Guia Completo de Cláusulas SQL (SQLite, PostgreSQL, MySQL, SQL Server)

Este documento apresenta exemplos de uso de cláusulas SQL, desde as mais simples até as mais avançadas, com foco principal em SQLite, comparações com PostgreSQL e inclusão de exemplos em MySQL e SQL Server, quando houver diferenças.

1 - SELECT

Exemplo 1 (SQLite, PostgreSQL, MySQL, SQL Server):

```
1. SELECT * FROM tabela;
2.
```

Descrição: Seleciona todas as colunas de uma tabela.

---

2 - WHERE

Exemplo 1 (SQLite, PostgreSQL, MySQL, SQL Server):

```
1. SELECT * FROM tabela WHERE coluna = 'valor';
2.
```

Descrição: Filtra os resultados com base em uma condição.

---

3 - INSERT

Exemplo 1 (SQLite, PostgreSQL, MySQL, SQL Server):

```
1. INSERT INTO tabela (coluna1, coluna2) VALUES ('valor1', 'valor2');
2.
```

Descrição: Insere valores em colunas específicas.

---

4 - UPDATE

Exemplo 1 (SQLite, PostgreSQL, MySQL, SQL Server):

```
1. UPDATE tabela SET coluna1 = 'novo_valor' WHERE coluna2 = 'valor';
2.
```

Descrição: Atualiza valores em uma tabela com base em uma condição.

---

5 - DELETE

Exemplo 1 (SQLite, PostgreSQL, MySQL, SQL Server):

```
DELETE FROM tabela WHERE coluna = 'valor';

```

Descrição: Remove linhas de uma tabela com base em uma condição.

---

6 - JOIN

Exemplo 1 (SQLite, PostgreSQL, MySQL, SQL Server):

```
SELECT a.coluna1, b.coluna2
FROM tabela1 a
JOIN tabela2 b ON a.id = b.id;

```

Descrição: Realiza um INNER JOIN entre duas tabelas.

---

7 - GROUP BY

Exemplo 1 (SQLite, PostgreSQL, MySQL, SQL Server):

```
SELECT coluna, COUNT(*)
FROM tabela
GROUP BY coluna;

```

Descrição: Agrupa os dados com base em uma coluna e aplica funções agregadas.

---

8 - HAVING

Exemplo 1 (SQLite, PostgreSQL, MySQL, SQL Server):

```
SELECT coluna, COUNT(*)
FROM tabela
GROUP BY coluna
HAVING COUNT(*) > 1;

```

Descrição: Filtra os grupos criados pelo GROUP BY.

---

9 - DISTINCT

Exemplo 1 (SQLite, PostgreSQL, MySQL, SQL Server):

SELECT DISTINCT coluna FROM tabela;

Descrição: Retorna valores únicos de uma coluna.

---

10 - LIMIT

Exemplo 1 (SQLite, PostgreSQL, MySQL):

```
1. SELECT * FROM tabela LIMIT 10;
2.
```

Descrição: Limita o número de linhas retornadas.

Exemplo 2 (SQL Server):

```
1. SELECT TOP 10 * FROM tabela;
2.
```

Descrição: O SQL Server utiliza TOP em vez de LIMIT.

---

11 - OFFSET

Exemplo 1 (SQLite, PostgreSQL, MySQL):

```
1. SELECT * FROM tabela LIMIT 10 OFFSET 5;
2.
```

Descrição: Pula um número específico de linhas antes de começar a retornar os resultados.

Exemplo 2 (SQL Server):

```
1. SELECT * FROM tabela
2. ORDER BY coluna
3. OFFSET 5 ROWS FETCH NEXT 10 ROWS ONLY;
4.
```

Descrição: O SQL Server utiliza OFFSET combinado com FETCH NEXT.

---

12 - UNION

Exemplo 1 (SQLite, PostgreSQL, MySQL, SQL Server):

```
1. SELECT coluna1 FROM tabela1
2. UNION
3. SELECT coluna1 FROM tabela2;
4.
```

Descrição: Combina os resultados de duas consultas, removendo duplicatas.

---

13 - CREATE TABLE

Exemplo 1 (SQLite, PostgreSQL, MySQL):

```
CREATE TABLE tabela (
    id INTEGER PRIMARY KEY,
    coluna1 TEXT,
    coluna2 INTEGER
);

```

Descrição: Cria uma nova tabela com colunas e tipos de dados definidos. O tipo INTEGER PRIMARY KEY é usado para auto incremento (compatível com SQLite, PostgreSQL e MySQL).

Exemplo 2 (SQL Server):

```
CREATE TABLE tabela (
    id INT IDENTITY(1,1) PRIMARY KEY,
    coluna1 NVARCHAR(MAX),
    coluna2 INT
);

```

Descrição: Utiliza IDENTITY para auto incremento e tipos de dados específicos como NVARCHAR. Observa-se que os tipos podem variar entre os bancos de dados, por exemplo, NVARCHAR no SQL Server para suportar texto Unicode.

```
CREATE TABLE tabela (
    id INTEGER PRIMARY KEY,
    coluna1 TEXT,
    coluna2 INTEGER
);

```

Descrição: Cria uma nova tabela com colunas e tipos de dados definidos.

Exemplo 2 (PostgreSQL):

```
CREATE TABLE tabela (
    id SERIAL PRIMARY KEY,
    coluna1 TEXT,
    coluna2 INTEGER
);

```

Descrição: Difere no tipo de dado SERIAL para auto incremento.

Exemplo 3 (SQL Server):

```
CREATE TABLE tabela (
    id INT IDENTITY(1,1) PRIMARY KEY,
    coluna1 NVARCHAR(MAX),
    coluna2 INT
);

```

Descrição: Utiliza IDENTITY para auto incremento e tipos de dados específicos como NVARCHAR.

---

14 - ALTER TABLE

Exemplo 1 (SQLite, MySQL, SQL Server):

```
1. ALTER TABLE tabela ADD coluna3 TEXT;
2.
```

Descrição: Adiciona uma nova coluna a uma tabela existente. No caso do SQLite, há limitações quanto a alterações mais complexas, como renomear colunas ou remover colunas existentes, que exigem a recriação da tabela. Em MySQL e SQL Server, comandos adicionais estão disponíveis para essas operações.

Exemplo 2 (PostgreSQL):

```
1. ALTER TABLE tabela ADD COLUMN coluna3 TEXT;
2.
```

Descrição: Similar ao SQLite, mas exige a palavra-chave COLUMN. O PostgreSQL também oferece suporte a uma gama maior de operações, como renomear ou alterar tipos de colunas diretamente.

Exemplo 2 (PostgreSQL):

```
1. ALTER TABLE tabela ADD COLUMN coluna3 TEXT;
2.
```

Descrição: Similar, mas exige a palavra-chave COLUMN.

Exemplo 3 (MySQL, SQL Server):

```
1. ALTER TABLE tabela ADD coluna3 TEXT;
2.
```

Descrição: Uso idêntico ao SQLite.

---

15 - DROP TABLE

Exemplo 1 (SQLite, PostgreSQL, MySQL, SQL Server):

```
1. DROP TABLE tabela;
2.
```

Descrição: Remove uma tabela e todos os seus dados.
