

Acesso à múltiplas tabelas (Joins)

Prof. Aparecido Vilela Junior

Junções

Obtendo Dados de Várias Tabelas

EMP

EMPNO	ENAME	...	DEPTNO
7839	KING	...	10
7698	BLAKE	...	30
...			
7934	MILLER	...	10

DEPT

DEPTNO	DNAME	LOC
10	ACCOUNTING	NEW YORK
20	RESEARCH	DALLAS
30	SALES	CHICAGO
40	OPERATIONS	BOSTON

```

EMPNO  DEPTNO  LOC
-----
7839      10  NEW YORK
7698      30  CHICAGO
7782      10  NEW YORK
7566      20  DALLAS
7654      30  CHICAGO
7499      30  CHICAGO
...
14 rows selected.

```

Junções

O Que É uma Junção?

Use uma junção para consultar dados a partir de uma ou mais tabelas.

```
SELECT    tabela1.coluna, tabela2.coluna  
FROM      tabela1, tabela2  
WHERE     tabela1.coluna1 = tabela2.coluna2;
```

- Criar uma condição de junção na cláusula **WHERE**.
- Prefixar o nome da coluna com o nome da tabela quando o mesmo nome da coluna aparecer em mais de uma tabela.

Junções - Diretrizes

- Ao criar uma instrução SELECT que una tabelas, anteceda o nome da coluna com o nome da tabela a fim de esclarecer e avançar o acesso ao banco de dados.
- Caso apareça o mesmo nome da coluna em mais de uma tabela, o nome da coluna deve estar prefixado com o nome da tabela.
- Para juntar n tabelas, é necessário um mínimo de $(n-1)$ condições de junção. Assim, para juntar quatro tabelas, é necessário um mínimo de três junções. Esta regra pode não se aplicar se sua tabela possuir uma chave primária concatenada, no caso de mais de uma coluna ser necessária para identificar exclusivamente cada linha.

Produto Cartesiano

- Um produto cartesiano é formado quando:
 - Uma condição de junção estiver omitida
 - Uma condição de junção estiver inválida
 - Todas as linhas na primeira tabela estão unidas a todas as linhas da segunda tabela
- Para evitar um produto Cartesiano, sempre inclua uma condição de junção válida em uma cláusula WHERE.

Produto Cartesiano

Gerando um Produto Cartesiano

EMP (14 linhas)

EMPNO	ENAME	...	DEPTNO
7839	KING	...	10
7698	BLAKE	...	30
...			
7934	MILLER	...	10

DEPT (4 linhas)

DEPTNO	DNAME	LOC
10	ACCOUNTING	NEW YORK
20	RESEARCH	DALLAS
30	SALES	CHICAGO
40	OPERATIONS	BOSTON

"Produto
Cartesiano: →
14*4=56 linhas"

ENAME	DNAME
KING	ACCOUNTING
BLAKE	ACCOUNTING
...	
KING	RESEARCH
BLAKE	RESEARCH
...	
56 rows selected.	

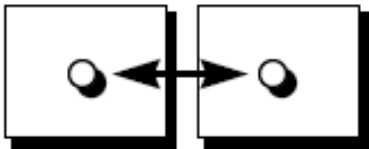
Produto Cartesiano

- Gera-se um produto Cartesiano caso uma condição de junção seja omitida.
- O exemplo do slide exhibe o nome do funcionário e do departamento a partir das tabelas EMP e DEPT.
 - Porque nenhuma cláusula WHERE foi especificada, todas as linhas (14 linhas) da tabela EMP são unidas a todas as linhas (4 linhas) na tabela DEPT, gerando dessa forma 56 linhas na saída.

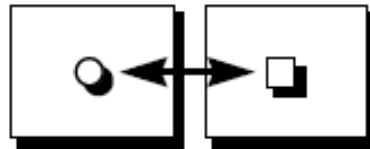
Junções

Tipos de Junções

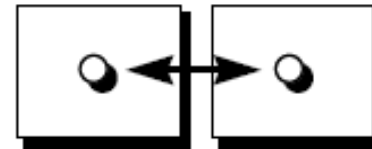
**Junção
idêntica**



**Junção
não-idêntica**



**Junção
externa**



Autojunção



Tipos de Junções

- Há dois tipos principais de condições de junção:
 - Junções idênticas
 - Junções não-idênticas
- Métodos de junção adicional incluem o seguinte:
 - Junções externas
 - Autojunções
 - Operadores de conjunto

Junções Idênticas

EMP

EMPNO	ENAME	DEPTNO
7839	KING	10
7698	BLAKE	30
7782	CLARK	10
7566	JONES	20
7654	MARTIN	30
7499	ALLEN	30
7844	TURNER	30
7900	JAMES	30
7521	WARD	30
7902	FORD	20
7369	SMITH	20
...		
14 rows selected.		

