Resumão BD

SQL

- Structured Query Language 5Linguagem de Consulta Estruturada);
- linguagem padrão adotada por diferentes SGBDs;
- criar manipular banco de dados relacionais;
- alguns SGBDs trazem variações, porém a lógica estrutural é a mesma;
- caracterizado pela utilização de comandos (instruções);

Classificações das instruções

- de acordo com sua função, normalmente em dois tipos;

1. DDL: Linguagem de Definição de Dados

Permitem criar / alterar a estrutura / excluir uma tabela.

- CREATE TABLE;
- DROP TABLE;
- ALTER TABLE;

2. DML: Linguagem de Manipulação de Dados

Permitem inserir / atualizar / deletar / selecionar registros em uma tabela.

- INSERT;
- UPDATE;
- DELETE;
- SELECT;

3. Outros Comandos

- DESC;

Comandos

CREATE DATABASE

```
CREATE DATABASE nome_do_banco_de_dados;
```

```
USE nome_do_banco_de_dados;
```

CREATE TABLE (DDL)

```
CREATE TABLE nome_da_tabela (
  nomeatributo1 tipodedado tipoatributo,
  nomeatributo2 tipodedado,
  nomeatributo3 tipodedado
);
```

Tipo de Dado	Descrição
VARCHAR (max)	Textos
INTEGER	Números Inteiros
FLOAT	Números Fracionados
DOUBLE	Números Fracionados
DATE	Datas
TIME	Horas
BLOB	Imagens e Arquivos

Tipo de Atributos	Descrição
PRIMARY KEY	Chave Primária
FOREIGN KEY	Chave Estrangeira
NOT NULL	Não Nulo
UNIQUE	Único
AUTO_INCREMENT	Incremento; Numérico Automático;

```
CREATE TABLE Endereco (
  id_end INTEGER PRIMARY KEY
```

```
  rua_end VARCHAR (200) NOT NULL,
  numero_end INTEGER,
  bairro_end VARCHAR (100)
);

CREATE TABLE Cliente (
  id_cli INTEGER PRIMARY KEY,

  nome_cli VARCHAR (100),
  cpf_cli VARCHAR(15),
  data_nasc_cli DATE,
  sexo_cli VARCHAR (20),
  email_cli VARCHAR (100),

  id_end_fk INTEGER,
  FOREIGN KEY (id_end_fk) REFERENCES Endereco (id_end)
);
```

DROP TABLE (DDL)

- Usado para deletar uma tabela no banco de dados;
- Não é possível excluir uma tabela que possua registros dependentes de registros de outra tabela;

```
DROP TABLE nome_da_tabela;
```

ALTER TABLE (DDL)

- Realiza alterações na estrutura de uma tabela

Tipos de Ações

`ADD` - Adicionar Atributo

```
ALTER TABLE nome_da_tabela ADD atributo tipodedado;
```

```
ALTER TABLE cliente ADD endereco_cli VARCHAR (100);
ALTER TABLE cliente ADD renda_cli FLOAT;
```

Posição

- FIRST

```
ALTER TABLE Aluno ADD renda_familiar_alu FLOAT FIRST;
```

- AFTER

```
ALTER TABLE Aluno ADD renda_familiar_alu FLOAT AFTER nome_alu;
```

`DROP` - Excluir Atributo

```
ALTER TABLE nome_da_tabela DROP atributo;
```

```
ALTER TABLE cliente DROP email_cli;
ALTER TABLE cliente DROP nome_cli;
```

`CHANGE` - Altera Atributo

```
ALTER TABLE nome_da_tabela CHANGE nome_do_atributo
novo_nome_do_atributo tipodedado;
```

```
# Exemplo para alterar apenas o NOME do atributo:
ALTER TABLE cliente CHANGE data_nasc_cli data_nascimento_cli DATE;

# Exemplo para alterar apenas o TIPO do atributo:
ALTER TABLE cliente CHANGE data_nascimento_cli data_nascimento_cli
VARCHAR (20);

# Exemplo para alterar o NOME e o TIPO do atributo:
ALTER TABLE cliente CHANGE data_nascimento_cli nascimento_cli DATE;
```

DESC

- Mostra a estrutura física da tabela, incluindo o nome dos atributos e os seus tipos de dados;

```
DESC nome_da_tabela;
```

```
DESC cliente;
```

INSERT (DML)

- Permite a insersão de registros em uma tabela;
- Há dois modos para tal: fazendo referencia ou não aos atributos.

