



Universidade Federal
de Campina Grande



Banco de Dados I

Unidade 5: A linguagem SQL - DML Parte 1

Prof. Cláudio de Souza Baptista, Ph.D.
Laboratório de Sistemas de Informação – LSI
UFCG

SQL-DML

- Esquemas do BD Empresa:
 - **Empregado**(matricula, nome, endereco, salario, supervisor, depto)
 - **Departamento**(coddep, nome, gerente, dataini)
 - **Projeto**(codproj, nome, local, depart)
 - **Alocacao**(matric, codproj, horas)

SQL-DML

- SQL interativo
- As operações de manipulação sem cursor são:
SELECT, INSERT, UPDATE, DELETE
- O comando Select:
 - A forma básica do comando Select é:

```
SELECT <lista atributos>  
FROM <lista tabelas>  
WHERE <condição>
```

SQL-DML - Exemplos

Q1. Obtenha o salário de José

```
SELECT salario  
FROM Empregado  
WHERE nome='José'
```

Obs.: Podemos renomear o nome da coluna no resultado

```
SELECT salario as SalarioJose  
FROM Empregado  
WHERE nome='José'
```

SQL-DML - Exemplos

Obs2: Podemos usar colunas como expressões:

```
SELECT matricula, salario, 0.15*salario as IR  
FROM Empregado
```

- Podemos inserir constantes na cláusula select se necessário

```
SELECT nome, 'marajá' as Marajá  
FROM Empregado  
WHERE salario > 10.000,00
```

SQL-DML - Exemplos

- **Q2.** Selecione o nome e o endereço de todos os empregados que trabalham no departamento de produção

```
SELECT e.nome, e.endereco  
FROM empregado e, departamento d  
WHERE d.nome = 'Produção' and d.coddep = e.depto
```

obs.: 'e' e 'd' são aliases (variáveis de dupla)

SQL-DML - Exemplos

- **Q.3** Para cada projeto em 'Fortaleza', liste o código do projeto, o departamento que controla o projeto e o nome do gerente com endereço e salário

```
SELECT p.codproj, d.nome, e.nome,  
       e.endereco, e.salario  
FROM Projeto p, Departamento d, Empregado e  
WHERE p.depart = d.coddep and  
       d.gerente = e.matricula and  
       p.local = 'Fortaleza'
```

SQL-DML - Exemplos

- **Q4.** Para cada empregado, recupere seu nome e o nome do seu supervisor

```
SELECT e.nome, s.nome  
FROM Empregado s, Empregado e  
WHERE e.matricula = s.supervisor
```

obs.: 'e' e 's' são variáveis tupla

- **Q5.** Selecione a matrícula de todos os empregados

```
SELECT matricula  
FROM Empregados
```


SQL-DML - Exemplos

- **Q6.** Faça o produto cartesiano, seguido de projeção de Empregados X Departamento retornando a matrícula do empregado e o nome do departamento

```
SELECT matricula, d.nome  
FROM Empregado, Departamento d
```

- **Q7.** Selecione todos os atributos de todos os empregados do departamento d5

```
SELECT *  
FROM Empregado  
WHERE depto = 'd5'
```

SQL-DML - Exemplos

- **Q8.** Selecione todos os atributos de todos os empregados do departamento pessoal

```
SELECT e.*  
FROM Empregado e, Departamento d  
WHERE d.nome = 'Pessoal' and d.coddep = e.depto
```

- **Q9.** Recupere os salários de cada empregado

```
SELECT salario  
FROM empregado
```

SQL-DML - Exemplos

- Algumas vezes surgem duplicatas como resposta a uma query. Podemos eliminá-las usando o comando DISTINCT na cláusula SELECT
- **Q10.** Selecione os diferentes salários pagos pela empresa aos empregados

```
SELECT DISTINCT salario  
FROM empregado
```

Operações de Conjuntos

- As operações de conjunto **union**, **intersect**, e **except** operam nas relações e correspondem às operações da álgebra relacional: \cup , \cap , $-$, respectivamente.
- Cada uma dessas operações elimina automaticamente duplicatas; para reter todas as duplicatas use **ALL: union all, intersect all e except all**.
- Suponha que uma tupla ocorre m vezes em r e n vezes em s , então, ela ocorre:
 - $m + n$ vezes em r union all s
 - $\min(m, n)$ vezes em r intersect all s
 - $\max(0, m - n)$ vezes em r except all s
- **OBS.: No Oracle ao invés de usar EXCEPT deve-se usar MINUS**