Chave estrangeira

DEPT

DEPTNO	DNAME	LOC
10	ACCOUNTING	NEW YORK
30	SALES	CHICAGO
10	ACCOUNTING	NEW YORK
20	RESEARCH	DALLAS
30	SALES	CHICAGO
30	SALES	CHICAGO
30	SALES	CHICAGO
30	SALES	CHICAGO
30	SALES	CHICAGO
20	RESEARCH	DALLAS
20	RESEARCH	DALLAS
...		
14 rows selected.		

Chave primária

Junções Idênticas

- Para determinar o nome do departamento de um funcionário, compare o valor na coluna DEPTNO na tabela EMP com os valores DEPTNO da tabela DEPT.
- O relacionamento entre as tabelas EMP e DEPT é uma *junção idêntica* — ou seja, os valores da coluna DEPTNO das duas tabelas devem ser iguais.
- Com frequência, essa junção envolve complementos de chave primária e estrangeira.

Junções Idênticas

```
SQL> SELECT  emp.empno,    emp.ename, emp.deptno,  
2           dept.deptno, dept.loc  
3 FROM      emp, dept  
4 WHERE     emp.deptno=dept.deptno;
```

EMPNO	ENAME	DEPTNO	DEPTNO	LOC
7839	KING	10	10	NEW YORK
7698	BLAKE	30	30	CHICAGO
7782	CLARK	10	10	NEW YORK
7566	JONES	20	20	DALLAS
...				

14 rows selected.

Junções Idênticas

- A cláusula SELECT especifica os nomes de coluna a recuperar:
 - nome do funcionário, número do funcionário e número do departamento, que são as colunas na tabela EMP
 - número do departamento, nome do departamento e localização, que são as colunas na tabela DEPT
- A cláusula FROM especifica as duas tabelas que o banco de dados deve acessar:
 - – tabela EMP
 - – tabela DEPT
- A cláusula WHERE especifica como as tabelas serão unidas:
 - EMP.DEPTNO=DEPT.DEPTNO
- Porque a coluna DEPTNO é comum às duas tabelas, ela deve estar prefixada pelo nome da tabela a fim de evitar ambiguidade.

Apelidos em tabelas

Simplifique consultas usando apelidos de tabela.

```
SQL> SELECT emp.empno, emp.ename, emp.deptno,  
2          dept.deptno, dept.loc  
3 FROM emp, dept  
4 WHERE emp.deptno=dept.deptno;
```

```
SQL> SELECT e.empno, e.ename, e.deptno,  
2          d.deptno, d.loc  
3 FROM emp e, dept d  
4 WHERE e.deptno= d.deptno;
```

Junções

Junções Não-idênticas

EMP

EMPNO	ENAME	SAL
7839	KING	5000
7698	BLAKE	2850
7782	CLARK	2450
7566	JONES	2975
7654	MARTIN	1250
7499	ALLEN	1600
7844	TURNER	1500
7900	JAMES	950
...		
14 rows selected.		

SALGRADE

GRADE	LOSAL	HISAL
1	700	1200
2	1201	1400
3	1401	2000
4	2001	3000
5	3001	9999

"o salário na tabela EMP está entre salário inferior e salário superior na tabela SALGRADE"

Junções

- O relacionamento entre a tabela EMP e a tabela SALGRADE é uma junção não-idêntica, o que significa que nenhuma coluna da tabela EMP corresponde diretamente a uma coluna da tabela SALGRADE.
- O relacionamento entre as duas tabelas é que a coluna SAL da tabela EMP está entre a coluna LOSAL e HISAL da tabela SALGRADE.
- O relacionamento é obtido usando um outro operador que não o igual (=).

