

Essencial Database

## **Funções de Grupo e Subconsultas**

Professora: Lúcia Contente Mós

Tutor: Marcelo Estruc

### **Introdução**

As funções de grupo operam em conjuntos de linhas para fornecer um resultado por grupo. As operações das funções de grupo podem envolver todas as linhas de uma tabela ou conjuntos de linhas.

Uma subconsulta é uma consulta utilizada dentro de uma instrução SQL. Pode ser utilizada dentro de instruções select, insert, delete update ou create table.

### **Objetivos da aula**

- Desenvolver consultas complexas com o uso de funções grupo;
- Implantar as restrições nos grupos de dados com a cláusula Having;
- Identificar e Realizar Subconsultas de uma única linha;
- Identificar e Realizar Subconsultas de várias linhas;
- Identificar e Realizar Subconsultas de várias colunas.

### **Resumo**

#### **Funções de Grupo**

As funções de grupo operam em conjuntos de linhas para fornecer um resultado por grupo. As operações das funções de grupo podem envolver todas as linhas de uma tabela ou conjuntos de linhas definidos por critérios pré-estabelecidos.

Assim como as funções de diversas outras linguagens ou aplicativos as funções disponíveis na SQL requerem argumentos para realização das operações e

retornam valores; neste caso os argumentos serão representados pelo nome da coluna: FUNÇÃO(coluna).

Algumas funções disponíveis:

**AVG ( )** – Retorna a média obtida entre os valores de um conjunto

**COUNT ( )** – Retorna a quantidade de ocorrências

**MAX ( )** – Retorna o maior valor de um conjunto

**MIN ( )** – Retorna o menor valor de um conjunto

**SUM( )** – Retorna a somatória dos valores de um conjunto

**VARIANCE( )** – Retorna a variância entre os valores de um conjunto

**Sintaxe:**

```
SELECT  [coluna,]  função_de_grupo(coluna)
FROM    tabela
[WHERE   condição]
[GROUP BY      coluna]
[ORDER BY      coluna];
```

onde:

[coluna,]

lista de colunas envolvidas na consulta, é opcional

função\_de\_grupo(coluna)

indica o nome da função que será utilizada e que os dados da coluna especificada serão passados como parâmetro para a função

FROM tabela

tabela ou tabelas utilizadas na consulta

[WHERE condição]

condição para realização da consulta, limita o conjunto de dados que irão compor o conjunto.

[GROUP BY *coluna*]

[HAVING condição]

Cria grupos de dados

limita os grupos a serem mostrados, é similar à cláusula where, mas aplica-se somente à colunas que tenham valores agrupados

[ORDER BY *coluna*];

A ordenação é por default ascendente.

- Todas as funções de grupo, exceto COUNT(\*), ignoram valores nulos. Para substituir um valor por valores nulos, use a função NVL.

O agrupamento simples envolve todo o conjunto de uma determinada tabela, isto é, considera todas as linhas que satisfazem um critério e cada função envolvida produz um único resultado para o conjunto.

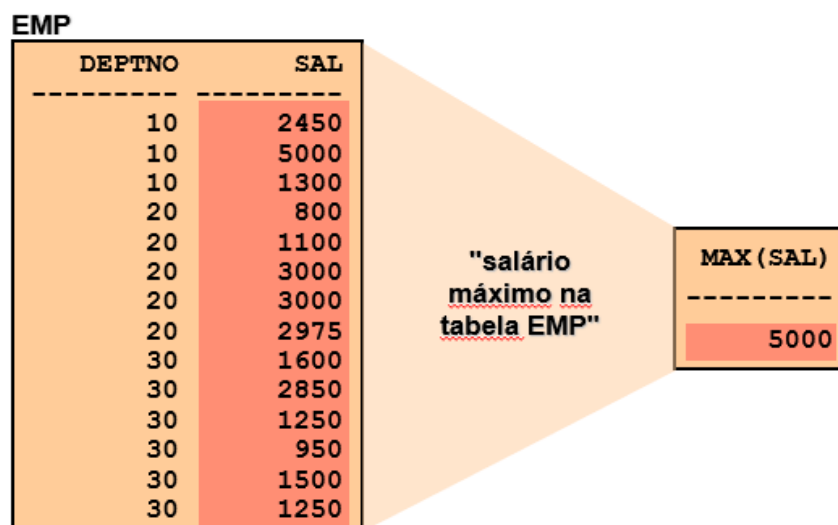


Figura 1 - Função de Grupo Max - Fonte: Goya, 2013.