SQL-DML - Exemplos

- **Q11.** Liste todos os nomes de projetos que envolvem o empregado 'Silva' como trabalhador ou como gerente do departamento que controla o projeto.

```
(SELECT p.nome
FROM Projeto p, Departamento d, Empregado e
WHERE d.coddep = p.depart and
      d.gerente = e.matricula and
      e.nome = 'Silva')

UNION

(SELECT p.nome
FROM Projeto p, Alocação a, Empregado e
WHERE p.codproj = a.codproj and e.matricula = a.matricula
and e.nome = 'Silva')
```

SQL-DML - Exemplos

- **Consultas Aninhadas:** consultas que possuem consultas completas dentro de sua cláusula **WHERE**.
- **Q12:** A consulta Q11 poderia ser reescrita da seguinte forma:

SQL-DML - Exemplos

```
SELECT DISTINCT nome
FROM Projeto
WHERE codproj IN (SELECT codproj
                  FROM Projeto p, Departamento d, Empregado e
                  WHERE p.depart = d.coddep and
                      d.gerente = e.matricula and
                      e.nome = 'Silva')
```

OR

```
codproj IN (SELECT codproj
            FROM Alocação a, Empregado e, Projeto p,
            WHERE p.codproj = a.codproj and
                  e.matricula = a.matricula and
                  e.nome = 'Silva')
```

SQL-DML - Exemplos

- **Q13.** Recupere o nome de cada empregado que tem um dependente com o mesmo nome e mesmo sexo

```
SELECT e.nome
FROM empregado e
WHERE e.matricula IN
      (SELECT matricula
       FROM dependente d
       WHERE d.matricula = e.matricula
              And d.nome = e.nome
              And d.sexo = e.sexo)
```

Obs.: Veja que **e.matricula**, **e.nome** e **e.sexo** são atributos de **empregado** da consulta externa.

SQL-DML - Exemplos

- Q14. Re-escrevendo a Q13 sem usar aninhamento

```
SELECT e.nome  
FROM empregado e, dependente d  
WHERE e.matricula = d.matricula and  
       e.nome = d.nome and e.sexo = d.sexo
```

SQL-DML - Exemplos

- A construção **EXISTS**

- A construção **exists** retorna o valor *true* se o argumento da subquery é não vazio.
- Seja R uma relação qualquer:
 - $\text{exists } R \Leftrightarrow R \neq \emptyset$
 - $\text{not exists } R \Leftrightarrow R = \emptyset$

SQL-DML - Exemplos

- A construção **EXISTS**

- A consulta Q13 poderia ser:

```
SELECT e.nome  
FROM empregado e  
WHERE EXISTS (SELECT *  
               FROM dependente d  
               WHERE e.matricula = d.matricula  
               and e.nome = d.nome and e.sexo = d.sexo)
```

- Podemos usar o **NOT EXISTS(Q)**

SQL-DML - Exemplos

- **Q.15** Recupere os nomes dos empregados que não têm dependentes

```
SELECT e.nome  
FROM empregado e  
WHERE NOT EXISTS (SELECT *  
                  FROM dependente d  
                  WHERE e.matricula = d.matricula)
```

- Podemos usar um conjunto de valores explícitos:
 - **Q16.** Selecione a matricula de todos os empregados que trabalham nos projetos 10, 20 ou 30

```
SELECT DISTINCT matric  
FROM alocao  
WHERE codproj IN (10,20,30)
```

SQL-DML - Exemplos

Q17. Mostre os empregados que trabalham em todos os projetos do empregado com mat = 800.

```
SELECT mat
FROM empregado e
WHERE NOT EXISTS (
    ( SELECT codproj FROM alocao WHERE mat = 800)
    EXCEPT
    ( SELECT codproj FROM alocao a WHERE a.mat = e.mat)
)
```

Note que $X - Y = \emptyset \Leftrightarrow X \subseteq Y$

OBS.: No Oracle o operador diferença é **minus**

SQL-DML - Exemplos

- Podemos verificar valores nulos através de

IS NULL e IS NOT NULL:

- **Q18.** Selecione os nomes de todos os empregados que não têm supervisores

```
SELECT nome  
FROM empregado  
WHERE supervisor IS NULL
```

SQL-DML - Exemplos

▪ Funções

□ SQL fornece 5 funções embutidas:

- **COUNT**: retorna o número de tuplas ou valores especificados numa query
- **SUM**: retorna a soma os valores de uma coluna
- **AVG**: retorna a média dos valores de uma coluna
- **MAX**: retorna o maior valor de uma coluna
- **MIN**: identifica o menor valor de uma coluna

OBS.: Estas funções só podem ser usadas numa cláusula SELECT ou numa cláusula HAVING (a ser vista depois)

SQL-DML - Exemplos

- **Q19.** Encontre o total de salários, o maior salário, o menor salário e a média salarial da relação empregados

```
SELECT SUM(salario), MAX(salario),  
MIN(salario), AVG(salario)  
FROM Empregado
```

- **Q20.** Encontre o maior e menor salário do departamento de Produção

```
SELECT MAX(salario), MIN(salario)  
FROM Empregado e, Departamento d  
WHERE e.depto = d.coddep and  
      d.nome = 'Produção'
```


SQL-DML - Exemplos

- **Q.21** Obtenha o número de empregados da empresa

```
SELECT COUNT(*)  
FROM Empregado
```

- **Q.22** Obter o número de salários distintos do departamento de Contabilidade

```
SELECT COUNT(DISTINCT salario)  
FROM empregado e, departamento d  
WHERE (e.depto = d.coddep and d.nome = 'Contabilidade')
```

- O que aconteceria se escrevêssemos **COUNT(salario)** ao invés de **COUNT(DISTINCT salario))**?

SQL-DML - Exemplos

- **Q.23** Obter o nome dos empregados que tenham 2 ou mais dependentes

```
SELECT e.nome  
FROM empregado e  
WHERE (SELECT COUNT(*)  
       FROM Dependente d  
       WHERE e.matricula = d.matricula) >= 2)
```

References

Ex.: Uso da função `max` numa query dentro de um `SELECT` de outra query:

```
SELECT mat, salario , (SELECT MAX(salario)
                        FROM empleado)
FROM empleado;
```

SQL-DML - Exemplos

- Cláusula **GROUP BY, HAVING**

- Usadas para lidar com grupos.

- **Q24.** Para cada departamento, obter o código do departamento, o número de empregados e a média salarial

```
SELECT depto, COUNT(*), AVG(salario)
FROM Empregado
GROUP BY depto
```

- ⇒ as tuplas de empregados são separadas em grupos (departamento) e as funções **COUNT** e **AVG** são aplicadas a cada grupo separadamente.

SQL-DML - Exemplos

- **Q25.** Para cada projeto, obter o código do projeto, seu nome e o número de empregados que trabalham naquele projeto

```
SELECT p.codproj, p.nome, COUNT(*)  
FROM Projeto p, Alocacao a  
WHERE p.codproj = a.codproj  
GROUP BY p.codproj, p.nome
```

⇒ o agrupamento e as funções são aplicadas após a junção.

Exemplo Group By

Como saber quantas pessoas estão em cada curso?!

Vamos pensar na lógica

1. Temos de contar os alunos

2. Temos de contar separado por curso

O resultado seria este:

Cr	NroAlunos
CC	3
MC	2
SI	4
ECA	1

ALUNOS				CURSOS			
Matr	Nome	Sexo	Cr	Cr	Nome	Depto	Coord
1	A	F	CC	CC	Teoria da Linguagem	DCC	RG
2	B	M	CC	MC	Programação Computacional	DCC	TN
3	C	M	CC	SI	Tratamento da Informação	DCC	LUJ
4	D	F	MC	ECA	Linguagem de Programação	ENG	XYZ
5	E	M	MC				
6	F	M	SI				
7	G	F	SI				
8	H	F	SI				
9	I	M	SI				
10	J	M	ECA				

MATRICULAS			
Matr	Disc	T	Sem
1	DCC011	Z	20162
1	DCC851	A	20162
1	DCC834	A	20161
2	DCC011	Z	20161
...

