

5. SQL

- Conjunto de Operações para Manipulação de Dados-

❑ Limitando o número de tuplas no resultado de uma consulta:

- LIMIT no MySQL

```
SELECT nome_da(s)_coluna(s)
```

```
FROM nome_tabela
```

```
WHERE condição
```

```
LIMIT num1 [,num2];
```

- Se somente num1 existe, o valor especifica o número de linhas a ser retornado do início do result set:
 - ➔ **SELECT * FROM tabela LIMIT 5; #Recupera os 5 primeiros registros**
- Se num2 existe, o valor de num1 especifica a posição inicial ($1^a=0$) e num2 o número de linhas a ser retornado a partir de num1:
 - ➔ **SELECT * FROM tabela LIMIT 5,10; #Recupera as linhas de 6 até 15**

5. SQL

- Conjunto de Operações para Manipulação de Dados-

❑ Limitando o número de tuplas no resultado de uma consulta:

- TOP no SQL Server

```
SELECT TOP número [percent] nome_da(s)_coluna(s)  
FROM nome_tabela  
WHERE condição;
```

- Ex1: “Selecionar os 3 primeiros clientes”

```
SELECT TOP 3 *  
FROM Clientes;
```

- Ex2: “Selecionar os 50% primeiros clientes do Brasil”

```
SELECT TOP 50 PERCENT *  
FROM Clientes  
WHERE País='Brasil';
```

5. SQL

- Conjunto de Operações para Manipulação de Dados-

❑ Limitando o número de tuplas no resultado de uma consulta:

- TOP no SQL Server (Cont.)
- Ex3: “Selecionar os primeiros clientes do Brasil entre o 10º e o 20º”

```
SELECT TOP 20 *  
FROM Clientes  
WHERE País='Brasil';  
  
EXCEPT  
  
SELECT TOP 10 *  
FROM Clientes  
WHERE País='Brasil';
```

5. SQL

- Conjunto de Operações para Manipulação de Dados-

❑ Limitando o número de tuplas no resultado de uma consulta:

- ROWNUM no Oracle:

- Ex1: “Selecionar os 3 primeiros clientes”

```
SELECT * FROM Clientes  
WHERE ROWNUM <= 3;
```

- Ex2: “Selecionar os 3 primeiros clientes do Brasil”

```
SELECT * FROM Clientes  
WHERE País='Brasil' AND ROWNUM <= 3;
```

- Ex3: “Selecionar os primeiros clientes do Brasil entre o 10º e o 20º”

```
SELECT * FROM Clientes  
WHERE País='Brasil' AND ROWNUM >= 10 AND ROWNUM <= 20;
```

5. SQL

- Conjunto de Operações para Manipulação de Dados-

❑ CASE no MySQL

➡ A cláusula CASE pode ser usada de 2 formas no MySQL:

1) CASE case_value

WHEN when_value THEN statement_list

[WHEN when_value THEN statement_list] ...

[ELSE statement_list]

END

onde **case_value** é uma expressão. Este valor é comparado a expressão **when_value** em cada cláusula **WHEN** até uma delas ser igual. Quando esta é encontrada, a correspondente cláusula **THEN** executa a **statement_list**. Se nenhuma **when_value** é igual, a cláusula **ELSE** executa seu **statement_list**, se houver.

PS: Se nenhum **when_value** igualar com o valor testado e a sentença **CASE** não contiver uma cláusula **ELSE**, ocorre um erro do tipo: “**Case not found for CASE statement**”.

5. SQL

- Conjunto de Operações para Manipulação de Dados-

❑ CASE no MySQL

2)CASE

```
WHEN search_condition THEN statement_list  
[WHEN search_condition THEN statement_list] ...  
[ELSE statement_list]  
END
```

Onde para cada cláusula **WHEN** a expressão **search_condition** é avaliada até uma ser **verdadeira**. Neste ponto é comparado a expressão **when_value** em cada cláusula **WHEN** até uma delas ser verdadeira. Quando isto ocorre, a correspondente cláusula **THEN** executa a **statement_list**. Se nenhuma **search_condition** é **verdadeira**, a cláusula **ELSE** executa seu **statement_list**, se houver.

