Revisão Aprofundada

DDL, DML e DCL - As 3 Linguagens SQL

Você domina as diferenças entre DDL, DML e DCL? 15 questões críticas para consolidar seu conhecimento.

Abertura: DDL, DML e DCL - As 3 Linguagens SQL

DDL (Data Definition Language)

Definição de Estrutura

COMANDOS
CREATE, ALTER, DROP, TRUNCATE

DEFINE
Tabelas, Colunas, Tipos, Restrições

REVERSIBILIDADE
Não reversível (sem backup)

DML (Data Manipulation Language)

Manipulação de Dados

COMANDOS
SELECT, INSERT, UPDATE, DELETE

DEFINE
Conteúdo (registros/linhas)

REVERSIBILIDADE
Reversível (com ROLLBACK)

DCL (Data Control Language)

Controle de Acesso

COMANDOS
GRANT, REVOKE

DEFINE
Permissões de usuários/papéis

REVERSIBILIDADE
Reversível (com REVOKE)

Questões DDL (Q1-Q3): TRUNCATE, Restrições e ALTER TABLE

Q1 **Qual comando DDL remove todos os dados de uma tabela, mas mantém a estrutura intacta? Qual é a diferença para o DELETE?**

RESPOSTA CHAVE

TRUNCATE TABLE é DDL e remove todos os dados sem registrar no log. DELETE é DML e registra cada linha.

Aspecto	TRUNCATE	DELETE
Tipo	DDL	DML
Velocidade	Muito rápido	Mais lento
Log	Não registra linhas	Registra cada linha
ROLLBACK	Não reversível	Reversível

Q2 **Qual restrição garante que um valor de coluna nunca estará ausente e qual garante que o valor é único?**

RESPOSTA CHAVE

NOT NULL (nunca ausente) e UNIQUE (valor único sem duplicatas)

```
-- NOT NULL: Coluna obrigatória
CREATE TABLE clientes (
  nome VARCHAR(100) NOT NULL
);

-- UNIQUE: Sem duplicatas
CREATE TABLE clientes (
  email VARCHAR(100) UNIQUE
);
```

Q3 **Como se adiciona uma nova coluna à tabela Clientes para armazenar o email (VARCHAR(100))?**

RESPOSTA CHAVE

ALTER TABLE Clientes ADD COLUMN email VARCHAR(100);

```
-- Adicionar coluna simples
ALTER TABLE clientes ADD COLUMN email VARCHAR(100);

-- Adicionar com restrição NOT NULL
ALTER TABLE clientes ADD COLUMN data_cadastro DATE NOT NULL DEFAULT CURRENT_DATE;
```

Questões DDL (Q4-Q5): Integridade Referencial e DROP TABLE

Q4 **O que acontece quando você tenta inserir um registro que viola uma Chave Estrangeira (FK)?**

RESPOSTA CHAVE

O SGBD retorna um erro, impedindo a inserção e mantendo a Integridade Referencial (um pedido não pode existir sem um cliente válido).

```
-- Criar tabelas com relacionamento
CREATE TABLE clientes (cliente_id INT PRIMARY KEY);
CREATE TABLE pedidos (pedido_id INT, cliente_id INT,
  FOREIGN KEY (cliente_id) REFERENCES clientes(cliente_id));

-- ✓ CORRETO: cliente_id 1 existe
INSERT INTO pedidos VALUES (101, 1);

-- ❌ ERRO: cliente_id 999 não existe
INSERT INTO pedidos VALUES (102, 999); -- FOREIGN KEY constraint failed
```

Q5 **Qual a diferença entre DROP TABLE e DELETE FROM Tabela?**

RESPOSTA CHAVE

DROP TABLE é DDL e destrói a estrutura completa (tabela e dados). DELETE é DML e remove apenas os dados (mantendo a estrutura).

