MySQL的DDL命令(create、drop、truncate、alter)及show命令

# 数据定义语言DDL（Data Define Language）

DDL就是用于描述**数据库中要存储的现实世界实体的语言**。

在一些特定情况下，如：金融、安全、税务等等部门，有一些数据库为了安全需要，**需要禁止执行DDL语句**。数据库的DDL命令主要包括：**create、drop、truncate（只对table）、alter**，为**创建、删除和更改（增删改）**。

由于**DDL的操作对象是数据库或表格**，所以在语句中应该声明目标是**database**还是**table**。而DML与DQL语句只是针对表格中的数据记录操作，所以不用加‘table’关键字。

## 对数据库（database）的操作：

对数据库的操作包括:创建数据库、删除数据库、修改数据库。

**create database** dbName chaset ...;

**drop database** dbName;

**alter database** dbName set ...;

## 对于表格（table）的操作：

对表格的操作包括：创建表格、删除表格、修改表格、查看表格。其中修改表格最为复杂，修改表格包括：对表格重命名、添加字段、删除字段、对字段重命名、修改字段数据类型和约束。

**create table** tbName(f1 type1,f2 type2,f3 type3...) charset ...; #创建表格#

**drop table** tbName; #删除表格#

**truncate (table)** tbName; #清空表格，‘table’可以省略。

**alter table** tbName **rename** ...;#对表格重命名#

**alter table** tbName **add ...**; #增加字段#

**alter table** tbName **drop** ...;#删除字段#

**alter table** tbName **modify** ...;#修改字段#

**alter table** tbName **change** ...;#对字段重命名#

# show与desc(describe)命令、use命令

## 显示所有的数据库与表格

**show databases;** 显示所有的数据库；

**use** dbName; 进入dbName数据库；

**show tables;** 显示该数据库的所有表格；

若**直接查看某个数据**的所有表格：(未进入某个数据库或在一个数据库查看另外数据库的表格)

**show tables from** dbName; #不能使用show dbName.tables;错误的

## 查看数据库属性及表格属性结构

**show create** **database** dbName; 显示创建数据库的语句;

**show create table** tbName; 显示创建表格的语句;

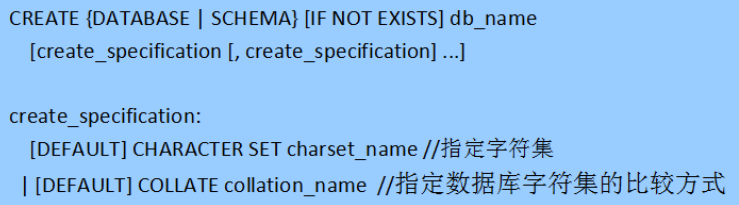
**desc(describe)** tbName; 显示tbName表格的结构（字段名称、数据类型、约束等）。

其他数据库的表格结构：

**desc dbName.tbName;** #查看dbName数据库中的tbName表格结构

# 对数据库的操作

## 创建数据库:

语法：

**create database** dbName; # 字符集、校对规则都是默认的

**create** **database** dbname [default] **character set** utf8; #字符集为utf-8，校对默认。

注意：utf8中间没有连接符-或\_。

**create database** dbName **character set** gbk **collate** gbk\_chinese\_ci;

**create database** dbname [default] **character set** utf8 **collate utf8\_general\_ci** ;

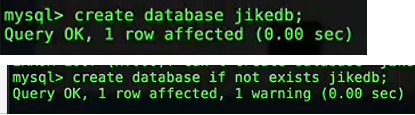
建议创建数据库的时候就指定字符集（字符编码方式）。CHARACTER SET GBK COLLATE gbk\_chinese\_ci 创建表格的时候，必要情况下，可以为每一个字段单独指定字符集（编码集）。

总结：一般创建数据库的时候指定一下字符集（Charset）和校对规则（Collate）

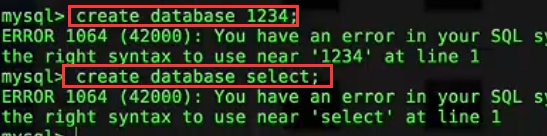
**GBK：create database 库名 default character set gbk collate gbk\_chinese\_ci;**

**UTF-8：create database 库名 default character set utf8 collate utf8\_general\_ci;**

**注意1**： 如果名为dbName的数据库已经存在，则会报错 Can't create database 'db1'; database exists。防止报错： **create database if not exists** dbName; 如果已经存在，则不创建，不会报错；若不存在，则创建。