PS.: Cada **statement_list** consiste de uma ou mais sentenças SQL. Não é permitido um **statement_list** vazio.

5. SQL

- Conjunto de Operações para Manipulação de Dados-

❑ CASE no MySQL

➡ Exemplo:

Foi feito um **Update sem Where** numa tabela “*products*” de uma empresa, de tal maneira que se perdeu a informação do preço dos produtos, ficando a mesma da seguinte forma:

products			
id	name	type	price
1	Monitor	B	0
2	Headset	A	0
3	PC Case	A	0
4	Computer Desk	C	0
5	Gaming Chair	C	0
6	Mouse	A	0

id	name	type	price
1	Monitor	B	0
2	Headset	A	0
3	PC Case	A	0
4	Computer Desk	C	0
5	Gaming Chair	C	0
6	Mouse	A	0

Porém, sabe-se que:

- Se o tipo do produto é igual A, o preço será 20.0
- Se o tipo do produto é igual B, o preço será 70.0
- Se o tipo do produto é igual C, o preço será 530.5

5. SQL

- Conjunto de Operações para Manipulação de Dados-

❑ CASE no MySQL

➡ Exercícios:

Deseja-se obter o nome e o preço de todos os produtos, mostrando primeiro todos os produtos do tipo A, segundo o tipo B, e por último o tipo C.

Além disso, os produtos de cada um dos tipos (A,B,C), devem estar ordenados pelo seu ID de forma decrescente.

Exemplo de Saída

type	price
Mouse	20.0
PC Case	20.0
Headset	20.0
Monitor	70.0
Gaming Chair	530.5
Computer Desk	530.5

5. SQL

- Conjunto de Operações para Manipulação de Dados-

❑ CASE no MySQL

➡ Solução 1:

```
SELECT `name`,(CASE `type`  
    WHEN 'A' THEN (20.0)  
    WHEN 'B' THEN (70.0)  
    WHEN 'C' THEN (530.5)  
    END) AS `price`  
  
FROM `products`  
ORDER BY `type`,  
        `id` DESC;
```

	name	price
▶	Mouse	20.0
	PC Case	20.0
	Headset	20.0
	Monitor	70.0
	Gaming Chair	530.5
	Computer Desk	530.5

5. SQL

- Conjunto de Operações para Manipulação de Dados-

❑ CASE no MySQL

➡ Solução 2:

```
SELECT `name`,(CASE  
    WHEN `type`='A' THEN (20.0)  
    WHEN `type`='B' THEN (70.0)  
    WHEN `type`='C' THEN (530.5)  
    END) AS `price`  
  
FROM `products`  
ORDER BY `type`,  
        `id` DESC;
```

	name	price
▶	Mouse	20.0
	PC Case	20.0
	Headset	20.0
	Monitor	70.0
	Gaming Chair	530.5
	Computer Desk	530.5

5. SQL

- Conjunto de Operações para Manipulação de Dados-

❑ Cláusula IF no MySQL

➡ A cláusula IF dentro de uma *Function* no MySQL:

```
1) IF search_condition THEN statement_list  
   [ELSEIF search_condition THEN statement_list] ...  
   [ELSE statement_list]  
END IF
```

Onde se a expressão **search_condition** é avaliada como **verdadeira**, a correspondente cláusula **THEN** ou **ELSEIF** executa a **statement_list**. Se nenhuma **search_condition** é **verdadeira**, a cláusula **ELSE** executa seu **statement_list**, se houver.

PS: Cada **statement_list** consiste de uma ou mais sentenças SQL. Não é permitido um **statement_list** vazio.

