# 数据库的基本操作

关于数据库的基本操作有创建数据库、查看数据库、修改数据库、删除数据库以及选择 数据库。

## 创建数据库

创建数据库：

**CREATE DATABASE 数据库名称;**

创建数据库成功后，会在MySQL的data目录中自动生成相应的文件夹，里面放置了该 数据库的配置文件。

## 查看数据库

1）查看系统中已存在的数据库：**SHOW DATABASES;**

2）查看某个数据库的信息：**SHOW CREATE DATABASE数据库名称;**

3）如果想要知道现在所处的是哪个数据库，其语法格式为：

**SELECT DATABASE( );**

## 修改数据库

1）修改数据库编码的基本语法格式如下所示：

**ALTER DATABASE 数据库名称 DEFAULT CHARACTER SET 编码方式;**

2）修改数据库校对规则的语法格式如下所示：

**ALTER DATABASE 数据库名称 COLLATE 校对规则\_bin;**

## 删除数据库

删除数据库只能将数据库系统中已存在的数据库删除，否则将删除失败。其语法格式为：  
 **DROP DATABASE 数据库名称;**

删除数据库后，数据库中的所有数据将被清除。

## 选择数据库

如果要操作某个数据库，则必须先进入该数据库，其语法格式为：

**USE 数据库名称;**

# MySQL数据类型

MySQL的数据类型包括整型、浮点型、定点型、日期和时间型、字符串型、二进制型 等。

## 整型

根据取值范围的不同，MySQL中的整数类型可分为5中，分别是TINYINT、SMALLINT、 MEDIUMINT、INT、BIGINT。

|  |  |  |  |
| --- | --- | --- | --- |
| 数据类型 | 字节数 | 无符号的取值范围 | 有符号的取值范围 |
| TINYINT | **1** | 0~255 | -128~127 |
| SMALLINT | **2** | 0~65535 | -32678~32767 |
| MEDIUMINT | **3** | 0~16777215 | -8388608~8388607 |
| INT | **4** | 0~4294967297 | -2147483648~2147483647 |
| BIGINT | **8** | 0~18446744073709551615 | -9223372036854755808~9223372036854775807 |

## 浮点型与定点型

在MySQL中，存储小数使用浮点型和定点型来表示。浮点型分为单精度浮点型（FLOAT） 和双精度浮点型（DOUBLE）；定点型使用DECIMAL表示。

|  |  |  |  |
| --- | --- | --- | --- |
| 数据类型 | 字节数 | 负值的取值范围 | 非负数的取值范围 |
| FLOAT | 4 | -3.402823466E+38~-1.175494351E-38 | 1.175494351E-38~3.402823466E+38和0 |
| DOUBLE | 8 | -1.7976931348623157E+308~-2.2250738585072014E-308 | -2.2250738585072014E-308~1.7977931348623157E+308和0 |
| DECIMAL(M,D) | M+2 | -1.7976931348623157E~-2.2250738585072014E-308 | 2.2250738585072014E-308~1.7976931348623154E+308和0 |

其中，定点型中的M表示的是数据的总长度（包括小数点）、D表示小数点后的长度。例如，将数据类型为DECIMAL（6,2）的数据3.1415插入数据后，显示的结果为3.14，多余位数会进行四舍五入。

## 日期与时间型

日期和时间型有5种，分别是YEAR、DATE、TIME、DATETIME、TIMESTAMP。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据类型 | 字节数 | 取值范围 | 日期格式 | 零值 |
| YEAR | 1 | 1901~2155 | YYYY | 0000 |
| DATE | 4 | 1000-01-01~9999-12-3 | YYYY-MM-DD | 0000-00-00 |
| TIME | 3 | -838:59:59~838:59:59 | HH:MM:SS | 00:00:00 |
| DATETIME | 8 | 1000-01-01 00:00:00~9999-12-31 23:59:59 | YYYY-MM-DD HH:MM:SS | 0000-00-00 00:00:00 |
| TIMESTAMP | 4 | 1970-01-01 00:00:01~2038-01-19 03:14:07 |  | 0000-00-00 00:00:00 |

