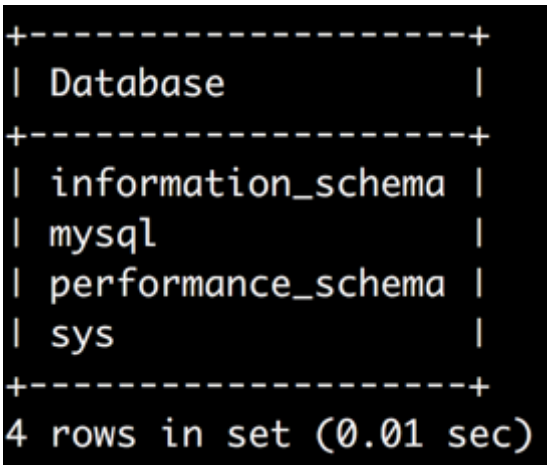
# 数据定义语言（DDL）

## 1、数据库操作

- 显示所有数据库

  命令：`show databases;`




  mysql数据库：用户权限相关数据

  information_schema数据库：MySQL本身架构相关数据

- 创建数据库
  - 格式

    utf-8：`CREATE DATABASE 数据库名称 DEFAULT CHARSET utf8 COLLATE utf8_general_ci;`

    gbk：`CREATE DATABASE 数据库名称 DEFAULT CHARACTER SET gbk COLLATE gbk_chinese_ci;`

  - 排序规则：对于mysql中那些字符类型的列,如VARCHAR,CHAR,TEXT类型的列,都需要有一个*COLLATE*类型来告知mysql如何对该列进行排序和比较。

| 值 | 说明 |
| --- | --- |
| utf8_general_ci | 校对速度快，但准确度稍差 |
| utf8_bin | 字符串每个字符用二进制数据编译存储 |
| utf8_unicode_ci | 准确度高，但校对速度稍慢。若数据库中有德语、法语或者俄语需求，需使用utf8_unicode_ci |
| gbk_chinese_ci | 特定中文时使用 |

  - 示例

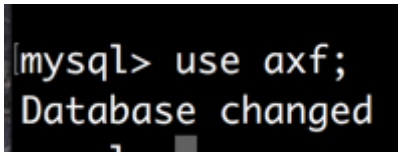
    `create database axf default charset utf8 collate utf8_general_ci;`

```
mysql> create  database  axf  default
Query OK, 1 row affected (0.00 sec)

mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| axf                |
| mysql              |
| performance_schema |
| sys                |
+--------------------+
5 rows in set (0.00 sec)
```

- 切换数据库

  格式：`use  数据库名;`

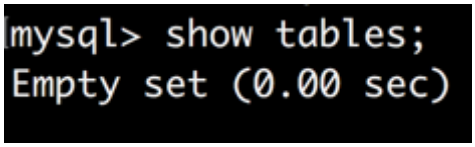



- 查看当前选择的数据库

  格式：`select  database();`

- 显示当前使用的数据库中所有表

  格式：`show tables;`




- 删除数据库

  格式：`drop  database  数据库名;`

## 2、表操作

**创建表**

```
create table student(name char(20), age int);
create table teacher(name char(20), age int);
```

- 数据完整性

  一个数据库就是一个完整的业务单元，可以包含多张表，数据被存储在表中。在表中为了更加准确的存储数据，保证数据的正确有效，可以在创建表的时候，为表添加一些强制性的验证，包括数据字段的类型、约束

- 格式

  create table 表名(列名 类型 [约束1[,约束2[,.....]]]) engine=引擎 default charset=编码;

- 名词解析

| 名词 | 解析 |
| --- | --- |
| 表名 | 给数据集合起的名字，默认表名是区分大小写的 |
| 列名 | 字段的名字，遵守标识符规则<br>列名与列的别名在所有的情况下均是忽略大小写的 |
| 编码 | 要与数据编码相同，可以解决存储编码错误的问题 |
| 引擎 | myisam：默认引擎，不支持事物<br>innodb：支持事物，原子性操作 |
| 类型 | 明确字段存储的数据类型 |
| 约束 | 对字段数据的进一步控制 |