**Exemplo 1:** Verificar o maior salário do conjunto; neste caso todas as linhas da tabela seriam avaliadas e apenas um valor retornaria como resultado;

```
Select MAX(sal)
from scott.EMP;
```

**Exemplo 2:** Calcular a média salarial, o maior salário, o menor salário e a somatória dos salários de todos os funcionários que possuem a cadeia de caracteres 'SALES' como parte do cargo.

```
SELECT AVG(sal), MAX(sal), MIN(sal), SUM(sal)
FROM      scott.emp
WHERE    job LIKE 'SALES%';
```

**Resultado:**

<b>AVG(SAL)</b>	<b>MAX(SAL)</b>	<b>MIN(SAL)</b>	<b>SUM(SAL)</b>
1400	1600	1250	5600

**Exemplo 3:** Exibir a quantidade de funcionários que trabalham no departamento 30.

```
SELECT  COUNT(*)
FROM    scott.emp
WHERE  deptno = 30;
```

A Função COUNT tem dois formatos: COUNT(\*) e COUNT(*expr*).

COUNT(\*) retorna o número de linhas em uma tabela, inclusive linhas duplicadas e linhas contendo valores nulos em qualquer uma das colunas. Se uma cláusula WHERE estiver incluída na instrução SELECT, COUNT(\*) retornará o número de linhas que satisfizer a condição na cláusula WHERE.

COUNT(*expr*) retorna o número de linhas não nulas na coluna identificada por *expr*.

## Agrupando Resultados:

DEPTNO	SAL
10	2450
10	5000
10	1300
20	800
20	1100
20	3000
20	3000
20	2975
30	1600
30	2850
30	1250
30	950
30	1500
30	1250

2916.6667

2175

1566.6667

"salário médio na tabela EMP para cada departamento"

DEPTNO	AVG (SAL)
10	2916.6667
20	2175
30	1566.6667

Figura 2 - Cláusula Group by - Fonte: Goya, 2013.

**Exemplo 4:** Exibir a média salarial por departamento.

```
SELECT deptno, AVG(sal)
FROM scott.emp
GROUP BY deptno;
```

**Resultado:**

DEPTNO	AVG(SAL)
10	2916.6667
20	2175
30	1566.6667

Para limitar o resultado de linhas que estarão envolvidas no agrupamento deve-se primeiro utilizar a cláusula WHERE e depois a cláusula GROUP BY.

Todas as colunas individuais envolvidas na consulta, isto é, que não estão participando de funções de grupo, devem ser incluídas na cláusula GROUP BY.

Não é possível usar o apelido de coluna na cláusula GROUP BY.

Por default, as linhas são classificadas por ordem crescente das colunas incluídas na lista GROUP BY. Isso pode ser sobreposto usando a cláusula ORDER BY.

A coluna GROUP BY não precisa estar na cláusula SELECT.

Pode-se utilizar a função de grupo na cláusula ORDER BY.

**Exemplo 5:** Exiba a somatória dos salários por departamento e cargo.

```
SELECT deptno, job, sum(sal)
```

```
FROM emp
```

```
GROUP BY deptno, job;
```

**Resultado:**

DEPTNO	JOB	SUM(SAL)
--------	-----	----------

-----	-----	-----
-------	-------	-------

10	CLERK	1300
----	-------	------

10	MANAGER	2450
----	---------	------

30	CLERK	950
----	-------	-----

30	MANAGER	2850
----	---------	------

30	SALESMAN	5600
----	----------	------

...

No exemplo 5 primeiro, as linhas são agrupadas pelo número do departamento. Em seguida, dentro dos grupos de números de departamentos, as linhas são agrupadas pelo cargo.

Dessa forma, a função SUM é aplicada à coluna de salários para todos os cargos dentro de cada grupo de números de departamentos.