如果插入的值不合法，系统会自动将对应的零值插入。

### YEAR型

YEAR型表示年份，使用以下三种方式指定YEAR值：

使用4为字符或数字表示。范围为‘1901~2155’或1901~2155。

### DATE型

DATE型表示日期，使用以下四种方式指定DATE值。

1. 以‘YYYY-MM-DD’或者‘YYYYMMDD’字符串表示。
2. 以YY-MM-DD或者YYMMDD数字格式表示。
3. 使用CURRENT\_DATE或者NOW( )表示当前系统时期。

### TIME型

TIME型表示时间，使用三种方式指定TIME值。

1. 以‘D HH:MM:SS’字符串表，其中D表示日，可以取0~34之间的值，插入数据时， 小时的值等于（D×24+HH）。
2. 以‘HHMMSS’字符串表示或者HHMMSS数字表示。
3. 使用CURRENT\_TIME或NOW( )插入当前系统时间。

### DATETIME型

DATETIME型表示日期时间型，使用四种格式指定DATETIME值。

1. 以字符串‘YYYY-MM-DD HH:MM:SS’或者‘YYYYMMDDHHMMSS’表示日期时间。

2）以数字YYYYMMDDHHMMSS表示日期时间。

3）使用NOW( )输入当前系统的日期时间。

### TIMESTAMP型

TIMESTAMP型表示日期时间，与DATATIME型相同，但取值范围比DATATIME小。

1. 使用CURRENT\_TIMESTAMP输入当前系统日期时间。
2. 输入NULL时，系统会输出输入系统当前日期时间。
3. 无任何输入时，系统会输入系统当前日期时间。

## 字符串型

使用CHAR型和VARCHAR型都可以表示字符串数据，不同的是VARCHAR可以存储可变 长度的字符串。其语法格式为：

**CHAR( M ) 或 VARCHAR( M )**

其中，M表示的是字符串的最大长度，单位字节；例如，CHAR( 4 )表示不管插入值的字 节数是多少，所占用的存储空间都是4个字节。而VARCHAR( 4 )所对应的数据所占用的 字节数为实际数据所占字节数+1。

## 二进制型

为了存储字符串、图片和声明等数据，MySQL提供了BINARY型和VARBINARY型表示二 进制型。定义格式为：

**BINARY( M ) 或 VARBINARY( M );**

M表示最大字节数。

BINARY型是固定的，如果数据的长度不足M，数据的后面将用“\0”补齐。

## ENUM型

ENUM型表示枚举型，定义格式为：

**ENUM (‘值1’ , ‘值2’ , ‘值3’ , …)**

其中(‘值1’ , ‘值2’ , ‘值3’ , …)表示枚举列表，ENUM型的数据只能从枚举列表中取一个。

枚举列表中的每一个值都有一个顺序编号，MySQL中存入的就是这个顺序编号，而不 是枚举列表中的值。

## SET型

SET型表示字符串对象，它的值可以由零个或多个，定义格式为：

**SET( ‘值1’ , ‘值2’ , ‘值3’ ,…)**

对于每一个值都有一个顺序编号，MySQL中存入的就是这个顺序编号，而不是列表中 的值。

## 大文本数据型

大文本数据型用来表示大文本的数据，如文章内容、评论等。它的类型分为四种，如下：

|  |  |  |  |
| --- | --- | --- | --- |
| 数据类型 | 存储范围 | 数据类型 | 存储范围 |
| TINYTEXT | 0~255个字节 | MEDIUMTEXT | 0~16777215个字节 |
| TEXT | 0~65535个字节 | LONGTEXT | 0~4294967295个字节 |