5. SQL

- Conjunto de Operações para Manipulação de Dados-

❑ Cláusula IF no MySQL

➡ A cláusula IF dentro de “Functions” no MySQL:

Ex:

```
DELIMITER //

CREATE FUNCTION SimpleCompare(n INT, m INT)
  RETURNS VARCHAR(20)

BEGIN
  DECLARE s VARCHAR(20);

  IF n > m THEN SET s = '>';
  ELSEIF n = m THEN SET s = '=';
  ELSE SET s = '<';
  END IF;

  SET s = CONCAT(n, ' ', s, ' ', m);

  RETURN s;
END //

DELIMITER ;
```

5. SQL

- Conjunto de Operações para Manipulação de Dados-

❑ Função IF no MySQL

➡ A função IF no MySQL:

1) IF(condition, statement_for_true_condition, statement_for_false_condition)

Onde se a expressão em **condition** é avaliada como **verdadeira**, a primeira cláusula após a vírgula (*statement_for_true_condition*) será executada. Caso contrário, a segunda cláusula após a vírgula (*statement_for_false_condition*) será executada.

PS: Não é permitido um **statement**... vazio.

5. SQL

- Conjunto de Operações para Manipulação de Dados-

❑ IF no MySQL

➡ A função IF no MySQL:

```
Ex: SELECT `name`,(IF(type='A', (20.0),  
IF(`type`='B', (70.0),  
IF(`type`='C', (530.5),  
0.0)  
)) AS `price`  
  
FROM `products`  
  
ORDER BY `type`,  
`id` DESC;
```

name	price
Mouse	20.0
PC Case	20.0
Headset	20.0
Monitor	70.0
Gaming Chair	530.5
Computer Desk	530.5

5. SQL

- Conjunto de Operações para Manipulação de Dados-

❑ Resumindo dados usando COMPUTE e COMPUTE BY

- ➡ Uma cláusula COMPUTE BY permite visualizar os detalhes e o resumo das linhas com uma instrução SELECT.
- ➡ Pode-se calcular os valores resumidos de subgrupos, ou um valor resumido de todo o conjunto de resultados.
- ➡ A cláusula COMPUTE usa as seguintes informações:
 - ⇒ A palavra-chave opcional BY. Com isso, calcula o agregado de linha especificado por coluna.
 - ⇒ Um nome de função de agregação de linha. Isso inclui SUM, AVG, MIN, MAX ou COUNT.
 - ⇒ Uma coluna na qual será executada a função de agregação de linha.

5. SQL

- Conjunto de Operações para Manipulação de Dados-

❑ Resumindo dados usando COMPUTE e COMPUTE BY

- ➡ Usamos a cláusula COMPUTE para obtermos visualmente dados totalizados. Já com a cláusula COMPUTE BY podemos visualizar os dados sem os mesmos serem agrupados.
- ➡ Por meio do COMPUTE BY, é possível visualizar tanto linhas de detalhe, como linhas de resumo com o uso de uma instrução SELECT.
- ➡ Importante ressaltar que o resultado gerado pela cláusula COMPUTE é apenas para visualização e não pode ser usado, ou seja, não possui característica relacional.
- ➡ Em conjunto com COMPUTE, podemos usar funções como MIN, MAX, AVG, COUNT ou SUM.

5. SQL

- Conjunto de Operações para Manipulação de Dados-

❑ Resumindo dados usando COMPUTE e COMPUTE BY

➡ Com base em uma tabela Produtos, vamos realizar uma totalização das quantidades de produtos, que no resultado estarão separados por tipo. Para isso, usaremos o seguinte comando:

```
SELECT Nome, Fabricante, Quantidade, Tipo  
FROM Produtos  
ORDER BY Tipo  
COMPUTE SUM (Quantidade) BY Tipo
```

5. SQL

- Conjunto de Operações para Manipulação de Dados-

❑ Resumindo dados usando COMPUTE e COMPUTE BY

➡ Com base em uma tabela Produtos, vamos realizar uma totalização das quantidades de produtos, que no resultado estarão separados por tipo. Para isso, usaremos o seguinte comando:

```
SELECT Nome, Fabricante, Quantidade, Tipo  
FROM Produtos  
ORDER BY Tipo  
COMPUTE SUM (Quantidade) BY Tipo
```

➡ No código acima temos a cláusula COMPUTE SUM (Quantidade) BY Tipo. Ele usará o tipo de produto como critério para obter subtotais das quantidades de produtos em estoque.

