MODELAGEM E DESENVOLVIMENTO DE BANCO DE DADOS

Fabrício Felipe Meleto Barboza

SAGAH
SOLUÇÕES EDUCACIONAIS INTEGRADAS

SQL: conceitos e funcionalidades

Objetivos de aprendizagem

Ao final deste texto, você deve apresentar os seguintes aprendizados:

- Identificar a linguagem SQL.
- Reconhecer os principais tipos de linguagem SQL.
- Relacionar os principais tipos de linguagem SQL.

Introdução

A linguagem SQL é utilizada para a realização de consultas em dados armazenados em um banco de dados. Ela é muito importante, pois permite a manipulação, de forma simples e poderosa, dos dados presentes em um Sistema de Gerenciamento de Banco de Dados.

Neste capítulo, você irá estudar sobre os conceitos e as funcionalidades da linguagem SQL. Verá, também, os diversos tipos de operadores suportados pela linguagem SQL e, com isso, como construir comandos SQL mais refinados e inteligentes.

Identificando a linguagem SQL

Segundo Elmasri e Navathe (2004), a linguagem SEQUEL nasceu unicamente para ser a linguagem de consulta do sistema System R da IBM. Sua evolução originou a linguagem SQL tanto difundida atualmente.

Saiba mais

Segundo Santos (2008), "a linguagem SQL surgiu em meados da década de 1970, sendo resultado de um estudo de E. F. Codd, membro do laboratório de pesquisa da IBM em San Jose, Califórnia. Esse estudo tinha foco em desenvolver uma linguagem que se adapta ao modelo relacional".

Para realizar a inserção, remoção, edição ou qualquer outra atividade com os dados que irão para o banco de dados ou que já estão nele, é necessária uma interação com o mesmo. Dessa forma, a linguagem SQL (*Structure Query Language* ou Linguagem de Consulta Estruturada) é a intermediária essa interação, criando a ponte necessária entre o que precisa ser feito e o banco de dados efetivamente. Assim, para que algo aconteça, é necessário que a linguagem SQL esteja com a sintaxe e os valores desejados corretos.

Pelo fato de ser de fácil utilização e aprendizado, essa linguagem foi rapidamente difundida pelo mundo e, assim, é utilizada para gerenciar os bancos de dados do tipo MariaDB (e MySQL), Microsoft SQLServer, PostgreSQL e Oracle. É a linguagem comumente usada por *database administrators* diariamente em suas jornadas de trabalho.

Perceba que o MongoDB não utiliza a linguagem SQL para criar suas consultas e manipular os dados necessários. Isso se dá pela forma com que ele organiza suas informações e também como as disponibiliza de volta ao sistema quando este solicita uma consulta, por exemplo. Assim, ele consegue desempenhar uma velocidade incrível de resposta a consultas se comparado aos bancos de dados mais tradicionais, que utilizam a linguagem de consulta estruturada (SQL). Portanto, o MongoDB é um banco de dados do tipo não relacional.

Fique atento

A linguagem SQL apresenta uma sintaxe bem estruturada. Um exemplo básico seria:

SELECT atributos (obrigatório)
FROM tabelas (obrigatório)
WHERE condições (opcional)

Assim, para retornar o nome e o CPF de todos os clientes que moram no Paraná cadastrados na tabela "Cliente" do Quadro 1 a seguir, faríamos:

SELECT Nome, CPF
FROM Cliente
WHERE Estado = PR

Uma justificativa para por que a linguagem SQL é tão utilizada e difundida em todo o mundo pode ser que:

> A linguagem possui muitos recursos, tanto para manipulação de dados (DML, *Data Manipulation Language*) como para a definição de dados (DDL, *Data Definition Language*). Por conta disso, SQL é a linguagem mais utilizada em sistemas de bancos de dados (KOMATSU, 2011, p. 20).

Saiba mais

"Em 1982, foi lançada a primeira versão padronizada da linguagem SQL, que veio ganhando melhorias de acordo com sua evolução e tornando-se, assim, a mais poderosa ferramenta para definição e manipulação de BDs. Hoje, é utilizada em grande parte dos BDs existente, tais como MySQL, SQLServer, Firebird, dentre outros". (SANTOS, 2008, documento on-line).

Para realizar as consultas e transações, a linguagem SQL conta com o auxílio de operadores lógicos, operadores relacionais, cláusulas condicionais e funções de agregação. Veremos cada um desses auxílios no texto que segue.

