



SQL

Linguagem de Manipulação de Dados

Profa. Maria Camila Nardini Barioni
camila.barioni@facom.ufu.br

Bloco B - sala 1B137

1º semestre de 2024

SQL DML – CONTINUAÇÃO...

SQL DML

◆ SELECT ... FROM ... WHERE ...

- lista atributos de uma ou mais tabelas de acordo com alguma condição

◆ INSERT INTO ...

- insere dados em uma tabela

◆ DELETE FROM ... WHERE ...

- remove dados de tabelas já existentes

◆ UPDATE ... SET ... WHERE ...

- altera dados específicos de uma tabela

SQL DML

◆ SELECT ... FROM ... WHERE ...

- lista atributos de uma ou mais tabelas de acordo com alguma condição

◆ INSERT INTO ...

- insere dados em uma tabela

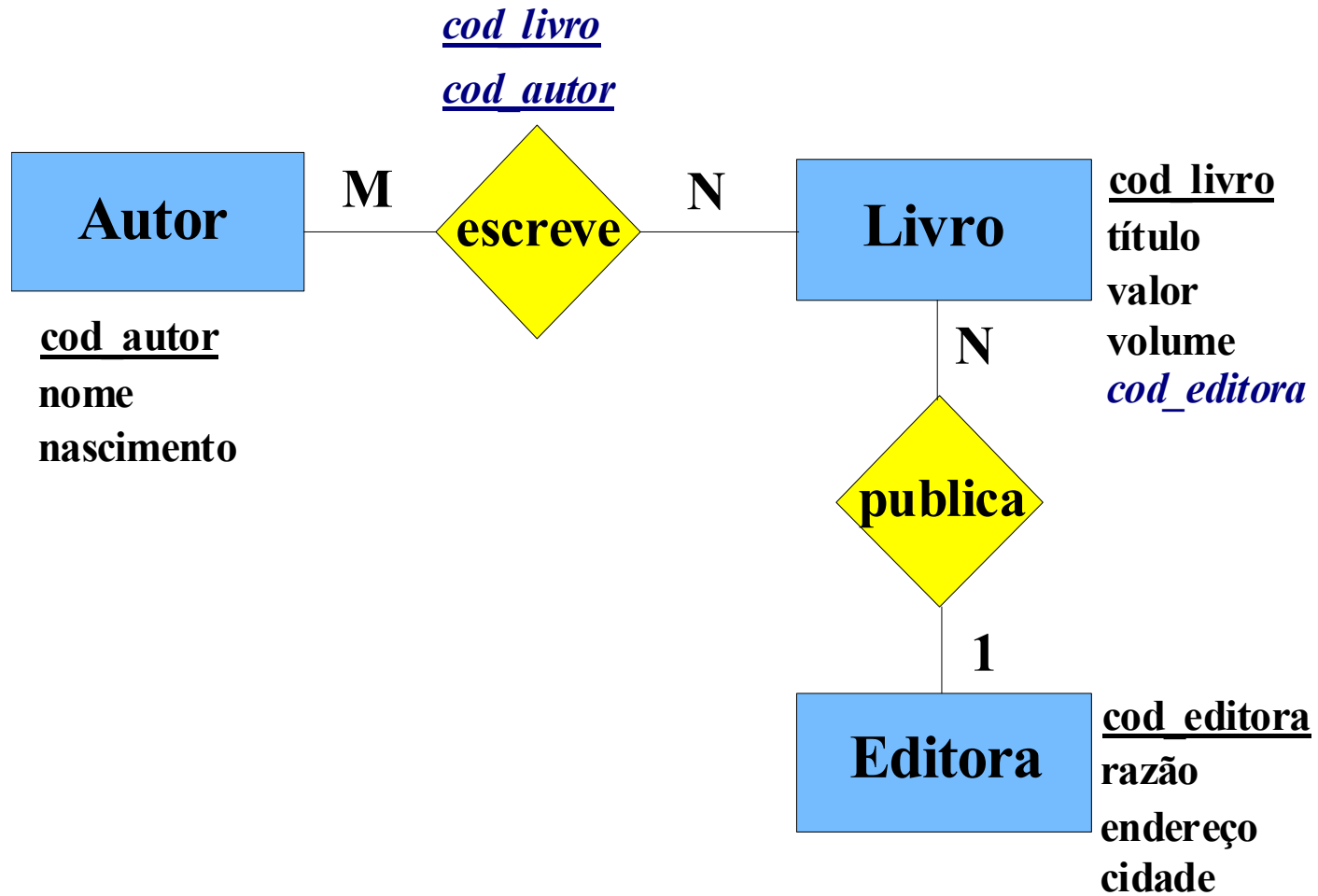
◆ DELETE FROM ... WHERE ...

