

MÓDULO

BASE DE DADOS MYSQL

UNIDADE

CRIAÇÃO DE BASES DE DADOS EM MYSQL

ÍNDICE

OBJETIVOS.....	3
INTRODUÇÃO.....	4
1. CRIAÇÃO DE BASE DE DADOS POR CÓDIGO	5
2. CRIAÇÃO DE TABELAS	10
2.1. INTRODUÇÃO	10
2.2. TIPOS DE DADOS	11
2.3. RELACIONAMENTOS, CHAVES PRIMÁRIAS E ESTRANGEIRAS.....	18
2.4. ÍNDICE	19
2.5. MOTORES DE ARMAZENAMENTO	21
2.6. AS TABELAS DA BASE DE DADOS DO FUTEBOL.....	23
CONCLUSÃO	29
AUTOAVALIAÇÃO	31
SOLUÇÕES	35
PROPOSTAS DE DESENVOLVIMENTO DO ESTUDO.....	36
BIBLIOGRAFIA	37

OBJETIVOS

Com esta unidade didática, pretende-se que desenvolva os seguintes objetivos de aprendizagem:

- Criar uma base de dados.
- Criar tabelas para as bases de dados.
- Trabalhar com o tipo de dados apropriado para cada coluna de uma tabela.
- Estabelecer relacionamentos entre as diferentes tabelas na base de dados.
- Estabelecer restrições nas tabelas da base de dados.

INTRODUÇÃO

Se lhe perguntarem o que é que considera mais importante numa biblioteca, se os livros ou as prateleiras onde estão arrumados, provavelmente responderá que os livros (as informações) são mais importantes. Mas acontece que, sem as estantes (as tabelas), não é possível guardar nem recuperar os livros da biblioteca (a base de dados). Como pode ver, todas as peças têm a sua importância, por isso, esta unidade será dedicada às bases de dados e às tabelas.

Ao longo da unidade aprenderá a criar bases de dados e tabelas, tanto na linha de comandos da Shell do XAMPP como no editor gráfico MySQL Workbench. Também serão abordados os tipos de dados e mecanismos de base de dados que pode usar, bem como as restrições de integridade que pode definir.

Depois de concluir esta unidade poderá criar as suas próprias bases de dados e tabelas. Será capaz de criar a estrutura para inserir os dados relativos ao que está a trabalhar em cada caso, quer seja a gestão de uma biblioteca, de uma loja ou qualquer outro projeto.

Nesta unidade continuará a trabalhar com a linha de comandos da Shell do XAMPP, por isso é essencial que tenha os conceitos da unidade anterior bem consolidados.

1. CRIAÇÃO DE BASE DE DADOS POR CÓDIGO

Irá criar uma base de dados cujo tema principal é o futebol e que será utilizada também nas próximas unidades. Esta base de dados será chamada futebol.

1. Faça login no MySQL com privilégios de administrador. Como viu na unidade anterior, terá de escrever:

```
mysql -u root -p
```

2. A seguir, contando com a ajuda fornecida pelo MySQL, execute `help contents`.

```
help contents;
```

3. Na lista de tópicos da ajuda, escolha aquele que é dedicado à definição de dados e escreva `help Data Definition`;. Aparecerá uma longa lista de tópicos de ajuda sobre comandos de criação, exclusão e modificação.

```
help Data Definition;
```

```

XAMPP for Windows - mysql -u root -p
MariaDB [(none)]> help
You asked for help about help category: "Data Definition"
For more information, type 'help <item>', where <item> is one of the following
topics:
  ALTER DATABASE
  ALTER EVENT
  ALTER FUNCTION
  ALTER LOGFILE GROUP
  ALTER PROCEDURE
  ALTER SEQUENCE
  ALTER SERVER
  ALTER TABLE
  ALTER TABLESPACE
  ALTER VIEW
  CONSTRAINT
  CREATE DATABASE
  CREATE EVENT
  CREATE FUNCTION
  CREATE INDEX
  CREATE PACKAGE
  CREATE PACKAGE BODY
  CREATE PROCEDURE
  CREATE SEQUENCE
  CREATE SERVER
  CREATE TABLE
  CREATE TABLESPACE
  CREATE TRIGGER
  CREATE VIEW
  DROP DATABASE
  DROP EVENT
  DROP FUNCTION
  DROP INDEX
  DROP PACKAGE
  DROP PACKAGE BODY
  DROP PROCEDURE
  DROP SEQUENCE
  DROP SERVER
  DROP TABLE
  DROP TABLESPACE
  DROP TRIGGER
  DROP VIEW
  
```

4. Por fim, escolha o tema dedicado à criação de bases de dados. Para isso, escreva e execute `help CREATE DATABASE;`, obtendo o resultado da imagem que se segue.

`help CREATE DATABASE;`

```

XAMPP for Windows - mysql -u root -p
TRUNCATE TABLE
MariaDB [(none)]> help CREATE DATABASE;
Name: 'CREATE DATABASE'
Description:
Syntax
-----
CREATE [OR REPLACE] {DATABASE | SCHEMA} [IF NOT EXISTS]
db_name
[create_specification] ...

create_specification:
[DEFAULT] CHARACTER SET [=] charset_name
| [DEFAULT] COLLATE [=] collation_name

Description
-----
CREATE DATABASE creates a database with the given name. To
use this statement, you need the CREATE privilege for the
database. CREATE SCHEMA is a synonym for CREATE DATABASE.

For valid identifiers to use as database names, see
Identifier Names.

OR REPLACE
The OR REPLACE clause was added in MariaDB 10.1.3

If the optional OR REPLACE clause is used, it acts as a
shortcut for:

DROP DATABASE IF EXISTS db_name;
CREATE DATABASE db_name ...;

IF NOT EXISTS

When the IF NOT EXISTS clause is used, MariaDB will return a
warning instead of an error if the specified database
already exists.
  
