Structured Query Language

Parte 2

Prof. Leandro Correia

Bancos de Dados Relacionais

SQL - DML

- SELECT - Recuperação/ busca de dados
- INSERT - Inclusão de tuplas/registros
- UPDATE - Alteração do conteúdo dos dados
- DELETE - Exclusão de tuplas/registros

```
SELECT [DISTINCT] lista_colunas
FROM {nome_tabela}
     [[,{nome_tabela}] ...]
[WHERE condição]
[GROUP BY clause]
[HAVING clause]
[ORDER BY clause]
```

Bancos de Dados Relacionais

SQL - DML

- Seleção simples

```
SELECT  num_matricula, nome
FROM aluno

SELECT * FROM aluno
```

- Seleção com filtro

```
SELECT    nome,  endereco
FROM      aluno
WHERE     codigo_curso = 1
AND       num_matricula > 100
```

Bancos de Dados Relacionais

SQL - DML

- Junções (Joins)
  - CROSS JOIN
    - Produto Cartesiano;
  - INNER JOIN
    - Igualdade entre os lados da comparação;
  - OUTER JOIN
    - Igualdade parcial entre os lados da comparação;
    - Pode ser left, right ou full outer join;
    - Quando não há igualdade, valores nulos são assumidos.

Aluno

cod_aluno	nome_aluno	cod_curso
1	Mario	52
2	Ana	51
3	Paula	52

Curso

cod_curso	nome_curso
51	Biologia
52	Computação
53	Direito

Cross Join

cod_aluno	nome_aluno	cod_curso	cod_curso	nome_curso
1	Mario	52	51	Biologia
1	Mario	52	52	Computação
1	Mario	52	53	Direito
2	Ana	51	51	Biologia
2	Ana	51	52	Computação
2	Ana	51	53	Direito
3	Paula	52	51	Biologia
3	Paula	52	52	Computação
3	Paula	52	53	Direito

Aluno

cod_aluno	nome_aluno	cod_curso
1	Mario	52
2	Ana	51
3	Paula	52

Curso

cod_curso	nome_curso
51	Biologia
52	Computação
53	Direito

Inner Join

cod_aluno	nome_aluno	cod_curso	cod_curso	nome_curso
1	Mario	52	52	Computação
2	Ana	51	51	Biologia
3	Paula	52	52	Computação
4	Carlos	52	52	Computação

Aluno

cod_aluno	nome_aluno	cod_curso
1	Mario	52
2	Ana	51
3	Paula	52

Curso

cod_curso	nome_curso
51	Biologia
52	Computação
53	Direito

Outer Join (right)

cod_aluno	nome_aluno	cod_curso	cod_curso	nome_curso
1	Mario	52	52	Computação
2	Ana	51	51	Biologia
3	Paula	52	52	Computação
4	Carlos	52	52	Computação
null	null	null	53	Direito

Bancos de Dados Relacionais

SQL - DML

- Junções (Joins)
  - CROSS JOIN

```
select * from aluno, curso
select * from aluno cross join curso
```

  - INNER JOIN

```
select * from aluno A, curso C
where A.cod_curso = C.cod_curso

select * from aluno A inner join curso C
   on A.cod_curso = C.cod_curso
```

  - OUTER JOIN

```
select * from aluno A right outer join curso C
   on A.cod_curso = C.cod_curso
```

Bancos de Dados Relacionais

SQL - DML

- Funções de Agregação
  - SUM(expressao)
    - Somatório dos valores da expressão;
  - AVG(expressao)
    - Média aritmética dos valores da expressão;
  - COUNT(expressao)
    - Quantidade de valores da expressão;
  - COUNT(*)
    - Número de linhas selecionadas;
  - MAX(expressao)
    - Maior valor da expressão;
  - MIN(expressao)
    - Menor valor da expressão.
  - Observação
    - SUM, AVG, COUNT, MAX, e MIN ignoram valores nulos;
    - COUNT(*) não ignora linhas nulas.

Bancos de Dados Relacionais

SQL - DML

- Funções de Agregação (exemplos)

SELECT **COUNT(*)** FROM aluno

SELECT **MAX(salario), MIN(salario)**
FROM empregado

SELECT **COUNT (DISTINCT city)** FROM S

SELECT **AVG(salario)** FROM empregado
WHERE codigo_depto = 01

SELECT **SUM(QTY)** FROM SP

SQL - DML

- Cláusula GROUP BY
  - Especifica os campos que servirão de critério para definição de grupos de valores sumarizados através de funções de agregação;
  - Quando utilizadas sem uma cláusula GROUP BY, as funções de agregação retornam apenas um valor para a consulta realizada.
  - Exemplo:

```
SELECT      D.cod_depto, AVG(E.salario)
FROM        empregado E, departamento D
WHERE       E.cod_depto = D.cod_depto
GROUP BY    D.cod_depto
```

SQL - DML

- Cláusula HAVING
  - Filtro aplicado sobre o resultado das funções de agregação;
  - Restrição aplicada após a operação de agrupamento (GROUP BY);
  - Pode fazer referência a qualquer coluna que esteja na lista do SELECT;
  - Exemplo:

```
SELECT      D.cod_depto, AVG(E.salario)
FROM        empregado E, departamento D
WHERE       E.cod_depto = D.cod_depto
GROUP BY    D.cod_depto
HAVING      AVG(E.salario) > 1000
```