## Restringindo resultados do grupo

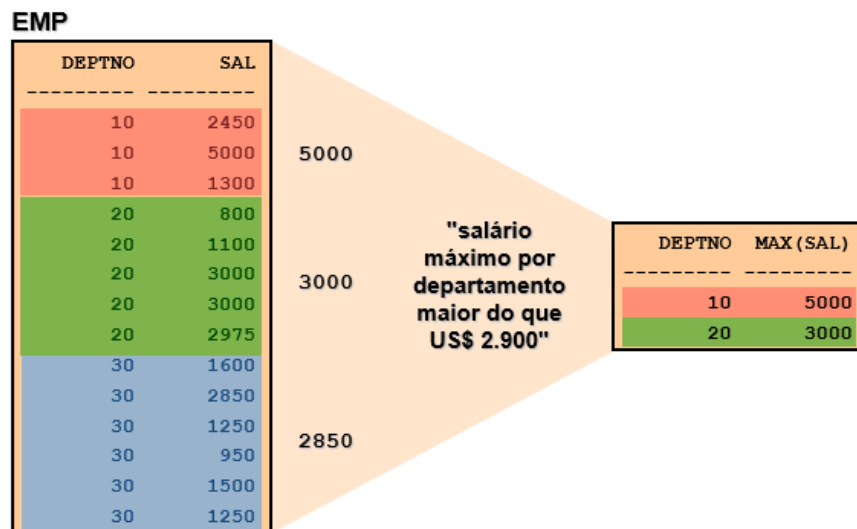


Figura 3 - Cláusula Having - Fonte: Goya, 2013.

Da mesma forma que se usa a cláusula WHERE para restringir as linhas que serão selecionadas, pode-se usar a cláusula HAVING para restringir grupos.

As seguintes etapas são executadas quando a cláusula HAVING é utilizada:

- As linhas são agrupadas.
- A função de grupo é aplicada ao grupo.
- Os grupos que correspondem aos critérios na cláusula HAVING são exibidos.

**Exemplo 6:** Exibir os números de departamentos e o salário máximo para os departamentos, cujo salário máximo seja maior do que 2.900.

```
SELECT deptno, max(sal)
FROM scott.emp
GROUP BY deptno
HAVING max(sal)>2900;
```

### Resultado:

DEPTNO	MAX(SAL)
10	5000
20	3000

## Subconsultas

Uma subconsulta é uma consulta utilizada dentro de uma instrução SQL. Pode ser utilizada dentro de instruções select, insert, delete update ou create table. Pode ser do tipo:

- **Subconsultas de uma única linha:** consultas que retornam somente uma linha da instrução SELECT interna.
- **Subconsultas de várias linhas:** consultas que retornam mais de uma linha da instrução SELECT interna.
- **Subconsultas de várias colunas:** consultas que retornam mais de uma coluna da instrução SELECT interna

### Subconsulta em Consultas

O uso de subconsultas em Consultas é útil quando a consulta principal requer valores desconhecidos, por exemplo: Suponha que seja necessário criar uma consulta para descobrir quem recebe um salário maior que o salário de Jones, neste caso **qual é o salário de Jones?**

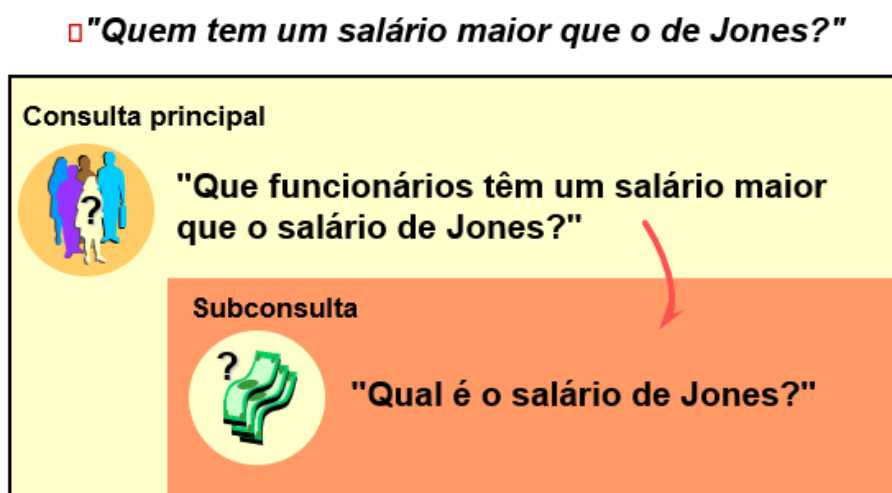


Figura 4 - Exemplo de Subconsulta - Goya, 2013.

Para resolver esse problema, são necessárias *duas* consultas: uma para descobrir quanto Jones recebe e outra para descobrir quem recebe mais do que ele.



**Sintaxe:**

```
SELECT  colunas,  
FROM    tabela  
WHERE   condição operador expr  
        (SELECT  select_list  
          FROM    tabela);
```

onde:

A condição envolve uma operação de comparação entre uma coluna e o resultado que será retornado pela subconsulta. A subconsulta é uma consulta, portanto, pode ser construída de acordo com o problema apontado e incluir condições, funções de grupo, várias colunas etc.

A subconsulta (consulta interna) é executada uma vez antes da consulta principal.