```


5. A partir da ajuda obtida, observe que para o MySQL as palavras DATABASE e SCHEMA são sinónimos. A base de dados só pode ser criada se não houver outra com o mesmo nome, e também há uma série de parâmetros opcionais que pode fornecer, como o conjunto de caracteres a usar. Por enquanto, não se preocupe com estes parâmetros opcionais, pois ao instalar o servidor foram selecionados os valores padrão, que são perfeitamente válidos.

Para implementar o comando solicitado à ajuda, pode escrever CREATE DATABASE futebol; ou CREATE SCHEMA futebol;, e a base de dados com o nome escolhido será criada.

```
URL: https://mariadb.com/kb/en/create-database/  
MariaDB [(none)]> CREATE DATABASE futebol;  
Query OK, 1 row affected (0.001 sec)
```

Verifique se a base de dados foi criada conforme esperado, através do comando show databases.

```
MariaDB [(none)]> SHOW DATABASES;  
+-----+  
| Database  
+-----+  
| escola  
| futebol  
| information_schema  
| mysql  
| performance_schema  
| phpmyadmin  
| test  
+-----+  
7 rows in set (0.001 sec)  
  
MariaDB [(none)]>
```

Verifique o que acontece se tentar criar uma base de dados com um nome que já existe antes – execute a instrução:

```
CREATE SCHEMA futebol;
```

Como seria de esperar o resultado é um erro, pois não pode haver duas bases de dados com o mesmo nome.

```
C:\> XAMPP for Windows - mysql -u root -p
MariaDB [(none)]> CREATE SCHEMA futebol;
ERROR 1007 (HY000): Can't create database 'futebol'; database exists
MariaDB [(none)]>
```

Depois, tente com a instrução seguinte:

```
CREATE SCHEMA IF NOT EXISTS futebol;
```


Observe que o resultado desta vez não deu nenhum erro, pois foi indicado que a base de dados só seria criada caso não existisse outra com o mesmo nome.

```
C:\> XAMPP for Windows - mysql -u root -p
+-----+
7 rows in set (0.001 sec)

MariaDB [(none)]> CREATE SCHEMA futebol;
ERROR 1007 (HY000): Can't create database 'futebol'; database exists
MariaDB [(none)]> CREATE SCHEMA IF NOT EXISTS futebol;
Query OK, 0 rows affected, 1 warning (0.001 sec)

MariaDB [(none)]>
```

Pode utilizar esta base de dados através do comando **USE futebol;** e verificar que tabelas possui através do **show tables;** Como pode ver na imagem seguinte, a base de dados existe, mas está completamente vazia, não possui tabelas nem dados. Ao longo desta unidade irá preencher essa lacuna na base de dados dedicada ao futebol.

 XAMPP for Windows - mysql -u root -p

```
MariaDB [futebol]> USE futebol;  
Database changed  
MariaDB [futebol]> SHOW TABLES;  
Empty set (0.001 sec)  
  
MariaDB [futebol]> _
```

2. CRIAÇÃO DE TABELAS

2.1. INTRODUÇÃO

Uma base de dados sem tabelas é como uma árvore sem frutos; simplesmente, não faria sentido. Não faria sentido, porque a única maneira de armazenar informações numa base de dados é fazê-lo em células de tabelas. Portanto, uma base de dados sem tabelas não pode armazenar dados, e para que serve uma base de dados sem informações?

As células são definidas por uma linha e uma coluna: a coluna corresponde a um campo da tabela e a linha corresponde ao registo correspondente. Segue-se um exemplo para clarificar melhor:

Nome	Apelido	Ano de nascimento
Daniel	Silveira	1936
Helena	Santos	1941
Marcelo	Martins	1940
Manuel	Silva	1938
Carlos	Lopes	1938

A primeira linha da tabela corresponde aos nomes das colunas ou campos comuns a todos os registos da tabela. Cada nova linha com os dados de cada uma das pessoas é um registo. O cruzamento de linhas e colunas produz células, neste caso como são cinco registos e três campos, então trata-se de $(5 \times 3 =)$ 15 células.

2.2. TIPOS DE DADOS

Conforme viu na tabela anterior, existem vários tipos de dados, como o ano de nascimento, que são números que correspondem a uma data, ou nomes e apelidos, que são dados alfanuméricos. Em termos gerais, existem os seguintes tipos de dados padrão em qualquer base de dados: strings, números inteiros, floats e datas. A implementação e os nomes de cada servidor de base de dados fazem com que sejam diferentes.

Para saber mais sobre os tipos de dados que pode utilizar com o MySQL, pode executar **help Data Types**; na linha de comandos. Desta forma será mostrada uma lista, por ordem alfabética, com os tipos de dados com os quais poderá trabalhar a partir do MySQL. Também poderá escrever **help** e o nome de algum deles para obter mais informações sobre o mesmo:

- AUTO_INCREMENT.
- BIGINT.
- BINARY.
- BIT.
- BLOB.
- BLOB DATA TYPE.
- BOOLEAN.
- CHAR.
- CHAR BYTE.
- DATE.
- DATETIME.
- DEC.

- DECIMAL.
- DOUBLE.
- DOUBLE PRECISION.
- ENUM.
- FLOAT.
- INT.
- INTEGER.
- LONGBLOB.
- LONGTEXT.
- MEDIUMBLOB.
- MEDIUMINT.
- MEDIUMTEXT.
- SET DATA TYPE.
- SMALLINT.
- TEXT.
- TIME.
- TIMESTAMP.
- TINYBLOB.
- TINYINT.
- TINYTEXT.
- VARBINARY.
- VARCHAR.
- YEAR DATA TYPE.