Exemplo

- SELECT E.ENAME, E.SAL, G.GRADE
- FROM EMP E, SALGRADE G
- WHERE E.SAL BETWEEN G.LOSAL AND G.HISAL

JOINS – TABELAS

- CLAUSULA **ON** - usado para deixar a junção mais legível.
 - SELECT e.employee_id, e.last_name,
d.department_id, d.department_name
 - FROM employees e JOIN departments d
 - **ON** (e.department_id = d.department_id);

JOINS – TABELAS

- JUNÇÃO COM 3 TABELAS NO PADRAO ANSI
 - SELECT e.employee_id, e.first_name,
d.department_name, l.city
 - FROM employees e JOIN departments d
 - ON d.department_id = e.department_id
 - JOIN locations l
 - ON l.location_id = d.location_id;

JOINS – TABELAS

- OUTER JOIN:
- LEFT - Exibe todas as informações da tabela a esquerda mesmo que não tenha a direita.
- Ex.
 - `SELECT e.first_name || ' ' || e.last_name "Nome",`
 - `d.department_name "Departamento"`
 - `FROM employees e`
 - `LEFT OUTER JOIN departments d`
 - `ON e.department_id = d.department_id;`
- A consulta acima exibe o nome do empregado e o nome do departamento de todos os empregados mesmo que não esteja alocado em nenhum departamento.

JOINS – TABELAS

- RIGHT - exibe todas as informações da tabela a direita mesmo que não tenha a esquerda.
 - SELECT e.last_name "Sobrenome",
 - e.job_id "Cargo", d.department_id "Depto",
 - d.department_name "Nome Depto"
 - FROM employees e RIGHT OUTER JOIN departments d
 - ON (e.department_id = d.department_id);
- A consulta acima exibe o sobrenome, o cargo, o id do departamento e o nome do departamento de todos os funcionários; inclusive dos departamentos que não possuem funcionário algum.

JOINS – TABELAS

- FULL - Exibe todas as linhas da esquerda e da direita mesmo que não haja correspondência.
- LEFT + RIGHT
 - SELECT e.first_name || ' ' || e.last_name "Nome", e.job_id "Cargo", d.department_id "Depto",
 - d.department_name "Nome Depto"
 - FROM employees e FULL OUTER JOIN departments d
 - ON e.department_id = d.department_id;
- A consulta acima exibe os funcionários que não trabalham em nenhum departamento e os departamentos que não possuem empregado.

Exemplo – INNER JOIN (ANSI)

- Deseja-se uma lista contendo o nome e departamento do gerente e nome e salário de todos os funcionários subordinados a ele. Apresente o resultado ordenado por departamento e salário (descendente).
- `SELECT G.CD_DEPTO, G.NM_FUNC "Gerente",`
- `F.NM_FUNC, F.VL_SAL`
- `FROM FUNCIONARIO G INNER JOIN DEPTO D ON G.CD_MAT =`
`D.CD_GERENTE`
- `JOIN FUNCIONARIO F ON F.CD_DEPTO = D.CD_DEPTO`
- `ORDER BY 1, 4 DESC;`

Exemplo – INNER JOIN

- SELECT G.CD_DEPTO, G.NM_FUNC "Gerente",
- F.NM_FUNC, F.VL_SAL
- FROM FUNCIONARIO G, FUNCIONARIO F,
DEPTO D
- WHERE G.CD_MAT = D.CD_GERENTE
- AND F.CD_DEPTO = D.CD_DEPTO
- ORDER BY 1, 4 DESC;

Exercícios

- 1) Crie uma consulta para exibir o nome, o número e o nome do departamento de todos os funcionários.
- 2) Crie uma lista única de todos os cargos existentes no departamento 30. Inclua a localização do departamento 30 na saída.
- 3) Crie uma consulta para exibir o nome do funcionário, o nome do departamento e a localização de todos os funcionários que recebem uma comissão.
- 4) Exiba o nome do funcionário e o nome do departamento para todos os funcionários que possuem um A em seus nomes.

Exercícios

- 5) Crie uma consulta para exibir o nome, o cargo, o número e o nome do departamento para todos os funcionários que trabalham em DALLAS.
- 6) Exiba o nome e o número do funcionário junto com o nome e o número do gerente. Coloque um label nas colunas Employee, Emp#, Manager e Mgr#, respectivamente.
- 7) Modifique o exercício anterior para exibir todos os funcionários incluindo King, que não possuem um gerente.
- 8) Crie uma consulta que exibirá o nome do funcionários, o número do departamento e todos os funcionários que trabalham no mesmo departamento de um determinado funcionário. Forneça a cada coluna um label apropriado.