5. SQL

- Conjunto de Operações para Manipulação de Dados-

- Resumindo dados usando COMPUTE e COMPUTE BY

SELECT Nome, Fabricante, Quantidade, Tipo

FROM Produtos

ORDER BY Tipo

COMPUTE SUM (Quantidade) BY Tipo

Results

Messages

	Nome	Fabricante	Quantidade	Tipo
1	Armário de Serviço	Aracaju	50.00	Armário

	sum
1	50.00

	Nome	Fabricante	Quantidade	Tipo
1	GT-I6220 Quad Band	Samsung	300.00	Celular

	sum
1	300.00

	Nome	Fabricante	Quantidade	Tipo
1	Playstation 3	Sony	100.00	Console
2	Playstation 2	Sony	100.00	Console
3	Xbox 360 120GB	Microsoft	350.00	Console
4	Wii 120GB	Nintendo	250.00	Console

	sum
1	800.00

5. SQL

- Conjunto de Operações para Manipulação de Dados-

❑ Resumindo dados usando COMPUTE e COMPUTE BY

➡ COMPUTE BY com JOIN

Podemos usar a cláusula COMPUTE BY junto de outra cláusula a fim de fazer a associação de várias tabelas. Essa outra cláusula é a JOIN.

A seguir, temos um exemplo do uso de COMPUTE BY com JOIN, usando as tabelas Funcionario, Cargo e Vendas

5. SQL

- Conjunto de Operações para Manipulação de Dados-

❑ Resumindo dados usando COMPUTE e COMPUTE BY

➡ COMPUTE BY com JOIN

```
SELECT F.NomeFuncionario AS [Nome], F.SalarioFuncionario AS  
[Salário], V.ValorPedido AS [Valor], C.NomeCargo AS [Cargo]  
FROM Funcionario AS F  
INNER JOIN Cargo AS C  
ON C.IdCargo = F.IdCargo  
INNER JOIN Vendas AS V  
ON V.IdFuncionario = F.IdFuncionario  
ORDER BY F.NomeFuncionario  
COMPUTE SUM (F.SalarioFuncionario) BY F.NomeFuncionario  
COMPUTE SUM (F.SalarioFuncionario)
```

5. SQL

- Conjunto de Operações para Manipulação de Dados-

❑ Resumindo dados usando COMPUTE e COMPUTE BY

➡ COMPUTE BY com JOIN

	Nome	Salário	Valor	Cargo
1	Marisa da Horta	3025.00	75.00	Programador Jr.
	sum			
1	3025.00			

	Nome	Salário	Valor	Cargo
1	Tinica	2750.00	150.00	Web Designer Pl.
	sum			
1	2750.00			

	sum			
1	5775.00			

➡ Com COMPUTE é possível obter valores de resumo para grupos, bem como calcular, para o mesmo grupo, mais de uma função agregada.

5. SQL

- Conjunto de Operações para Manipulação de Dados-

❑ COMPUTE BY no MySQL

➡ A cláusula COMPUTE BY foi deprecada no MySQL.

➡ Pode-se calcular os valores resumidos de subgrupos, ou um valor resumido de todo o conjunto de resultados, utilizando-se variáveis locais, conforme o exemplo a seguir:

Dada a tabela “products_prices”, tendo o seguinte esquema e a seguinte instância (povoamento):

products_prices (esquema)

Field	Type	Null	Key	Default
id	decimal(10,0)	NO	PRI	NULL
name	varchar(50)	YES		NULL
type	char(1)	YES		NULL
price	decimal(10,2)	YES		NULL

products_prices (instância)

id	name	type	price
1	Monitor	B	70.00
2	Headset	A	20.00
3	PC Case	A	20.00
4	Computer Desk	C	530.50
5	Gaming Chair	C	530.50
6	Mouse	A	20.00

5. SQL

- Conjunto de Operações para Manipulação de Dados-

❑ Uso de variáveis locais para simular “COMPUTE BY” no MySQL

➡ Calcular os somatórios de preços totais e parciais por tipos de produtos.