Aspecto	DROP TABLE	DELETE FROM
Tipo	DDL	DML
Remove	Estrutura + Dados	Apenas Dados
Espaço	Liberado	Mantido
Reversibilidade	Não reversível	Reversível
Risco	Crítico	Alto

Questões DML (Q6-Q8): WHERE, INSERT e LIMIT

Q6 **Qual é a cláusula de segurança mais crítica ao executar os comandos UPDATE e DELETE?**

RESPOSTA CHAVE

A cláusula WHERE. Sem ela, todos os registros da tabela serão afetados (erro catastrófico).

```
-- ❌ PERIGO: UPDATE sem WHERE
UPDATE clientes SET salario = 5000; -- Todos recebem 5000!

-- ✓ CORRETO: UPDATE com WHERE
UPDATE clientes SET salario = 5000 WHERE departamento = 'Vendas';

-- ❌ PERIGO: DELETE sem WHERE
DELETE FROM clientes; -- Todos deletados!

-- ✓ CORRETO: DELETE com WHERE
DELETE FROM clientes WHERE status = 'inativo';
```

Q7 **Cite as duas formas básicas de escrita do comando INSERT.**

RESPOSTA CHAVE

1. Especificando colunas (RECOMENDADO); 2. Sem especificar colunas (implícito - menos seguro).

```
-- Forma 1: Especificando colunas (RECOMENDADO)
INSERT INTO clientes (cliente_id, nome, email)
VALUES (1, 'João Silva', 'joao@email.com');

-- Forma 2: Sem especificar colunas (implícito)
INSERT INTO clientes
VALUES (2, 'Maria Santos', 'maria@email.com');

-- Múltiplos registros
INSERT INTO clientes (cliente_id, nome, email) VALUES
  (3, 'Pedro Costa', 'pedro@email.com'),
  (4, 'Ana Silva', 'ana@email.com');
```

Q8 **Se você tem 10.000 registros, mas o comando SELECT retorna apenas 10, qual cláusula foi usada para limitar o número de registros exibidos?**

RESPOSTA CHAVE

LIMIT (MySQL/PostgreSQL) ou FETCH FIRST N ROWS ONLY (SQL Server/Oracle).

```
-- MySQL/PostgreSQL: LIMIT
SELECT * FROM clientes LIMIT 10; -- Primeiros 10
SELECT * FROM clientes LIMIT 10 OFFSET 20; -- 10 a partir do 20º

-- SQL Server/Oracle: FETCH FIRST
SELECT * FROM clientes FETCH FIRST 10 ROWS ONLY;

-- Com ORDER BY (recomendado)
SELECT * FROM clientes ORDER BY cliente_id DESC LIMIT 10;
```

Questões DML (Q9-Q10): Filtro Lógico e UPDATE

Q9 **Você precisa listar apenas os clientes que moram em 'São Paulo' e têm mais de 30 anos. Quais cláusulas você usaria?**

RESPOSTA CHAVE

WHERE combinada com o operador lógico AND. AND exige que AMBAS as condições sejam verdadeiras.

```
-- AND: Ambas as condições devem ser verdadeiras
SELECT * FROM clientes
WHERE cidade = 'São Paulo' AND idade > 30;

-- OR: Pelo menos uma condição deve ser verdadeira
SELECT * FROM clientes
WHERE cidade = 'São Paulo' OR cidade = 'Rio de Janeiro';

-- Combinando AND e OR com parênteses
SELECT * FROM clientes
WHERE (cidade = 'São Paulo' OR cidade = 'RJ')
  AND idade > 30;
```

Q10 **Como você alteraria o preço do ProdutoA para 50.00?**

RESPOSTA CHAVE

UPDATE Produtos SET preco = 50.00 WHERE nome_produto = 'ProdutoA';

```
-- UPDATE simples com WHERE
UPDATE produtos SET preco = 50.00
WHERE nome_produto = 'ProdutoA';

-- UPDATE com múltiplas colunas
UPDATE produtos SET preco = 50.00, estoque = 100
WHERE nome_produto = 'ProdutoA';

-- UPDATE com cálculo (aumentar 10%)
UPDATE produtos SET preco = preco * 1.10
WHERE categoria = 'Eletrônicos';

-- UPDATE com múltiplas condições
UPDATE produtos SET preco = 50.00
WHERE nome_produto = 'ProdutoA' AND categoria = 'Eletrônicos';
```