Exemplo Group By

```
select Cr, count(*) as NroAlunos from ALUNOS  
group by Cr
```

Cr	NroAlunos
CC	3
MC	2
SI	4
ECA	1

ALUNOS				CURSOS			
Matr	Nome	Sexo	Cr	Cr	Nome	Depto	Coord
1	A	F	CC	CC	Contabilidade	DCC	RG
2	B	M	CC	MC	Marketing	DCC	TN
3	C	M	CC	SI	Sistemas de Informação	DCC	CDJ
4	D	F	MC	ECA	Engenharia de Computação	ENG	XYZ
5	E	M	MC				
6	F	M	SI				
7	G	F	SI				
8	H	F	SI				
9	I	M	SI				
10	J	M	ECA				

MATRICULAS			
Matr	Disc	T	Sem
1	DCC011	7	20162
1	DCC851	A	20162
1	DCC834	A	20161
2	DCC011	Z	20161
...

Exemplo Group By

```
select Cr, count(*) as NroAlunos from ALUNOS  
group by Cr
```

ALUNOS

Matr	Nome	Sexo	Cr
1	A	F	CC
2	B	M	CC
3	C	M	CC
4	D	F	MC
5	E	M	MC
6	F	M	SI
7	G	F	SI
8	H	F	SI
9	I	M	SI
10	J	M	ECA

Cr	NroAlunos
CC	3
MC	2
SI	4
ECA	1

Exemplo Group By

```
select Cr, Sexo, count(*) as NroAlunos from ALUNOS  
group by Cr, Sexo
```

ALUNOS

Matr	Nome	Sexo	Cr
1	A	F	CC
2	B	M	CC
3	C	M	CC
4	D	F	MC
5	E	M	MC
6	F	M	SI
7	G	F	SI
8	H	F	SI
9	I	M	SI
10	J	M	ECA

Cr	Sexo	NroAlunos
CC	F	1
CC	M	2
MC	F	1
MC	M	1
SI	F	2
SI	M	2
ECA	M	1

SQL-DML - Exemplos

- **HAVING**

- usada em conjunto com **GROUP BY** para permitir a inclusão de condições nos grupos

- **Q.26.** Para cada projeto que possui mais de 2 empregados trabalhando, obter o código do projeto, nome do projeto e número de empregados que trabalha neste projeto

```
SELECT p.codproj, p.nome, COUNT(*)  
FROM Projeto p, Alocacao a  
WHERE p.codproj = a.codproj  
GROUP BY p.codproj, p.nome  
HAVING COUNT(*) > 2
```

- Uma query é avaliada primeiro aplicando a cláusula **WHERE** e depois **GROUP BY** **HAVING**

Ex. Group by com Having

Como saber quantas pessoas estão em cada curso 👍
Apenas cursos com mais de 3 matrículas???

```
select Cr, count(*) as NroAlunos from ALUNOS  
group by Cr  
having NroAlunos > 3
```

Ou seja, **having** adiciona uma condição para o grupo entrar no resultado da consulta!

Ainda de outra forma: apenas grupos com **having true** entram no resultado

ALUNOS				CURSOS			
Matr	Nome	Sexo	Cr	Cr	Nome	Depto	Coord
1	A	F	CC	CC	Curso de Engenharia	DCC	RG
2	B	M	CC	MC	Curso de Matemática	DCC	IN
3	C	M	CC	SI	Curso de Sistemas de Informação	DCC	CDI
4	D	F	MC	ECA	Engenharia de Computação e Automação	ENG	XYZ
5	E	M	MC				
6	F	M	SI				
7	G	F	SI				
8	H	F	SI				
9	I	M	SI				
10	I	M	ECA				

MATRICULAS			
Matr	Disc	T	Sem
1	DCC011	Z	20162
1	DCC851	A	20162
1	DCC834	A	20161
2	DCC011	Z	20161

SQL-DML - Exemplos

- Operadores de Comparação e Aritméticos

- BETWEEN:**

- Sintaxe:

expressão [NOT] BETWEEN expressão AND expressão

- Ex.:

y BETWEEN x AND Z equivale a $x \leq y \leq z$

- Q.27** Selecione os nomes dos empregados que ganham mais de 1000 e menos de 2000 reais

```
SELECT nome
FROM Empregado
WHERE salario BETWEEN 1000 AND 2000
```

SQL-DML - Exemplos

- **LIKE:**

- Permite comparações de substrings. Usa dois caracteres reservados ‘%’ (substitui um número arbitrário de caracteres) e ‘_’ (substitui um único caracter).