COM Referência

- Pode-se especificar quais dados serão inseridos no registro.
- Os outros atributos terão por padrão o valor nulo (`NULL`).

```
INSERT INTO nome_da_tabela (atributo1, atributo2, atributo4) VALUES
(valorAtributo1, valorAtributo2, valorAtributo4);
```

```
INSERT INTO Cliente (id_cli, nome_cli) VALUES (1, 'José da Silva');

INSERT INTO Cliente (id_cli, nome_cli, telefone_cli) VALUES (2, 'Ana
Maria', '69 9999 8888');

INSERT INTO Cliente (id_cli, nome_cli, email_cli) VALUES (3, 'Gustavo
Silva', 'gustavo@gmail.com');
```

SEM Referência

- Todos os atributos deverão receber um valor, mesmo que seja `NULL`.

```
INSERT INTO nome_da_tabela VALUES (valorAtributo1, valorAtributo2,
valorAtributo3, valorAtributo4);
```

```
INSERT INTO Cliente VALUES (1, 'José da Silva', null, null, null);
INSERT INTO Cliente VALUES (2, 'Ana Maria', null, null, '69 9999
8888');
INSERT INTO Cliente VALUES (3, 'Gustavo Silva', null,
'gustavo@gmail.com', null);
INSERT INTO Cliente VALUES (4, 'Marcos Pereira', '123.456.789-15',
'pereira@gmail.com', '69 98888 7777');
```

Regras

1. Atributos que tenham o tipo `VARCHAR`, `DATE`, `TIME` etc. são inseridos entre aspas simples.
2. Chaves primarias devem ser únicas (sem repetição entre os registros).
3. O uso de `auto_increment` na declaração de um atributo primário fará com que o mesmo não necessite de um valor fornecido na inserção do registro, de maneira a aceitar valores `NULL`.
4. Caso o atributo tenha `not null`, seu valor **deverá** ser preenchido - o SGBD não permitirá a insersão. Os atributos sem `not null` são opcionais.
5. Valores de chaves extrangeiras devem referenciar registros que existam na tabela de origem.

Atenção!

- DATE* ao contrário e dentro de aspas simples (`'2022-10-26'`).
- INTEGER, FLOAT ou DOUBLE* sem aspas (`920`, `13.90`, `9463.82`).

UPDATE (DML)

- Permite alterar os registros que já estejam em uma tabela.

```
UPDATE nome_da_tabela SET atributo = valor, atributo2 = valor2 WHERE condição;
```

> Em cada registro, a comparação será realizada. Caso a condição seja verdadeira, o comando será executado naquele registro. Por favor, dê uma olhada no tópico Condições, que está dentro de Conceitos.

```
UPDATE Cliente SET renda_cli = renda_cli + 1000 WHERE (idade_cli > 30);
```

DELETE (DML)

- Permite apagar um ou vários registros em uma tabela.

- Não existe o comando DELETE de somente um atributo específico; Para isso, veja o comando UPDATE.
-

```
DELETE FROM nome_da_tabela WHERE (condição);
```

```
DELETE FROM Cliente WHERE (id_cli = 1);
```

- Comandos sem uma condição 5WHERE5 excluem todos os registros de uma tabela.

```
DELETE FROM Cliente;
```

Registros Vinculados

- Não é possível excluir registros que possuam chaves primárias vinculadas a chaves estrangeiras em outras tabelas.