**注意2**：数据库和表格的名字不能是纯数字，也不能是MySQL中的**关键字**。



## 数据库命名规则：

### 由字母、数字、下划线、@、#和$符号组成。

### 首字母不能是数字和$符号；

### 标识符不允许是MySQL的保留字；

### 不允许有空格和特殊字符；

### 长度小于128位。

## 删除数据库:drop database dbName;

## 修改数据库: alter database dbName ...;

**主要是修改数据库的字符集问题。**

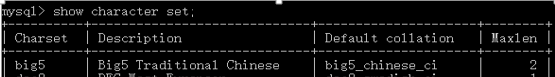
**直接把创建语句中的create改为alter就可以了。**

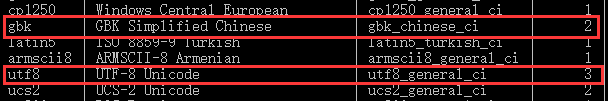
**alter database dbName character set utf8 collate utf8\_general\_ci;**

# 数据库及表格的字符集问题

## 利用show character set; 语句查询数据的字符集及collation

**show character set;** 或者**show charset**;或者**show char set**；



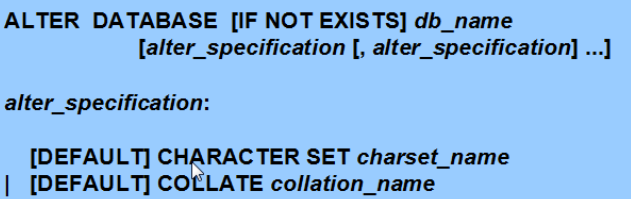


## 查看某个数据库的字符集

**show create database dbname；**



## 修改数据库的字符集和校对规则



**只需要把create改成alter即可**。不再赘述，使用较少。

# 表的操作

在MySQL中，表是一种很重要的数据库对象，是组成数据库的基本元素。表由若干个字段组成，主要用来实现存储数据记录。表的操作包括（增删改查）创建表、删除表、修改表和查看表。表中的数据库对象包括：列、索引、触发器。

# 创建表：create table

## 语法:

**create table** tbName(

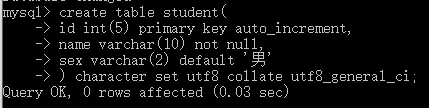
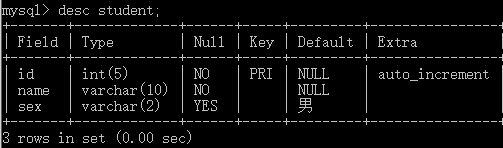
field1 type1 primary key auto\_increment,

field2 type2 not null,

field3 type3 default 0

) **character set** utf8 **collate** utf8\_general\_ci;

## 示例:

# drop：删除数据库或表格(简单)、truncate清空表格

## 删除数据库：drop database dbname;#删除一个数据库（无法一次删除多个数据库）

## 删除表格：drop table tbname; #删除一个表格

**drop table** tb1,tb2,tb3,...;#删除多个表格。

## 清空表格：truncate (table) tbname; #一次只可以清空一个表格（无法多个）

truncate 只针对表格，所以table可以省略。

功能：先利用**drop table** tbName;删除表格，然后再按照原表格的数据结构创建**create table** ...一个新的表格。**清空表格内容：delete from tbname;与truncate table tbName;。**

# 查看表

**查看表的基本结构：desc** tbName;

**查看表的详细结构：show create table** tbName;

# alter命令：

## 概述

**alter命令**主要是用来**修改数据库、表格的字符集、对表格重命名**以及**对表格添加字段、删除字段、对字段重命名、修改字段数据类型等**。

## 修改数据库、表格的字符集

**alter database** dbName **[default] character set** utf8 **collate** utf8\_general\_ci; 修改数据库字符集

**alter table** tbName **character set** utf8 **collate** utf8\_general\_ci;修改表格字符集

**创建数据库的模板**（utf8或gbk）

**create database 库名 default character set gbk collate gbk\_chinese\_ci;**

**create database 库名 default character set utf8 collate utf8\_general\_ci;**

## 对表重命名

**alter + rename**：对表格重命名；

**alter table** tbName **rename (to)** newTbName;#to可以省略

或者：**rename talbe** tbName **to** newTbName;#to不可以省略

## 对表格字段的增删改查（add、drop、change、modify、desc）

**alter + add**：添加字段; 位置：第一个、最后一个、某字段之后

**添加字段的位置：**①默认情况下在最后位置添加新的字段；②若要在最前端添加新字段，需要语句最后添加first，如**add** field type **first**; ③在**表的指定字段之后**添加字段，最后添加**after field**；如**add** field1 type **after** field2;