Outra forma de obter estas informações, mas de forma mais ordenada, é visitar a seguinte página web:

- <http://dev.mysql.com/doc/refman/5.1/en/data-types.html>

Nesta página, os tipos de dados estão separados pelos dados que armazenam, sejam números, datas ou textos. Na tabela a seguir, por exemplo, poderá ver os tipos de dados para armazenamento de números inteiros e as suas principais características.

Tipo	Bytes	Alcance com sinal	Alcance sem sinal
TINYINT	1	-128 a 127	De 0 a 255
SMALLINT	2	-32.768 a 32.767	De 0 a 65.535
MEDIUMINT	3	-8.388.608 a 8.388.607	De 0 a 16.777.215
INT	4	-2.147.483.648 a 2.147.483.647	De 0 a 4.294.967.295
BIGINT	8	-9.223.372.036.854.775.808 a 9.223.372.036.854.775.807	De 0 a 18.446.744.073.709.551.615

Para trabalhos com números reais que requeiram o uso de decimais, podem ser utilizados os tipos FLOAT, DOUBLE, DECIMAL e NUMERIC, que é utilizado como sinónimo de DECIMAL. Os tipos FLOAT e DOUBLE representam números decimais aproximados, mas ocupam menos espaço no disco; os tipos DECIMAL e NUMERIC permitem armazenar valores exatos, mas ocupam mais espaço. A diferença entre os tipos FLOAT e DOUBLE é que o DOUBLE permite armazenar mais valores, mas também ocupa mais espaço. Todos eles são capazes de lidar com o valor 0. Nas imagens a seguir pode ver as características de cada tipo e como defini-las corretamente.

```
Selecionar Linha de comandos - mysql

Name: 'FLOAT'
Description:
FLOAT[(M,D)] [UNSIGNED] [ZEROFILL]

A small (single-precision) floating-point number. Permissible values
are -3.402823466E+38 to -1.175494351E-38, 0, and 1.175494351E-38 to
3.402823466E+38. These are the theoretical limits, based on the IEEE
standard. The actual range might be slightly smaller depending on your
hardware or operating system.

M is the total number of digits and D is the number of digits following
the decimal point. If M and D are omitted, values are stored to the
limits permitted by the hardware. A single-precision floating-point
number is accurate to approximately 7 decimal places.

UNSIGNED, if specified, disallows negative values.

Using FLOAT might give you some unexpected problems because all
calculations in MySQL are done with double precision. See
http://dev.mysql.com/doc/refman/5.7/en/no-matching-rows.html.

URL: http://dev.mysql.com/doc/refman/5.7/en/numeric-type-overview.html
```

```
Linha de comandos - mysql -h 127.0.0.1 -P 8889 -u root -proot

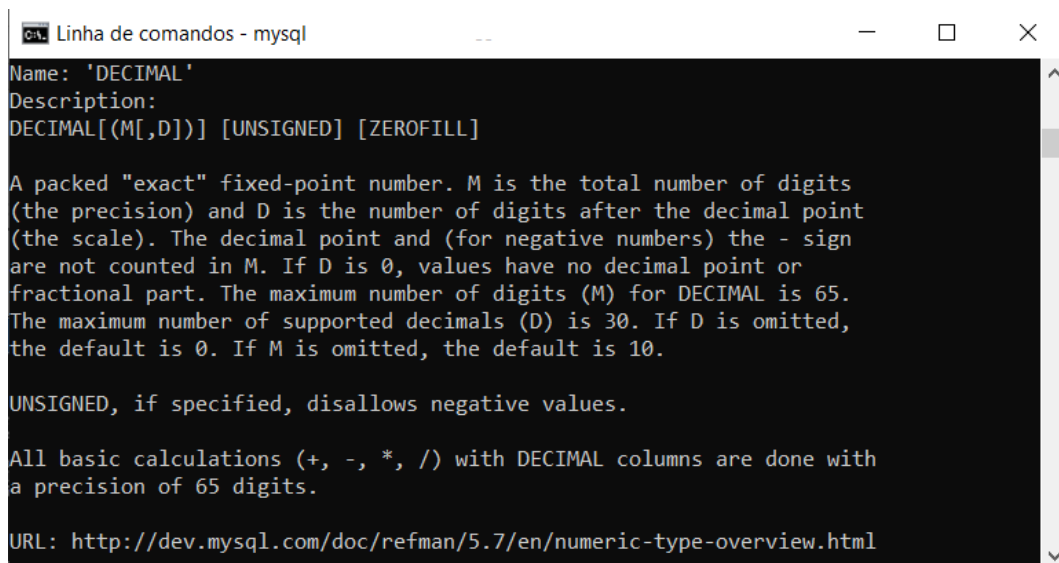
Name: 'DOUBLE'
Description:
DOUBLE[(M,D)] [UNSIGNED] [ZEROFILL]

A normal-size (double-precision) floating-point number. Permissible
values are -1.7976931348623157E+308 to -2.2250738585072014E-308, 0, and
2.2250738585072014E-308 to 1.7976931348623157E+308. These are the
theoretical limits, based on the IEEE standard. The actual range might
be slightly smaller depending on your hardware or operating system.

M is the total number of digits and D is the number of digits following
the decimal point. If M and D are omitted, values are stored to the
limits permitted by the hardware. A double-precision floating-point
number is accurate to approximately 15 decimal places.

UNSIGNED, if specified, disallows negative values.

URL: http://dev.mysql.com/doc/refman/5.7/en/numeric-type-overview.html
```

Como pode observar nas imagens anteriores, as palavras UNSIGNED e ZEROFILL aparecem como opcionais na definição dos dados deste tipo. A palavra UNSIGNED permite indicar que apenas valores positivos serão usados, permitindo então, no caso de inteiros, obter o dobro dos valores naturais possíveis em troca de sacrificar o sinal negativo.