É possível colocar a subconsulta em várias cláusulas SQL:

- cláusula WHERE
  - cláusula HAVING
  - cláusula FROM
- 
- As subconsultas devem estar entre parênteses e ao lado direito do operador de comparação.
  - Não utilizar uma cláusula ORDER BY a uma subconsulta.
  - Utilizar operadores de uma única linha com subconsultas de uma única linha.
  - Utilizar operadores de várias linhas com subconsultas de várias linhas.

**Subconsultas de uma linha**

Podem ser utilizados os operadores relacionais <, >, <>, >=, <= e = para subconsultas que retornam uma linha.

Nos exemplos 1, 2, 3 e 4, são apresentadas diversas situações para subconsultas de uma linha.

Operador	Significado
=	Igual a
>	Maior do que
>=	Maior do que ou igual a
<	Menor do que
<=	Menor ou igual a
<>	Diferente de

Fonte: autoral, 2023.

**Exemplo 1** - Utilizando subconsultas em condições: Exiba o nome de todos os funcionários cujo salário é maior do que o salário do funcionário 7566.

```
SELECT ename
FROM scott.emp
WHERE sal >
      (SELECT sal
       FROM scott.emp
       WHERE empno = 7566);
```

**Exemplo 2** - Utilizando subconsultas em condições compostas: Exiba o nome e o cargo dos funcionários que cujo cargo é igual ao cargo do funcionário 7369 e o salário é maior do que o salário do funcionário 7876.

```
SELECT ename, job
FROM scott.emp
WHERE job =
      (SELECT job
       FROM scott.emp
       WHERE empno = 7369)
AND sal >
      (SELECT sal
       FROM scott.emp
       WHERE empno = 7876);
```

**Exemplo 3** - Utilizando subconsultas com funções de grupo: Exiba o nome, cargo e salário de todos os funcionários com o salário igual ao menor salário recebido pelos funcionários.

```
SELECT    ename, job, sal
FROM      scott.emp
WHERE     sal =
          (SELECT  MIN(sal)
           FROM scott.emp);
```

**Exemplo 4** - Utilizando subconsultas com funções de grupo na cláusula having: Exiba o menor salário por departamento quando o menor salário for menor do que o menor salário do departamento 20.

```
SELECT deptno, MIN(sal)
FROM scott.emp
GROUP BY deptno
HAVING MIN(sal) >
      (SELECT MIN(sal)
       FROM scott.emp
       WHERE deptno = 20);
```

Um erro comum em subconsultas ocorre quando se utiliza um operador simples para um retorno de várias linhas.

### **Operadores para Subconsulta de várias linhas**

Operador	Significado
IN	Igual a qualquer membro na lista
ANY	Compare o valor a cada valor retornado pela subconsulta
ALL	Compare o valor a todo valor retornado pela subconsulta

Figura 6 - Operadores de subconsulta para várias linhas - Fonte: Goya, 2013.

### Exemplo de Uso do Operador Any em subconsulta de várias linhas

A leitura desse operador é “qualquer valor” retornado pela consulta interna. Observe, que no exemplo abaixo, serão selecionados os funcionários que ganham mais que qualquer funcionário que tenha o cargo clerk.

SQL>	SELECT	empno, ename, job	1300
2	FROM	emp	1100
3	WHERE	sal < ANY	800
4			950
5		(SELECT	sal
6		FROM emp	
7	AND	WHERE job = 'CLERK')	

EMPNO	ENAME	JOB
7654	MARTIN	SALESMAN
7521	WARD	SALESMAN

Figura 7- Uso do Operador Any - Fonte: PUGA, 2013.

### Exemplo de Uso do Operador All em subconsulta de várias linhas

A leitura desse operador é “todos os valores” retornados pela consulta interna. Observe, que no exemplo abaixo, serão selecionados os funcionários que ganham mais que todas as médias salariais por departamento.

SQL>	SELECT	empno, ename, job	1566.6667
2	FROM	emp	2175
3	WHERE	sal > ALL	2916.6667
4		(SELECT avg(sal)	
5		FROM emp	
6		GROUP BY deptno);	

EMPNO	ENAME	JOB
7839	KING	PRESIDENT
7566	JONES	MANAGER
7902	FORD	ANALYST
7788	SCOTT	ANALYST

Figura 8-Uso do Operador ALL - Fonte: PUGA, 2013.

## Exemplo de Uso da subconsulta com o comando create table (“cópia de tabela”)

Perceba, que no exemplo abaixo, a tabela copia emp é criada com toda a estrutura e os dados da tabela emp. É importante ressaltar que esse comando não copia as constraints.

```
create table copia_emp as (select * from emp);
```

```
select * from copia_emp;
```

Resultado da Consulta x Saída do Script x

Tarefa concluída em 1,068 segundos

Table COPIA\_EMP criado.

