[MAC0426] Sistemas de Bancos de Dados
[IBI5013] Bancos de Dados para Bioinformática

Aula 17
Linguagem SQL (Parte 4)

Consultas com Agrupamento/Agregação e Junções
e
Comandos para a Modificação de Dados

29 de maio de 2017

Profa. Kelly Rosa Braghetto

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

Álgebra Relacional Estendida

$\mathcal{F}$ = agrupamento e agregação

e

Junções Externas (*Outer joins*):
mantém no resultado "tuplas soltas" → tuplas que não se "juntam" a nada

Funções de agregação

- Funções de agregação não são operadores da álgebra relacional clássica
- Elas se aplicam a colunas inteiras de uma relação e produzem um único resultado
- Os exemplos mais importantes são: **SOMA (SUM)**, **MÉDIA (AVG)**, **CONT(COUNT)**, **MIN** e **MAX**

Exemplo: agregação

R =

(A	B)
1	3
3	4
3	2

SUM(A) = 7
COUNT(A) = 3
MAX(B) = 4
AVG(B) = 3

Operador de agrupamento

- R1 $\leftarrow {}_{L}\mathcal{F}_{A}$ (R2)

  onde:

  - *L* é uma lista de atributos individuais (agrupadores) de R2
  - *A* é uma lista de operadores de agregação aplicados a atributos em *R2*

Aplicando ${}_{L}\mathcal{F}_{A}(R)$

- Agrupa *R* segundo todos atributos agrupadores em *L*
  - Ou seja: forma um grupo para cada combinação de valor distinta para esses atributos em *R*
- Dentro de cada grupo, computa *A*
- O resultado tem uma tupla para cada grupo, contendo:
  1. Os atributos em L
  2. E as agregações dos grupos

Exemplo: agrupamento/agregação

R =

(A	B	C)
1	2	3
4	5	6
1	2	5

$_{A,B}\mathcal{F}_{AVG(C)}$ (R) = ??

Primeiro, agrupa *R por A* e *B* :

A	B	C
1	2	3
1	2	5
4	5	6

Então, calcula a média de *C* dentro dos dois grupos:

A	B	AVG_C
1	2	4
4	5	6

Junção externa

- Considere a junção $R \bowtie_C S$.
- Uma tupla de *R* que não possui uma tupla em S para realizar a junção é chamada de *solta*.
  - O mesmo vale para uma tupla de *S*.
- Uma **junção externa** preserva as tuplas soltas, “complementando-as” com NULL.

Exemplos junção externa

R =

(A	B)
1	2
4	5

S =

(B	C)
2	3
6	7

Em uma junção natural, (1,2) junta com (2,3), mas as duas outras tuplas são “soltas”.

R Junção Externa S =

A	B	C
1	2	3
4	5	NULL
NULL	6	7

Agora, de volta ao SQL

Cada uma dessas operações tem um comando equivalente em SQL

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 o preço médio de Fanfa:

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

Eliminando duplicações em uma agregação

- Pode-se usar o DISTINCT dentro de uma agregação
- Exemplo: encontre o número de preços *diferentes* cobrados pela Fanfa:

```
SELECT COUNT(DISTINCT preço)
FROM Venda
WHERE nome_refri = 'Fanfa';
```

Valores NULL são ignorados na agregação

- Um NULL nunca contribui para uma soma, média ou contagem, e nunca pode ser nem o mínimo, nem o máximo de uma coluna
- Mas se não existir valores não nulos em uma coluna, então o resultado da agregação é NULL
  - Exceção: COUNT de um conjunto vazio é 0

Exemplo: efeito de NULLs

```
SELECT count(*)
FROM Venda WHERE
nome_refri = ’Fanfa’;
```

O número de lanchonetes que vendem Fanfa.

```
SELECT count(preço)
FROM Venda WHERE
nome_refri = ’Fanfa’;
```

O número de lanchonetes que vendem Fanfa a um preço conhecido.