- remove dados de tabelas já existentes

◆ UPDATE ... SET ... WHERE ...

- altera dados específicos de uma tabela

Exemplo ME-R



Exemplo Modelo Relacional

AUTOR = { COD AUTOR, NOME, NASCIMENTO }



ESCREVE = { COD AUTOR, COD LIVRO }



LIVRO = { COD LIVRO, TITULO, VALOR,
VOLUME, COD_EDITORA }



EDITORIA = { COD EDITORA, RAZAO, ENDERECO,
CIDADE }

Funções de Agregação

Function	Argument Type(s)	Return Type	Description
<code>array_agg(expression)</code>	any	array of the argument type	input values, including nulls, concatenated into an array
<code>avg(expression)</code>	smallint, int, bigint, real, double precision, numeric, or interval	numeric for any integer-type argument, double precision for a floating-point argument, otherwise the same as the argument data type	the average (arithmetic mean) of all input values
<code>bit_and(expression)</code>	smallint, int, bigint, or bit	same as argument data type	the bitwise AND of all non-null input values, or null if none
<code>bit_or(expression)</code>	smallint, int, bigint, or bit	same as argument data type	the bitwise OR of all non-null input values, or null if none
<code>bool_and(expression)</code>	bool	bool	true if all input values are true, otherwise false
<code>bool_or(expression)</code>	bool	bool	true if at least one input value is true, otherwise false
<code>count(*)</code>		bigint	number of input rows
<code>count(expression)</code>	any	bigint	number of input rows for which the value of <i>expression</i> is not null
<code>every(expression)</code>	bool	bool	equivalent to <code>bool_and</code>
<code>max(expression)</code>	any array, numeric, string, or date/time type	same as argument type	maximum value of <i>expression</i> across all input values
<code>min(expression)</code>	any array, numeric, string, or date/time type	same as argument type	minimum value of <i>expression</i> across all input values
<code>string_agg(expression, delimiter)</code>	(text, text) or (bytea, bytea)	same as argument types	input values concatenated into a string, separated by delimiter
<code>sum(expression)</code>	smallint, int, bigint, real, double precision, numeric, or interval	bigint for smallint or int arguments, numeric for bigint arguments, double precision for floating-point arguments, otherwise the same as the argument data type	sum of <i>expression</i> across all input values
<code>xmlagg(expression)</code>	xml	xml	concatenation of XML values (see also Section 9.14.1.7)

<https://www.postgresql.org/docs/current/functions-aggregate.html>

Funções de Agregação

Detalhamento e exemplos

- ◆ **MIN()** : Retorna o mínimo de um conjunto de valores contido em um campo especificado em uma consulta.
- ◆ **MAX()** : Retorna o máximo de um conjunto de valores contido em um campo especificado em uma consulta.

-- Verificar o valor do livro mais caro

```
SELECT MAX (VALOR)
```

```
FROM LIVRO;
```

-- E se quisermos saber o título do livro mais caro?

Funções de Agregação

Detalhamento e exemplos

- E se quisermos saber o título do livro mais caro?
- Alguns alunos poderiam ter pensado na seguinte
- solução

```
SELECT TITULO FROM LIVRO WHERE VALOR = MAX(VALOR) ;  
-- Funciona?
```

Funções de Agregação

Detalhamento e exemplos

-- E se quisermos saber o título do livro mais caro?
-- Alguns alunos poderiam ter pensado na seguinte
-- solução

```
SELECT TITULO FROM LIVRO WHERE VALOR = MAX(VALOR) ;
```

-- Funciona?

-- A função MAX não pode ser usada na cláusula WHERE

-- e agora?

```
SELECT TITULO, VALOR
```

```
    FROM LIVRO
```

```
    WHERE VALOR = (SELECT MAX(VALOR) FROM LIVRO) ;
```

-- OK!