Estado

id_est	nome_est
1	Acre
2	Rondônia
3	Amazonas

Cidade

id_cid	nome_cid	id_est_fk
1	Ji-Paraná	2
2	Rio Branco	1
3	Manaus	3

- Cenário 1 6 `Correto`

Os registros que dependem de Rondônia foram excluidos.

```
DELETE FROM Cidade WHERE (nome_cid = 'Ji-Paraná');
DELETE FROM Estado WHERE (id_est = 2);
```

- Cenário 2 ↳ `Erro`

Há cidades que dependem do Acre.

```
DELETE FROM Estado WHERE (id_est = 1);
```

SELECT (DML)

- Consulta os registros armazenados nas tabelas.
- Não modifica nenhum registro.

```
SELECT atributo1, atributo2, ... FROM nome_da_tabela WHERE
(condição);
```

Com o * no lugar atributos, todos os atributos serão selecionados.

```
SELECT * FROM nome_da_tabela WHERE (condição);
```

SELECT / ORDER BY

- Ordena os registros em uma consulta a partir de um atributo.
- Usado no final do comando SELECT.

```
SELECT atributo1, atributo2, ... FROM nome_tabela WHERE (condição)
ORDER BY atributo;
```

```
SELECT * FROM Cliente WHERE (renda_cli > 1000) ORDER BY nome_cli;
```

Palavra chave	Ordenamento
`DESC`	Decrescente
`ASC`	Ascendente (padrão)

```
SELECT * FROM Cliente WHERE (renda_cli > 1000) ORDER BY nome_cli
DESC;
```

SELECT / GROUP BY

- Agrupa registros com um valores iguais em um atributo.
- Mostra apenas os registros diferentes de um determinado atributo.
- Usado no final do comando SELECT;

```
SELECT atributo1 FROM nome_tabela WHERE (condição) GROUP BY
atributo1;
```

```
SELECT idade_cli FROM Cliente GROUP BY idade_cli;
```

`ORDER BY` vem após de `GROUP BY`.

```
SELECT idade_cli FROM Cliente GROUP BY idade_cli ORDER BY
idade_cli;
```

SELECT: Funções Especiais

```
SELECT Nome_Função(atributo) FROM nome_tabela WHERE (condição);
```

Função	Descrição
COUNT(atributo)	quantidade total de registros não nulos de um atributo;
COUNT5DISTINCT atributo)	quantidade total de registros diferentes não nulos de um atributo;
SUM(atributo)	soma dos valores de um atributo;
AVG(atributo)	média dos valores de um atributo;
MIN(atributo)	menor valor de um atributo
MAX(atributo)	maior valor de um atributo
CURTIME55	horário atual do sistema operacional
CURDATE55	data atual do sistema operacional
DATE_FORMAT(atributo, mascara)	data no formato da máscara definida

Função	Descrição
EXTRACT(parte FROM atributo)	extrai uma parte da data
WEEKDAY(atributo)	dia da semana da data

COUNT (atributo)

```
SELECT COUNT(id_cli) FROM Cliente;
```

COUNT (DISTINCT atributo)

```

```

SUM(atributo)

AVG(atributo)

MIN(atributo)

MAX(atributo)

- maior valor de um atributo

CURTIME()

CURDATE()

```
select curdate();
```

DATE_FORMAT(atributo, mascara)

```
SELECT DATE_FORMAT(data_nascimento_cli, '%d/%m/%Y') from cliente;
```

EXTRACT(parte FROM atributo)

Símbolo	Parte
year	ano
month	mês
day	dia

```
SELECT nome_cli, EXTRACT(year FROM datanasc_cli) FROM cliente;
```

ROUND(atributo, casas decimais)

```
SELECT ROUND(valor_total_vend, 2) FROM venda;
```

WEEKDAY(atributo)

Valor	Dia da Semana
0	Segunda-feira
1	Terça-feira
2	Quarta-feira
3	Quinta-feira
4	Sexta-feira
5	Sábado
6	Domingo

```
SELECT WEEKDAY(data_nascimento_cli) FROM cliente;
```

1. Funções não podem ser usadas em conjunto com atributos no mesmo SELECT.

Funções não ser usadas dentro de condições. Para isso, veja Sub Consulta

3. As Funções A UNT e SUM retornam apenas um valor.
4. AS Funções M IN podem retornam um ou vários valores.
5. Funções só p er utilizadas com valores numéricos.

