BANCOS DE DADOS

LINGUAGEM SQL

BANCOS DE DADOS

- **CONTEÚDO PROGRAMÁTICO:**

  - 1 - SQL (Structure Query Language);
  - 1.2 - Linguagem de Definição de Dados (DDL);
  - 1.2.1 Comandos de criação, alteração e exclusão de objetos do banco;
  - 1.3 Operadores SQL.

BANCOS DE DADOS

- **1.1 - Conceitos de Bancos de dados e SQL;**

A linguagem padrão de bancos de dados relacionais é SQL que é dividida em 5 grupos:

1 - **DDL** - Linguagem de Definição de Dados.
2 - **DML** - Linguagem de Manipulação de Dados.
3 - **DCL** - Linguagem de Controle de Dados.
4 - **DQL** - Linguagem de Consulta de Dados.
5 - **DTL** - Linguagem de Transação de Dados.

BANCOS DE DADOS

- **1.2 - Linguagem de Definição de Dados (DDL):**

Uma DDL permite ao utilizador definir tabelas novas e elementos associados. Os principais comandos deste grupo são:

1 - CREATE
2 - ALTER
3 - DROP

BANCOS DE DADOS

- **1.2 - Linguagem de Definição de Dados (DDL):**

COMANDO CREATE:

- Criar uma base de dados:

```
Create Database Nome_da_Base_de_Dados;

Create Database Minha_Base_de_Dados;
```

- Conectar-se a uma base de dados:

```
USE Nome_da_Base_de_Dados;

USE Minha_Base_de_Dados;
```

BANCOS DE DADOS

- **1.2 - Linguagem de Definição de Dados (DDL):**

COMANDO CREATE:

- Criar uma tabela:

```
CREATE TABLE Nome_Da_Tabela
(
nome_da_coluna_1 tipo_de_dados(tamanho),
nome_da_coluna_2 tipo_de_dados(tamanho),
);

CREATE TABLE Minha_Tabela
(
coluna_1 int(5),
coluna_2 varchar(45)
);
```

BANCOS DE DADOS

- **1.2 - Linguagem de Definição de Dados (DDL):**

- Verificar as bases de dados criadas:

```
show databases;
```

- Verificar as tabelas criadas:

```
show tables;
```

- Verificar a estrutura das tabelas:

```
describe Nome_da_Tabela;

describe MINHA_TABELA;
```

BANCOS DE DADOS

- **1.2 - Linguagem de Definição de Dados (DDL):**

COMANDO ALTER:

- Adicionar uma coluna na tabela:

```
ALTER TABLE NOME_DA_TABELA
ADD NOME_DA_COLUTA Tipo_de_dados(Tamanho);

ALTER TABLE MINHA_TABELA
ADD COLUNA_3 DATETIME;
```

BANCOS DE DADOS

- **1.2 - Linguagem de Definição de Dados (DDL):**

COMANDO ALTER:

- Remover uma coluna na tabela:

```
ALTER TABLE Nome_da_Tabela
DROP COLUMN Nome_da_Coluna;

ALTER TABLE MINHA_TABELA
DROP COLUMN coluna_3;
```

BANCOS DE DADOS

- **1.2 - Linguagem de Definição de Dados (DDL):**

COMANDO ALTER:

- Alterar a coluna de uma tabela:

```
ALTER TABLE Nome_da_Tabela
CHANGE Nome_da_Coluna tipo_de_dados;

ALTER TABLE Minha_Tabela
CHANGE Coluna_2 Coluna_2 year;
```

- Alterar o nome da tabela:

```
rename table minha_tabela to novo_nome;
```

BANCOS DE DADOS

- **1.2 - Linguagem de Definição de Dados (DDL):**

COMANDO DROP:

- Remover uma tabela:

```
DROP TABLE Nome_Da_Tabela;

DROP TABLE Minha_Tabela;
```

- Remover uma base de dados:

```
DROP DATABASE Nome_da_Base_de_Dados;

DROP DATABASE Minha_Base_de_Dados;
```