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)
FROM Venda
GROUP BY nome_refri;
```

nome_refri	AVG(preço)
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):

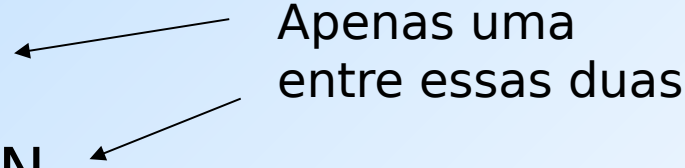
```
SELECT *
FROM Cliente JOIN Frequentador ON
    nome = nome_cliente;
```

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 ao 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).

Modificações no estado do banco de dados

- Um comando de modificação não devolve um resultado (como uma consulta faz), ele modifica o estado do BD de alguma forma
- Existem 3 tipos de modificações:

1. Inserção de tupla(s)
2. Remoção de tupla(s)
3. Modificação do(s) valor(es) dos componentes de tupla(s) existente(s)

Inserção

- Para inserir uma única tupla:

  INSERT INTO <relação>

  VALUES (<lista de valores>);

- Exemplo: adicione a Apreciador(nome_cliente, nome_refri) o fato de que Ana gosta de Fanfa.

```
INSERT INTO Apreciador
VALUES('Ana', 'Fanfa');
```

Especificando atributos no INSERT

- Nós podemos adicionar ao nome da relação uma lista de atributos.
- Há duas razões para se fazer isso:
  1. Quando esquecemos a ordem padrão dos atributos da relação.
  2. Quando não temos valores para todos os atributos e queremos que o SGBD preencha os componentes faltantes com NULL ou um valor default.

Exemplo: especificando atributos

- Outra forma de adicionar o fato de que Ana gosta de Fanfa a Apreciador(Nome_cliente, nome_refri):

```
INSERT INTO
 Apreciador(nome_refri, nome_cliente)
VALUES('Fanfa', 'Ana');
```

Adicionando valores padrão

- Em um comando CREATE TABLE, podemos indicar um valor padrão para um atributo, por meio da cláusula DEFAULT.
- Quando uma tupla inserida não possui valor para esse atributo, o valor padrão será usado.

Exemplo: valores padrão

```
CREATE TABLE Cliente (
   nome CHAR(30) PRIMARY KEY,
   endereco CHAR(50)
        DEFAULT 'Av. Paulista, 123',
   telefone CHAR(16)
);
```

Exemplo: valores padrão

```
INSERT INTO Cliente(nome)
VALUES('Ana');
```

Tupla resultante:

nome	endereco	telefone
Ana	Av. Paulista 123	NULL

Inserção de várias tuplas

- Podemos inserir o resultado todo de uma consulta em uma relação usando a forma:

  INSERT INTO <relação>

  (<subconsulta>);

Exemplo: inserção de subconsultas

- Usando Frequentador(nome_cliente, nome_refri), insira em uma nova relação Colegas(nome) todos os colegas “em potencial” da Ana, i.e., os clientes que frequentam pelo menos uma lanchonete frequentada pela Ana.

Solução

Nome dos potenciais colegas

Pares de tuplas de Clientes onde a primeira é para Ana e a segunda é para um outro cliente qualquer, e as lanchonetes são a mesma.

```
INSERT INTO Colegas
(SELECT f2.nome_cliente
 FROM Frequentador f1, Frequentador f2
 WHERE f1.nome_cliente = 'Ana' AND
   f2.nome_cliente <> 'Ana' AND
   f1.nome_lanch = f2.nome_lanch
);
```

Remoção

- Para remover tuplas que satisfazem uma condição em uma relação:

  DELETE FROM <relação>

  WHERE <condição>;

Exemplo: remoção

- Remova de Apreciador(nome_cliente, nome_refri) o fato de que Ana gosta de Fanfa:

```
DELETE FROM Apreciador
WHERE nome_cliente = 'Ana'
   AND nome_refri = 'Fanfa';
```

Exemplo: remoção de todas as tuplas

- Esvazie a relação Apreciador:

```
DELETE FROM Apreciador;
```

- Observe que nenhuma cláusula WHERE é necessária.

Exemplo: remoção de algumas tuplas

- Remova de Refrigerante(nome, fabricante) todos os refris para os quais existe um outro refri feito pelo mesmo fabricante.