Funções de Agregação

Detalhamento e exemplos

◆ **SUM()**: Retorna a soma de um conjunto de valores contido em um campo especificado em uma consulta. A função `SUM` ignora os registros que contenham campos Null

```
-- calcular o preço total dos livros de Paulo Coelho
```

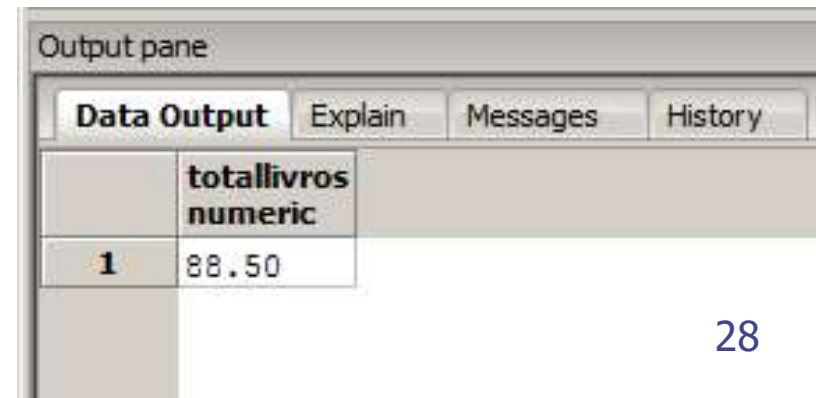
Funções de Agregação

Detalhamento e exemplos

- ◆ **SUM()**: Retorna a soma de um conjunto de valores contido em um campo especificado em uma consulta. A função SUM ignora os registros que contenham campos Null

-- calcular o preço total dos livros de Paulo Coelho

```
SELECT SUM(V valor) AS TotalLivros
FROM LIVRO, AUTOR, ESCREVE
WHERE NOME = 'PAULO COELHO' AND
      COD_AUTOR = COD_AUTOR_ESC AND
      COD_LIVRO_ESC = COD_LIVRO;
```



Output pane	
Data Output	
Explain	
Messages	
History	
	totallivros numeric
1	88.50

Funções de Agregação

Detalhamento e exemplos

- ◆ **AVG()**: Calcula a média aritmética de um conjunto de valores contido em um campo especificado em uma consulta

-- calcular o preço médio dos livros da livraria

```
SELECT AVG (VALOR) AS Media  
FROM LIVRO;
```

Funções de Agregação

Detalhamento e exemplos

◆ **COUNT()**: Calcula o número de registros retornado por uma consulta. A função COUNT não conta registros que tenham campos NULL, exceto quando expr for o caractere curinga asterisco (*)

-- contar quantos autores estão cadastrados

```
SELECT COUNT(*) FROM AUTOR;
```

-- contar quantos livros possuem informação de volume

```
SELECT COUNT(VOLUME)
```

```
FROM LIVRO;
```

-- observe que registros nos quais VOLUME é NULL não são considerados

Funções de Agregação

Detalhamento e exemplos

◆ COUNT()