Questões DCL (Q11-Q12): GRANT/REVOKE e Concessão de Leitura

Q11 **Qual é o comando DCL usado para conceder permissões de acesso? E qual remove?**

RESPOSTA CHAVE

GRANT concede privilégios. REVOKE remove privilégios.

```
-- GRANT: Conceder privilégios
GRANT SELECT ON clientes TO role_vendedor;
GRANT SELECT, INSERT ON vendas TO role_vendedor;
GRANT ALL ON * TO role_admin;

-- REVOKE: Remover privilégios
REVOKE SELECT ON clientes FROM role_vendedor;
REVOKE DELETE ON vendas FROM role_vendedor;
REVOKE ALL ON * FROM role_admin;
```

Q12 **Você precisa que o usuário analista_junior consiga ver os dados da tabela Pedidos, mas não possa alterá-los. Qual comando você usaria?**

RESPOSTA CHAVE

GRANT SELECT ON Pedidos TO analista_junior;

```
-- Conceder apenas SELECT (leitura)
GRANT SELECT ON pedidos TO analista_junior;

-- Conceder SELECT em colunas especificas
GRANT SELECT (pedido_id, cliente_id, valor) ON pedidos TO analista_junior;

-- Conceder SELECT em múltiplas tabelas
GRANT SELECT ON pedidos, clientes TO analista_junior;

-- Remover privilégio de leitura
REVOKE SELECT ON pedidos FROM analista_junior;
```

Questões DCL (Q13-Q14): Princípio do Menor Privilégio e Risco

Q13 **(A Questão Mais Importante) Qual é o princípio mais importante na segurança de um banco de dados: dar acesso a todos ou menor privilégio?**

RESPOSTA CHAVE

O Princípio do Menor Privilégio (Principle of Least Privilege - PoLP). Um usuário ou sistema deve ter apenas o nível de acesso e as permissões estritamente necessárias para realizar sua função.

APLICAÇÃO DO POLP

Vendedor: SELECT clientes, INSERT vendas. Analista BI: SELECT *. DBA: ALL. Nunca conceder tudo a todos.

```
-- ❌ ERRADO: Conceder tudo a todos
GRANT ALL ON * TO role_vendedor; -- Vendedor pode deletar tabelas!

-- ✓ CORRETO: Apenas o necessário
GRANT SELECT ON clientes TO role_vendedor;
GRANT INSERT ON vendas TO role_vendedor;
-- Vendedor pode ler clientes e registrar vendas, nada mais
```

Q14 **Qual é o risco de conceder a um Analista de BI permissão de DROP TABLE?**

RESPOSTA CHAVE

Risco Crítico. Um erro acidental ou intencional pode levar à perda irreversível da estrutura do banco de dados e de todos os dados.

```
-- ❌ PERIGO: Conceder DROP a um Analista de BI
GRANT DROP ON * TO role_analista_bi; -- Risco Crítico!
DROP TABLE clientes; -- Perda irreversível!

-- ✓ CORRETO: Apenas SELECT para Analista
GRANT SELECT ON * TO role_analista_bi; -- Analista pode ler, não deletar

-- ✓ CORRETO: DROP apenas para DBAs
GRANT DROP ON * TO role_dba; -- Apenas DBAs podem deletar
```

Questão DCL (Q15): Papéis (Roles) e Administração

Q15 **O que são Papéis (Roles) e por que são melhores do que dar permissões diretamente a usuários individuais?**

RESPOSTA CHAVE

Papéis são grupos de permissões que simplificam a administração. É mais eficiente gerenciar 5 papéis do que 500 usuários individuais, garantindo consistência e auditoria.

ADMINISTRAÇÃO CENTRALIZADA

Modificar privilégios de um papel afeta todos os usuários do papel automaticamente.

CONSISTÊNCIA DE PERMISSÕES

Todos os vendedores têm exatamente os mesmos privilégios. Sem inconsistências.

FACILITA AUDITORIA

Rastrear quem tem acesso é simples: basta verificar os papéis, não 500 usuários.