```
DELETE FROM Refrigerantes r
WHERE EXISTS (
    SELECT nome FROM Refrigerante
    WHERE fabricante = r.fabricante AND
        nome <> r.nome);
```

Refris com o mesmo Fabricante e um nome diferente do nome do refri representado pela tupla r.

Semântica da remoção (1)

- Suponha que a Cola-Coca produza somente Fanfa e Fanfa Diet.
- Suponha que passemos primeiro pela tupla da Fanfa.
- A subconsulta é não vazia, por causa da tupla da Fanfa Diet, então removeremos a Fanfa.
- Agora, quando r é a tupla para Fanfa Diet, nós removeremos essa tupla também?

Semântica da remoção (2)

- Resposta: Fanfa Diet será removida também!
- A razão para isso é o fato da remoção acontecer em dois estágios:
  1. Marcação de todas as tuplas para as quais a condição WHERE é satisfeita.
  2. Remoção das tuplas marcadas.

Alterações

- Para mudar alguns atributos em algumas tuplas de uma relação:

  UPDATE <relação>
  SET <lista de atribuições a atributos>
  WHERE <condição sobre as tuplas>;

Exemplo: alteração

- Mude o número do telefone do cliente John para 555-1212:

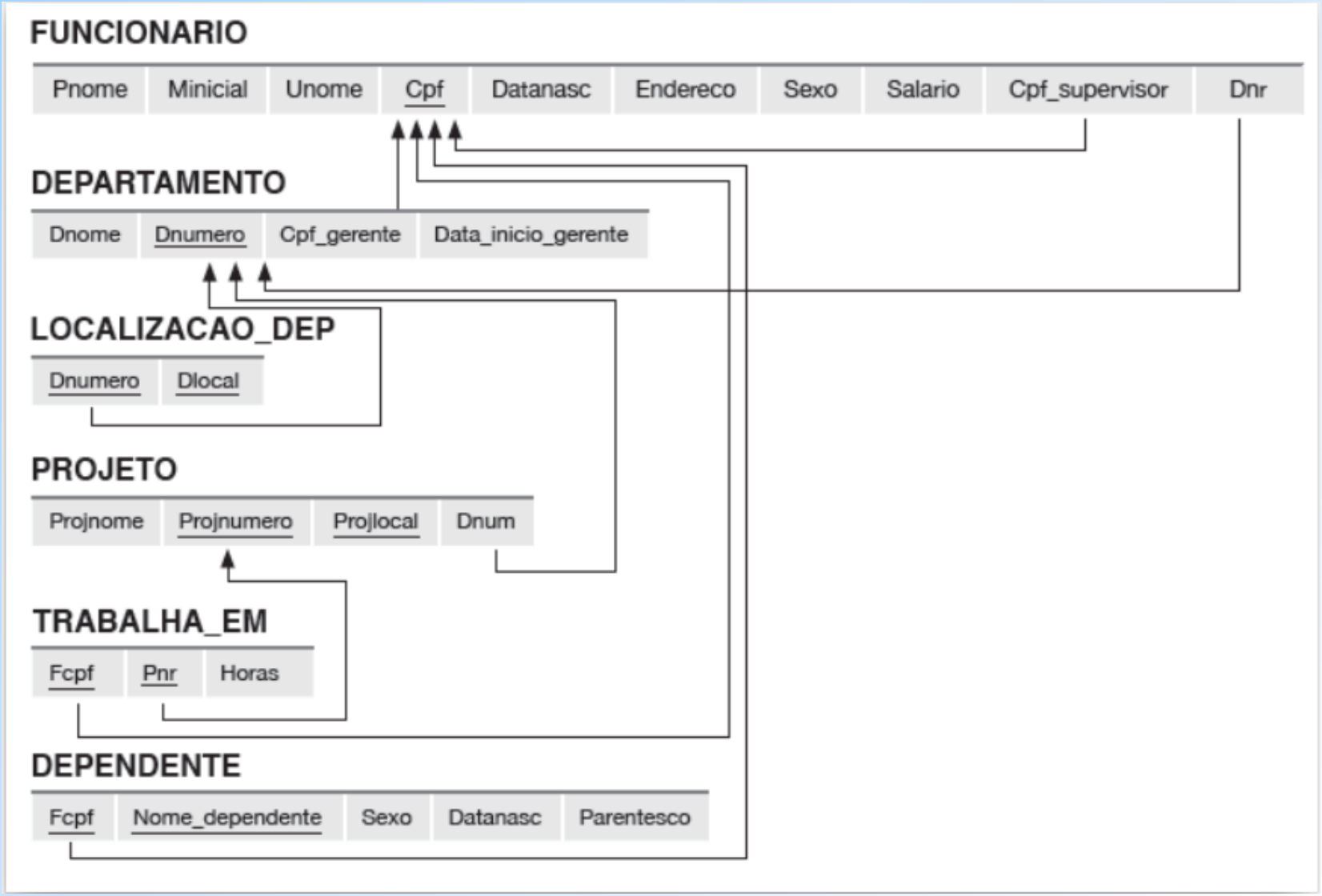
```
UPDATE Cliente
SET telefone = '555-1212'
WHERE nome = 'Jonn';
```

Exemplo: modificação de várias tuplas

- Faça com que R$4 seja o preço máximo para um refrigerante:

