No vídeo anterior, aprendemos como ordenar resultados. Agora, vamos estudar como agrupá-los. **Agrupar** é juntar campos que são repetidos e, nos casos de campos numéricos, em meio a essa junção temos a opção de aplicar fórmulas matemáticas, que podem ser de soma, de média, de máximo ou mínimo valor, entre outros.

Para ficar mais fácil de entender, vamos ver em um exemplo. Observe a tabela a seguir, em que a coluna "X" apresenta letras e a coluna "Y" mostra valores numéricos:

O comando que vamos aprender é o GROUP BY, que apresentará o resultado agrupando valores numéricos por uma chave de critério:

Texto

O conteúdo gerado por IA pode estar incorreto.

O GROUP BYserá responsável por agrupar as linhas em que "X" tem o mesmo valor. Por exemplo: as linhas 1 e 2 serão unidas em uma só, e o campo "Y" terá o valor "3", por causa da fórmula de soma SUM(Y) aplicada na consulta. O retorno dessa seleção será o seguinte:

Acabamos de conhecer a fórmula SUM (soma), vamos ver mais quatro opções. A fórmula MAX (máximo) exibirá o maior valor:

Texto

O conteúdo gerado por IA pode estar incorreto.

Na tabela original, a letra "A" aparece duas vezes. Os valores de "Y" nessas ocorrências são 2 e 1. Como estamos usando a fórmula MAX(Y), o valor exibido será o maior, no caso, o número 2. O mesmo vale para a letra "B", que tem três ocorrências cujos valores de "Y" são 2, 3 e 1, nessa ordem. Com o uso de MAX(Y), o valor exibido será 3, e assim em diante.

Já a fórmula MIN(mínimo) fará o contrário - exibirá o menor valor:



A fórmula AVG (do inglês, *average*) fará uma **média** dos valores:



Ou seja, no caso da letra "A", o MySQL verá quais são os valores de "Y" (2 e 1) e fará a média deles, chegando a 1,5. No caso de "B", serão computados os valores 2, 3 e 1 e, feita a média, o resultado exibido será 2 e assim por diante.

E a fórmula COUNT mostrará o número de ocorrência:

No caso de "A", temos duas linhas, então o valor exibido é 2. No caso de "B", temos três linhas, então o resultado é 3, e assim em diante. Então, essas são as cinco fórmulas que podemos usar com o GROUP BY.

Um detalhe importante: quando omitimos os campos de agregação e utilizamos apenas a fórmula, não é necessário colocar a cláusula GROUP BY. Por exemplo:

Interface gráfica do usuário, Aplicativo

O conteúdo gerado por IA pode estar incorreto.

O MySQL irá somar todos os campos de "Y" e simplesmente mostrar o resultado. Se, em vez disso, usássemos MIN, o retorno seria 1, pois o menor valor de "Y" na tabela é 1. Com MAX, teríamos 5. Com AVG, a média daria 2.22. Com COUNT obteríamos 9.

Então, vamos ao MySQL Workbench criar alguns exemplos na base "sucos\_vendas". Criaremos um novo script e começaremos consultando toda a tabela de clientes:

Texto

O conteúdo gerado por IA pode estar incorreto.

Em seguida, vamos restringir um pouco essa consulta, selecionando apenas o estado e o limite de crédito na tabela:

Texto

O conteúdo gerado por IA pode estar incorreto.

Ao fazer essa consulta, veremos uma lista em que cada linha mostra o estado e o limite de crédito de um cliente. Se nosso objetivo for saber o total de limite de crédito de cada estado, usaremos o GROUP BY:

**SELECT** ESTADO, SUM(LIMITE\_DE\_CREDITO) **as** LIMITE\_TOTAL **FROM** tabela\_de\_clientes **GROUP** **BY** ESTADO;

Sempre que uma fórmula for usada, é necessário usar um *alias* ("apelido") para o campo. No nosso caso, definimos "LIMITE\_TOTAL" como *alias* da soma dos limites de crédito.

O retorno mostrará os limites de crédito somados agrupados por estado: em São Paulo, o limite total é R$810.000,00 e, no Rio de Janeiro, é R$995.000,00.