Utilizar o UNSIGNED com números reais não fornece a capacidade de lidar com mais valores positivos, apenas indica que números negativos não podem ser usados. ZEROFILL permite que todos os dados de uma mesma coluna de números tenham o mesmo comprimento, preenchendo zeros à esquerda; neste caso, o mesmo resultado é dado tanto em números inteiros quanto em números reais. Por exemplo, uma coluna de valores inteiros de 3 posições: o valor 8 seria 008, o 23 seria 023, etc.

No caso de datas e horários, são várias as opções disponíveis. É possível que se recorde de algo que ficou conhecido como o "efeito 2000", um problema que ocorria no tratamento de datas em alguns sistemas, em que, para economizar espaço, os anos eram armazenados com 2 dígitos em vez de 4. A tabela a seguir mostra os principais tipos de dados que permitem representar a data. Mais informações podem ser encontradas na seguinte página web:

- <http://dev.mysql.com/doc/refman/5.1/en/date-and-time-types.html>

Tipo	Formato	Valores permitidos
DATE	AAAA-MM-DD	Desde "1000-01-01" até "9999-12-31".
DATETIME	AAAA-MM-DD hh:mm:ss	Desde "1000-01-01 00:00:00" até "9999-12-31 23:59:59".
TIMESTAMP	AAAA-MM-DD hh:mm:ss	Desde "1970-01-01 00:00:01" UTC até "2038-01-19 03:14:07" UTC.
YEAR	AAAA	Desde "1901" até "2155".
	AA	Desde "1970" até "2069" utilizando os últimos dois números.
TIME	hh:mm:ss	Desde "-838:59:59" até "838:59:59".

Com tipos de dados referentes às strings de caracteres, os valores mais comuns são CHAR para as strings de comprimento fixo e VARCHAR para as strings de comprimento variável. No caso do CHAR, a limitação do número máximo de caracteres é 255, e no caso do VARCHAR é 65535, embora seja aconselhável consultar a documentação para evitar surpresas com valores elevados. Em geral, os tipos de comprimento variável permitem otimizar o espaço ocupado na base de dados, enquanto os de comprimento fixo permitem melhorar o tempo de acesso aos dados.

A imagem seguinte mostra a documentação oficial sobre o tipo de dados VARCHAR, e na tabela estão os principais tipos de dados que permitem armazenar strings de texto e os comprimentos máximos de texto que estes toleram.

```

C:\> Linha de comandos - mysql
Name: 'VARCHAR'
Description:
[NATIONAL] VARCHAR(M) [CHARACTER SET charset_name] [COLLATE
collation_name]

A variable-length string. M represents the maximum column length in
characters. The range of M is 0 to 65,535. The effective maximum length
of a VARCHAR is subject to the maximum row size (65,535 bytes, which is
shared among all columns) and the character set used. For example, utf8
characters can require up to three bytes per character, so a VARCHAR
column that uses the utf8 character set can be declared to be a maximum
of 21,844 characters. See
http://dev.mysql.com/doc/refman/5.7/en/column-count-limit.html.

MySQL stores VARCHAR values as a 1-byte or 2-byte length prefix plus
data. The length prefix indicates the number of bytes in the value. A
VARCHAR column uses one length byte if values require no more than 255
bytes, two length bytes if values may require more than 255 bytes.

*Note*:

MySQL follows the standard SQL specification, and does not remove
trailing spaces from VARCHAR values.

VARCHAR is shorthand for CHARACTER VARYING. NATIONAL VARCHAR is the
standard SQL way to define that a VARCHAR column should use some
predefined character set. MySQL uses utf8 as this predefined character
set. http://dev.mysql.com/doc/refman/5.7/en/charset-national.html.
NVARCHAR is shorthand for NATIONAL VARCHAR.

URL: http://dev.mysql.com/doc/refman/5.7/en/string-type-overview.html

```

Tipo	Número máximo de caracteres
TINYTEXT	255
TEXT	65.535
MEDIUMTEXT	16.777.215
LONGTEXT	4.294.967.295

Informações sobre outros tipos de dados, como BINARY, VARBINARY, BLOB, ENUM e SET estão incluídas na página web oficial:

- <http://dev.mysql.com/doc/refman/5.1/en/string-types.html>

A documentação considera-os como strings, mas, na verdade, não são estritamente strings. Por esta razão, não lidará com este tipo de dados nesta unidade, embora seja aconselhado que leia sobre eles na documentação oficial do MYSQL.

2.3. RELACIONAMENTOS, CHAVES PRIMÁRIAS E ESTRANGEIRAS

As bases de dados relacionais ligam os valores de algumas tabelas com os valores de outras. No caso de uma base de dados de futebol poderia encontrar várias relações entre as tabelas, por exemplo, entre a tabela da equipa e a tabela dos jogadores. Um jogador é sempre um funcionário de uma equipa; isso implica um relacionamento entre os dois na vida real e, portanto, um relacionamento também na base de dados. Aqui estão mais alguns exemplos de relacionamentos que podem ocorrer:

- Os jogadores inscritos numa equipa durante a temporada.
- As equipas que jogam durante a temporada.
- Os árbitros que arbitram as partidas disputadas pelos jogadores da equipa nos estádios durante a temporada.

Como pode ver, a vida real oferece vários relacionamentos entre as diferentes entidades existentes. Árbitros, jogadores, jogos, equipas, temporadas, todas essas entidades estão relacionadas umas com outras. As entidades da base de dados são convertidas em tabelas. O que se pretende representar é a realidade, e antes de um jogo teria de se listar um árbitro principal, dois jogadores atacantes, quatro árbitros auxiliares, a equipa da casa e a equipa visitante.

Tanto no caso de árbitros como no caso de equipas, teria de ser possível diferenciar de uma forma única os árbitros e as equipas umas das outras. Estas são precisamente as chaves primárias, os valores que permitem que cada elemento de uma determinada tabela seja diferenciado de maneira única. Por exemplo, no caso de uma pessoa, a chave primária numa tabela pode ser o seu número de identificação ou o seu número de contribuinte, ou qualquer outra coisa que o identifique de forma exclusiva e única. As chaves primárias são exclusivas numa tabela, necessárias para cada registo e não podem ser nulas em nenhum momento.