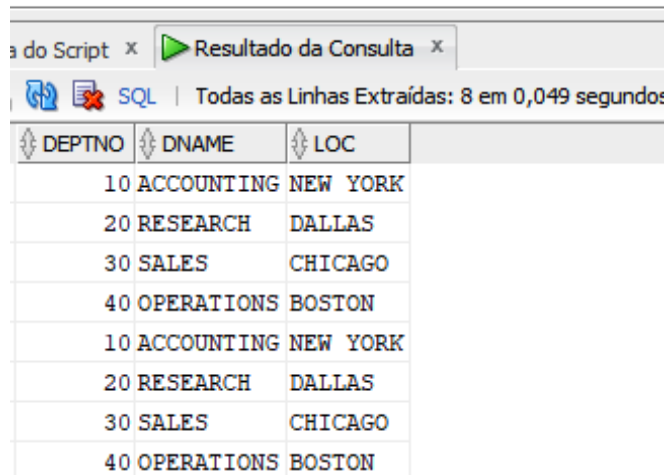
EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7369	SMITH	CLERK	7902	17/12/80	800		20
7499	ALLEN	SALESMAN	7698	20/02/81	1600	300	30
7521	WARD	SALESMAN	7698	22/02/81	1250	500	30
7566	JONES	MANAGER	7839	02/04/81	2975		20
7654	MARTIN	SALESMAN	7698	28/09/81	1250	1400	30
7698	BLAKE	MANAGER	7839	01/05/81	2850		30
7782	CLARK	MANAGER	7839	09/06/81	2450		10
7788	SCOTT	ANALYST	7566	19/04/87	3000		20
7839	KING	PRESIDENT		17/11/81	5000		10

Figura 9 - Uso de Subconsulta no Create table - Fonte: Autor, 2023.

## Exemplo de Uso da subconsulta com o comando insert (“cópia de registros”)

Perceba, que no exemplo abaixo, os registros da tabela dept são inseridos na tabela cópia dept.

```
insert into copia_dept (select * from dept);  
select * from copia_dept;
```



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

Figura 10 - Uso de Subconsulta com o Insert - Fonte: Autor, 2023.

## Conteúdo bônus

### Gerando Subtotais com o operador Rollup

Para gerar subtotais nas colunas da cláusula group by, utilize o operador Rollup.

```
SELECT deptno, job, sum(sal)
FROM scott.emp
GROUP BY ROLLUP(deptno, job);
```

DEPTNO	JOB	SUM(SAL)
10	CLERK	1300
10	MANAGER	2450
10	PRESIDENT	5000
10		8750
20	ANALYST	6000
20	CLERK	1900
20		7900
...		

9 rows selected.

Figura 11 - Operador Rollup - Fonte: Goya, 2013.

## Gerando Subtotais com o operador Cube

O operador cube gera subtotais em todas as possibilidades das colunas citadas na cláusula group by.

<b>SELECT deptno, job, sum(sal)</b>		
<b>FROM scott.emp</b>		
<b>GROUP BY CUBE(deptno, job);</b>		
DEPTNO	JOB	SUM(SAL)
		29025
	CLERK	4150
	ANALYST	6000
	MANAGER	8275
	SALESMAN	5600
	PRESIDENT	5000
10		8750
10	CLERK	1300
10	MANAGER	2450
10	PRESIDENT	5000
20		10875
20	CLERK	1900
20	ANALYST	6000
20	MANAGER	2975
30		9400
30	CLERK	950
30	MANAGER	2850
30	SALESMAN	5600

Figura 12 - Operador Cube - Fonte: PUGA, 2013.

Os operadores rollup e cube são muito utilizados em operações de mineração de dados.

## Referências Bibliográficas

DATE, C. J. **Introdução a sistemas de banco de dados**. Rio de Janeiro. Ed. Campus, 1991.

CHEN, Peter. **Modelagem de dados: a abordagem entidade-relacionamento para projeto lógico**. São Paulo: Makron Books, 1990.

MEDEIROS, L. F. **Banco de dados, princípios e práticas**, 1ª. ed., Ed. Intersaberes, 2013.

PUGA, S., França E., GOYA M., **Banco de dados: Implementação em SQL, PL/SQL e Oracle 11g**, Ed. Pearson, 2013.

ELMASRI R., NAVATHE, S. **Sistemas de Banco de Dados**, 4ª ed. Ed. Pearson, 2005.