SET @gt:=0;#Contador de valor total

SET @pr:=0;#Contador de valor parcial

SET @st:=0;#Contador de subtotal

SET @pg:="";#Grupo anterior

SELECT `type`, `price`, (@gt := @gt + `price`) AS Total,

(@pr := IF(@pg != `type`, `price`, @pr + `price`)) AS Parcial,

(@st := IF(@pg != `type`, 1, @st + 1)) AS SubTotal, @pg:=`type` AS Tipo

FROM `products_prices`

ORDER BY `type` ASC;

id	name	type	price
1	Monitor	B	70.00
2	Headset	A	20.00
3	PC Case	A	20.00
4	Computer Desk	C	530.50
5	Gaming Chair	C	530.50
6	Mouse	A	20.00

type	price	Total	Parcial	SubTotal	Tipo
A	20.00	20.00	20.00	1	A
A	20.00	40.00	40.00	2	A
A	20.00	60.00	60.00	3	A
B	70.00	130.00	70.00	1	B
C	530.50	660.50	530.50	1	C
C	530.50	1191.00	1061.00	2	C

5. SQL

- Conjunto de Operações para Manipulação de Dados-

❑ Uso de variáveis locais para simular “COMPUTE BY” no MySQL

➡ Poderia ainda mostrar somente os preços totais por tipos de produtos.

SET @gt:=0;#Contador de valor total

SET @pr:=0;#Contador de valor parcial

SET @st:=0;#Contador de subtotal

SET @pg:="";#Grupo anterior

SELECT Tipo, ROUND(max(Parcial),1) AS TotalPorTipo

FROM (SELECT `type`, `price`, (@gt := @gt + `price`) AS Total,

(@pr := IF(@pg != `type`, `price`, @pr + `price`)) AS Parcial,

(@st := IF(@pg != `type`, 1, @st + 1)) AS SubTotal, @pg:=`type` AS Tipo

FROM `products_prices`

ORDER BY `type` ASC) AS TiposParciais

GROUP BY Tipo;

type	price	Total	Parcial	SubTotal	Tipo
A	20.00	20.00	20.00	1	A
A	20.00	40.00	40.00	2	A
A	20.00	60.00	60.00	3	A
B	70.00	130.00	70.00	1	B
C	530.50	660.50	530.50	1	C
C	530.50	1191.00	1061.00	2	C

Tipo	TotalPorTipo
A	60.0
B	70.0
C	1061.0

5. SQL

- Conjunto de Operações para Manipulação de Dados-

❑ Funções RAND() e RAND(N) no MySQL: (p.1445)

1) RAND()

Retorna um valor v ponto flutuante randômico, onde $0 \leq v < 1.0$.

2) RAND(N)

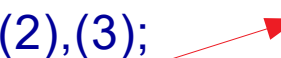
Se um argumento inteiro constante N é especificado, o mesmo é usado como valor semente, que produz uma sequência repetível de valores de colunas.

Exemplos de uso:

```
CREATE TABLE t (i INT);
```

```
INSERT INTO t VALUES(1),(2),(3);
```

```
SELECT i, RAND() FROM t;
```



i	RAND()
1	0.61914388706828
2	0.93845168309142
3	0.83482678498591

3 rows in set (0.00 sec)

5. SQL

- Conjunto de Operações para Manipulação de Dados-

❑ Funções RAND() e RAND(N) no MySQL: (p.1445)

ORDER BY RAND() combinado com a cláusula LIMIT é usado para selecionar um número randômico (amostragem) de um conjunto de linhas:

Ex:

Selecionar, aleatoriamente, 1000 registros de junção entre as tabelas *table1* e *table2*:

Solução:

```
SELECT * FROM table1, table2 WHERE a=b AND c<d ORDER BY RAND()  
LIMIT 1000;
```

Referências



- ❑ Notas de Aula – Prof. Angelo Brayner
- ❑ [https://docs.microsoft.com/pt-br/previous-versions/sql/sql-server-2008-r2/ms190452\(v=sql.105\)](https://docs.microsoft.com/pt-br/previous-versions/sql/sql-server-2008-r2/ms190452(v=sql.105))
- ❑ <https://www.devmedia.com.br/conceitos-e-exemplos-do-compute-by-compute-by-e-case-sql-server-2008-parte-1/21575>



FIM