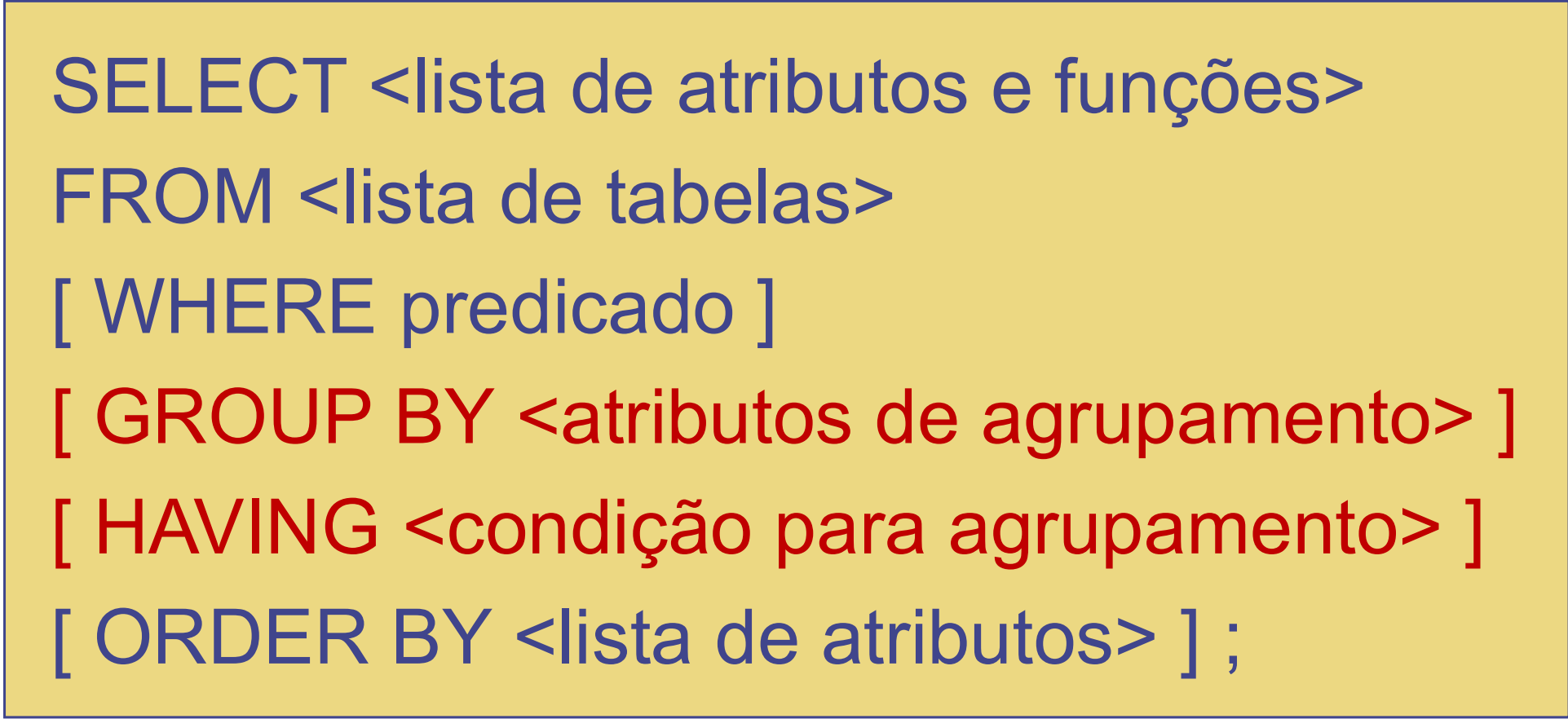
- é possível usar a palavra chave DISTINCT. Desta forma, os valores duplicados nas colunas não são contados.

```
/* qual o número de autores que possuem livros na livraria */  
SELECT COUNT(DISTINCT COD_AUTOR_ESC) AS AUTORES  
FROM ESCREVE;
```

Output pane		
	Data Output	Explain Messages History
	cod_livro_esc smallint	cod_autor_esc smallint
1	31	1
2	55	1
3	63	1
4	14	2
5	13	3

	Data Output	Explain Messages History
	autores bigint	
1	3	

SELECT



```
SELECT <lista de atributos e funções>  
FROM <lista de tabelas>  
[ WHERE predicado ]  
[ GROUP BY <atributos de agrupamento> ]  
[ HAVING <condição para agrupamento> ]  
[ ORDER BY <lista de atributos> ] ;
```

<http://www.postgresql.org/docs/9.2/static/sql-select.html>

Cláusula GROUP BY

- ◆ Podemos dividir o conjunto de tuplas de uma relação em grupos de acordo com algum critério, baseado nos valores dos atributos
 - Por exemplo, na tabela abaixo as tuplas podem ser agrupadas de acordo com o nome do autor
 - Exemplo: PAULO COELHO, MACHADO DE ASSIS e JOSÉ MARIA

Output pane

	titulo character(35)	nome character varying(20)
1	MAKTUB	PAULO COELHO
2	BRIDA	PAULO COELHO
3	HISTÓRIAS PARA PAIS, FILHOS E NETOS	PAULO COELHO
4	DOM CASMURRO	MACHADO DE ASSIS
5	SQL	JOSÉ MARIA

Cláusula GROUP BY

-- autores de todos os livros da livraria

```
SELECT NOME
FROM LIVRO, AUTOR, ESCREVE
WHERE COD_AUTOR = COD_AUTOR_ESC AND
      COD_LIVRO_ESC = COD_LIVRO
GROUP BY NOME;
```

Data Output	Explain	Messa
		nome character varying(20)
1		PAULO COELHO
2		MACHADO DE ASSIS
3		JOSÉ MARIA

- ◆ Observe que na cláusula SELECT só podem constar os atributos presentes no GROUP BY
- ◆ Isso faz sentido pois, por exemplo, existem 3 livros do PAULO COELHO cadastros, qual deles apareceria no resultado?
- ◆ Tente adicionar o campo TITULO na cláusula SELECT do comando acima e observe o resultado

