SQL

Linguagem de consulta estruturada

Você sabia que seu material didático é interativo e multimídia? Isso significa que você pode interagir com o conteúdo de diversas formas, a qualquer hora e lugar. Na versão impressa, porém, alguns conteúdos interativos ficam desabilitados. Por essa razão, fique atento: sempre que possível, opte pela versão digital. Bons estudos!

A programação em banco de dados depende do conhecimento das instruções SQL. A Linguagem SQL é a forma de tratarmos os vários aspectos de um banco de dados, desde sua criação e criação de suas tabelas, inserção dos dados e, principalmente, manipulação desses dados para que sua correlação nos gere informações úteis para uso.

SQL (*Structured Query Language*)

A Linguagem de Consulta Estruturada estabeleceu-se como a linguagem padrão dos sistemas gerenciadores de bancos de dados (SGBD).

MySQL

O MySQL é uma poderosa ferramenta gráfica desenvolvida para o ambiente Windows, e realiza de maneira gráfica todas as tarefas possíveis de serem realizadas na versão de linha de comando. Tem seu código aberto, é distribuído e suportado pela Oracle Corporation, e trabalha com várias plataformas, tendo sido escrito em C e C++.

Fonte: Shutterstock.

SAIBA MAIS

Com o MySQL pode-se definir instruções SQL incorporadas de maneira embutida ou dinâmica na maioria das linguagens de programação como, Node.js, PHP, C#, C++, Java, Android, Swift para citar apenas algumas delas (ORACLE, 2018).

Na linguagem SQL destacam-se cinco subconjuntos de instruções:

- DDL – Data Definition Language
- DML – Data Manipulation Language
- DQL – Data Query Language
- DCL – Data Control Language
- DTL – Data Transaction Language

No MySQL as instruções SQL foram implementadas também em subconjuntos, sempre seguindo os requisitos de confiabilidade, performance, integridade e disponibilidade.

Linguagem de Definição de Dados (DDL – Data Definition Language)

Conjunto de instruções SQL para definição dos dados e sua estrutura.

CREATE - criar banco de dados, tabelas, colunas.

DROP – excluir banco de dados, tabelas, colunas.

ALTER – alterar banco de dados, tabelas, colunas.

TRUNCATE – esvaziar toda a tabela.

Linguagem de Manipulação dos Dados (DML – Data Manipulation Language)

Conjunto de instruções SQL para inserção e manutenção dos dados.

INSERT – inserir dados em uma tabela.

UPDATE – atualizar os dados existentes em uma tabela.

DELETE – excluir registros de uma tabela.

Linguagem de Consulta a Dados (DQL – Data Query Language)

Conjunto de instruções SQL para consultas de todos os dados armazenados e suas relações, e ajuda para comandos de sintaxe.

SELECT – principal instrução de consulta do SQL.

SHOW – exibir todas as informações além dos dados (metadata).

HELP – exibir informações do Manual de referência do MySQL.

Linguagem de Controle de Dados (DCL – Data Control Language)

Conjunto de instruções SQL para controle de autorizações de acesso e seus níveis de segurança.

GRANT – esta instrução concede privilégios às contas de usuário.

REVOKE – esta instrução permite revogar os privilégios da conta de usuário.

Linguagem de Transação de Dados (DTL – Data Transaction Language)

Conjunto de instruções para o controle de transações lógicas que são agrupadas e executadas pela DML.

START TRANSACTION – iniciar uma nova transação.

SAVEPOINT – identificação de um determinado ponto numa transação.

COMMIT – instrução de entrega ao SGBD, fazendo com que todas as alterações sejam permanentes.

ROLLBACK [TO SAVEPOINT] – instrução ao SGBD de reverter toda a transação, cancelando todas as alterações ou até determinado ponto da transação.

RELEASE SAVEPOINT – instrução para remoção de um SAVEPOINT

Estrutura básica das consultas SQL

A principal instrução básica, consiste em três cláusulas:

SELECT – identificação dos campos desejados numa consulta.

FROM – lista as tabelas que deverão ser lidas.

WHERE – consiste em expressões lógicas envolvendo os campos das tabelas da cláusula FROM.

As consultas são resultados de um produto cartesiano das tabelas especificadas na cláusula FROM, sendo realizada através de uma condição ou mais condições da cláusula WHERE aplicando os resultados nos campos da cláusula SELECT (MACHADO, 2014).

1 - A cláusula SELECT

O resultado de uma consulta SQL é uma tabela. Considerando o banco de dados mundo, suas tabelas e respectivos campos:

- cidade (Id, Nome, CodigoPais, Estado, Populacao)
- pais (Codigo, Nome, Continente, Regiao, Area, Populacao)
- linguapais (CodigoPais, Lingua, Oficial, Porcentagem)

Exemplos:

Consulta simples

```
SELECT Nome FROM cidade;
```

Consulta precisa

```
SELECT nome, nascimento, cpf FROM clientes WHERE cpf = '12345678901';
```

Consulta imprecisa

```
SELECT * FROM clientes;
```

Conheça mais sobre as instruções básicas de consulta.