Muitas vezes são usadas chaves formadas artificialmente que podem ser constituídas por um número; esse número aumenta à medida que os registos são inseridos na tabela. Isso deve-se ao facto de poder haver, por exemplo, várias pessoas com o mesmo nome e apelido, mas que obviamente não são a mesma pessoa, e seria necessário distinguir cada pessoa de uma forma única.

Para completar os relacionamentos, o que se faz é incluir as chaves estrangeiras na tabela que está relacionada às outras. Por exemplo, os árbitros e as equipes têm cada um a sua chave primária, e o valor dessa chave primária será usado como uma chave estrangeira nas colunas `arbitroPrinc`, `jogadorEsq`, `jogadorDrt`, `arbitro4`, `equipaCasa` e `equipaVist`. Desta forma, a questão dos relacionamentos será resolvida.

2.4. ÍNDICE

Os índices permitem otimizar a velocidade com que são recuperadas as informações; assim como ao abrir um livro com um índice no início, é possível localizar mais rapidamente as informações de que precisa. Além das chaves primárias e estrangeiras já vistas, que também eram índices, existem índices que permitem restringir uma coluna para que tenha valores únicos através das palavras `UNIQUE INDEX`. E outros que simplesmente permitem otimizar a velocidade de pesquisa.

```

Linha de comandos - mysql
Name: 'CREATE INDEX'
Description:
Syntax:
CREATE [UNIQUE|FULLTEXT|SPATIAL] INDEX index_name
    [index_type]
    ON tbl_name (index_col_name,...)
    [index_option]
    [algorithm_option | lock_option] ...

index_col_name:
    col_name [(length)] [ASC | DESC]

index_option:
    KEY_BLOCK_SIZE [=] value
    | index_type
    | WITH PARSER parser_name
    | COMMENT 'string'

index_type:
    USING {BTREE | HASH}

algorithm_option:
    ALGORITHM [=] {DEFAULT|INPLACE|COPY}

lock_option:
    LOCK [=] {DEFAULT|NONE|SHARED|EXCLUSIVE}

CREATE INDEX is mapped to an ALTER TABLE statement to create indexes.
See [HELP ALTER TABLE]. CREATE INDEX cannot be used to create a PRIMARY
KEY; use ALTER TABLE instead. For more information about indexes, see
http://dev.mysql.com/doc/refman/5.7/en/mysql-indexes.html.

```

Como pode ver na imagem anterior, as ordenações crescente e decrescente podem ser utilizadas para a coluna na qual o índice é aplicado, com a ideia de otimizar as pesquisas. E existem várias outras opções, contudo, para não complicar a questão por enquanto, ficará a conhecer apenas as de valor único. Se pretender saber mais, visite:

- <http://dev.mysql.com/doc/refman/5.1/en/create-index.html>
- <http://dev.mysql.com/doc/refman/5.1/en/mysql-indexes.html>

Ninguém gosta de restrições e proibições, mas se as mesmas não existissem, todos fariam o que quisessem e provavelmente o mundo seria um caos. Para que isso não aconteça com as bases de dados, também existem restrições. As restrições podem ser aplicadas a uma coluna para manter a integridade referencial e a consistência dos dados numa base de dados. São as seguintes:

- PRIMARY KEY: restrição de chave primária.
- FOREIGN KEY: restrição de chave estrangeira.
- UNIQUE: restrição de valor único.
- NOTNULL: restrição à não admissão de valores nulos.

A restrição de chave primária implica que o valor é único em toda a tabela e nunca pode assumir um valor nulo. A restrição de chave estrangeira implica que esta só pode aceitar valores existentes na chave primária da tabela relacionada. A restrição de valor único implica que não pode haver nenhum valor repetido para essa coluna da tabela. A restrição de não admitir valores nulos implica a obrigação de incluir sempre informações específicas na coluna.

Essas restrições ajudam a manter a integridade referencial das bases de dados, forçando a chave primária a reconhecer cada registo de maneira única; forçando, por meio de chaves estrangeiras, o uso apenas dos valores disponíveis na chave primária da tabela relacionada; restringindo o uso de valores únicos para evitar erros; e evitando a existência de muitos valores nulos.

2.5. MOTORES DE ARMAZENAMENTO

Ao contrário de outros programas de gestão de base de dados, o MySQL permite o uso de diferentes mecanismos de armazenamento na mesma base de dados. Mais informações sobre os motores podem ser encontradas na página web <http://dev.mysql.com/doc/refman/5.1/en/storage-engines.html> e na lista a seguir:

- **MyISAM:** o mecanismo de armazenamento padrão, a menos que a configuração inicial tenha sido alterada durante a instalação. Fornece armazenamento e recuperação rápida de dados. MyISAM é uma evolução do ISAM (já descontinuado), que significa Indexed Sequential Access Method (Método de Acesso Sequencial Indexado) e refere-se à maneira como o MySQL acede aos seus dados. As tabelas que utilizam este motor não permitem a utilização de transações.
- **InnoDB:** um mecanismo de armazenamento transacional (compatível com ACID), com recursos para confirmar e cancelar transações. A sigla ACID significa Atomicity, Consistency, Isolation & Durability, o que indica que as operações de uma transação são realizadas de uma só vez ou de nenhuma, garantindo assim a integridade dos dados. Também permite o uso de chaves estrangeiras, algo essencial para manter a integridade da base de dados.
- **MERGE:** este motor permite agrupar tabelas do tipo MyISAM com as características idênticas sob o mesmo nome.
- **MEMORY:** as tabelas deste tipo são armazenadas em RAM, o que beneficia a velocidade. Por outro lado, os dados são perecíveis, pois estão armazenados numa memória volátil, e desaparecem quando o computador é desconectado de onde estava. Este mecanismo era conhecido anteriormente como HEAP.
- **EXAMPLE:** um mecanismo de armazenamento que não permite armazenar ou recuperar dados. O seu objetivo é mostrar aos programadores o código necessário para escrever o seu próprio mecanismo de armazenamento.
- **FEDERATED:** permite o acesso à tabela a partir de servidores remotos, dando a possibilidade de ter uma base de dados lógica de diferentes servidores.