Cláusula GROUP BY

-- autores de todos os livros da livraria

```
SELECT NOME
FROM LIVRO, AUTOR, ESCRIVE
WHERE COD_AUTOR = COD_AUTOR_ESC AND
      COD_LIVRO_ESC = COD_LIVRO
GROUP BY NOME;
```

Data Output	Explain	Messa
		nome character varying(20)
1		PAULO COELHO
2		MACHADO DE ASSIS
3		JOSÉ MARIA

- ◆ Tente adicionar o campo TITULO na cláusula SELECT do comando acima e observe o resultado
- ◆ ERRO: coluna "livro.titulo" deve aparecer na cláusula GROUP BY ou ser utilizada em uma função de agregação

Cláusula GROUP BY

- ◆ Para os próximos exemplos execute o script de inserção do arquivo novasinsercoes.txt disponível no Moodle (novasinsercoes.txt)

Cláusula GROUP BY

- ◆ Mais de um atributo pode ser usado no agrupamento

	titulo character(35)	nome character varying(20)	valor numeric(7,2)
1	SQL	JOSÉ MARIA	10.90
2	DOM CASMURRO	MACHADO DE ASSIS	10.90
3	O ALIENISTA	MACHADO DE ASSIS	10.90
4	CONTOS FLUMINENSES	MACHADO DE ASSIS	24.00
5	CONTOS	MACHADO DE ASSIS	24.00
6	MAKIB	PAULO COELHO	24.00
7	BRIDA	PAULO COELHO	29.50
8	HISTÓRIAS PARA PAIS, FILHOS E NETOS	PAULO COELHO	35.00

-- agrupando os livros por autor/preço

```
SELECT NOME, VALOR FROM LIVRO, AUTOR, ESCRITOR  
WHERE COD_AUTOR = COD_AUTOR_ESC AND  
      COD_LIVRO_ESC = COD_LIVRO  
GROUP BY NOME, VALOR  
ORDER BY NOME
```

Cláusula GROUP BY

- ◆ Mais de um atributo pode ser usado no agrupamento

	titulo character(35)	nome character varying(20)	valor numeric(7,2)
1	SQL	JOSÉ MARIA	10.90
2	DOM CASMURRO	MACHADO DE ASSIS	10.90
3	O ALIENISTA	MACHADO DE ASSIS	10.90
4	CONTOS FLUMINENSES	MACHADO DE ASSIS	24.00
5	CONTOS	MACHADO DE ASSIS	24.00
6	MAKTUB	PAULO COELHO	24.00
7	BRIDA	PAULO COELHO	29.50
8	HISTÓRIAS PARA PAIS, FILHOS E NETOS	PAULO COELHO	35.00

-- agrupando os livros por autor/pr

```
SELECT NOME, VALOR FROM LIVRO, AU
WHERE COD_AUTOR = COD_AUTOR_ESC
      COD_LIVRO_ESC = COD_LIVRO

GROUP BY NOME, VALOR
ORDER BY NOME
```

	nome character varying(20)	valor numeric(7,2)
1	JOSÉ MARIA	10.90
2	MACHADO DE ASSIS	10.90
3	MACHADO DE ASSIS	24.00
4	PAULO COELHO	24.00
5	PAULO COELHO	29.50
6	PAULO COELHO	35.00

Cláusula GROUP BY

- ◆ As funções agregadas (e.g., COUNT, MIN, MAX, AVG) podem ser usadas para cálculos com subgrupos de tuplas definidos pela cláusula GROUP BY

- ◆

	titulo character(35)	nome character varying(20)	valor numeric(7,2)
1	SQL	JOSÉ MARIA	10.90
2	DOM CASMURRO	MACHADO DE ASSIS	10.90
3	O ALIENISTA	MACHADO DE ASSIS	10.90
4	CONTOS FLUMINENSES	MACHADO DE ASSIS	24.00
5	CONTOS	MACHADO DE ASSIS	24.00
6	MAKTUB	PAULO COELHO	24.00
7	BRIDA	PAULO COELHO	29.50
8	HISTÓRIAS PARA PAIS, FILHOS E NETOS	PAULO COELHO	35.00

-- qual o número de livros por autor?