Resultados com duplicidade

```
SELECT ALL Nome FROM cidade;
```

ou

```
SELECT * FROM cidade;
```

Resultados sem duplicidade

```
SELECT DISTINCT Nome FROM cidade;
```

Sempre que necessário utilize o DISTINCT para remover duplicidades nas consultas.

Com expressões aritméticas

```
SELECT Nome, Populacao/2 FROM cidade;
```

Será exibida uma tabela com os campos Nome e Populacao, porém o valor de Populacao está dividido por 2.

Utiliza-se os operadores +, -, * e /.

Existem operações com datas, outros tipos especiais e outras funções aritméticas.

2 - A cláusula WHERE

Para entendermos a cláusula WHERE, vamos supor a seguinte consulta:

"Encontre todos os nomes de cidades que tenham uma população inferior a 100000":

```
SELECT Nome, Populacao
FROM cidade
WHERE Populacao < 100000;
```

BETWEEN

Para simplificar algumas operações de comparação existe o BETWEEN:

```
SELECT Nome, Populacao
FROM cidade
WHERE Populacao BETWEEN 90000 AND 100000;
```

Ao invés de:

```
SELECT Nome, Populacao
FROM cidade
WHERE Populacao >= 90000 AND Populacao <= 100000;
```

SAIBA MAIS

Os conectivos AND, OR e NOT podem também ser utilizados. Também pode-se utilizar operadores de comparação <, <=, >, >=, = e <>. Podemos também ainda usar o operador de comparação NOT BETWEEN.

3 - A cláusula FROM

Ela produz um produto cartesiano das tabelas especificadas na cláusula.

Exemplos:

"Encontre todas os nomes e população das cidades e línguas do seu país":

```
SELECT cidade.Nome, cidade.Populacao, linguapais.Linguage
FROM cidade, pais, linguapais
WHERE cidade.CodigoPais = pais.Codigo AND pais.Codigo = linguapais.CodigoPais;
```

Pode-se ainda verificar a mesma consulta para as cidades que falem “Português”.

“Encontre todas os nomes e população das cidades e dos países que falem Português”:

```
 SELECT cidade.Nome, cidade.Populacao, linguapais.Linguage
 FROM cidade, pais, linguapais
 WHERE cidade.CodigoPais = pais.Codigo AND pais.Codigo = linguapais.CodigoPais AND
linguaPais.Linguagem = "Português";
```

Operação de renomeação e variáveis do registro

Os campos podem ser renomeados, e para isso usa-se a cláusula AS, da seguinte forma: **`nome-antigo`** AS **`nome-novo`**

Esta cláusula poderá ser utilizada na cláusula SELECT e FROM.

Utilizando alguns exemplos anteriores:

```
SELECT Nome, Populacao as PopulacaoDaCidade
FROM cidade
```

O resultado é uma tabela com o campo `Populacao` sendo renomeado para `PopulacaoDaCidade`.

A cláusula AS pode ser utilizada também para definição de variáveis de registro e deverão estar associadas a uma determinada tabela. Elas são definidas na cláusula FROM.

Exemplo:

“Encontre todas os nomes e população das cidades e línguas do seu país”:

```
SELECT C.Nome, C.Populacao, L.Linguagem
FROM cidade as C, pais as P, linguapais as L
WHERE C.CodigoPais = P.Codigo AND P.Codigo = L.CodigoPais;
```

Operações de String e Ordenação

O operador LIKE determina a correspondência de padrões, e estes são descritos usando caracteres especiais:

Porcentagem (%): corresponde a qualquer substring.

Sublinhado (_): corresponde a qualquer caractere.

Exemplo:

“Encontre os nomes de todas as cidades na tabela cidade onde os nomes iniciem por ‘Sor’”:

```
SELECT Nome
FROM cidade
WHERE Nome like ‘Sor%’;
```

SAIBA MAIS

Considere os seguintes exemplos:

'Sor%' localizará qualquer string iniciando com "Sor".

'%or%' localizará qualquer string iniciando contendo "or".

'___' localizará qualquer string com exatamente três caracteres.

'___%' localizará qualquer string com pelo menos três caractere.s

ORDER BY

A cláusula ORDER BY faz com que os registros sejam ordenados pelo campo especificado. Exemplo:

"Encontre os nomes de todas as cidades na tabela cidade ordenados por nome":

```
SELECT Nome
FROM cidade
ORDER BY Nome;
```

Esta cláusula ordena em ordem crescente por padrão, mas você pode explicitar utilizando ASC, ou também ordenar de maneira decrescente com DESC. Exemplo:

"Encontre os nomes de todas as cidades na tabela cidade ordenados por nome em ordem decrescente":

```
SELECT Nome
FROM cidade
ORDER BY Nome DESC;
```

O resultado de uma consulta é o produto da quantidade de registros de cada tabela. Tabelas que tenham milhares ou milhões de registros poderão ter como resultado um número absurdo de registros. Mesmo sendo uma simples consulta, sempre devemos prestar atenção ao resultado que se espera, pois pode ocorrer uma degradação de performance do servidor ao ponto de quebrar uma conexão.

Fonte: Shutterstock.