INSTALAÇÃO

Donwload versão free (*community edition*): https://dev.mysql.com/downloads/
No caso do curso, será utilizado a versão “MySQL on Windows (Installer & Tools)

INTRODUÇÃO

*O MySQL náo é *case sensitive.*

campos: Colunas;
registros: Linhas;
chave primária: Mostra que só é possível ter um campo daquele tipo na tabela;
chave estrangeira: Mostra que só é possível inserir um dado do tipo da chave em outra tabela caso o dado exista na tabela que contenha a chave;
esquemas: Grupamento de tabelas de um tipo parecido;
view: Tabela virtual, composta pelo resultado de uma consulta a outras tabelas ou views, não sendo uma tabela física mas sim existindo dinamicamente;
procedures: Rotinas definidas por um nome que podem ser chamadas, podendo receber parâmetros, instruções e até retornar valores;
funções: Pode retornar um valor ao ser passado parâmetros para ela;
trigger: Aviso caso algo aconteça no banco de dados ou tabela, como “foi adicionado um novo campo”.

CRIANDO UM BANCO DE DADOS

-Database e schema são sinonimos, então pode-se escolher:

> *CREATE {DATABASE} [IF NOT EXISTS] db_name*
> [create_specification]...
ou
> *CREATE {SCHEMA} [IF NOT EXISTS] db_name*
> [create_specification]...

[create_especification]**:**
[DEFAULT] CHARACTER SET [=] charset_name //ex. UTF8
| [DEFAULT] COLLATE [=] collation_name //padrão do conj. de caracteres
| DEFAULT ENCRYPTION [=] {‘Y’ | ‘N’}

APAGAR O BANCO DE DADOS

> DROP {DATABASE | SCHEMA} [IF EXISTS] db_name

SELECIONAR QUAL DATABASE USAR

> *use db_name*
Ou por exemplo, para criar a tabela “tbproduto” no banco de dados “sucos”:

> CREATE TABLE `sucos`.`tbproduto`

TIPOS DE DADOS

NUMÉRICOS:

Inteiros:

Tipo	Valor em Bytes	Menor Valor (Com Sinal) - Signed	Menor Valor (Sem snal) - Unsigned	Maior Valor (Com sinal) - Signed	Maor valor (Sem sinal) - Unsigned
TINYINT	1	-128	0	127	255
SMALLINT	2	-32768	0	32767	65535
MEDIUMINT	3	-8388608	0	8388607	16777215
INT	4	-2147483648	0	2147483647	4294967295
BIGINT	8	-2xE63	0	2xE63-1	2xE64 - 1

Propriedade UNSIGNED:

Não permite sinal no número. Por isso o conjunto de valores válidos aumentam.

Decimais:

FLOAT - Precisão simples de 4 bytes
ex. Um número com 7 digitos e 4 casas decimais: FLOAT(7,4) //321,1234
DOUBLE -Precisão dupla de 8 bytes

Fixos:

DECIMAL - Até 65 digitos
ex. Um número com 10 digitos e 2 casas decimais: DECIMAL(10,2)
NUMERIC - Até 65 digitos

Único:

BIT - Até 64 Bits
ex. BIT(1) - Pode ser 1 ou 0
BIT(2) - Pode ser 01 ou 10 ou 00 ou 11

ATRIBUTOS DOS CAMPOS NÚMERICOS:

SIGNED ou UNSIGNED - Vai possuir ou não sinal do número;
ZEROFILL - Preenche com Zeros os espaços;
AUTO_INCREMENT - Sequencia auto incrementada;

Data e hora:

DATE - 1000-01-01 até 9999-12-31
DATETIME - 1001-01-01 00:00:00 até (...)
TIMESTAMP - 1970-01-01 00:00:01 UTC até (...)
TIME - -838:59:59 até 839:59:59
YEAR - 1901- 2155 (de 2 ou 4 dígitos)

TIPOS STRINGS:

CHAR - Cadeira de caracteres com valor fixo (de 0 a 255).
ex. CHAR(4) - “aa” - “ aa” //note os espaços vazios
VARCHAR - Cadeira de caracteres com valor variado (de 0 a 255)
ex. VARCHAR94) - “aa” - “aa”

strings binário:

BINARY - Cadeira de caracteres com valor fixo (de 0 a 255). Expressos em binário.
VARBINARY - Cadeira de caracteres com valor variado(de 0 a 255). Em binário.
BLOB - binários longo, dividido em: //ex. de uso: gravar os bytes de uma foto
TINYBLOB
BLOB
MEDIUMBLOB
LONGBLOG
TEXT - Texto longo, dividido em:
TINYTEXT
TEXT
MEDIUMTEXT
LONGTEXT
ENUM - Permite armazenar uma lista pré-definida devalores.
ex. Size ENUM (‘x-small’,’small’, ‘medium’, ‘large’, ‘x-large’), que só permitirá que se armazene dados do tipo estabelecido, como ‘large’.
SET e COLLATE - “Que tipo de conjunto de caracteres serão suportados”.
SPACIAL:
GEOMETRY
POINT
LINESTRING
POLYGON

CRIANDO UMA TABELA

Cadastro de Clientes

- CPF do cliente
- O nome completo
- Endereço (Rua, bairro, cidade, estado e CEP)
- Data de nascimento
- A idade
- O sexo
- O limite de crédito para ele comprar produtos
- O volume mínimo de sucos que ele pode comprar
- Se ele já realizou a primeira compra

```
> CREATE TABLE tbCliente
> (CPF VARCHAR(11),
> NOME VARCHAR(100),
```

> ENDERECO1 VARCHAR(150),
> ENDERECO2 VARCHAR(150),
> BAIRRO VARCHAR(50),
> CIDADE VARCHAR(50),
> ESTADO VARCHAR(50),
> CEP VARCHAR(8),
> IDADE SMALLINT,
> SEXO VARCHAR(1),
> `LIMITE CREDITO` FLOAT,
> `VOLUME COMPRA` FLOAT,
> PRIMEIRA_COMPRA BIT(1))

APAGANDO TABELAS:

> DROP TABLE tabela_name;

INSERINDO DADOS NA TABELA

> INSERT INTO nome_tabela (
> nomes_das_colunas, ex_coluna_2) VALUES (valores_que_quero inserir, ex_valor_2)

ALTERANDO REGISTROS

ex. trocar a embalagem de ‘pet’ para ‘lata’ e o preço de 3,20 para 2,50:

> UPDATE nome_tabela SET embalagem = ‘Lata’, preco_lista = 2.46
> WHERE produto = ‘544931’;

EXCLUINDO REGISTROS

> DELETE FROM tabela_nome WHERE produto = ‘1078680’

INCLUINDO A CHAVE PRIMÁRIA

> ALTER TABLE nome_tabela ADD PRIMARY KEY (nome_coluna);

MANIPULANDO DATAS E CAMPOS LÓGICOS

> ALTER TABLE nome_tabela ADD PRIMARY KEY (CPF);
> ALTER TABLE nome_tabela ADD COLUMN (data_nasc DATE);
> INSERT INTO nome_tabela (
> CPF, nome, end1, end2, bairro, cidade, estado, CEP, idade, sexo, limite_credito,
> volume_compra, primeira_compra, data_nasc)
> VALUES (‘03033990’, ‘João’,’Rua x’, ‘ ‘, ‘Vila Y’, ‘Cidade Z’, ‘Estado W’, ‘1111111’, ‘20’, ‘M’,
> 123000.00,1000, 0, ‘1989-10-24’);

SELECIONANDO DADOS DA TABELA

> SELECT CPF, nome, end1
> FROM nome_tabela;

> SELECT CPF AS cpf_cliente *//renomeia um campo para selecioná-lo*

FILTRANDO REGISTROS

> SELECT * FROM clientes WHERE cidade = ‘Rio de Janeiro’;

> UPDATE nome_tabela SET SABOR = ’Cítrico’ WHERE sabor = ‘Limão’

> SELECT * FROM tabela_nome WHERE idade > 22; *//o símbolo de diferente é <>*

> SELECT * FROM tabela_nome WHERE NOME> ‘F’; *//lista os nomes com letras depois de F*

*Não é possível buscar por números float exatos, nesse caso, seria mais interessante usar o tipo DECIMAL para procurar por ex, 10.008, ou então, utilizar a seguinte estratégia:
>> SELECT * FROM tabela_nome WHERE preco BETWEEN 16.007 AND 16.009;

FILTRANDO DATAS

> SELECT * FROM tabela_nome WHERE data_nasc <= ‘1995-01-13’; *//quem nasceu antes ou nessa data*

> > SELECT * FROM tabela_nome WHERE YEAR(data_nasc) = 1995; //também pode-se usar MONTH

FILTROS COMPOSTOS

> SELECT * FROM tabela_nome WHERE (idade > 22 AND idade <32) OR sexo = ‘M’;