- **ARCHIVE:** permite armazenar grandes quantidades de informações sem usar nenhum índice. Como consequência, é muito rápido a guardar informações e lento a recuperá-las.
- **CSV:** permite guardar dados em ficheiros de texto utilizando o formato Comma Separated Values.
- **BLACKHOLE:** este mecanismo aceita dados, mas não os armazena nem devolve, por isso tem o nome de “buraco negro”. Normalmente, é utilizado com bases de dados distribuídas em vários servidores.
- **NDBCLUSTER:** este mecanismo é usado em ambientes distribuídos e de alta disponibilidade.

Dos motores anteriores, os principais são MyISAM e InnoDB. Deverá sempre utilizar o InnoDB para várias propriedades consideradas importantes e valiosas, como a possibilidade de usar chaves estrangeiras. A tabela seguinte apresenta as ações possíveis de alguns dos principais mecanismos de armazenamento.

Feature	MyISAM	Memory	InnoDB	Archive	NDB
Storage limits	256TB	RAM	64TB	None	384EB
Transactions	No	No	Yes	No	Yes
Locking granularity	Table	Table	Row	Row	Row
MVCC	No	No	Yes	No	No
Geospatial data type support	Yes	No	Yes	Yes	Yes
Geospatial indexing support	Yes	No	No	No	No
B-tree indexes	Yes	Yes	Yes	No	Yes
Hash indexes	No	Yes	No	No	Yes
Full-text search indexes	Yes	No	No	No	No
Clustered indexes	No	No	Yes	No	No
Data caches	No	N/A	Yes	No	Yes
Index caches	Yes	N/A	Yes	No	Yes
Compressed data	Yes ^[a]	No	Yes ^[a]	Yes	No
Encrypted data ^[a]	Yes	Yes	Yes	Yes	Yes
Cluster database support	No	No	No	No	Yes
Replication support ^[a]	Yes	Yes	Yes	Yes	Yes
Foreign key support	No	No	Yes	No	No
Backup / point-in-time recovery ^[a]	Yes	Yes	Yes	Yes	Yes
Query cache support	Yes	Yes	Yes	Yes	Yes
Update statistics for data dictionary	Yes	Yes	Yes	Yes	Yes

Fonte: <http://dev.mysql.com/doc/refman/5.1/en/storage-engines.html>.

Se deseja saber a lista de engines que podem ser utilizados no seu sistema, pode executar a instrução **SHOW ENGINES;** na linha de comandos do MySQL, e aparecerá uma lista indicando o nome do motor, um comentário sobre o mesmo, a disponibilidade ou não de transações e algumas outras características de interesse.

2.6. AS TABELAS DA BASE DE DADOS DO FUTEBOL

Para armazenar os dados na base de dados de futebol é necessário ter algumas tabelas. Podem ser mais do que as apresentadas a seguir, mas, para a elaboração de uma primeira base de dados, estas bastam. Cada tabela da base de dados corresponde a uma entidade existente na vida real, deste modo poderá reconhecê-las.

A seguir são apresentadas algumas das afirmações que levaram a optar por este projeto (certamente conseguirá pensar em mais algumas que permitiriam criar uma base de dados mais completa):

- Um campeonato é disputado durante uma determinada temporada.
- Uma temporada tem um ano inicial e um ano final.
- Um campeonato implica que há uma equipa que ganha o campeonato e outra que é vice-campeã.
- Uma equipa está baseada num determinado país.
- As equipas campeãs e vice-campeãs de um campeonato são as de uma competição específica, numa temporada específica e em datas específicas.

Irá construir as tabelas da base de dados através destas premissas. De momento, as entidades que pode localizar são: equipa, temporada, país, competição e campeonato. E de cada uma delas poderá deduzir uma série de propriedades associadas; por exemplo, uma equipa tem um nome, um presidente, um estádio onde a equipa está instalada, um hino, entre outros.

Em última análise, cada entidade pode ser composta por muitas outras entidades; se esse modelo de realidade é mais ou menos completo, dependerá das necessidades que tiver em cada caso. Modelos muito completos implicam maior complexidade e tempo de desenvolvimento, mas permitem extrair muito mais informações.

Como fez até agora, começará por pesquisar ajuda para a criação de tabelas escrevendo na linha de comandos **help CREATE TABLE;**. Também pode encontrar informações no website <http://dev.mysql.com/doc/refman/5.1/en/create-table.html>.

A seguir, apresenta-se a tabela e o código correspondente para a criar (leia toda esta secção antes de experimentar fazê-la).

