10 – Introdução

- Até agora, todas as consultas que fizemos foram feitas com base em condições verificadas linha por linha, ou seja, se uma determinada linha da tabela estivesse dentro das condições de pesquisa, elas seriam relacionadas no resultado.

- Entretanto, pode ser necessário a execução de operações envolvendo um grupo de registros. Por exemplo, para fazer uma consulta que retorna o valor total da folha de pagamento de uma empresa.

- Para realizar essa operação, teremos que somar o salário de todos os funcionários da empresa, ou seja, somar a coluna salário da tabela de empregados de todos os registros ativos da tabela.

- O SQL oferece funções para esse tipo de consulta, além de uma série de outros recursos de agregação e agrupamento.

10.1 – Funções de Agregação

- As funções de agregação realizam uma operações específicas nas colunas de um determinado grupo. As principais funções de agregação são:

Função	Descrição
MIN( )	Retorna o menor valor de um conjunto de valores estabelecidos na consulta.
MAX( )	Retorna o maior valor de um conjunto de valores estabelecidos na consulta.
AVG( )	Retorna a média de um conjunto de valores estabelecidos na consulta.
SUM( )	Retorna a soma de valores de um conjunto de valores estabelecidos na consulta.
COUNT( )	Retorna a quantidade de valores de um conjunto

10.1 – Funções de Agregação

- Vamos imaginar um caso onde seja necessário uma consulta na tabela de produtos, onde se deseja exibir como resultado a soma do preço de custo de todos os produtos, o produto mais barato e mais caro, a média do preço de custo dos produtos e quantos produtos temos cadastrado no banco de dados:

id_produto	descricao	preco_custo	preco_venda	id_categoria	quantidade
1	Processador i7 2.8GHz	200.00	400.00	2	30
2	Processador i5 2.0GHz	150.00	300.00	2	30
3	Processador i3 2.2GHz	100.00	200.00	2	30
4	Memória Kingston 8GB	100.00	200.00	1	20
5	Memória Kingston 16GB	200.00	300.00	1	*NULL*
6	Memória Scandisk 4GB	80.00	120.00	1	20
7	HD 1TB Samsung	200.00	300.00	4	40
8	HD 2TB Samsung	300.00	400.00	4	40
9	HD SSD 100GB Samsung	500.00	800.00	4	15
10	HD SSD 200GB Seagate	1000.00	1500.00	4	15
11	Placa Mãe Intel P8	300.00	600.00	3	35
12	Placa Mãe ASUS XM	340.00	650.00	3	35
13	Mouse Wireless Multilaser	30.00	60.00	5	30
14	Mouse com fio Multilaser	20.00	40.00	5	30
15	Teclado Wireless Dell	30.00	60.00	5	30
16	Teclado com fio Multilaser	10.00	20.00	5	30
17	Monitor LG 15	300.00	600.00	6	20
18	Monitor Dell 14	200.00	500.00	6	20

10.1 – Funções de Agregação

- O exemplo abaixo retorna a soma total do preço de custo dos produtos, o produto mais barato e mais caro, o preço de custo médio dos produtos e número de produtos cadastrados:

```
SELECT SUM(preco_custo) AS Total_Custo, MIN(preco_custo) AS Menor_Custo, MAX(preco_custo) AS Maior_Custo, FORMAT (AVG(preco_custo),2) AS Media_Custo, COUNT(preco_custo) AS Num_Produto FROM produto;
```

Total_Custo	Menor_Custo	Maior_Custo	Media_Custo	Num_Produto
4060.00	10.00	1000.00	225.56	18

10.1 – Funções de Agregação

- Também é possível acrescentar filtros nas consultas que possuem função de agregação. Como no exemplo abaixo, onde deseja-se exibir o total de produtos da categoria 3(Monitor):

```
SELECT produto.id_categoria, categoria.descricao, SUM(produto.quantidade) AS Total_Estoque
FROM produto JOIN categoria USING(id_categoria) WHERE produto.id_categoria=6;
```

id_categoria	descricao	Total_Estoque
6	Monitores	40

10.1 – Funções de Agregação

- As funções de agregação permitem que operações aritméticas sejam feitas, como no exemplo abaixo, onde se deseja calcular o total do valor de estoque (soma da quantidade de cada produto da categoria multiplicado pelo valor unitário) de cada uma das categorias:

```
SELECT SUM(quantidade*preco_custo) AS Valor_Estoque FROM produto;
```

Valor_Estoque
94700.00

10.2 – Funções de Agrupamento

- O agrupamento de dados na linguagem **SQL** é obtido com uma cláusula **GROUP BY** combinada com o comando **SELECT** após o uso da cláusula **WHERE** e antes da cláusula **ORDER BY.**

- O comando GROUP BY faz o agrupamento de uma determinada coluna, agrupando os valores armazenados nestas, de forma que operações de agregação possam ser realizadas.