ESCALABILIDADE

Adicionar novo usuário é rápido: apenas atribua um papel existente.

```
-- Criar papéis
CREATE ROLE role_vendedor;
CREATE ROLE role_analista_bi;
CREATE ROLE role_dba;

-- Conceder privilégios a papéis
GRANT SELECT ON clientes TO role_vendedor;
GRANT SELECT, INSERT ON vendas TO role_vendedor;
GRANT SELECT ON * TO role_analista_bi;
GRANT ALL ON * TO role_dba;

-- Atribuir usuários a papéis
GRANT role_vendedor TO usuario_joao;
GRANT role_vendedor TO usuario_maria;
GRANT role_analista_bi TO usuario_pedro;
GRANT role_dba TO usuario_admin;

-- Modificar privilégios do papel (afeta todos os usuários)
GRANT UPDATE (preco) ON produtos TO role_vendedor;
```

Atividade Prática: Aplicação de Conceitos

Cenário: Loja Virtual

Você é o DBA de uma loja virtual. Precisa criar tabelas, inserir dados, e definir permissões de acesso para diferentes papéis (Vendedor, Analista BI, DBA).

1 Criar Tabelas (DDL)

Crie 3 tabelas: clientes, produtos, vendas. Defina colunas, tipos de dados e restrições (NOT NULL, UNIQUE, PRIMARY KEY, FOREIGN KEY).

2 Inserir Dados (DML)

Insira 5 registros em cada tabela usando INSERT INTO. Teste com WHERE para filtrar dados. Teste UPDATE e DELETE com segurança.

3 Definir Papéis (DCL)

Crie 3 papéis: role_vendedor, role_analista_bi, role_dba. Conceda privilégios apropriados com GRANT (sem DROP para vendedor/analista).

4 Validar Segurança

Teste se cada papel pode fazer apenas o que deve. Vendedor não deve poder deletar tabelas. Analista não deve poder alterar preços.

EXEMPLO: ESTRUTURA DE TABELAS

```
-- Tabela Clientes (DDL)
CREATE TABLE clientes (
  cliente_id INT PRIMARY KEY,
  nome VARCHAR(100) NOT NULL,
  email VARCHAR(100) UNIQUE
);

-- Inserir dados (DML)
INSERT INTO clientes VALUES (1, 'João Silva', 'joao@email.com');

-- Definir permissões (DCL)
GRANT SELECT, INSERT ON clientes TO role_vendedor;
```

Checklist de Validação

- ✓ Todas as 3 tabelas foram criadas com restrições apropriadas
- ✓ Dados foram inseridos e testados com SELECT, UPDATE, DELETE
- ✓ 3 papéis foram criados com privilégios diferenciados
- ✓ Vendedor NÃO pode fazer DROP ou ALTER
- ✓ Analista BI pode ler todos os dados (SELECT)

Conclusão: Consolidação e Próximos Passos

DDL (Q1-Q5)

- Q1: TRUNCATE vs DELETE
- Q2: NOT NULL vs UNIQUE
- Q3: ALTER TABLE
- Q4: Integridade Referencial
- Q5: DROP TABLE vs DELETE

DML (Q6-Q10)

- Q6: Cláusula WHERE
- Q7: Sintaxe INSERT
- Q8: LIMIT/FETCH
- Q9: Filtro Lógico (AND/OR)
- Q10: UPDATE com WHERE

DCL (Q11-Q15)

- Q11: GRANT/REVOKE
- Q12: Concessão SELECT
- Q13: Princípio PoLP
- Q14: Risco de DROP
- Q15: Papéis (Roles)

CHECKLIST DE APRENDIZADOS

- ✓ Entendo as diferenças entre DDL, DML e DCL
- ✓ Compreendo o Princípio do Menor Privilégio (PoLP)
- ✓ Entendo Integridade Referencial e restrições
- ✓ Consigo classificar comandos SQL corretamente
- ✓ Sei quando usar WHERE em UPDATE/DELETE
- ✓ Consigo aplicar GRANT/REVOKE com segurança

"Dominar DDL, DML e DCL é dominar SQL." *Cada linguagem tem seu propósito e risco. DDL estrutura, DML manipula, DCL protege. Juntas, formam a base da segurança e eficiência em banco de dados. Continue praticando!*