**alter table** tbName **add [column]** newField type default 0; #**最后位置**添加一个字段

**alter table** tbName **add [column]** newField type default 0 **first**; #**第一位置**添加一个字段

**alter table** tbName **add [column]** newField type **after field**; #**某字段之后位置**添加一个字段

**alter table** tbName **add [column]** newField1 type default 0，**add** **[column]** newField2 type not null,**add [column]** ...; 默认都在最后位置添加多个字段，可以分别为每个新字段指定添加的位置。

**alter + drop**：删除字段；（easy）

**alter table** tbName **drop** **[column]** field1;#删除一个字段

**alter table** tbName **drop** **[column]** field1,**drop** **[column]** field2,**drop [column]** ...;#删除多个字段

**alter + change**：对**字段重命名**、**修改数据类型及约束**

**alter table** tbName **change** oldField newField type [default 0];修改一个字段

**alter table** tbName **change** oldField1 newField1 type [default 0],**change** oldField2 newField2 type [default 0],**change** ...; 修改多个字段

注意：后面一定要带**数据类型**，否则报错。旧名字与新名字可以相同，名字没有改变，可以改变数据类型及约束，功能同alter+modify。

**alter + modify：**修改字段的数据类型及约束、**修改字段的顺序**

**alter table** tbName **modify** field newType [default 0]; 修改1个字段

**alter table** tbName **modify** field1 newType [default 0],**modify** field2 newType2 [default 0],**modify** field3 newType3 [default 0],**modify** ...;修改多个字段

**调整字段的顺序**：

**alter table** tbName **modify** field type **first**;#将该字段调整到第一个位置；

**alter table** tbName **modify** field1 type **after** field2;#将该字段调整到另一个字段之后；

**alter table** tbName **modify** field1 type **after** field2, **modify** field3 type **after** field4,modify...;同时可以调整多个字段的位置

**示例：alter table** user **modify** name varchar(10) **after** sex,**modify** score int(10) **first**;

注意：**不能随意修改数据类型**，如非数字的varchar类型不可以转成int类型，数字的varchar类型可以转换为int类型，int类型也可以转换为varchar类型。

**desc** tbName**; 或者 show create table** tbName**;**查询字段

## alter table tbName rename to 、add、drop、change、modify...等可以混合使用

**alter table** user **rename to** people,**add** address varchar(100),**drop** Fd,**change** sex gender char(4),**modify** score int(6); 混合使用

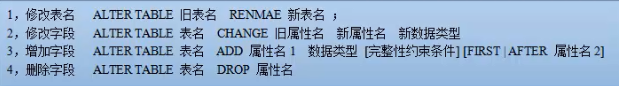
## 增添约束add constraint

一个表中只有一个主键，主键可以为单字段主键，也可以为多字段主键（联合主键）。

**alter table** tbName **add constraint primary key**(field);#单字段主键

**alter table** tbName **add constraint primary key**(field1,field2,...);#多字段主键

**alter table** tbName **add constraint unique**(field1,field2,...);#为这些键设置唯一约束



# 实现ID自增

一般情况下，一个表格都会有一个id，用来设置主键，并且设置为自增。

## 作为主键默认为NOT NULL、UNIQUE，非空唯一

这个ID的作用就是**用来唯一标识一条数据记录**。一般从1开始。

在插入记录的时候，不需要知道当前id需要是多少，直接插入数据，数据库会自动分配一个id号。

## 方法：重要

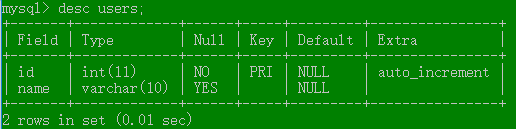
创建表格时：

create table students(**id int primary key auto\_increment**,name varchar(10));

或者 create table students2(id int,name varchar(10), **primary key(id)**);

或者 create table sss(id int,name varchar(10));

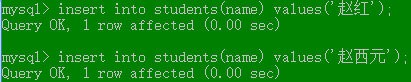
**alter table students3 modify id int primary key;**



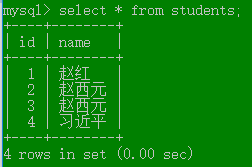
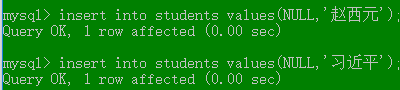
## 插入数据方法

### 第一种插入数据方法：

对于id，不用添加值，自动增加。



### 第二种插入数据的方法：



## 注意：

### 如果人为插入一个id号，且不是连续的，则设置成功，且以后自动的id号都会以最大的id为基准进行增加。（设置已经存在的id号时是错误的，因为id是主键，是唯一的，不允许重复）

**insert into** students **values**(NULL,’习平’);

### 在插入新的记录之前，把最后一条记录删除，插入新的记录的id会在删除记录的基础上继续自增。