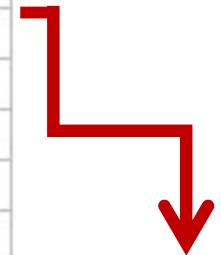
```
SELECT NOME, COUNT(*) FROM AUTOR, ESCRIVE
WHERE COD_AUTOR = COD_AUTOR_ESC
GROUP BY NOME
```


Cláusula GROUP BY

- ◆ As funções agregadas (e.g., COUNT, MIN, MAX, AVG) podem ser usadas para cálculos com subgrupos de tuplas definidos pela cláusula GROUP BY

- ◆

	titulo character(35)	nome character varying(20)	valor numeric(7,2)
1	SQL	JOSÉ MARIA	10.90
2	DOM CASMURRO	MACHADO DE ASSIS	10.90
3	O ALIENISTA	MACHADO DE ASSIS	10.90
4	CONTOS FLUMINENSES	MACHADO DE ASSIS	24.00
5	CONTOS	MACHADO DE ASSIS	24.00
6	MAKTUB	PAULO COELHO	24.00
7	BRIDA	PAULO COELHO	29.50
8	HISTÓRIAS PARA PAIS, FILHOS E NETOS	PAULO COELHO	35.00



-- qual o número de livros por autor

```
SELECT NOME, COUNT(*) FROM AUTOR,  
WHERE COD_AUTOR = COD_AUTOR_ESC  
GROUP BY NOME
```

	nome character varying(20)	count bigint
1	PAULO COELHO	3
2	MACHADO DE ASSIS	4
3	JOSÉ MARIA	1

Cláusula HAVING

- ◆ **HAVING:** é semelhante à cláusula WHERE. HAVING elimina tuplas *agrupadas que não satisfazem a uma determinada condição*
- ◆ **Diferença com WHERE:** WHERE filtra tuplas individuais antes da aplicação do GROUP BY, enquanto HAVING filtra grupo de tuplas criadas por GROUP BY. As condições de filtragem do HAVING devem ser feitas baseando-se nos atributos agrupados por GROUP BY

Cláusula HAVING

-- Editoras cujo total de publicações é maior que 1

```
SELECT COD_EDITORA AS EDITORA
FROM LIVRO
GROUP BY COD_EDITORA
HAVING COUNT(COD_EDITORA) > 1;
```

-- ERRO comum

```
SELECT COD_EDITORA AS EDITORA
FROM LIVRO
GROUP BY COD_EDITORA
HAVING COUNT(EDITORA) > 1;
```

-- ERRO: coluna "editora" não existe

LINE 4: HAVING COUNT(EDITORA) > 1;

-- Selecionar o nome das editoras cujo total de publicações é maior que 1 ?

Cláusula HAVING

-- Selecionar o nome das editoras cujo total de publicações é maior que 1 ?

-- Solução

```
SELECT RAZAO, L.COD_EDITORA
FROM EDITORA E, LIVRO L
WHERE E.COD_EDITORA = L.COD_EDITORA
GROUP BY RAZAO, L.COD_EDITORA
HAVING COUNT(L.COD_EDITORA) > 1;
```

Subconsultas: SELECTs aninhados

- ◆ São blocos SELECT...FROM...WHERE completos dentro da cláusula WHERE de outra consulta
- ◆ Essa construção possibilita a realização de consultas sobre os resultados obtidos em outras consultas
- ◆ Podem aparecer na cláusulas
 - SELECT
 - FROM
 - WHERE

Subconsultas: SELECTs aninhados

- ◆ Na cláusula FROM: visão inline
- ◆ Exemplo:

```
/* Listar informações sobre livros e editoras*/  
SELECT *  
FROM LIVRO L JOIN (SELECT * FROM EDITORA) E ON  
(L.COD_EDITORA = E.COD_EDITORA)
```

Subconsultas: SELECTs aninhados

- ◆ Na cláusula FROM: visão inline
- ◆ Exemplo:

/ Listar informações sobre livros e editoras*/*

SELECT *

**FROM LIVRO L JOIN (SELECT * FROM EDITORA) E ON
(L.COD_EDITORA = E.COD_EDITORA)**

Output pane									
	cod_livro smallint	titulo character(35)	valor numeric(7,2)	volume smallint	cod_editora smallint	cod_editora smallint	razao character varying(20)	endereco character(20)	cidade character(70)
1	31	MAKTUB	24.00		1	1	ROCCO	R. RODRIGO S	RIO DE JANEI
2	55	BRIDA	29.50		1	1	ROCCO	R. RODRIGO S	RIO DE JANEI
3	63	HISTÓRIAS PA	35.00		2	2	GLOBO		RIO DE JANEI
4	14	DOM CASMURRO	10.90		3	3	ATICA		SÃO PAULO
5	13	SQL	10.90		4	4	USP/ICMC		

