

Linguagem SQL

Comando SELECT

Agrupamento de Resultados

- [Introdução](#)
- [Funções de Agregação Básicas](#)
 - [COUNT](#)
 - [SUM](#)
 - [MIN](#)
 - [MAX](#)
 - [AVG](#)
- [Cláusulas de Agrupamento](#)
 - [GROUP BY](#)
 - [HAVING](#)

▪ Interesse

- Obter **informação agregada** (i.e., **resumida**) ...
... sobre **grupos** de **registos** do **resultado** de um SELECT.

▪ Exemplo

- Tabela Empregado

	IDEMPREGADO	NOME	LOCAL	LOCALIDADE	SALARIO
1		1 Nico Seixal	Avenida da Glória 1000	Lisboa	6200
2		2 Ana Rita	Avenida da Glória 1200	Lisboa	5980
3		3 Joana Gonçalves	Rua Antunes Almeida 114	Porto	6200
4		4 Paula Silva	Rua Pereira 186 1º DTO	Porto	1520
5		5 António Castro	Rua Direita 25 2º ESQ	Porto	2100
6		6 Maria Santos	Rua da Liberdade 150	Braga	1520
7		7 Francisco Vale	Avenida do Monte 1240	Braga	650
8		8 Patricia Cunha	Rua da Velhota 233 5º ESQ	Braga	745

Grupo Lisboa

Grupo Porto

Grupo Braga

- Total de empregados por localidade

LOCALIDADE	TOTAL_EMPREGADOS
Porto	3
Braga	3
Lisboa	2

```
SELECT localidade, COUNT(*) AS total_empregados
FROM empregado
GROUP BY localidade;
```

Cláusula **GROUP BY**:

- Especifica agrupamento pela localidade.
- Resultado **dividido em grupos** pela localidade.

Função de Agregação **COUNT**:

- Conta os registos de cada grupo.

Funções

- Aplicadas a uma simples coluna numa tabela.
- Retornam um simples valor.

Interesse

- Obter informação agregada (i.e., resumida) sobre:
 - Grupo único de registos
 - Grupos de registos ... especificados na cláusula GROUP BY

Funções de Agregação podem ser aplicadas a:

- grupo único (sem GROUP BY)
- vários grupos (com GROUP BY)

Exemplo

- Função COUNT
 - Tabela: EMPREGADO (..., NOME, ..., LOCALIDADE, ...)
 - Informação agregada de todos os registos (grupo único)

```
-- N° Total de Empregados  
SELECT COUNT(*)  
FROM empregado;
```

COUNT(*)
8

NULL
ignorado

```
SELECT COUNT(*) AS total_empregados  
FROM empregado;
```

TOTAL_EMPREGADOS
8

Nome
coluna
alterado

- Informação agregada de cada grupo de registos

```
SELECT localidade, COUNT(*) AS total_empregados  
FROM empregado  
GROUP BY localidade;
```

LOCALIDADE	TOTAL_EMPREGADOS
Porto	3
Braga	3
Lisboa	2

■ Tipos Mais Vulgares

Função	Formato	Resultado	Observação
COUNT	COUNT(*)	Número total de registos do resultado/grupo de resultados do SELECT	
	COUNT(coluna)	Número de valores <u>diferentes de NULL</u> na coluna especificada.	Ignora NULL
	COUNT (DISTINCT coluna)	Número de valores diferentes na coluna especificada.	<ul style="list-style-type: none">▪ Não conta repetidos▪ Ignora NULL
MAX	MAX(coluna)	Maior valor da coluna especificada.	<ul style="list-style-type: none">▪ Aplicada também a colunas do tipo <i>string</i>, usando uma comparação alfabética.▪ Ignora NULL
MIN	MIN(coluna)	Menor valor da coluna especificada.	
SUM	SUM(coluna)	Soma dos valores da coluna especificada.	<ul style="list-style-type: none">▪ Aplicáveis apenas a colunas numéricas.▪ Ignora NULL
AVG	AVG(coluna)	Média dos valores da coluna especificada.	

Funções aplicadas a uma **simples** coluna

Palavra-reservada **DISTINCT**

- Para eliminar duplicados antes de aplicar função
- Pode ser usado em **todas as funções**
- Especificada antes do nome da coluna
 - Como no exemplo do COUNT

■ Exemplos

```
-- N° de salários diferentes  
SELECT COUNT(DISTINCT salario)  
FROM empregado;
```

COUNT(DISTINCTSALARIO)
6

DISTINCT ignora NULL

```
-- Salários máximo e mínimo  
SELECT MAX(salario), MIN(salario)  
FROM empregado;
```

MAX(SALARIO)	MIN(SALARIO)
6200	650

- Possível aplicar múltiplas funções.
- Mostrada um linha de resultados.

```
-- Diferença entre os salários máximo e mínimo  
SELECT MAX(salario)-MIN(salario)  
FROM empregado;
```

MAX(SALARIO)-MIN(SALARIO)
5550

```
-- 10% do total dos salários  
SELECT SUM(salario) * 0.1  
FROM empregado;
```

SUM(SALARIO)*0.1
2491,5

- Legal nas Cláusulas

- SELECT
- HAVING

- Illegal

- Na cláusula **WHERE** ... como operando

- Exemplo

```
61  -- Empregados com o menor salário
62  SELECT nome
63  FROM empregado
64  WHERE salario = MIN(salario);
```

```
ORA-00934: função de grupo não é aqui permitida
00934. 00000 - "group function is not allowed here"
*Cause:
*Action:
Error at Line: 64 Column: 18
```

