**Bruno Campagnol de Oliveira**

**OBS:Não consegui entender filtros quando eles são datas**

1 - Escreva uma consulta de agregação no MongoDB para calcular o total de vendas na coleção "vendas", considerando o campo "valor" em cada documento.

db.vendas.aggregate([

{

$group: {

\_id: null,

total\_vendas: { $sum: "$VALOR" }

}

}

])

Texto

Descrição gerada automaticamente com confiança média

2 - Escreva uma consulta de agregação no MongoDB para contar o número de vendas na coleção "vendas" agrupadas por categoria.

db.vendas.aggregate([

{

$group: {

\_id: "$CATEGORIA",

total\_vendas: { $sum: 1 }

}

}

])

Texto

Descrição gerada automaticamente

3 - Escreva uma consulta de agregação no MongoDB para calcular o valor médio de vendas na coleção "vendas" agrupadas por marca

db.vendas.aggregate([

{

$group: {

\_id: "$MARCAS",

valor\_medio: { $avg: "$VALOR" }

}

}

])

Texto

Descrição gerada automaticamente

4 - Escreva uma consulta de agregação no MongoDB para calcular o total de produtos vendidos na coleção "vendas" agrupadas por código do cliente

db.vendas.aggregate([

{

$group: {

\_id: "$COD\_CLIENTE",

total\_produtos: { $sum: "$QTD" }

}

}

])

Texto

Descrição gerada automaticamente

5 - Escreva uma consulta de agregação no MongoDB para identificar o produto mais vendido na coleção "vendas", com base no campo "QTD" em cada documento.

db.vendas.aggregate([

{

$group: {

\_id: "COD\_PRODUTO",

total\_vendas: { $sum: "$QTD" }

}

},

{

$sort: { total\_vendas: -1 }

},

{

$limit: 1

}

])

Texto

Descrição gerada automaticamente

6 - Escreva uma consulta de agregação no MongoDB para calcular o total de vendas na coleção "vendas" agrupadas por data de venda.

db.vendas.aggregate([

{

$group: {

\_id: "$DATA\_VENDA",

total\_vendas: { $sum: "$VALOR" }

}

}

])

Texto

Descrição gerada automaticamente

7 - Escreva uma consulta de agregação no MongoDB para calcular o total de produtos vendidos na coleção "vendas" agrupadas por código do cliente e ordene os resultados em ordem decrescente do total de produtos vendidos.

db.vendas.aggregate([

{

$group: {

\_id: "$COD\_CLIENTE",

total\_produtos: { $sum: "$QTD" }

}

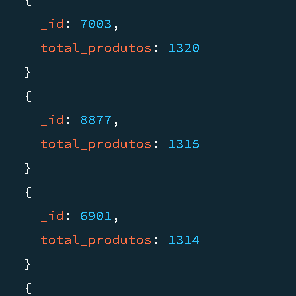
},

{

$sort: { total\_produtos: -1 }

}

])



• 8 - Escreva uma consulta de agregação no MongoDB para calcular o valor médio de vendas na coleção "vendas" agrupadas por código do cliente.

db.vendas.aggregate([

{

$group: {

\_id: "$COD\_CLIENTE",

valor\_medio: { $avg: "$VALOR" }

}

}

])

Texto

Descrição gerada automaticamente

9 - Escreva uma consulta de agregação no MongoDB para contar o número de vendas na coleção "vendas" agrupadas por descrição do produto

db.vendas.aggregate([

{

$group: {

\_id: "$DESCRICAO",

total\_vendas: { $sum: 1 }

}

}

])

Interface gráfica do usuário, Texto

Descrição gerada automaticamente

10 - Escreva uma consulta de agregação no MongoDB para obter o total de produtos vendidos agrupados por descrição e categoria na coleção "vendas"

db.vendas.aggregate([

{

$group: {

\_id: "$DESCRICAO",

total\_qtd: { $sum: "$QTD" },

total\_valor: { $sum: "$VALOR" },

total\_multiplicacao: {

$sum: {

$multiply: ["$QTD", "$VALOR"]

}

}

}

}

])

Texto

Descrição gerada automaticamente

11. Média do valor das vendas por categoria.

db.vendas.aggregate([

{

$group: {

\_id: "$CATEGORIA",

valor\_medio: { $avg: "$VALOR" }

}

}

])

O \_id categoria serve para mostrar o campo que será usado como base de agrupamento e o AVG(operador que acumula valores e mostra a média dos mesmos) será feito usando os valores da coluna VALOR de cada registro

Texto

Descrição gerada automaticamente

12. Valor mínimo e máximo das vendas por categoria.

db.vendas.aggregate([

{

$group: {

\_id: "$CATEGORIA",

valor\_minimo: { $min: "$VALOR"},

valor\_maximo: { $max: "$VALOR" },

}

}

])

Usando o $MIN e $MAX, podemos receber o valor menor e maior do agrupamento escolhido, que no caso foi a coluna CATEGORIA

Texto

Descrição gerada automaticamente

13.Soma do valor das vendas por marca e categoria.

db.vendas.aggregate([

{

$group: {

\_id: { categoria: "$CATEGORIA", marcas:

"$MARCAS" },

total\_valors: { $sum: "$VALOR"},

}

}

]

Texto

Descrição gerada automaticamenteUsei o \_id para selecionar qual o agrupamento que eu preciso, sendo assim coloquei as categorias e as marcas por categoria e depois usei o Operador SUM para somar os valores da coluna VALOR da tabela e mostrar os registros somados no TOTAL\_VALOR

14.Quantidade total de produtos vendidos por marca e categoria.

db.vendas.aggregate([

{

$group: {

\_id: { categoria: "$CATEGORIA", marcas:

"$MARCAS" },

quat\_total: { $sum: "$QTD"},

}

}

])

A mesma logica do exercício anterior só que usando a coluna QTD para fazer a soma da quantidade vendida

Texto

Descrição gerada automaticamente

15.Ordenar as vendas por data de venda em ordem crescente.

db.vendas.aggregate(

[

{ $sort : { DATA\_VENDA : 1 } }

]

)

Usei o SORT para fazer a ordenação de todas as vendas, usando o :1 significa que a ordem deve ser crescente. Não usei \_id pois eu quero todos os campos ordenados, e não apenas um grupo de vendas

Texto

Descrição gerada automaticamente

16.Selecione as vendas com quantidade maior ou igual a 5.

db.vendas.aggregate([

{

$match: { QTD: { $gte: 5 } }

}

])

Usei o Operador MATCH para mostrar os registros baseado na condição que eu especifiquei, como no caso, mostrar todas as vendas com a quantidade

Texto

Descrição gerada automaticamente

17.Valor total das vendas por ano.

db.vendas.aggregate([

{

$group: {

\_id: { data\_ano: "$DATA\_VENDA"},

SOMA\_TOTAL\_ANO: {$sum: "$VALOR"}

}

}

])

18.Média do valor das vendas por ano.

19.Selecione as vendas com valor entre 10 e 30.

db.vendas.aggregate([

{

$match: { VALOR: { $gte: 10, $lte: 30 } }

}

])

Texto

Descrição gerada automaticamente

Usei o Operado MATCH para ele me retornar todos os valores baseado no parâmetro que eu passar, eu juntei ele com o GTE(maior ou igual) e o LTE(menor ou igual) para que ele limitasse os registros no resultado que tenha colunas VALOR entre os GTE e LTE que eu passei.

20.Selecione as 5 vendas com maior valor.

$group: {

\_id: "$COD\_CLIENTE",

valor\_total: { $sum: "$VALOR" }

}

},

{

$sort : { valor\_total : -1 }

},

{

$limit : 5

}

])

Usei a ordenação decrescente para poder visualizar os maiores valores em cima e depois usei $LIMIT para visualizar somente os 5 primeiros registros

Texto

Descrição gerada automaticamente

21.Selecione as 5 vendas com menor valor.

db.vendas.aggregate([

{

$group: {

\_id: "$COD\_CLIENTE",

valor\_total: { $sum: "$VALOR" }

}

},

{

$sort : { valor\_total : 1 }

},

{

$limit : 5

}

])

Igual a questão anterior, porém usando o $SORT: 1, que faz com que ordene de forma crescente, sendo assim os menores valores primeiro na lista. Também usei o $LIMIT para me mostrar apenas os 5 primeiros registros

22. Quantidade total de produtos vendidos por mês.

23. Valor total das vendas por mês.

24. Valor total das vendas por cliente em ordem decrescente.

db.vendas.aggregate([

{

$group: {

\_id: "$COD\_CLIENTE",

valor\_total: { $sum: "$VALOR" }

}

},

{

$sort : { valor\_total : -1 }

}

])

Usei o COD\_CLIENTE para definir qual seria o agrupamento, ou seja, ele seria pelo código de cada cliente, usei o $SUM para somar os valores contidos dentro do campo VALOR de cada venda registrada e após isso $SORT para ordenar a exposição dos valores no resultado, com -1 significando decrescente

Texto

Descrição gerada automaticamente

25. Média do valor das vendas por mês.

26. Selecione as vendas com valor maior ou igual a 50.

db.vendas.aggregate([

{

$match: { VALOR: { $gte: 50 } }

}

])

Operador $MATCH recebe os valores da coluna VALOR e filtra apenas registros igual ou maior que 50

Texto

Descrição gerada automaticamente

27. Valor total das vendas por dia da semana.

28. Média do valor das vendas por dia da semana.

29. Quantidade total de produtos vendidos por dia da semana.

30. Média do valor das vendas por marca.

db.vendas.aggregate([

{

$group: {

\_id: "$MARCAS",

valor\_medio\_marca: { $avg: "$VALOR" }

}

}

])

Agrupei usando o \_id $MARCAS, que é o nome do campo que quero fazer a filtragem, e após isso pelo para fazer a média dos valores compostos no campo VALOR usando a função $AVG para isso

Texto

Descrição gerada automaticamente

31. Selecione as vendas que ocorreram em fevereiro.

32. Ordene as vendas por valor e pelo código do cliente em ordem crescente.

db.vendas.aggregate([

{

$sort : {COD\_CLIENTE: 1, VALOR : 1}

}

])

Não sei se está correto a forma de pensamento, mas eu ordenei inicialmente pelo código do cliente e dentro disso, suas vendas em ordem crescente

Texto

Descrição gerada automaticamente

33. Soma do valor das vendas para cada combinação de categoria e marca.

34. Selecione as vendas que ocorreram no primeiro semestre do ano.

35. Média do valor das vendas no primeiro semestre do ano.

36. Quantidade total de produtos vendidos no primeiro semestre do ano.

37. Ordene as vendas por categoria em ordem alfabética e pelo valor em ordem decrescente.

db.vendas.aggregate([

{

$sort : {CATEGORIA: 1, VALOR : -1}

}

])

Parecido com o exercício 32, para ordenar em ordem alfabética também podemos

38. Selecione as vendas com valor igual a 10, 20 ou 30.

db.vendas.aggregate([

{

$match: { VALOR: { $eq: 10, $eq: 20, $eq: 30 } }

}

])

Texto

Descrição gerada automaticamente

39. Selecione as vendas que ocorreram no mês de dezembro.