## BLOB型

BLOB型是一种特殊的二进制型，用于表示数据量很大的二进制数据。分为四种：

|  |  |  |  |
| --- | --- | --- | --- |
| 数据类型 | 存储范围 | 数据类型 | 存储范围 |
| TINYBLOB | 0~255个字节 | MEDIUMBLOB | 0~16777215个字节 |
| BLOB | 0~65535个字节 | LONGBLOB | 0~4294967295个字节 |

## BIT型

BIT型表示小二进制型。定义格式为：

**BIT( M )**

M表示每个值的位数，范围1~64。

如果数据长度小于M位，将在数据的左边用0补齐。如BIT( 6 )时，存入011实则为000011。

# 数据表的基本操作

数据表的基本操作有创建数据表、查看数据表、修改数据表、删除数据表。

## 创建数据表

在创建数据表之前应使用“USE 数据库名”指定要操作的数据库。其创建格式为：

**CREATE TABLE 表名(**

**字段名,数据类型 [约束条件] ,**

**.......**

**字段名n,数据类型 [约束条件]**

**)**

字段名不能重复。

## 查看数据表

1）查看当前数据库中的所有数据表：**SHOW TABLES;**

2）查看某个数据表的创建信息：**SHOW CREATE TABLE 表名;**

3）查看某个数据表的字段信息：**DESC 表名;**

4）查看某个数据表的状态：**SHOW TABLE STATUS LIKE ‘表名’\G**

**（\G表示格式化显示）**

## 修改数据表

1）修改表名：**ALTER TABLE 旧表名 RENAME TO 新表名;**

2）修改字段名：**ALTER TABLE 表名 CHANGE 旧字段名 新字段名 该字段的数据类型;**

3）修改字段的数据类型：**ALTER TABLE 表名 MODIFY 字段名 新数据类型;**

4）添加字段：**ALTER TABLE 表名 ADD 新字段名 数据类型 [约束条件] [FIRST|AFTER 已 存在字段名];**

其中，FRIST表示将要添加的字段设置为表的第一个字段，AFTER表示在指定的字段后面添加。

5）删除字段：**ALTER TABLE 表名 DROP 字段名;**

6）修改字段的排列位置：**ALTER TABLE 表名 MODIFY 字段名1 数据类型 FIRST|AFTER 字段名2;**

## 删除数据表

删除数据库中已存在的表，语法为：

**DROP TABLE 表名;**

这种语法只能删除没有被其他表关联的数据表。

# 表的约束

1）表的约束指的是对表的字段进行限制，其实质是对字段的取值进行约束，从而保证 数据的正确性和唯一性。

2）表的约束分为主键约束、非空约束、默认约束、唯一约束、自增增加约束、外键 约束。

## 主键约束

主键约束是使用PRIMARY KEY定义的约束。主键约束分为两种，单字段主键和 多字段 主键（多字段主键也称联合主键）。

1. 单字段主键指的是由一个字段构成的主键，其语法格式为：

**字段名 数据类型 PRIMARY KEY,**

1. 多字段主键指的是多个字段组合而成的主键，其语法格式为：

**PRIMARY KEY( 字段1, 字段2 ,…),**

每个数据表中最多只有一个主键约束，主键约束的字段值不能为空（NULL），也不能 重复（必须唯一）。

## 非空约束

非空约束是使用NOT NULL定义的约束。其语法格式为：

**字段名 字段类型 NOT NULL,**

非空约束的字段的值不能为NULL。

## 唯一约束

唯一约束是使用UNIQUE定义的约束，其语法格式为：

**字段名 字段类型 UNIQUE,**

唯一约束的字段的值不能重复，必须唯一。

## 默认约束

默认约束是使用DEFAULT定义的约束，其基本语法格式为：

**字段名 字段类型 DEFAULT,**

默认约束用于设置字段的默认值。

## 自动增加约束

自动增加约束是使用AUTO\_INCREMENT定义的约束。其语法格式为：

**字段名 数据类型 PRIMARY KEY AUTO\_INCREMENT,**

AUTO\_INCREMENT约束只能定义在任何整型的字段上并且该字段定义了主键约束或唯一约束。自动增加约束的字段值如果没有给它赋值，那么该字段的值将从已存在的最大值开始递增+1。