- Solução

- Com SELECT encaixado (*subquery*)

```
-- Empregados com o menor salário
SELECT nome
  FROM empregado
 WHERE salario = ( SELECT MIN(salario)
                   FROM empregado );
```

NOME
Francisco Vale

- Na cláusula **SELECT**

- Lista de colunas (fora de funções) juntamente com funções de agregação sem GROUP BY para agrupar os dados, conjuntamente.
- Exemplo:

```
72  SELECT nome, COUNT(salario)
73  FROM empregado;
```

```
ORA-00937: não é uma função de grupo de grupo-único
00937. 00000 - "not a single-group group function"
*Cause:
*Action:
Error at Line: 72 Column: 8
```

▪ Cláusulas

- GROUP BY // divide resultado de *query* em múltiplos grupos de registos
- HAVING // filtra grupos da tabela resultante da *query*

```
SELECT [ ALL | DISTINCT ] <lista select>
FROM <lista de tabela>
[WHERE <condição_where>]
[GROUP BY <lista groupby>]
[HAVING <condição having>]
[ORDER BY <lista orderby> [ DESC | ASC ] ]
```


▪ **Sintaxe**

```
SELECT [ ALL | DISTINCT ] <lista select>
FROM <lista de tabela>
[WHERE <condição_where>]
[GROUP BY <lista groupby>]
[HAVING <condição having>]
[ORDER BY <lista orderby> [ DESC | ASC ] ]
```

<lista groupby> ::= { *table_name* | *view_name* | *table_alias* }. * | { *column_name expression* }, [, ...n]

▪ **Interesse**

- Agrupar informação
- Dividir o resultado de um SELECT ... em grupos de resultados ...
... para serem processados por funções de agregação.

▪ **Deve Especificar**

- Todas as colunas especificadas no SELECT ... **exceto** colunas em funções de agregação
 - Razão: SGBD precisa de saber como agrupar o resultado do SELECT

▪ **Cada Elemento do SELECT**

- Deve ser um **simples** valor ... para **cada** grupo

▪ **Valores NULL**

- Também são agrupados numa coluna

Colunas do GRUPO BY:

- Não têm de existir no SELECT.

▪ **Com Cláusula WHERE**

- WHERE é executado em 1º
- Grupos formados depois com os registos selecionados pelo WHERE

■ Exemplos

```
-- N° de empregados por localidade
SELECT localidade, COUNT(*) AS total_empregados
  FROM empregado
 GROUP BY localidade;
```

LOCALIDADE	TOTAL_EMPREGADOS
Porto	3
Braga	3
Lisboa	2

Localidade em
SELECT permitido
porque tem 1 valor
por grupo

```
-- N° de empregados por localidade com salário superior a 3000
SELECT localidade, COUNT(*) AS total_empregados
  FROM empregado
 WHERE salario > 3000
 GROUP BY localidade;
```

LOCALIDADE	TOTAL_EMPREGADOS
Porto	1
Lisboa	2

```
-- Total recebido por cada empregado
SELECT nome, salario + SUM(valor) AS total
  FROM empregado NATURAL JOIN premio
 GROUP BY nome, salario
```

NOME	TOTAL
Nico Seixal	6870
Joana Gonçalves	6350
Paula Silva	2020
Francisco Vale	700

Tabela PREMIO		
ID_PREMIO	ID_EMPREGADO	VALOR
1	1	200
2	1	250
3	1	220
4	3	150
5	4	500
6	7	50

- **Criada**

- Para trabalhar **conjuntamente** com GROUP BY

- **Sintaxe**

```
SELECT [ ALL | DISTINCT ] <lista select>
FROM <lista de tabela>
[WHERE <condição_where>]
[GROUP BY <lista groupby>]
[HAVING <condição having>]
[ORDER BY <lista orderby> [ DESC | ASC ] ]
```

<condição having> ::= condição com expressões, constantes ou colunas **incluídas** na <lista groupby>

- **Interesse**

- Restringir os grupos que surgem no resultado final
 - i.e. filtrar grupos

- **Aplicada**

- Sobre os resultados dos grupos

- **Nomes de colunas**

- Incluídas em GROUP BY
 - **Ou** ... usadas em funções de agregação

Colunas do GRUPO BY:

- Não têm de existir no SELECT.

- **Condição**

- Inclui sempre ... pelo menos uma **função** de agregação
 - Senão... pode ser colocada em WHERE

■ Exemplos

```
-- N° de empregados por localidade com mais de 2 empregados
SELECT localidade, COUNT(*) AS total_empregados
  FROM empregado
  GROUP BY localidade
 HAVING COUNT(localidade)>2;
```

LOCALIDADE	TOTAL_EMPREGADOS
Porto	3
Braga	3

```
-- N° de empregados por localidade com mais de 2 empregados
SELECT localidade, COUNT(*) AS total_empregados
  FROM empregado
  GROUP BY localidade
 HAVING COUNT(*)>2;
```

```
-- Localidades com mais de 2 empregados com salário superior a 3000
SELECT localidade, COUNT(*) AS total_empregados
  FROM empregado
  WHERE salario > 3000
  GROUP BY localidade
 HAVING COUNT(*)>1;
```

LOCALIDADE	TOTAL_EMPREGADOS
Lisboa	2