Para facilitar o seu entendimento, os exemplos de SQLs representados na sequência de cada quadro estarão baseados na tabela Clientes a seguir:

Quadro 1. Visualização da "Tabela Clientes"

ClienteNome	Idade	Estado	E-mail
João	15	São Paulo	
Maria	22	Paraná	
Felipe	53	Rio de Janeiro	felipe@aaa.com
Conceição	56	Rio de Janeiro	conceicao@aaa.com
Sophie	24	São Paulo	sophie@aaa.com
Augusto	40	Minas Gerais	

Operadores lógicos

Os operadores lógicos são empregados para realizar uma combinação ou descombinação de associações da consulta. Veja um exemplo no Quadro 2.

Quadro 2. Operadores lógicos

Operador	Ação
AND	Retorna apenas quando as duas condições forem verdadeiras
OR	Retorna quando uma das duas condições for verdadeira
NOT	Retorna o contrário da expressão

Exemplos práticos:

1. Consultar os nomes dos clientes que sejam do estado do Paraná ou de São Paulo:

```
SELECT ClienteNome
FROM clientes
WHERE
Estado = 'Paraná'
OR
'São Paulo';
```

2. Consultar os nomes dos clientes que sejam do estado Rio de Janeiro e tenham nome Felipe:

```
SELECT ClienteNome
FROM clientes
WHERE
Estado = 'Paraná'
AND
ClienteNome = 'Felipe';
```

Operadores relacionais

Os operadores relacionais são utilizados para realizar comparações entre valores e estrutura do Sistema de Gerenciamento de Banco de Dados. Veja um exemplo no Quadro 3.

Quadro 3. Operadores relacionais

Operador	Ação
=	Igual
<>	Diferente
>	Maior
>=	Maior ou igual
<	Menor
<=	Menor ou igual
BETWEEN	Buscar em um intervalo
NOT BETWEEN	Buscar o que não está em um intervalo
LIKE	Buscar com caractere curinga '%'
NOT LIKE	Buscar o que não está no caractere curinga '%'
IN	Buscar em uma lista
NOT IN	Buscar o que não está em uma lista

Exemplos práticos:

1. Consultar os nomes dos clientes que tenham idade maior que 50 anos:
```
SELECT ClienteNome
FROM clientes
WHERE
Idade > 50;
```
2. Consultar os nomes dos clientes que tenham idade entre 50 e 60 anos:
```
SELECT ClienteNome
FROM clientes
WHERE
Idade BETWEEN 50 AND 60;
```
3. Consultar os nomes dos clientes cujo nome comece com a letra J:
```
SELECT ClienteNome
FROM clientes
WHERE
ClienteNome LIKE 'J%';
```
4. Consultar os nomes dos clientes cuja idade é menor ou igual a 30 anos:
```
SELECT ClienteNome
FROM clientes
WHERE
Idade <= 30;
```

Cláusulas condicionais

As cláusulas condicionais servem para condicionar os dados que serão apresentados por meio de uma consulta ou, ainda, deletados ou editados.

Veja um exemplo no Quadro 4.

Quadro 4. Cláusulas condicionais

Operador	Ação
FROM	Indica qual tabela irá receber o comando
WHERE	Especifica as condições necessárias
GROUP BY	Agrupa os resultados semelhantes
HAVING	É utilizada para expressar a condição de satisfação para cada grupo
ORDER BY	Ordena o resultado
DISTINCT	Retorna dados sem repetição
UNION	Combina duas consultas SQL sem repetição de dados
UNION ALL	Combina duas consultas SQL com repetição de dados, se houver

Exemplos práticos:

1. Consultar os nomes dos clientes, ordenando pelo nome:
   ```
   SELECT ClienteNome
   FROM clientes
   ORDER BY ClienteNome;
   ```
2. Consultar os estados dos clientes sem repetição:
   ```
   SELECT DISTINCT(Estado)
   FROM clientes;
   ```
3. Contar quantos clientes cada estado tem:
   ```
   SELECT COUNT(*)
   FROM clientes
   GROUP BY Estado;
   ```

Funções de agregação

As funções de agregação são responsáveis por extrair informações de forma rápida e precisa da consulta realizada, poupando tempo do responsável.

Veja um exemplo no Quadro 5.

