WILDCARD - CONCAT – ALIAS – LIMIT - GROUP BY

Prof. Ewerton

Wildcards

Wildcards são caracteres especiais usados para substituir um ou mais caracteres em uma string. Eles são usados com o operador LIKE em uma cláusula WHERE para pesquisar um padrão especificado em uma coluna. Existem dois tipos de wildcards no MySQL: o caractere % que representa zero ou mais caracteres e o caractere _ que representa um único caractere.

LIKE Operator	Description
WHERE CustomerName LIKE 'a%'	Encontra todos os valores que começam com "a"
WHERE CustomerName LIKE '%a'	Encontra todos os valores que terminam com "a"
WHERE CustomerName LIKE '%or%'	Localiza quaisquer valores que tenham “or" em qualquer posição
WHERE CustomerName LIKE '_r%'	Finds any values that have "r" in the second position
WHERE CustomerName LIKE 'a__%'	Encontra todos os valores que começam com "a" e têm pelo menos 3 caracteres de comprimento
WHERE ContactName LIKE 'a%o'	Encontra todos os valores que começam com "a" e terminam com "o"

CONCAT

Para concatenar valores no MySQL, você pode usar a função CONCAT(). Ela retorna a sequência que resulta da concatenação dos argumentos, sendo que pode ter um ou mais argumentos. Para concatenar os valores da coluna nome e sobrenome, basta executar o seguinte código:

```
UPDATE tabela set nome = concat (nome, " ", sobrenome);
```

Outra forma de concatenar valores é utilizando a função CONCAT_WS(). Ela funciona de forma semelhante à função CONCAT(), mas permite especificar um separador entre os valores concatenados. Por exemplo:

```
SELECT CONCAT_WS('-', nome, sobrenome) AS nome_completo FROM tabela;
```

Aliases

No MySQL, um alias é um nome alternativo que pode ser dado a uma tabela (será mostrado com INNER JOIN) ou coluna em uma consulta. Ele é usado para tornar o resultado da consulta mais fácil de entender ou para simplificar a sintaxe da consulta. Para criar um alias para uma coluna ou tabela em uma consulta SELECT, você pode usar a cláusula AS. Por exemplo:

```
SELECT tme_nome AS nome FROM times;
```

Esta consulta retorna todos os registros da tabela “times” e renomeia a coluna “tme_nome” como “nome”.

LIMIT

A cláusula LIMIT é usada para limitar o número de linhas retornadas por uma consulta no MySQL. A sintaxe é a seguinte:

SELECT * FROM tabela LIMIT n;

Esta consulta retorna as primeiras “n” linhas da tabela “tabela”. Você também pode especificar um deslocamento para começar a retornar linhas a partir de uma posição diferente. A sintaxe é a seguinte:

SELECT * FROM tabela LIMIT m, n;

Esta consulta retorna as próximas “n” linhas da tabela “tabela” começando na posição “m”.

GROUP BY

O comando GROUP BY é usado para agrupar registros semelhantes em uma tabela. A seleção é feita de acordo com os critérios definidos na cláusula WHERE, caso ela seja utilizada, e conforme o campo indicado para o agrupamento no comando GROUP BY. Com o GROUP BY podemos agrupar os valores de uma coluna e também realizar cálculos sobre esses valores. Desta forma, ao realizarmos uma consulta, os valores encontrados nas linhas são agrupados e então uma função de agregação pode ser aplicada sobre esses grupos. A sintaxe básica é:

SELECT colunas, função_agregação() FROM tabela WHERE filtro GROUP BY colunas;

Selecionar o número de vendas realizadas por cada vendedor:

```
SELECT id_vendedor, COUNT(*) FROM vendas GROUP BY id_vendedor;
```

Selecionar o número de vendas realizadas por cada vendedor e ordenar pelo número de vendas em ordem decrescente:

```
SELECT id_vendedor, COUNT(*) FROM vendas GROUP BY id_vendedor
ORDER BY COUNT(*) DESC;
```

Selecionar o número de vendas realizadas por cada vendedor e exibir apenas aqueles que realizaram mais de 10 vendas:

```
SELECT id_vendedor, COUNT(*) FROM vendas GROUP BY id_vendedor
HAVING COUNT(*) > 10;
```

Selecionar o número de vendas realizadas por cada vendedor e exibir apenas aqueles que realizaram mais de 10 vendas, ordenando pelo número de vendas em ordem decrescente:

```
SELECT id_vendedor, COUNT(*) FROM vendas GROUP BY id_vendedor HAVING COUNT(*) > 10 ORDER BY COUNT(*) DESC;
```

Selecionar o número de vendas realizadas por cada vendedor em cada mês:

```
SELECT id_vendedor, MONTH(data_venda), COUNT(*) FROM vendas GROUP BY id_vendedor, MONTH(data_venda);
```

Selecionar o número de vendas realizadas por cada vendedor em cada mês e exibir apenas aqueles que realizaram mais de 10 vendas em um determinado mês:

```
SELECT id_vendedor, MONTH(data_venda), COUNT(*) FROM vendas GROUP BY id_vendedor, MONTH(data_venda) HAVING COUNT(*) > 10;
```