# 索引

1）索引是数据库中专门用于快速查询数据的一种数据结构。类似于字典中的目录。

2）数据库的索引好比字典的目录表，是对数据库表中一列或多列的值进行排序后的一 种结构，其作用是提高查询表中数据的速度和效率。

3）索引可以提高查询数据的速度和效率，但索引会占用一定的磁盘空间，并且在创建 和维护索引时，其消耗的时间是随着数据量的增加而增加的。因此，使用索引时， 应该综合考虑索引的优点和缺点。

## 索引的分类

索引分为普通索引、唯一索引、全文索引、单列索引、多列索引、空间索引。

1. 普通索引是使用INDEX定义的索引。普通索引时MySQL中的基本索引，可以 创建在任何数据类型中，其值是否唯一和非空由字段的本身约束条件所决定。所有

的其他类型索引都要在普通索引的基础上创建。

1. 唯一索引是使用UNIQUE定义的索引，拥有唯一索引的字段值必须是唯一的。
2. 全文索引是使用FULLTEXT定义的索引，只能创建在CHAR、VARCHAR、TEXT类型的 字段上。并且只有MyISAM存储引擎支持全文索引（表默认的存储引擎是InnoDB）。
3. 单列索引指的是在表中单个字段上创建的索引，可以是普通索引、唯一索引、全文 索引，只要保证该索引只对应表中的一个字段。
4. 多列索引指的是在表中多个字段上创建索引。但只有在查询条件中使用了这些字段 的第一个字段时，该索引才会被使用。
5. 空间索引是使用SPATIAL定义的索引，空间索引只能创建在空间数据类型（GEOMETRY、 POINT、LINESTRING和POLYGON）的字段上。创建空间索引的字段需将其声明为NOT NULL，并且只有MyISAM存储引擎支持空间索引

## 创建索引

创建索引的方式有两种。

1. 创建表的时候创建索引。
2. 使用CREATE INDEX语句在已存在的表上创建索引。

### 创建表的时候创建索引

在创建表的时候直接创建索引，其语法格式为：

**CREATE TABLE 表名(**

**字段名 数据类型 [约束条件] ,**

**......**

**字段名 数据类型 [约束条件] ,**

**[UNIQUE|FULLTEXT|SPATIAL] INDEX [ 别名] (字段名1[(长度)] [ASC|DESC])**

**);**

1. UNIQUE：可选参数，表示唯一索引。
2. FULLTEXT：可选参数，表示全文索引。
3. SPATIAL：可选参数，表示空间索引。
4. INDEX：必选参数，表示字段的索引
5. 别名：可选参数，表示创建的索引的名称。如未设置，则以字段名为索引名。
6. 字段名1：必选参数，用于指定索引对应字段的名称，多个字段时使用逗号隔开。
7. ASC或DESC：可选参数，ASC表示升序排列，DESC表示降序排列。

8）如果没有为索引命名，则索引名将使用字段名作为索引名，也可以重新别名。

### 使用CREATE INDEX语句创建索引

使用CREATE INDEX语句在已存在的表中创建索引的语法格式为：

**CREATE [UNIQUE|FULLTEXT|SPATIAL] INDEX 索引名ON 表名 (字段名[(长度)] [ASC|DESC]);**

其中，UNIQUE、FULLTEXT、SPATILA都是可选参数，表示唯一性索引、全文索引、空间 索引。

ASC表示升序排列，DESC表示降序排列。

## 删除索引

由于索引会占用一定的磁盘空间。因此，为了避免影响数据库的性能，应该及时删除不再使用的索引。删除索引的语法格式为：

**DROP INDEX 索引名 ON 表名;**