# Linguagem SQL

Comando SELECT

Agrupamento de Resultados

# Sumário

- <u>Introdução</u>
- Funções de Agregação Básicas
  - COUNT
  - SUM
  - MIN
  - <u>MAX</u>
  - AVG
- Cláusulas de Agrupamento
  - GROUP BY
  - HAVING

# Agrupamento de Resultados

#### Interesse

- Obter informação <u>agregada</u> (i.e., <u>resumida</u>) ...
  - ... sobre grupos de registos do resultado de um SELECT.
- Exemplo
  - Tabela Empregado

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

Total de empregados por localidade

<b>\$ LOCALIDADE</b>	
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 <u>dividido em grupos</u> pela localidade.

## Função de Agregação **COUNT**:

Conta os registos de cada grupo.

# Funções de Agregação

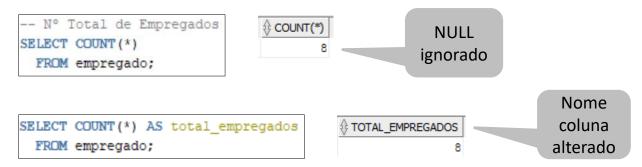
- Funções
  - Aplicadas a uma simples coluna duma tabela.
  - Retornam um <u>simples</u> valor.
- Interesse
  - Obter informação agregada (i.e., resumida) sobre:
    - Grupo único de registos
    - Grupos de registos ... especificados na cláusula GROUP BY

# Funções de Agregação podem ser aplicadas a:

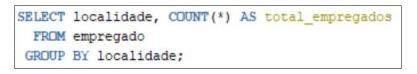
- grupo único (<u>sem</u> GROUP BY)
- vários grupos (<u>com</u> GROUP BY)

### Exemplo

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



Informação agregada de cada grupo de registos



	♦ TOTAL_EMPREGADOS
Porto	3
Braga	3
Lisboa	2

# Funções de Agregação

### Tipos Mais Vulgares

Função	Formato	Resultado	Observação	
COUNT	COUNT(*)	<b>Número</b> total de registos do resultado/grupo de resultados do SELECT		
	COUNT(coluna)	<b>Número</b> de valores <u>diferentes de NULL</u> na coluna especificada.	Ignora NULL	
	COUNT ( <b>DISTINCT</b> coluna)	<b>Número</b> de valores diferentes na coluna especificada.	<ul><li>Não conta repetidos</li><li>Ignora NULL</li></ul>	
MAX	MAX(coluna)	Maior valor da coluna especificada.	<ul> <li>Aplicada também a</li> </ul>	
MIN	MIN(coluna)	Menor valor da coluna especificada.	colunas do tipo string, usando uma comparação alfabética. • Ignora NULL	
SUM	SUM(coluna)	Soma dos valores da coluna especificada.	<ul> <li>Aplicáveis apenas a</li> </ul>	
AVG	AVG(coluna)	Média dos valores da coluna especificada.	colunas numéricas.  Ignora NULL	

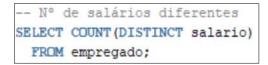
Funções aplicadas a uma <u>simples</u> coluna

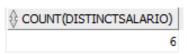
#### Palavra-reservada **DISTINCT**

- 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

# Funções de Agregação

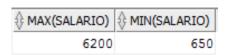
### Exemplos





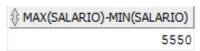
**DISTINCT** ignora NULL

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



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

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



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

```
$SUM(SALARIO)*0.1
2491,5
```

- Legal nas Cláusulas
  - SELECT
  - HAVING
- Ilegal
  - 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)

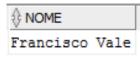
```
-- Empregados com o menor salário

SELECT nome

FROM empregado

WHERE salario = ( SELECT MIN(salario)

FROM empregado );
```



- Na cláusula SELECT
  - Lista de colunas (fora de funções) juntamente com funções de agregação <u>sem GROUP BY</u> 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 de Agrupamento

#### 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 <u>simples</u> valor ... para <u>cada</u> 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

# Cláusula GROUP BY

### Exemplos

```
-- N° de empregados por localidade

SELECT localidade, COUNT(*) AS total_empregados

FROM empregado

GROUP BY localidade;
```

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;
```

	↑ 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	<b>∜ TOTAL</b>
Nico Seixal	6870
Joana Gonçalves	6350
Paula Silva	2020
Francisco Vale	700

Tabela PREMIO			
		∜ 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
- <u>Ou</u> ... usadas em funções de agregação

### Condição

- Inclui sempre ... pelo menos uma <u>função</u> de agregação
  - Senão... pode ser colocada em WHERE

Colunas do GRUPO BY:

Não têm de existir no SELECT.

### 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;
```

```
-- 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;
```

Lisboa	2