Consultas simples

As consultas em SQL são utilizadas para recuperar informações armazenadas em um banco de dados. É possível realizar consultas complexas utilizando a linguagem SQL, o que torna essa ferramenta muito útil para obter informações relevantes de um grande volume de dados.

As consultas em SQL são amplamente utilizadas no dia a dia de diversas empresas e organizações para obter informações de seus bancos de dados. Por exemplo, em um e-commerce, as consultas em SQL podem ser usadas para recuperar informações sobre os produtos, pedidos e clientes. Em uma empresa de serviços financeiros, as consultas em SQL podem ser usadas para recuperar informações sobre as transações, contas e clientes.

A estrutura básica de uma consulta em SQL é a seguinte:

```
SELECT  coluna1,
        coluna2,
        ...
    FROM tabela
    WHERE condição;
```

A cláusula "SELECT" é usada para especificar as colunas que serão exibidas na consulta. A cláusula "FROM" é usada para especificar a tabela da qual as informações serão recuperadas. A cláusula "WHERE" é usada para especificar uma condição que deve ser satisfeita pelos dados a serem recuperados.

Por exemplo, suponha que temos a tabela "clientes" com as seguintes informações:

id	nome	email	idade
1	João Silva	joao@email.com	25
2	Maria Santos	maria@email.com	35
3	Carlos Souza	carlos@gmail.com	30
4	Ana Oliveira	anaoliveira@yahoo.com	28

Se quisermos recuperar os nomes e e-mails dos clientes que têm mais de 30 anos, podemos fazer o seguinte:

```
SELECT  nome,
        email
```

```
    FROM clientes
    WHERE idade > 30;
```

O resultado será o seguinte:

nome	email
Maria Santos	maria@email.com
Carlos Souza	carlos@gmail.com

Outro exemplo é quando queremos recuperar as informações de um cliente a partir do seu ID:

```
SELECT  *
    FROM clientes
    WHERE id = 1
```

O resultado será:

id	nome	email	idade
1	João Silva	joao@email.com	25

Veja que o símbolo `*` quer dizer o mesmo que "todas as colunas", ou seja, quando quisermos buscar todas as colunas de uma tabela, podemos utilizá-lo.

O WHERE

O WHERE é uma cláusula importante na linguagem SQL, que permite filtrar os resultados de uma consulta baseado em uma ou mais condições. Existem diversos sinais que podem ser utilizados no WHERE para comparar valores em uma tabela. A seguir, descreveremos alguns dos principais sinais.

Sinal de igual (=): O sinal de igual é usado para comparar se um valor é igual a outro. Por exemplo, a consulta a seguir retorna os clientes que possuem o nome "João":

```
SELECT * FROM clientes WHERE nome = 'João';
```

Sinal de diferente (!=): O sinal de diferente é usado para comparar se um valor é diferente de outro. Por exemplo, a consulta a seguir retorna os clientes que não possuem o nome "João":

```
SELECT * FROM clientes WHERE nome != 'João';
```

Sinal de maior que (>): O sinal de maior que é usado para comparar se um valor é maior que outro. Por exemplo, a consulta a seguir retorna os clientes que possuem idade maior que 30 anos:

```
SELECT * FROM clientes WHERE idade > 30;
```

Sinal de menor que (<): O sinal de menor que é usado para comparar se um valor é menor que outro. Por exemplo, a consulta a seguir retorna os clientes que possuem idade menor que 30 anos:

```
SELECT * FROM clientes WHERE idade < 30;
```

Sinal de maior ou igual (>=): O sinal de maior ou igual é usado para comparar se um valor é maior ou igual a outro. Por exemplo, a consulta a seguir retorna os clientes que possuem idade maior ou igual a 30 anos:

```
SELECT * FROM clientes WHERE idade >= 30;
```

Sinal de menor ou igual (<=): O sinal de menor ou igual é usado para comparar se um valor é menor ou igual a outro. Por exemplo, a consulta a seguir retorna os clientes que possuem idade menor ou igual a 30 anos:

```
SELECT * FROM clientes WHERE idade <= 30;
```

BETWEEN: A cláusula BETWEEN é usada para comparar se um valor está dentro de um intervalo de valores. Por exemplo, a consulta a seguir retorna os clientes que possuem idade entre 25 e 35 anos:

```
SELECT * FROM clientes WHERE idade BETWEEN 25 AND 35;
```

IN: O operador IN é uma cláusula utilizada em consultas SQL que permite selecionar registros em que um determinado campo tenha um valor que esteja dentro de uma lista de valores específicos. O uso do operador IN pode ser mais simples e conveniente do que a utilização de múltiplas cláusulas OR.

