil

Junção Natural

- Forma da junção natural: **R NATURAL JOIN S**
- A condição de junção é a igualdade sobre os pares de atributos das duas relações que possuem o mesmo nome

  Venda(nome_lanch, nome_refri, preço)

  Apreciador(nome_cliente, nome_refri)
- Exemplo:

```
SELECT * FROM Apreciador NATURAL JOIN Venda;
```

- Equivale a:

```
SELECT A.nome_cliente, V.*
FROM Apreciador A, Venda V
WHERE A.nome_refri = Venda.nome_refri
```

Junção Teta

- **R JOIN S ON <condição>**
- Exemplo: usando Cliente(nome, endereço) e Frequentador(nome_cliente, nome_lanch):

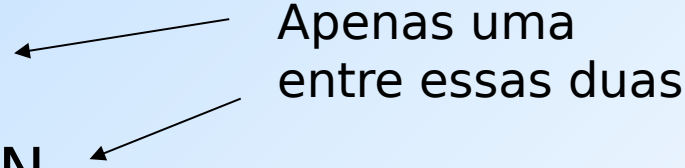
```
SELECT *
FROM Cliente JOIN Frequentador ON
    nome = nome_cliente;
```

nos dá todas quádruplas (c, *e*, *c*, *l*) tais que cliente ***c*** mora no endereço ***e*** e frequenta a lanchonete ***l***.

Junção Externa (Outer Join)

- **R OUTER JOIN S** é o núcleo de uma expressão de junção externa. Ele pode ser modificado por três cláusulas opcionais:

1. **NATURAL** antes de OUTER
2. **ON** <condição> depois de JOIN
3. **LEFT**, **RIGHT**, ou **FULL** antes de OUTER
   - LEFT = inclui apenas as tuplas soltas de R
   - RIGHT = inclui apenas as tuplas soltas de S
   - FULL = inclui as tuplas soltas de ambas



Exemplo:

SELECT * FROM R LEFT OUTER JOIN S ON (B=D AND C=E);

Exemplo: $R \ltimes_{B=D,C=E} S$

A	B	C
1	2	3
4	5	6
7	8	9

Relação R

D	E	F
2	3	10
2	3	11
6	7	12

Relação S

A	B	C	D	E	F
1	2	3	2	3	10
1	2	3	2	3	11
4	5	6	NULL	NULL	NULL
7	8	9	NULL	NULL	NULL

Resultado de $R \ltimes_{B=D,C=E} S$

Exemplo:

SELECT * FROM R RIGHT OUTER JOIN S ON (B=D AND C=E);

Exemplo: $R \rtimes_{B=D,C=E} S$

A	B	C
1	2	3
4	5	6
7	8	9

Relação R

D	E	F
2	3	10
2	3	11
6	7	12

Relação S

A	B	C	D	E	F
1	2	3	2	3	10
1	2	3	2	3	11
NULL	NULL	NULL	6	7	12

Resultado de $R \rtimes_{B=D,C=E} S$

Exemplo:

SELECT * FROM R FULL OUTER JOIN S ON (B=D AND C=E);

Exemplo: $R \bowtie_{B=D,C=E} S$

A	B	C
1	2	3
4	5	6
7	8	9

Relação R

D	E	F
2	3	10
2	3	11
6	7	12

Relação S

A	B	C	D	E	F
1	2	3	2	3	10
1	2	3	2	3	11
4	5	6	NULL	NULL	NULL
7	8	9	NULL	NULL	NULL
NULL	NULL	NULL	6	7	12

Resultado de $R \bowtie_{B=D,C=E} S$

Exemplo:

SELECT * FROM R NATURAL LEFT OUTER JOIN S ;

A	B	C
1	2	3
4	5	6
7	8	9

Relação R

B	C	D
2	3	10
2	3	11
6	7	12

Relação S

A	B	C	D
1	2	3	10
1	2	3	11
4	5	6	NULL
7	8	9	NULL

Resultado da junção natural externa à esquerda

Exemplo:

SELECT * FROM R NATURAL RIGHT OUTER JOIN S ;

A	B	C
1	2	3
4	5	6
7	8	9

Relação R

B	C	D
2	3	10
2	3	11
6	7	12

Relação S

A	B	C	D
1	2	3	10
1	2	3	11
NULL	6	7	12

Resultado da junção natural externa à direita

Exemplo:

SELECT * FROM R NATURAL FULL OUTER JOIN S ;

A	B	C
1	2	3
4	5	6
7	8	9

Relação R

B	C	D
2	3	10
2	3	11
6	7	12

Relação S

A	B	C	D
1	2	3	10
1	2	3	11
4	5	6	NULL
7	8	9	NULL
NULL	6	7	12

Resultado da junção natural externa completa

Exemplo: junção externa

- Exemplo: usando Cliente(nome, endereço) e Frequentador(nome_cliente, nome_lanch)

```
SELECT *
FROM Cliente LEFT OUTER JOIN Frequentador
                ON nome = nome_cliente;
```

nos dá uma lista de tuplas, onde cada tupla associa os dados de um cliente a um nome de lanchonete que ele frequenta. Se um cliente não frequenta nenhuma lanchonete, seus dados aparecerão na lista associados ao valor NULL (para o nome de lanchonete).

Exercícios

Aluno(<u>nroAluno</u>, nomeAluno, formação, nivel, idade)

Curso(<u>nome</u>, horario, sala, idProf)

Matriculado(<u>nroAluno, nomeCurso</u>)

Professor(<u>idProf</u>, nomeProf, idDepto)

7) Encontre todos os nomes de pessoas que aparecem no BD.

9) Para cada valor de nível que aparece em Aluno, imprima o nível e idade média dos alunos desse nível.

10) Para cada valor de nível que aparece em Aluno exceto os níveis que possuem menos de 6 alunos, imprima o nível e idade média dos alunos desse nível.

11) Encontre os nomes dos professores para os quais a quantidade de alunos na lista de matriculados de ao menos um dos cursos que eles ministram é menor do que 5.

Exercícios

Aluno(<u>nroAluno</u>, nomeAluno, formação, nivel, idade)

Curso(<u>nome</u>, horario, sala, idProf)

Matriculado(<u>nroAluno, nomeCurso</u>)

Professor(<u>idProf</u>, nomeProf, idDepto)

12) Para cada professor que ministra cursos apenas na sala *R128*, imprima seu nome e o número total de cursos que ele ou ela ministra.

13) Encontre os nomes dos alunos matriculados no número máximo de cursos.

14) Para cada valor de idade que aparece em Aluno, encontre o valor do nível que aparece com mais frequência. Por exemplo, se houver mais alunos no nível *FR* com idade 18 do que os alunos com idade 18 dos níveis *SR*, *JR* ou *SO*, você deve imprimir o par (18,*FR*).

15) Liste todos os pares (*nomeAluno*, *nomeCurso*) onde *nomeCurso* corresponde a um curso no qual o aluno *nomeAluno* está matriculado. Um aluno tem que aparecer na listagem mesmo se não tiver nenhuma matrícula (nesse caso, ele deve aparecer com NULL em *nomeCurso*).

Referências bibliográficas

- *A First Course in Database Systems,*
  Ullman e Widom. 1997.
  Capítulo 5
- *Database Systems – The Complete Book*, Garcia-Molina, Ullman e Widom. 2002.
  Capítulo 6
- *Sistemas de Bancos de Dados* (6ª edição), Elmasri e Navathe. 2010.
  Capítulos 4 e 5