Exercícios

- 9) Mostre a estrutura da tabela SALGRADE. Crie uma consulta que exiba o nome, o cargo, o nome do departamento, o salário e a classificação de todos os funcionários.
- 10) Crie uma consulta para exibir o nome e a data de admissão de qualquer funcionário admitido após o funcionário Blake.
- 11) Exiba todos os nomes de funcionários e as datas de admissão junto com o nome e a data de admissão do gerente para todos os funcionários admitidos antes de seus gerentes. Coloque um label nas colunas Employee, Emp Hiredate, Manager e Mgr Hiredate, respectivamente

Exercícios

- 1) Crie uma consulta para exibir o nome, o número e o nome do departamento de todos os funcionários.

```
SQL> SELECT      e.ename, e.deptno, d.dname
  2  FROM        emp e, dept d
  3  WHERE       e.deptno = d.deptno;
```

Exercícios

- 2) Crie uma lista única de todos os cargos existentes no departamento 30. Inclua a localização do departamento 30 na saída.

```
SQL> SELECT      DISTINCT e.job, d.loc  
      2  FROM      emp e, dept d  
      3  WHERE     e.deptno = d.deptno  
      4  AND       e.deptno = 30;
```

Exercícios

- 3) Crie uma consulta para exibir o nome do funcionário, o nome do departamento e a localização de todos os funcionários que recebem uma comissão.

```
SQL> SELECT      e.ename, d.dname, d.loc
2  FROM          emp e, dept d
3  WHERE         e.deptno = d.deptno
4  AND           e.comm IS NOT NULL;
```

Exercícios

- 4) Exiba o nome do funcionário e o nome do departamento para todos os funcionários que possuem um A em seus nomes.

```
SQL> SELECT      e.ename, d.dname  
      2  FROM      emp e, dept d  
      3  WHERE     e.deptno = d.deptno  
      4  AND       e.ename LIKE '%A%';
```

Exercícios

- 5) Crie uma consulta para exibir o nome, o cargo, o número e o nome do departamento para todos os funcionários que trabalham em DALLAS.

```
SQL> SELECT      e.ename, e.job, e.deptno, d.dname
  2  FROM          emp e, dept d
  3  WHERE         e.deptno = d.deptno
  4  AND           d.loc = 'DALLAS';
```


Exercícios

- 6) Exiba o nome e o número do funcionário junto com o nome e o número do gerente. Coloque um label nas colunas Employee, Emp#, Manager e Mgr#, respectivamente.

```
SQL> SELECT      e.ename "Employee", e.empno "Emp#",  
2               m.ename "Manager", m.empno "Mgr#"  
3               FROM          emp e, emp m  
4               WHERE          e.mgr = m.empno;
```

Exercícios

- 7) Modifique o exercício anterior para exibir todos os funcionários incluindo King, que não possuem um gerente.

```
SELECT  e.ename "Employee", e.empno "Emp#",  
        m.ename "Manager", m.empno "Mgr#"  
FROM    emp  e, emp m  
WHERE   e.mgr = m.empno(+)
```

Exercícios

- 8) Crie uma consulta que exibirá o nome do funcionários, o número do departamento e todos os funcionários que trabalham no mesmo departamento de um determinado funcionário. Forneça a cada coluna um label apropriado.

```
SQL> SELECT      e.deptno department, e.ename employee,  
2               c.ename colleague  
3 FROM          emp e, emp c  
4 WHERE         e.deptno = c.deptno  
5 AND           e.empno <> c.empno  
6 ORDER BY     e.deptno, e.ename, c.ename;
```

Exercícios

- 9) Mostre a estrutura da tabela SALGRADE. Crie uma consulta que exiba o nome, o cargo, o nome do departamento, o salário e a classificação de todos os funcionários.

```
SQL> SELECT e.ename, e.job, d.dname, e.sal, s.grade  
2   FROM   emp e, dept d, salgrade s  
3   WHERE  e.deptno = d.deptno  
4   AND    e.sal BETWEEN s.losal AND s.hisal;
```

Exercícios

- 10) Crie uma consulta para exibir o nome e a data de admissão de qualquer funcionário admitido após o funcionário Blake.

```
SELECT emp.ename, emp.hiredate  
FROM   emp, emp blake  
WHERE  blake.ename = 'BLAKE'  
AND    blake.hiredate < emp.hiredate;
```

Exercícios

- 11) Exiba todos os nomes de funcionários e as datas de admissão junto com o nome e a data de admissão do gerente para todos os funcionários admitidos antes de seus gerentes. Coloque um label nas colunas Employee, Emp Hiredate, Manager e Mgr Hiredate, respectivamente

```
SELECT e.ename "Employee", e.hiredate "Emp Hiredate",  
       m.ename "Manager", m.hiredate "Mgr Hiredate"  
FROM   emp e, emp m  
WHERE  e.mgr = m.empno  
AND    e.hiredate < m.hiredate;
```