SQL - DML

- Cláusula ORDER BY
  - Ordena o resultado da consulta por colunas;
  - Permite a especificação de múltiplas colunas;
  - Colunas que não aparecem na lista do SELECT podem ser utilizadas;
  - Exemplo:

```
SELECT      D.cod_depto, AVG(E.salario)
FROM        empregado E, departamento D
WHERE       E.cod_depto = D.cod_depto
GROUP BY    D.cod_depto
HAVING      AVG(E.salario) > 1000
ORDER BY    AVG(E.salario) DESC
```

Bancos de Dados Relacionais

SQL - DML

- Operador UNION
  - Combina o resultado de duas ou mais consultas em um único resultado consistindo de todas as linhas que pertencem a todas as consultas no UNION;
  - EXEMPLO:

```
SELECT    data_lancamento, valor_lancamento
FROM      lancamento2000
UNION
SELECT    data_lancamento, valor_lancamento
FROM      lancamento2001
ORDER BY  valor_lancamento DESC
```

SQL - DML

- Operador LIKE
  - Caracteres especiais
    - "%" String com qualquer tamanho;
    - "_" Único caracter
  - Exemplo:

```
SELECT num_matricula, nome
FROM empregado WHERE nome LIKE 'A%'

SELECT nome, endereco
FROM empregado
WHERE endereco LIKE '_a%'
```

SQL - DML

- Operador BETWEEN
  - Especifica um intervalo como critério, incluindo os valores determinados;
  - Exemplo:

```
SELECT num_matricula, nome
FROM empregado
WHERE num_matricula BETWEEN  10 AND 100
```

SQL - DML

- Sub-consultas

```
SELECT      e.num_matricula, e.nome
FROM        empregado e
WHERE       e. cod_departamento = 1
AND salario > (     SELECT AVG(salario)
                    FROM     empregado e
                    WHERE e.cod_departamento = 1    )
```

- Operadores IN / NOT IN
  - Comparam um valor com um conjunto válido explicitamente informado ou gerado através de uma consulta;
- Operadores EXISTS / NOT EXISTS
  - Funcionam como um teste de existência;
  - A sub-consulta retorna TRUE ou FALSE.

SQL - DML

- Operadores IN / NOT IN

```
SELECT S#, SNAME FROM S
WHERE STATUS IN (10, 15, 20)

SELECT num_matricula, nome
FROM empregado
WHERE cod_departamento IN
        (SELECT cod_departamento FROM departamento
         WHERE nome_departamento = "Informatica")

SELECT nome_departamento FROM departamento
WHERE cod_departamento NOT IN
        (SELECT cod_departamento FROM empregado)
```

Bancos de Dados Relacionais

SQL - DML

- Operadores EXISTS / NOT EXISTS

```
SELECT num_matricula, nome
FROM empregado e
WHERE EXISTS
        (SELECT cod_departamento FROM departamento d
         WHERE e.cod_departamento = d.cod_departamento
        AND         nome_departamento = "Informatica")

SELECT e.num_matricula, e.nome
FROM empregado e, departamento d
WHERE e.cod_departamento = d.cod_departamento
AND    nome_departamento = "Informatica"

SELECT nome_departamento FROM departamento d
WHERE NOT EXISTS
        (SELECT cod_departamento FROM empregado e
        WHERE e.cod_departamento = d.cod_departamento)
```

Bancos de Dados Relacionais

SQL - DML

- Inserção de registros (INSERT)

INSERT [INTO] {nome_tabela} [(lista_colunas)]
{lista_valores | comando_select}

- lista_colunas: Lista com os campos da tabela que serão incluídos;
- lista_valores: Lista de valores dos dados. Representada pela cláusula VALUES;
- comando_select: Comando SELECT cujo resultado será inserido na tabela em questão;

- Exemplos:

```
INSERT INTO aluno
VALUES (5, 'Paulo Andre', 'Rua A,  100 / 1001',
          '226-4099', 12)

INSERT INTO aluno (num_matricula, nome, cod_curso)
VALUES (5, 'Paulo Andre', 12)
```

SQL - DML

- Alteração de campos em registros existentes (UPDATE)

```
UPDATE  {nome_tabela}
    SET  { nome_coluna = {expressao |  NULL}
    [[,] column_name = {expressao  |  NULL} ]
[WHERE condicao]
```

– Exemplos:

```
UPDATE empregado
SET salario = salario * 2
WHERE cod_departamento = 1

UPDATE empregado
SET salario = (SELECT AVG(salario) FROM empregado)
WHERE salario IS NULL
```

SQL - DML

- Alteração de campos em registros existentes (UPDATE)
  - Exemplos:

```
UPDATE departamento
SET status = 0
WHERE cod_departamento IN
        ( SELECT cod_departamento FROM empregado)
```

Bancos de Dados Relacionais

SQL - DML

- Exclusão de registros (DELETE)

DELETE [FROM] {nome_tabela}
[WHERE condicao]

– Exemplos:

```
DELETE empregado
WHERE num_matricula =2334456

DELETE empregado
WHERE cod_departamento IN
        ( SELECT cod_departamento
          WHERE situacao_departamento= 'INATIVO')
```