- Como exemplo, é possível pensar na situação onde deseja-se saber a quantidade de funcionários em determinado departamento. Neste caso, será preciso usar o GROUP BY na coluna 'id_departamento' da tabela de funcionários, de forma que os registros sejam agrupados quando possuírem o mesmo valor. Com o comando COUNT é feita a contagem no grupo. O quadro abaixo ilustra a sintaxe:

```
SELECT <campos> FROM <tabela> WHERE <condição>
GROUP BY <campos> [ASC | DESC]
ORDER BY <campos> [ASC | DESC]
```

10.2 – Funções de Agrupamento

- O quadro abaixo mostra a implementação da consulta que retorna a quantidade de funcionários existente em cada departamento da empresa.

```
SELECT id_departamento, COUNT(*) AS 'Número de Funcionários' FROM
funcionario GROUP BY (id_departamento);
```

id_departamento	Número de Funcionários
1	1
2	2
3	2
4	4
5	1

10.2 – Funções de Agrupamento

- O quadro abaixo mostra a implementação da consulta que retorna a quantidade e o valor total (soma do preço de custo com quantidade) dos produtos existentes em cada categoria:

```
SELECT categoria.descricao, SUM(preco_custo*quantidade), SUM(quantidade)
FROM produto JOIN categoria USING(id_categoria) GROUP BY
(produto.id_categoria) ORDER BY categoria.descricao
```

descricao ▲ 1	sum(preco_custo*quantidade)	SUM(quantidade)
Acessórios	2700.00	120
Disco	42500.00	110
Memórias	3600.00	40
Monitores	10000.00	40
Placa Mãe	22400.00	70
Processadores	13500.00	90

10.2 – Funções de Agrupamento

- O quadro abaixo mostra uma consulta que retorna o total da folha de pagamento (soma dos salários de todos os funcionários) e a quantidade de funcionários do departamento de Vendas:

```
SELECT id_departamento, departamento.nome, SUM(salario_fixo) AS 'Folha de Pagamento', COUNT(*) AS 'Total de Funcionários' FROM funcionario JOIN departamento USING(id_departamento) WHERE departamento.nome='Vendas' GROUP BY (funcionario.id_departamento);
```

id_departamento	nome	Folha de Pagamento	Total de Funcionários
4	Vendas	5000.00	4

10.2 – Funções de Agrupamento

- A consulta abaixo mostra o total vendido por cada funcionário (soma do total de todos os pedidos) no ano de 2017:

```
SELECT nome, sum(total) AS 'Total Vendas' FROM funcionario JOIN pedido
USING(id_funcionario) WHERE YEAR(data)=2017 GROUP BY(id_funcionario)
ORDER BY SUM(total);
```

nome	Total Vendas
João da Garrincha	2800.00
John Rambo	3200.00

10.3 – Cláusula HAVING

- A cláusula **HAVING** determina uma condição (determina critérios) para que o agrupamento ocorra. Este comando pode ser usado apenas em conjunto com o **GROUP BY.**

- Outra observação importante é que não se deve usar funções de agregação no **WHERE**. Quando for necessário utilizar uma agregação como condição, deve-se colocá-la no HAVING.

- O HAVING é diferente do WHERE. O WHERE restringe os resultados obtidos sempre após o uso da cláusula FROM, ao passo que a cláusula HAVING determina condições para o agrupamento.

- Como exemplo, caso seja necessário mostrar a quantidade de funcionários de todos os departamentos que possuem no mínimo 2 funcionários. Neste caso, será necessário colocar o 'COUNT(*)>=2' como condição de agrupameto. Ele deve ser colocado no HAVING.

10.3 – Cláusula HAVING

- O exemplo abaixo mostra uma consulta que retorna todos os departamentos com no mínimo dois funcionários:

```
SELECT departamento.nome, count(*) FROM funcionario JOIN departamento
USING(id_departamento) GROUP BY(id_departamento) HAVING
COUNT(*)>=2;
```

nome	count(*)
Diretoria	2
TI	2
Vendas	4

10.3 – Cláusula HAVING

- O exemplo abaixo mostra uma consulta que retorna as categorias que possuem um valor total de estoque com valor superior a 5000 reais:

```
SELECT categoria.descricao, SUM(preco_custo*quantidade) AS 'Valor de Estoque' FROM produto JOIN categoria USING(id_categoria) GROUP BY (produto.id_categoria) HAVING(SUM(preco_custo*quantidade)>=5000) ORDER BY categoria.descricao
```

descricao ▲ 1	Valor de Estoque
Disco	42500.00
Monitores	10000.00
Placa Mãe	22400.00
Processadores	13500.00