BANCOS DE DADOS

- ## 1.3 - Operadores Aritméticos:

OPERADOR	DESCRIÇÃO
+	Adição
-	Subtração
*	Multiplicação
/	Divisão

- ## 1.3 - Operadores Relacionais:

OPERADOR	DESCRIÇÃO
=	Igual
<> , !=	Diferente
<=	Menor ou igual
<	Menor
>=	Maior ou igual
>	Maior
<=>	Igual a nulo

BANCOS DE DADOS

1.3 - Operadores Relacionais:

OPERADOR	DESCRIÇÃO
ISNULL	Retorna 1 para nulo ou 0 se não for nulo.
IN	Retorna `1` se é qualquer dos valores na lista `IN`, senão retorna `0`.
`COALESCE(lista)`	Retorna o primeiro elemento não **`NULL`** na lista:
`INTERVAL(N,N1,N2,N3,...)`	Retorna **`0`** se **`N`** < **`N1`**, **`1`** se **`N`** < **`N2`** e assim por diante ou **`-1`** se **`N`** é **`NULL`**.
`BETWEEN min AND max`	Se é maior que ou igual a **`min`** e é menor que ou igual a **`max`**, **`BETWEEN`** retorna **`1`**, senão é retornado **`0`**.

BANCOS DE DADOS

1.3 - Operadores Lógicos:

OPERADOR	DESCRIÇÃO
NOT, !	NOT logico. Avalia como **1** se o operador é **0**, como **0** se o operador é diferente de zero, e **NOT NULL** retorna **NULL**.
AND, &&	AND lógico. Avalia como **1** se todos os operandos são diferentes de zero e não é **NULL**, como **0** se um ou mais operandos são **0**, senão retorna **NULL**.
OR, ||	OR lógico. Avalia como **1** se algum operando é diferente de zero e como **NULL** se algum operando for **NULL**, senão **0** é retornado.
XOR	XOR lógico. Retorna **NULL** se o operando também é **NULL**. Para operandos não **NULL**, avalia como **1** se um número ímpar de operandos é diferente de zero, senão **0** é retornado.

BANCOS DE DADOS

- **1.4 - Tipos de dados e Atributos:**

É a definição dos tipos de dados que podem conter um campo, podem-se agrupar em três grandes grupos:

- **Tipos Numéricos**
- **Tipos de Data/Hora**
- **Tipos de Caracteres**

BANCOS DE DADOS

- **Tipos de Numéricos**

TIPO	INTERVALO	DESCRIÇÃO
TINYINT[(M)]	-127 a 128; ou 0 a 25	inteiros muitos pequenos
BIT		o mesmo que TINYINT
BOOL		o mesmo que TINYINT
SMALLINT[(M)]	-32768 a 32767	inteiros pequenos
MEDIUMINT[(M)]	-8388608 a 8388607; ou 0 a 16777215	inteiros de tamanho médio
INT[(M)]	-213 a 231-1; ou 0 a 232-1	inteiros regulares
INTEGER[(M)]		o mesmo que INT
BIGINT[(M)]	-2^{63} a 2^{63}-1; ou 0 a 2^{64}-1	inteiros grandes
FLOAT[(M,D)]	1.175494351E-38 a ±3.402823466E+38	números de ponto flutuante de precisão simples. O mesmo que FLOAT(4)
DOUBLE[(M,D)]	±1.7976931348623157E+308 a ±2.2250738585072014E-308	números de ponto flutuante de precisão dupla. O mesmo que FLOAT(8)
DOUBLE		O mesmo que DOUBLE[(M,D)]
PRECISION[(M,D)]		O mesmo que DOUBLE[(M,D)]
REAL[(M,D)]		O mesmo que DOUBLE[(M,D)]
DECIMAL[(M,D)]	variável	número de ponto flutuante armazenado como `char`
NUMERIC[(M,D)]		O mesmo que DECIMAL
DEC[(M,D)]		O mesmo que DECIMAL