Faremos, agora, uma consulta relativa às embalagens e aos preços de lista presentes na tabela de produtos:

Uma imagem contendo Texto

O conteúdo gerado por IA pode estar incorreto.

Vamos supor que precisamos descobrir qual é o preço mais caro de cada tipo de embalagem (PET, garrafa e lata):

**SELECT** EMBALAGEM, MAX(PRECO\_DE\_LISTA) **as** MAIOR\_PRECO **FROM** tabela\_de\_produtos **GROUP** **BY** EMBALAGEM;

Com essa consulta, constatamos que o produto mais caro que vem em garrafas tem o valor de R$13,312. O mais caro com embalagem PET é R$38,012. E o de lata custa R$4,56.

Além disso, é possível criar seleções com o comando COUNT:

**SELECT** EMBALAGEM, COUNT(\*) **as** CONTADOR **FROM** tabela\_de\_produtos **GROUP** **BY** EMBALAGEM;

O retorno nos mostrará a quantidade de produtos que existem com cada tipo de embalagem: 11 tem embalagem garrafa, 15 vem em PET e 5 em lata.

Vale lembrar que podemos aplicar **critérios de filtro** juntamente do ORDER BYe do GROUP BY. Como exemplo, primeiramente vamos selecionar os limites de crédito agrupados por bairro:

**SELECT** BAIRRO, SUM(LIMITE\_DE\_CREDITO) **as** LIMITE **FROM** tabela\_de\_clientes **GROUP** **BY** BAIRRO;

E, para filtrar a consulta, usaremos uma cláusula WHERE para ver apenas os limites de crédito dos bairros da cidade do Rio de Janeiro:

**SELECT** BAIRRO, SUM(LIMITE\_DE\_CREDITO) **as** LIMITE **FROM** tabela\_de\_clientes **WHERE** CIDADE = 'Rio de Janeiro' **GROUP** **BY** BAIRRO;

Ademais, é possível usar mais de um campo no GROUP BY:

**SELECT** ESTADO, BAIRRO, SUM(LIMITE\_DE\_CREDITO) **as** LIMITE **FROM** tabela\_de\_clientes **GROUP** **BY** ESTADO, BAIRRO;

Note que adicionamos o campo "ESTADO" no início da seleção (SELECT ESTADO) e também no fim, depois de GROUP BY. Na primeira ocorrência, estamos determinando como exibiremos os dados, já na segunda indicamos como queremos que sejam agrupados.

Em seguida, aplicaremos um filtro que irá restringir a consulta somente à cidade do Rio de Janeiro:

**SELECT** ESTADO, BAIRRO, SUM(LIMITE\_DE\_CREDITO) **as** LIMITE **FROM** tabela\_de\_clientes **WHERE** CIDADE = 'Rio de Janeiro' **GROUP** **BY** ESTADO, BAIRRO;

Ou seja, estamos selecionando estado, bairro e a soma dos limites de crédito somente da cidade do Rio de Janeiro, agrupando esses dados por estado e bairro.

E, por fim, se nossa meta for visualizar esse resultado de forma **ordenada**, podemos ainda incluir o ORDER BY:

**SELECT** ESTADO, BAIRRO, SUM(LIMITE\_DE\_CREDITO) **as** LIMITE **FROM** tabela\_de\_clientes

**WHERE** CIDADE = 'Rio de Janeiro'

**GROUP** **BY** ESTADO, BAIRRO

**ORDER** **BY** BAIRRO;

Dessa forma, veremos que o retorno respeitará a ordem alfabética dos bairros, começando por Água Santa, Barra da Tijuca, Cidade Nova e assim por diante. Essa última consulta é mais complexa, pois estamos agrupando, filtrando e ordenando dados em uma única seleção.

Aviso, quando realizamos o agrupamento é importante destacar que podemos receber o erro 1055 que acontece porque o MySQL tem um modo de operação chamado "only\_full\_group\_by", o qual determina que todas as colunas contidas no SELECT também estejam presentes na cláusula GROUP BY (que realiza o processo de agregação dos registros).