Tabela temporada		Motor InnoDB	
Campo	Tipo	Nulo	Observações
temporadaID	INT	NOT NULL	Chave primária autoincremental
anoInicio	YEAR	NOT NULL	Único
anoFim	YEAR	NOT NULL	Único

```
CREATE TABLE temporada (  
  temporadaID INT NOT NULL AUTO_INCREMENT,  
  anoInicio YEAR NOT NULL,  
  anoFim YEAR NOT NULL,  
  PRIMARY KEY (temporadaID),  
  UNIQUE INDEX anoInicio_UNIQUE (anoInicio ASC),  
  UNIQUE INDEX anoFim_UNIQUE (anoFim ASC)  
)  
ENGINE = InnoDB;
```

Tabela competição		Motor InnoDB	
Campo	Tipo	Nulo	Observações
competicaoID	INT	NOT NULL	Chave primária autoincremental
competicaoNome	VARCHAR(50)	NOT NULL	Único

```
CREATE TABLE competicao (
  competicaoID INT NOT NULL AUTO_INCREMENT,
  competicaoNome VARCHAR(50) NOT NULL,
  PRIMARY KEY (competicaoID),
  UNIQUE INDEX competicaoNome_UNIQUE(competicaoNome ASC)
)
ENGINE = InnoDB;
```

Tabela país		Motor InnoDB	
Campo	Tipo	Nulo	Observações
paisID	CHAR(3)	NOT NULL	Chave primária
paisNome	VARCHAR(50)	NOT NULL	Único

```
CREATE TABLE pais (
  paisID CHAR(3) NOT NULL,
  paisNome VARCHAR(50) NOT NULL,
  PRIMARY KEY (paisID),
  UNIQUE INDEX paisNome_UNIQUE (paisNome ASC)
)
ENGINE=InnoDB;
```

Tabela equipa		Motor InnoDB	
Campo	Tipo	Nulo	Observações
equipaID	INT	NOT NULL	Chave primária autoincremental
equipaNome	VARCHAR(50)	NOT NULL	Único
paisID	CHAR(3)	NULL	Chave estrangeira

```
CREATE TABLE equipa (
```

```

equipaID INT NOT NULL AUTO_INCREMENT,
equipaNome VARCHAR(50) NOT NULL,
paisID CHAR(3),
PRIMARY KEY (equipaID),
UNIQUE INDEX equipaNome_UNIQUE (equipaNome ASC),
INDEX equipaPais_FK (paisID ASC),
CONSTRAINT equipaPais_FK
FOREIGN KEY (paisID)
REFERENCES pais (paisID)
ON DELETE SET NULL
ON UPDATE CASCADE
)
ENGINE=InnoDB;

```

Tabela campeonato		Motor InnoDB	
Campo	Tipo	Nulo	Observações
campeonatoID	INT	NOT NULL	Chave primária autoincremental
competicaoID	INT	NOT NULL	Chave estrangeira
temporadaID	INT	NOT NULL	Chave estrangeira
campeao	INT	NOT NULL	Chave estrangeira
subcampeao	INT	NULL	Chave estrangeira
data	DATE	NULL	
cronica	TEXT	NULL	

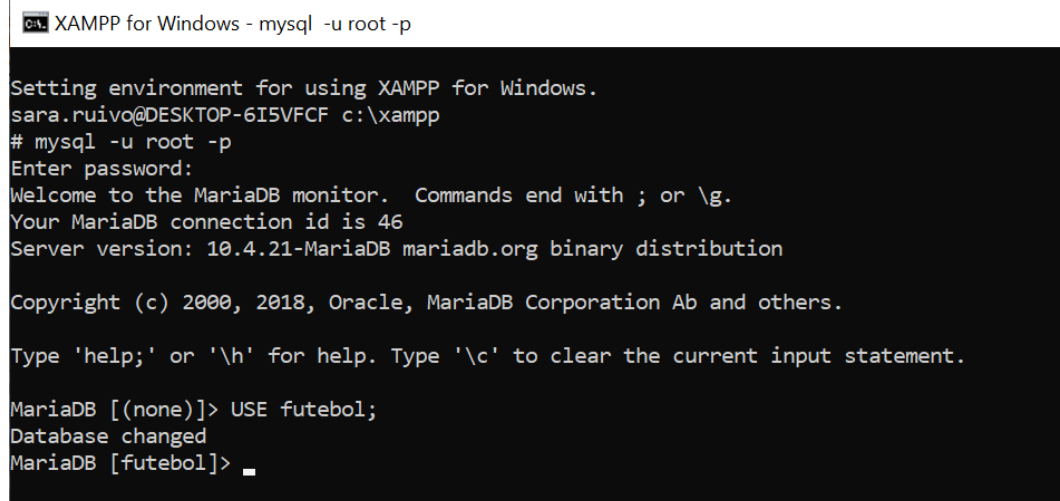
```

CREATE TABLE campeonato (
campeonatoID INT NOT NULL AUTO_INCREMENT,
competicaoID INT NOT NULL,
temporadaID INT NOT NULL,
campeao INT NOT NULL,
subcampeao INT,

```

```
data DATE,  
cronica TEXT,  
PRIMARY KEY (campeonatoID),  
CONSTRAINT competicao_campeonato_FK FOREIGN KEY (competicaoID)  
REFERENCES competicao (competicaoID),  
CONSTRAINT temporada_campeonato_FK FOREIGN KEY (temporadaID)  
REFERENCES temporada (temporadaID),  
CONSTRAINT equipa_campeao_campeonato_FK FOREIGN KEY (campeao)  
REFERENCES equipa (equipaID),  
CONSTRAINT equipa_subcampeao_campeonato_FK FOREIGN KEY (subcampeao)  
REFERENCES equipa (equipaID)  
)  
ENGINE=InnoDB;
```

Para criar estas tabelas, deve fazer login na linha de comandos do MySQL com o utilizador root e, de seguida, indicar a base de dados que irá usar através da instrução **USE futebol;**



```
C:\> XAMPP for Windows - mysql -u root -p  
  
Setting environment for using XAMPP for Windows.  
sara.ruivo@DESKTOP-6I5VFCF c:\xampp  
# mysql -u root -p  
Enter password:  
Welcome to the MariaDB monitor.  Commands end with ; or \g.  
Your MariaDB connection id is 46  
Server version: 10.4.21-MariaDB mariadb.org binary distribution  
  
Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.  
  
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.  
  
MariaDB [(none)]> USE futebol;  
Database changed  
MariaDB [futebol]> _
```