Subconsultas: SELECTs aninhados

◆ Na cláusula SELECT: função inline

◆ Exemplo:

/ Listar informações sobre livros e o horário da consulta*/*

```
SELECT *, (SELECT now())  
FROM LIVRO L
```

Output pane

	cod_livro smallint	titulo character(35)	valor numeric(7,2)	volume smallint	cod_editora smallint	now timestamp with time zone
1	31	MAKTUB	24.00		1	2017-04-26 07:09:01.83
2	55	BRIDA	29.50		1	2017-04-26 07:09:01.83
3	63	HISTÓRIAS P/	35.00		2	2017-04-26 07:09:01.83
4	14	DOM CASMURRO	10.90		3	2017-04-26 07:09:01.83
5	13	SQL	10.90		4	2017-04-26 07:09:01.83

Subconsultas: SELECTs aninhados

- ◆ Na cláusula WHERE: subconsulta aninhada
 - não correlacionada; ou
 - correlacionada: avaliada 1 vez para cada linha processada pela consulta superior

- ◆ Exemplo:

```
/* Selecionar o título com valor mais alto */  
SELECT TITULO, VALOR  
FROM LIVRO  
WHERE VALOR IN (  
    SELECT MAX (VALOR)  
    FROM LIVRO );
```


Subconsultas: SELECTs aninhados

- ◆ Exemplo – subconsulta não correlacionada:

/ Selecionar o título com valor mais alto */*

Query - postgres on postgres@localhost:5432 *

File Edit Query Favourites Macros View Help

SQL Editor Graphical Query Builder

Previous queries

```
set search_path to livraria;  
  
SELECT TITULO, VALOR  
FROM LIVRO  
WHERE VALOR IN (SELECT MAX(VALOR)  
                FROM LIVRO );
```

Output pane

Data Output Explain Messages History

	max numeric
1	35.00

Query - postgres on postgres@localhost:5432 *

File Edit Query Favourites Macros View Help

SQL Editor Graphical Query Builder

Previous queries

```
set search_path to livraria;  
  
SELECT TITULO, VALOR  
FROM LIVRO  
WHERE VALOR IN (SELECT MAX(VALOR)  
                FROM LIVRO );
```

Output pane

Data Output Explain Messages History

	título character(35)	valor numeric(7,2)
1	HISTÓRIAS PARA PAIS,	35.00

Subconsultas: SELECTs aninhados

◆ Outros exemplos:

/ Selecionar os títulos mais baratos do que o título cujo código é 31 */*

```
SELECT TITULO, VALOR
FROM LIVRO
WHERE VALOR < (SELECT VALOR
                FROM LIVRO
                WHERE COD_LIVRO = 31) ;
```

Subconsultas: SELECTs aninhados

◆ Outros exemplos:

/ Selecionar os títulos mais baratos do que o título cujo código é 31 */*

```
SELECT TITULO, VALOR
FROM LIVRO
WHERE VALOR < (SELECT VALOR
FROM LIVRO
WHERE COD_LIVRO = 31) ;
```

Output pane	
Data Output	Explain
Messages	History
valor numeric(7,2)	
1	24.00

Subconsultas: SELECTs aninhados

◆ Outros exemplos:

/ Selecionar os títulos mais baratos do que o título cujo código é 31 */*

```
SELECT TITULO, VALOR
FROM LIVRO
WHERE VALOR < (SELECT VALOR
                FROM LIVRO
                WHERE COD_LIVRO = 31) ;
```

Output pane			
Data Output		Explain	Messages
	titulo character(35)	valor numeric(7,2)	
1	DOM CASMURRO	10.90	
2	SQL	10.90	

Subconsultas: SELECTs aninhados

◆ Outros exemplos:

/ Selecionar os títulos de livros da editora com
COD_EDITORA = 3 que possuam valor menor do que todos
os livros da editora com COD_EDITORA = 1*/*

```
SELECT TITULO, VALOR
FROM LIVRO
WHERE COD_EDITORA = 3 AND
      VALOR < ALL(SELECT VALOR
                  FROM LIVRO
                  WHERE COD_EDITORA = 1);
```

Subconsultas: SELECTs aninhados

◆ Outros exemplos:

/ Selecionar os títulos de livros da editora com
COD_EDITORA = 3 que possuam valor menor do que todos
os livros da editora com COD_EDITORA = 1*/*

```
SELECT TITULO, VALOR
FROM LIVRO
WHERE COD_EDITORA = 3 AND
      VALOR < ALL (SELECT VALOR
                    FROM LIVRO
                    WHERE COD_EDITORA = 1);
```

Output pane		
Data Output		
Explain		
Messages		
History		
	valor numeric(7,2)	
1	24.00	
2	29.50	

Output pane		
Data Output		
Explain		
Messages		
History		
	titulo character(35)	valor numeric(7,2)
1	DOM CASMURRO	10.90

Subconsultas: SELECTs aninhados

◆ Pode-se utilizar com os operadores ANY e ALL

- >
- >=
- <
- <=
- =
- <>

◆ Curiosidade: = ANY equivale ao IN

Subconsultas: SELECTs aninhados

◆ Consultas aninhadas correlacionadas

- Quando condição na cláusula WHERE de uma consulta aninhada **referencia** algum atributo de uma relação declarada na consulta externa

Subconsultas: SELECTs aninhados

◆ Exemplo – consultas aninhadas correlacionadas

```
CREATE TABLE FUNCIONARIO (  
  COD INTEGER PRIMARY KEY,  
  NOME VARCHAR(20) );
```

```
CREATE TABLE DEPENDENTE (  
  COD INTEGER PRIMARY KEY,  
  NOME VARCHAR(20),  
  COD_FUNC INTEGER REFERENCES FUNCIONARIO)
```

```
INSERT INTO FUNCIONARIO VALUES (1, 'JOSÉ'), (2,  
  'MARIA'), (3, 'JOÃO');
```

```
INSERT INTO DEPENDENTE VALUES (1, 'MARIA', 2);
```

Subconsultas: SELECTs aninhados

◆ Exemplo - consultas aninhadas correlacionadas

/ Recuperar o nome de cada funcionário que tem um dependente com o mesmo nome do funcionário */*

```
SELECT F.NOME
      FROM FUNCIONARIO F, DEPENDENTE D
     WHERE F.COD = D.COD_FUNC AND
           F.NOME = D.NOME
```

```
SELECT F.NOME
      FROM FUNCIONARIO F
     WHERE F.COD IN (SELECT D.COD_FUNC
                     FROM DEPENDENTE D
                     WHERE F.NOME = D.NOME)
```

Subconsultas: SELECTs aninhados

- ◆ A função EXISTS em SQL é usada para verificar se o resultado de uma consulta aninhada é vazio (não contém tuplas) ou não
 - O resultado de EXISTS é TRUE se o resultado tiver pelo menos uma tupla

Subconsultas: SELECTs aninhados

◆ Exemplo - consultas aninhadas correlacionadas

/ Recuperar o nome de cada funcionário que tem um dependente com o mesmo nome do funcionário */*

```
SELECT F.NOME
FROM FUNCIONARIO F
WHERE EXISTS (SELECT D.COD_FUNC
               FROM DEPENDENTE D
               WHERE F.COD = D.COD_FUNC AND
                     F.NOME = D.NOME)
```

para cada tupla de FUNCIONARIO,
avaliar a consulta aninhada, que
recupera todas as tuplas
DEPENDENTE com os mesmos
COD_FUNC e NOME que a tupla
FUNCIONARIO;

Subconsultas: SELECTs aninhados

◆ Exemplo - consultas aninhadas correlacionadas

/ Recuperar os nomes dos funcionários que não tem dependentes */*

```
SELECT F.NOME
FROM FUNCIONARIO F
WHERE NOT EXISTS (SELECT *
                  FROM DEPENDENTE D
                  WHERE F.COD = D.COD_FUNC)
```

Output pane	
Data Output	
nome character varying(20)	
1	JOSÉ
2	JOÃO

Bibliografia

- ◆ Elmasri, Ramez; Navathe, Shamkant B. **Sistemas de banco de dados**. 4 ed. São Paulo: Addison Wesley, 2005, 724 p. Bibliografia: p. [690]-714.

Leitura complementar para casa

- ◆ Capítulo 4 do livro: Elmasri, Ramez; Navathe, Shamkant B. Sistemas de banco de dados. 6ª edição
- ◆ Manual do SGBD PostgreSQL
 - <http://www.postgresql.org/docs/manuals/>
 - Explorar as outras particularidades dos comandos apresentados