SELECT: Sub sultas

```
 renda_cli as 'Menor Renda'
FROM
 Cliente
WHERE
 (renda_cli = (SELECT MIN(renda_cli) FROM cliente));
```

Conceitos

Ajustes no Workbench

Permitir a alteração de vários registro de uma só vez.

- Edit ⁄ Preferences ⁄ SQL Editor
- Desmarque a caixa de seleção da opção “Safe Updates” na última linha da caixa de diálogo.

Condições

- Representa uma expressão lógica.
- É formada por valores e operadores.
- Retorna um valor, podendo ser `Verdadeiro` 5V5 ou `Falso` 5F5.

> Os comandos `UPDATE` e `DELETE` aceitam essa definição declarado no `WHERE`. Em cada registro, a comparação será realizada. Caso a condição seja verdadeira, o comando será executado naquele registro.

Comparação

```
(valor1 operador valor2)
```

Os valores a serem comparados podem reais ou representados por um atributo.

```
(12 > 40)                    # Falso
('Jonathan' = 'Jon')         # Falso
('1956-03-19' < '2022-04-25') # Verdadeiro
(56 = 56)                    # Verdadeiro
```

```
(media_alu > 60) # = ?
```

Os tipos de valores que podem ser comparados incluem:

- Números Inteiros;
- Números com Casa Decimal;
- Datas;
- Horas;
- Textos.

Comparadores

Operador	Descrição	Exemplo
=	Igual	`(atributo = 'Música');`
>	Maior	`(atributo > 143);`
<	Menor	`(atributo < 15.78);`
>=	Maior ou Igual	`(atributo >= '10:20');`
<=	Menor ou Igual	`(atributo <= '2022-10-10');`
<>	Diferente	`(atributo <> 10);`
BETWEEN	Entre	`(atributo BETWEEN 500 AND 100);`
Like	Parece com	`(atributo LIKE '%Jos_%');`
IS NULL	É nulo	`(atributo IS NULL)`
IS NOT NULL	É não-nulo	`(atributo IS NOT NULL)`

BETWEEN

- Usado para verificar se o valor de um atributo está em um intervalo de valores;

```
(renda_cli BETWEEN 500 AND 1000);
```

LIKE

- Pode ser usado em valores entre aspas simples (`VARCHAR`, `TIME` e `DATE`) para comparar cadeias de caracteres usando padrões de

comparação para um ou mais caracteres.

Símbolo Coringa	Utilidade
`%`	substitui um ou mais caracteres desconhecidos
`_`	substitui um único caractere desconhecido

```
(nome_cli LIKE 'Jackson%');
(nome_cli LIKE '%Silva%');
(nome_cli LIKE '%Bezerra');
(data_nasc_cli LIKE '1987%');
(hora_nasc_cli LIKE '21%');
```

IS NULL / IS NOT NULL

- O operador IS NULL é usado em uma condição para selecionar valores que sejam NULL (nulos) ou NOT NULL (não nulos);
- Os valores NULL não podem ser comparados com o operador = (igual), somente com o IS;

```
SELECT * FROM Cliente WHERE (nome_cli IS NULL);
```

Múltiplas Condições

- É possível por meio de operadores lógicos.
- A ordem importa. Defina-a por meio de parênteses, assim como é feito na matemática.

AND

- Todas as condições devem ser verdadeiras.

```
(condição) AND (condição)
```

Condição A	Condição B	Resultado `AND`
V	V	V
V	F	F

Condição A	Condição B	Resultado `AND`
F	V	F
F	F	F

OR

- Apenas uma das condições precisa ser verdadeira.

```
(condição) OR (condição)
```

Condição A	Condição B	Resultado `OR`
V	V	V
V	F	V
F	V	V
F	F	F

Considerações Finais

Este documento foi feito com base nos documentos sob propriedade intelectual do Professor Jackson, disponibilizados à turma via AVA.

Resumo elaborado por Gabriel R. Antunes - gabrielrodantunes@gmail.com.