```
UPDATE Venda
SET preco = 4.00
WHERE preco > 4.00;
```

Exercícios


FUNCIONARIO
Pnome
Minicial
Unome
Cpf
Datanasc
Endereco
Sexo
Salario
Cpf_supervisor
Dnr
DEPARTAMENTO
Dnome
Dnumero
Cpf_gerente
Data_inicio_gerente
LOCALIZACAO_DEP
Dnumero
Dlocal
PROJETO
Projnome
Projnumero
Projlocal
Dnum
TRABALHA_EM
Fcpf
Pnr
Horas
DEPENDENTE
Fcpf
Nome_dependente
Sexo
Datanasc
Parentesco

Exercícios (agreg./agrup.)

1. Achar a soma dos salários dos funcionários do departamento ‘Pesquisa’, o salário máximo, o salário mínimo e média dos salários.
2. Para cada departamento, recuperar o número do departamento, o número de funcionários no departamento e o seu salário médio.
3. Para cada departamento, recuperar o número do departamento, o nome do departamento, o número de funcionários no departamento e o seu salário médio.
4. Contar o número de valores de salários distintos no BD.
5. Mostrar o nome dos funcionários que têm três ou mais dependentes.
6. Para cada departamento que tem mais do que 5 funcionários, mostrar o núm. do departamento e a quantidade de funcionários que estão ganhando mais do que R$40.000,00.

Exercícios (modificações de dados)

1. Criar uma tabela “temporária” que contém o sobrenome do funcionário, o nome do projeto e as horas por semana para cada funcionário que trabalha em projeto.
2. Conceder um aumento de 10% a todos os funcionários do departamento ‘Pesquisa’.
3. Remover todos os projetos controlados pelo departamento ‘Vendas’.

Exercícios (“para casa”) com Agrupamento/Agregação

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)

1) Para cada valor de nível que aparece em Aluno, imprima o nível e idade média dos alunos desse nível.
2) Para cada valor de nível que aparece em Aluno exceto os níveis que possuem menos de 10 alunos, imprima o nível e idade média dos alunos desse nível.
3) 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.
4) Para cada professor que ministra cursos apenas na sala *R128*, imprima seu nome e o número total de cursos que ele ou ela ministra.
5) Encontre os nomes dos alunos matriculados no número máximo de cursos.
6) 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*).

Referências Bibliográficas

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