- **Q.28** Obter os nomes de empregados cujos endereços estão em Natal, RN

```
SELECT nome  
FROM empregado  
WHERE endereco LIKE '%Natal,RN%'
```

- Existem várias outras funções para se trabalhar com Strings: **SUBSTRING()**, **UPPER()**, **LOWER()**, ...

SQL-DML - Exemplos

- **UPPER (e LOWER):**

- Os dados em SQL são Case sensitive
 - Assim: 'Zé' <> 'zé' <> 'ze' <> 'ZE'
- Colocam o string em maiúsculo (e minúsculo)

- **Q.29** Obter os nomes de empregados cujos endereços estão em Natal, RN

```
SELECT nome  
FROM empregado  
WHERE UPPER (endereço) LIKE  
'%NATAL,RN%'
```

```
SELECT nome  
FROM empregado  
WHERE LOWER (endereço) LIKE  
'%natal,rn%'
```

SQL-DML - Exemplos

■ REGEX:

- Implementam expressões regulares (busca por padrões em strings)
- Alguns SGBDs seguem o padrão POSIX de REGEX
 - <http://www.opengroup.org/onlinepubs/007908799/xbd/re.html>
- Podem usar:
 - Metacaracteres (operadores que especificam o algoritmo de busca)
 - Literais (caracteres a serem buscados)
 - Ex: A expressão regular: **(f|ht)tps?:**
casa com os strings: **http:**, **https:**, **ftp:** e **ftps:**

SQL-DML - Exemplos

▪ REGEX no Oracle

- ❑ **REGEXP_LIKE (condição)**: usado na cláusula WHERE (ou em um check), retorna as linhas que casam com a expressão regular passada na condição.
- ❑ Ex. Filtrar a coluna nome por 'Steven' ou 'Stephen':

```
SELECT ... WHERE REGEXP_LIKE( nome, '^Ste[v|ph]en$')
```
- ❑ **REGEXP_REPLACE (função)**: troca a ocorrência de uma padrão em um string por um novo padrão passado como parâmetro.
- ❑ Pode ser usado num SELECT
- ❑ Ex. Colocar um espaço entre cada caracter da coluna nome_país:

```
SELECT REGEXP_REPLACE (nome_pais, '(.)', '\1 ') FROM Países
```

Veja que o '.' significa qualquer caracter

E o '\1' substitui a primeira subexpressão (o primeiro grupo de parêntesis no padrão).

SQL-DML - Exemplos

- **REGEX no Oracle (cont.)**

- Ex. Forçando que os números de telefones tenham o formato (XXX) XXX-XXXX

```
CREATE TABLE Contatos (  
    nome VARCHAR(30),  
    telefone VARCHAR(30),  
    CONSTRAINT formato_fone  
        CHECK ( REGEXP_LIKE ( telefone, '^(\d{3}) \d{3}-\d{4}$' ) )  
);
```

Documentação sobre REGEX no Oracle:

https://docs.oracle.com/cd/B19306_01/B14251_01/adfns_regexp.htm

SQL-DML - Exemplos

- **Q30.** Queremos ver o efeito de dar aos empregados que trabalham no ProdutoX um aumento de 10%

```
SELECT e.nome, 1.1*salario
FROM empregado e, alocao a, projeto p
WHERE e.matricula = a.matricula and
      a.codproj = p.codproj and
      p.nome = 'ProdutoX'
```

SQL-DML - Exemplos

■ Ordenação

- O operador **ORDER BY** permite ordenar o resultado de uma query por um ou mais atributos.
- **Q.31** Obter uma lista de empregados e seus respectivos departamentos e projetos, listando ordenado pelo nome do departamento

```
SELECT d.nome, e.nome, p.nome
FROM departamento d, empregado e, projeto p,
alocação a
WHERE d.coddep = e.depto AND
      e.matricula = a.matricula AND
      a.codproj = p.codproj
ORDER BY d.nome, e.nome
```

SQL-DML - Exemplos

▪ Ordenação

⇒ A ordem default é ascendente (**ASC**) caso queiramos ordem decrescente usamos **DESC**

□ Ex.

```
ORDER BY d.nome DESC, e.nome ASC
```