**数据类型**

- 数值类型

| 类型 | 大小 | 范围（有符号） | 范围（无符号） | 用途 |
| --- | --- | --- | --- | --- |
| TINYINT | 1 byte | (-128，127) | (0，255) | 小整数值 |
| SMALLINT | 2 bytes | (-32 768，32 767) | (0，65 535) | 大整数值 |
| MEDIUMINT | 3 bytes | (-8 388 608，8 388 607) | (0，16 777 215) | 大整数值 |
| INT或INTEGER | 4 bytes | (-2 147 483 648，2 147 483 647) | (0，4 294 967 295) | 大整数值 |
| BIGINT | 8 bytes | (-9,223,372,036,854,775,808，9 223 372 036 854 775 807) | (0，18 446 744 073 709 551 615) | 极大整数值 |
| FLOAT | 4 bytes | (-3.402 823 466 E+38，-1.175 494 351 E-38)，0，(1.175 494 351 E-38，3.402 823 466 351 E+38) | 0，(1.175 494 351 E-38，3.402 823 466 E+38) | 单精度<br>浮点数值 |
| DOUBLE | 8 bytes | (-1.797 693 134 862 315 7 E+308，-2.225 073 858 507 201 4 E-308)，0，(2.225 073 858 507 201 4 E-308，1.797 693 134 862 315 7 E+308) | 0，(2.225 073 858 507 201 4 E-308，1.797 693 134 862 315 7 E+308) | 双精度<br>浮点数值 |
| DECIMAL | 对DECIMAL(M,D)，如果M>D，为M+2否则为D+2 | 依赖于M和D的值 | 依赖于M和D的值 | 小数值 |

- 日期和时间类型

| 类型 | 大小 ( bytes) | 范围 | 格式 | 用途 |
| --- | --- | --- | --- | --- |
| DATE | 3 | 1000-01-01/9999-12-31 | YYYY-MM-DD | 日期值 |
| TIME | 3 | '-838:59:59'/'838:59:59' | HH:MM:SS | 时间值或持续时间 |
| YEAR | 1 | 1901/2155 | YYYY | 年份值 |
| DATETIME | 8 | 1000-01-01 00:00:00/9999-12-31 23:59:59 | YYYY-MM-DD HH:MM:SS | 混合日期和时间值 |
| TIMESTAMP | 4 | 1970-01-01 00:00:00/2038<br>结束时间是第 **2147483647** 秒，北京时间 **2038-1-19 11:14:07**，格林尼治时间 2038年1月19日 凌晨 03:14:07 | YYYYMMDD HHMMSS | 混合日期和时间值，时间戳 |

- 字符串类型

| 类型 | 大小 | 用途 |
| --- | --- | --- |
| CHAR | 0-255 bytes | 定长字符串 |
| VARCHAR | 0-65535 bytes | 变长字符串 |
| TINYBLOB | 0-255 bytes | 不超过 255 个字符的二进制字符串 |
| TINYTEXT | 0-255 bytes | 短文本字符串 |
| BLOB | 0-65 535 bytes | 二进制形式的长文本数据 |
| TEXT | 0-65 535 bytes | 长文本数据 |
| MEDIUMBLOB | 0-16 777 215 bytes | 二进制形式的中等长度文本数据 |
| MEDIUMTEXT | 0-16 777 215 bytes | 中等长度文本数据 |
| LONGBLOB | 0-4 294 967 295 bytes | 二进制形式的极大文本数据 |
| LONGTEXT | 0-4 294 967 295 bytes | 极大文本数据 |