BANCOS DE DADOS

- **Tipos de Data/Hora:**

TIPO	INTERVALO	FORMATO
DATE	1000-01-01 a 9999-12-31	YYYY-MM-DD
TIME	-838:59:59 a 838:59:59	HH:MM:SS
DATETIME	1000-01-01 00:00:00 a 9999-12-31 23:59:59	YYYY-MM-DD HH:MM:SS
TIMESTAMP[(M)]	1970-01-01 00:00:00 a algum momento em 2037.	TIMESTAMP YYYYMMDDHHMMSS TIMESTAMP(14)YYYYMMDDHHMMSS TIMESTAMP(12) YYMMDDHHMMSS TIMESTAMP(10) YYMMDDHHMM TIMESTAMP(8) YYYYMMDD TIMESTAMP(6) YYMMDD TIMESTAMP(4) YYMM TIMESTAMP(2) YY
YEAR[(2)]	70 a 69 (1970 a 2069)	YY
YEAR[(4)]	1901 a 2155	YYYY

BANCOS DE DADOS

- **Tipos de Caractere:**

TIPO	INTERVALO	DESCRIÇÃO
CHAR	1	Apenas 1 caracter
VARCHAR	1 a 255	Até 255 caracteres
TINYBLOB	0 a 2^8 - 1 (255)	BLOB pequeno
TINYTEXT	0 a 2^8 - 1 (255)	TEXT pequeno
BLOB	0 a 2^{16} - 1 (65535)	BLOB normal
TEXT	0 a 2^{16} - 1 (65535)	TEXT normal
MEDIUMBLOB	0 a 2^{24} - 1 (16777215)	BLOB médio
MEDIUMTEXT	0 a 2^{24} - 1 (16777215)	TEXT médio
LONGBLOB	0 a 2^{32} - 1 (4294967295)	BLOB longo
LONGTEXT	0 a 2^{32} - 1 (4294967295)	TEXT longo
ENUM('valor1','valor2',...)	0 a 65535	armazenam um dos valores listados ou NULL
SET('valor1','valor2',...)	0 a 64	armazenam um ou mais dos valores listados ou NULL

BANCOS DE DADOS

- **1.4 - Tipos de dados e Atributos:**

- Constraints:

São restrições estabelecidas ao tipo de dados que será salvo em determinada coluna de uma tabela com a finalidade de melhor a a integridade dos dados.

BANCOS DE DADOS

- ## 1.4 - Tipos de dados e Atributos:

- Constraints:

CONSTRAINT	DESCRIÇÃO
NOT NULL	A coluna não pode conter valores nulos.
UNIQUE	A coluna não pode conter valores repetidos.
PRIMARY KEY	Os registros da coluna são chaves primárias.
FOREIGN KEY	Os registros da coluna são chaves estrangeiras.
CHECK	Os registros da coluna devem ter uma condição.
DEFAULT	Os registros da coluna devem ter um valor padrão.
AUTO INCREMENT	Incrementa +1 automático no valor.
UNSIGNED	Esse atributo é usado para permitir somente valores positivos em uma coluna.
ZEROFILL	Preenche espaços vazios da coluna com o número zero.

BANCOS DE DADOS

- **1.4 - Tipos de dados e Atributos:**

- Adicionar atributos:

```
alter table novo nome
modify colunax year not null;

alter table novo_nome
modify coluna1 int(5) unique;
```

BANCOS DE DADOS

- **1.4 - Tipos de dados e Atributos:**

- Adicionar restrições(PK e FK):

```
alter table novo_nome
add constraint pk primary key(coluna1);

alter table outra_tabela
add constraint pk primary key(colunaA);

alter table outra_tabela
add constraint fk foreign key(colunaB)
references novo_nome(coluna1);
```

BANCOS DE DADOS

- **1.4 - Tipos de dados e Atributos:**

- Adicionar restrições (condição e valor padrão):

```
alter table novo_nome
add constraint condicao check(colunax>90);

ALTER TABLE novo_nome
MODIFY COLUMN colunax year(2) DEFAULT 99;
```