Quadro 5. Funções de agregação

Operador	Ação
AVG	Média entre os valores
COUNT	Retorna quantos registros possui
SUM	Soma os valores
MAX	Retorna o maior valor
MIN	Retorna o menor valor

Exemplos práticos:

1. Consultar a média de idade dos clientes:
```
SELECT AVG(Idade)
FROM clientes;
```
2. Contar quantos clientes são do Rio de Janeiro:
```
SELECT COUNT(*)
FROM clientes
WHERE
Estado = 'Rio de Janeiro';
```
3. Somar a idade de todos os clientes:
```
SELECT SUM(Idade)
FROM clientes;
```
4. Pegar o cliente mais velho:
```
SELECT MAX(Idade)
FROM clientes;
```
5. Pegar o cliente mais novo:
```
SELECT MIN(Idade)
FROM clientes;
```

Reconhecendo e relacionando os principais tipos de linguagens SQL

O papel da linguagem SQL para os Sistemas de Gerenciamento de Bancos de Dados é de crucial importância. Ela pode ser dividida em quatro subgrupos, que estudaremos separadamente a seguir.

Linguagem de manipulação de dados ou *data manipulation language* (DML)

A linguagem de manipulação de dados ou *data manipulation language* (DML) refere-se a um grupo secundário dentro da linguagem SQL e que pode realizar operações de manipulação de dados, tal como adição, remoção, exibição ou edição.

Exemplos de comandos:

- INSERT
- DELETE
- SELECT
- UPDATE

Exemplos práticos:

1. Inserir nova entrada na tabela Clientes:

```
INSERT INTO
clientes
VALUES (
'Gregório',
'45',
'Amazonas',
'gregorio@aaa.com'
);
```

2. Deletar entrada na tabela Clientes:

```
DELETE FROM
clientes
WHERE
NomeCliente = 'Gregório';
```

3. Editar entrada na tabela Clientes:

```
UPDATE
clientes
SET
Estado = 'Goiás'
WHERE
NomeCliente = 'Gregório';
```

Linguagem de definição de dados ou *data definition language* (DDL)

A linguagem de definição de dados ou *data definition language* (DDL) é a encarregada de trabalhar com as tabelas do Sistema de Gerenciamento de Banco de Dados e suas propriedades; portanto, seu uso contempla chaves primárias, por exemplo.

Exemplos de comandos:

- CREATE
- DROP
- ALTER

Exemplos práticos:

1. Criar nova tabela Fornecedores:
```
CREATE TABLE fornecedores (
    Nome varchar(255),
    Telefone varchar(255),
    Endereco varchar(255)
);
```
2. Deletar tabela Fornecedores:
```
DROP TABLE fornecedores;
```

Linguagem de controle de dados ou *data control language* (DCL)

A linguagem de controle de dados ou *data control language* (DCL) é responsável pelas permissões dos usuários, fornecendo, ou não, o retorno do comando executado pelo usuário. Dessa forma, a DCL integra uma das camadas de segurança do Sistema de Gerenciamento de Banco de Dados.

Os comandos para manipulação das permissões são:

- GRANT: para permitir o acesso pelo usuário
- REVOKE: para retirar as permissões do usuário

Linguagem de transação de dados ou *data transaction language* (DTL)

A linguagem de transação de dados ou *data transaction language* (DTL) serve de guia para realizar trabalhos no banco de dados e salvar apenas após a conclusão total do mesmo ou, ainda, retornar ao estado anterior ao último ponto salvo.

Exemplos de comandos:

- BEGIN WORK ou START TRANSACTION
- COMMIT: gravar as modificações no banco
- ROLLBACK: retornar o banco a um estado anterior ao último COMMIT

Linguagem de consulta de dados ou *data query language* (DQL)

A linguagem de consulta de dados ou *data query language* (DCL) contempla o comando que permite a consulta pelos dados presentes no Sistema de Gerenciamento de Banco de Dados.

Exemplo de comando:

- SELECT

Referências

ELMASRI R.; NAVATHE S. B. *Fundamentals of database systems*. 4. ed. Boston: Addison-Wesley, 2004.

KOMATSU, A. S. V. *Interpretador de linguagens de consulta relacionais*. 2011. 44 f. Trabalho de Conclusão de Curso (Graduação em Informática) – Universidade de São Paulo, São Paulo, 2011. Disponível em: <https://bcc.ime.usp.br/tccs/2011/andresuzuki/monografia.pdf>. Acesso em: 19 jul. 2018.

Leitura recomendada

SANTOS, J. C. *Entendendo a linguagem SQL*. 2008. Disponível em: <https://www.devmedia.com.br/entedendo-a-linguagem-sql/7775>. Acesso em: 19 jul. 2018.

Encerra aqui o trecho do livro disponibilizado para esta Unidade de Aprendizagem. Na Biblioteca Virtual da Instituição, você encontra a obra na íntegra.

Conteúdo:

SAGAH
SOLUÇÕES EDUCACIONAIS INTEGRADAS