**注意**：char(n) 和 varchar(n) 中括号中 n 代表字符的个数，并不代表字节个数，比如 CHAR(30) 就可以存储 30 个字符。

修改表结构：

```
# 添加列`alter table 表名 add 列名 类型;`
ALTER TABLE student ADD score INT(20);

- 删除列  `alter table 表名 drop  列名;`
ALTER TABLE student DROP score;

- 修改列
    类型：`alter table 表名 modify column 列名 类型;`
    ALTER TABLE student MODIFY COLUMN age INT(20);

    列名&类型：`alter table 表名 change 原列名 新列名 类型;`
    ALTER TABLE student CHANGE age score INT(20);
```

修改表名称：

```
# RENAME TABLE 原表名 TO 新表名;
RENAME TABLE student TO students;
```

清空表数据

`delete from 表名;`

`truncate table 表名;`

删除表

`drop  table  表名;`

示例**

```
create table student(
    id int(11) not null AUTO_INCREMENT primary key,
    sex int(11),
```

```
    age int(11),
    name VARCHAR(20),
    description varchar(100),
    birthday date
)ENGINE=InnoDB DEFAULT CHARSET=utf8;

等价于如下：  可用命令查看：SHOW CREATE TABLE student;

CREATE TABLE `student` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `sex` int(11) DEFAULT NULL,
  `age` int(11) DEFAULT NULL,
  `name` varchar(20) DEFAULT NULL,
  `description` varchar(100) DEFAULT NULL,
  `birthday` date DEFAULT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

```
insert into student values(0,1,18,'tom','tom is a good man','1999-01-01');
```

**约束**

- primary key

  作用：约束唯一标识数据库表中的每条记录

  注意：

  主键必须包含唯一的值

  主键列不能包含 NULL 值

  每个表都应该有一个主键，并且每个表只能有一个主键

- not null

  作用：约束强制列不接受 NULL 值

  注意：约束强制字段始终包含值，这意味着如果不向字段添加值，就无法插入新记录或者更新记录

- null

  作用：允许为空

- auto_increment

  作用：自增长

  注意：对于自增长列，必须是索引(含主键)

- default

  作用：用于向列中插入默认值

- unique

  作用：约束唯一标识数据库表中的每条记录

  与primary key的区别：

  - UNIQUE 和 PRIMARY KEY 约束均为列或列集合提供了唯一性的保证
  - PRIMARY KEY 拥有自动定义的 UNIQUE 约束
  - 每个表可以有多个 UNIQUE 约束，但是每个表只能有一个 PRIMARY KEY 约束

  使用：unique 唯一索引名称 (列名[, .....])

- binary(不讲)

作用：设置字段大小写敏感

- foreign key

  说明：一个表中的 FOREIGN KEY 指向另一个表中的 PRIMARY KEY

**示例**

```
CREATE TABLE student(
    id INT(11) NOT NULL AUTO_INCREMENT PRIMARY KEY,
    sex INT(11),
    age INT(11),
    NAME VARCHAR(20),
    description VARCHAR(100),
    birthday DATE,
    UNIQUE(NAME)
)ENGINE=INNODB DEFAULT CHARSET=utf8;
```

```
insert into student values(0,1,18,'tom','tom is a good man','1999-01-01');
insert into student values(0,1,18,'tom','tom is a good man','1999-01-01');
```

就会报错。

- 添加主键

  `alter table 表名 add primary key(列名);`

  eg:ALTER TABLE huazi.student ADD PRIMARY KEY(age);

- 删除主键

  `alter table 表名 drop primary key;`

  列名后的类型会修改列名的类型：`alter table 表名 modify 列名 int,drop primary key;`

- 添加外键

  `alter table 从表 add constraint 外键名称（形如：FK_从表_主表） foreign key 从表(外键字段) references 主表(主键字段);`

- 删除外键

  `alter table 表名 drop foreign key 外键名称`

- 修改默认值

  `alter table 表名 alter 列名 set default 值;`

- 删除默认值

  `alter table 表名 alter 列名 drop default;`