Depois, basta executar o código de criação de cada uma das tabelas apresentado anteriormente. A imagem a seguir serve de exemplo.

```

C:\> Linha de comandos - mysql -h 127.0.0.1 -P 8889 -u root -proot
mysql> use futebol;
Database changed
mysql> CREATE TABLE temporada (
  -> temporadaID INT NOT NULL AUTO_INCREMENT,
  -> anoInicio YEAR NOT NULL,
  -> anoFim YEAR NOT NULL,
  -> PRIMARY KEY (temporadaID),
  -> UNIQUE INDEX anoInicio_UNIQUE (anoInicio ASC),
  -> UNIQUE INDEX anoFim_UNIQUE (anoFim ASC)
  -> )
  -> ENGINE = InnoDB;
Query OK, 0 rows affected (0.39 sec)

mysql>

```

Para verificar se as tabelas foram criadas conforme o esperado, pode utilizar os comandos SHOW e DESCRIBE abordados anteriormente.

```

C:\> XAMPP for Windows - mysql -u root -p
# mysql -u root -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 48
Server version: 10.4.21-MariaDB mariadb.org binary distribution

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> SHOW TABLES;
ERROR 1046 (3D000): No database selected
MariaDB [(none)]> USE futebol
Database changed
MariaDB [futebol]> SHOW TABLES;
+-----+
| Tables_in_futebol |
+-----+
| campeonato         |
| competicao          |
| equipa             |
| pais                |
| temporada          |
+-----+
5 rows in set (0.001 sec)

MariaDB [futebol]> DESCRIBE campeonato;
+-----+-----+-----+-----+-----+-----+
| Field      | Type      | Null | Key | Default | Extra      |
+-----+-----+-----+-----+-----+-----+
| campeonatoID | int(11)   | NO   | PRI | NULL    | auto_increment |
| competicaoID  | int(11)   | NO   | MUL | NULL    |               |
| temporadaID  | int(11)   | NO   | MUL | NULL    |               |
| campeao      | int(11)   | NO   | MUL | NULL    |               |
| subcampeao   | int(11)   | YES  | MUL | NULL    |               |
| data         | date      | YES  |     | NULL    |               |
| cronica      | text      | YES  |     | NULL    |               |
+-----+-----+-----+-----+-----+-----+
7 rows in set (0.008 sec)

```

CONCLUSÃO

Aprendeu a utilizar a linha de comandos e o MySQL Workbench com o intuito de criar as tabelas para uma base de dados dedicada ao futebol. A criação das tabelas na base de dados foi o resultado de uma reflexão prévia, que permitiu descobrir quais os dados importantes, de que tipo de dados se tratava, de que modo alguns dados estavam relacionados a outros e uma série de outros pontos que relacionados com a criação de bases de dados.

AUTOAVALIAÇÃO

1. **Com qual das instruções a seguir é que se pode criar uma base de dados chamada “Desporto”?**
 - a) CREATE NEW SCHEMA desporto;.
 - b) CREATE TABLE desporto;.
 - c) CREATE SCHEMA desporto;.
 - d) CREATE DATABASE desporto;.

2. **Qual dos seguintes mecanismos de base de dados permite o uso de chaves e transações estrangeiras?**
 - a) MyISAM.
 - b) EXAMPLE.
 - c) MEMORY.
 - d) InnoDB.

3. **Qual dos seguintes mecanismos de base de dados não armazena registos no disco?**
 - a) MyISAM.
 - b) EXAMPLE.
 - c) MEMORY.
 - d) InnoDB.

4. Com qual das seguintes afirmações pode obter-se uma lista dos motores de base de dados disponíveis no sistema?
- a) SHOW ENGINES;.
 - b) SHOW DATABASE MOTORS;.
 - c) SHOW MOTOR ENGINES;.
 - d) SHOW DATABASE ENGINES;.
5. Qual dos seguintes tipos de dados está relacionado com datas?
- a) INT.
 - b) DATETIME.
 - c) FLOAT.
 - d) VARCHAR.
6. Qual dos seguintes tipos de dados é do tipo string de comprimento fixo?
- a) CHAR.
 - b) YEAR.
 - c) DOUBLE.
 - d) VARCHAR.
7. Com qual dos seguintes tipos de dados inteiros podem ser alcançados os valores mais altos?
- a) TINYINT.
 - b) SMALLINT.
 - c) MEDIUMINT.
 - d) BIGINT.

8. Com qual das seguintes opções é possível que um valor numérico seja preenchido com zeros à esquerda até que o seu comprimento seja concluído?
- a) UNSIGNED.
 - b) ZEROFILL.
 - c) NULL.
 - d) CONSTRAINT.
9. Com qual dos seguintes tipos de dados relacionados com a data não é possível trabalhar com o ano 1955?
- a) DATE.
 - b) DATETIME.
 - c) TIMESTAMP.
 - d) YEAR.
10. O que é que significa a sigla ACID?
- a) Atomicity, Consistency, Insolation & Durability.
 - b) Atomicity, Consistency, Isolation & Durability.
 - c) Availability, Consistency, Insolation & Durability.
 - d) Availability, Consistency, Isolation & Durability.

SOLUÇÕES

1.	c	2.	d	3.	c	4.	a	5.	b
6.	a	7.	d	8.	b	9.	c	10.	b

PROPOSTAS DE DESENVOLVIMENTO DO ESTUDO

Para ler mais sobre tabelas consulte as seguintes páginas web (em inglês):

- <https://dev.mysql.com/doc/refman/8.0/en/create-table.html>
- https://www.w3schools.com/sql/sql_create_table.asp

BIBLIOGRAFIA

- Beaulieu, A. (2006), *Aprende SQL*. Madrid: AnayaMultimedia.
- Gutiérrez Gallardo, J. D. (2009), *MySQL 5.1*. Madrid: AnayaMultimedia.
- Oracle (2021), "Reference Manual". Disponível em:
<http://dev.mysql.com/doc/refman/5.1/en/>. Consultado a 7 de abril de 2021.