Por exemplo, suponha que temos uma tabela de clientes e queremos selecionar apenas aqueles que moram em algumas cidades específicas. Podemos utilizar o operador IN para escrever uma consulta como esta:

```
SELECT  *
    FROM clientes
    WHERE cidade IN ('São Paulo', 'Rio de Janeiro', 'Belo Horizonte');
```

Essa consulta irá retornar apenas os registros da tabela de clientes em que o campo "cidade" tenha um valor igual a "São Paulo", "Rio de Janeiro" ou "Belo Horizonte".

O operador IN pode ser usado com qualquer tipo de valor, incluindo strings, números e datas. Além disso, também é possível utilizar subconsultas no lugar da lista de valores, como no exemplo a seguir:

```
SELECT  *
    FROM clientes
```

```
    WHERE cidade IN (
        SELECT cidade FROM estados WHERE sigla = 'SP'
    );
```

Neste caso, a consulta seleciona todos os clientes que moram em cidades do estado de São Paulo, utilizando uma subconsulta para obter a lista de cidades.

Além de todos estes comandos, ainda é possível realizar a união de dois ou mais filtros, utilizando as cláusulas `AND` e `OR`.

O operador AND é usado para combinar duas ou mais condições em uma cláusula WHERE. Ele retorna verdadeiro somente se todas as condições forem verdadeiras. Por exemplo, a seguinte consulta retorna todas as linhas da tabela "clientes" onde o país é "Brasil" e a idade é maior que 18:

```
SELECT * FROM clientes WHERE pais = 'Brasil' AND idade > 18;
```

Já o operador OR, por outro lado, é usado para combinar duas ou mais condições em uma cláusula WHERE, e retorna verdadeiro se pelo menos uma das condições for verdadeira. Por exemplo, a seguinte consulta retorna todas as linhas da tabela "clientes" onde o país é "Brasil" ou a idade é maior que 18:

```
SELECT * FROM clientes WHERE pais = 'Brasil' OR idade > 18;
```

Além disso, é possível combinar operadores AND e OR em uma mesma cláusula WHERE para construir condições mais complexas. Por exemplo, a seguinte consulta retorna todas as linhas da tabela "clientes" onde o país é "Brasil" e a idade é maior que 18, ou onde o país é "Argentina" e a idade é maior que 21:

```
SELECT  *
    FROM clientes
    WHERE (pais = 'Brasil' AND idade > 18)
        OR (pais = 'Argentina' AND idade > 21);
```

LIKE

O operador LIKE merece um espaço só para ele, pois com o LIKE podemos fazer diversos filtros em atributos do tipo texto.

O operador LIKE é uma cláusula utilizada em consultas SQL que permite fazer correspondência de padrões em valores de texto. O LIKE é usado para buscar por padrões de caracteres em uma coluna específica de uma tabela, e é geralmente combinado com caracteres especiais, chamados de caracteres curinga, para tornar a busca mais flexível.

Os caracteres curinga mais comuns usados com o operador LIKE são:

- % (porcentagem): representa zero, um ou vários caracteres.

_ (sublinhado): representa um único caractere.

Por exemplo, se quisermos selecionar todos os clientes cujos nomes comecem com "J", podemos utilizar a cláusula LIKE da seguinte forma:

```
SELECT * FROM clientes WHERE nome LIKE 'J%';
```

Nesse caso, o caractere % é usado para representar zero, um ou vários caracteres após a letra "J". A consulta retornará todos os clientes cujos nomes começam com a letra "J".

Neste exemplo, seriam retornados clientes como: Jorge, João e Joana.

Outra forma de utilizar o operador LIKE é utilizando o sublinhado, que representa um único caractere. Por exemplo, para selecionar todos os clientes cujos nomes tenham quatro letras e comecem com a letra "A", podemos utilizar a cláusula LIKE da seguinte forma:

```
SELECT * FROM clientes WHERE nome LIKE 'A___';
```

Nesse caso, os três sublinhados representam três caracteres quaisquer após a letra "A". A consulta retornará todos os clientes cujos nomes começam com a letra "A" e possuam mais três caracteres.

Neste exemplo, seriam retornados clientes como: Ana, Amo e Ada.

> É importante lembrar que o uso do operador LIKE pode ser sensível a maiúsculas e minúsculas, dependendo das configurações do banco de dados. Além disso, existem outras variações do operador LIKE, como o ILIKE (insensitive LIKE), que é insensível a maiúsculas e minúsculas, e o NOT LIKE, que retorna todos os registros que não correspondem ao padrão especificado.

Referências e materiais complementares

https://www.postgresql.org/docs/current/sql-select.html