# 停止与开启服务

【windows】管理员模式启动命令行

net stop mysql

net start mysql

【linux】

# SQL语句分类DDL、DML、DCL

【DDL】（Data Definition Languages）

数据定义语句，这些语句定义了不同的数据段、数据库、表、列、索引等数据库对象。常⽤的语句关键字主要包括create、drop、alter等。

【DML】（Data Manipulation Language）

数据操纵语句，用于添加、删除、更新和查询数据库记录，并检查数据完整性。常⽤的语句关键字主要包括 insert、delete、update和select等。

【DCL】（Data Control Language）

数据控制语句，用于控制不同数据段直接的许可和访问级别的语句。这些语句定义了数据库、表、字段、用户的访问权限和安全级别。主要的语句关键字包括grant、revoke等。

# 特殊作用的语句

【重启数据库】

管理员模式进入命令行

停止：输入 net stop mysql

启动：输入 net start mysql

【选择数据库】

USE db\_name;

【显示授予用户的权限】

SHOW GRANTS;

【执行sql脚本】

source F:\hello world\niuzi.sql

# 增删改查1——DDL

DDL针对表本身的定义、结构

------------------------【库】------------------------

【增·库】

CREATE DATABASE db\_name;

【删·库】

DROP DATABASE db\_name;

【改·库】

【查·库】

SHOW DATABASES;

------------------------【表】------------------------

【增·表】

CREATE TABLE table\_name(

column\_name1 type1 约束条件,

column\_name2 type2 约束条件,

…);

【删·表】

DROP TABLE table\_name;

【改·表】

【修改表名】

ALTER TABLE table\_name RENAME [TO] new\_name;

【查·表】

【查看库中的表名】

SHOW TABLES;

SHOW TABLES FROM db\_name;

【查看表的定义】

DESCRIBE table\_name;

DESC table\_name;

【查看表的详细定义】

SHOW CREATE TABLE table\_name \G;

\G是按字段竖向排列显示

------------------------【列】------------------------

【增·列】

ALTER TABLE table\_name ADD [COLUMN] column\_name type;

【删·列】

ALTER TABLE table\_name DROP [COLUMN] column\_name;

【改·列】

【修改类型】

ALTER TABLE table\_name MODIFY [COLUMN] column\_name new\_type;

新的类型可以和原来的相同

【修改列名和类型】

ALTER TBALE table\_name CHANGE COLUMN column\_name new\_name new\_type;

新的列名和新的类型可以和原来的相同

【修改列的排列顺序】

上面的ADD、MODIFY、CHANGE语句后面可接[FIRST|AFTER column\_name]

用于调整列的位置

ALTER TABLE table\_name MODIFY COLUMN column\_name new\_type FIRST;

【查·列】

SHOW COLUMNS FROM table\_name;

# 增删改查2——DML

DML针对数据库中的数据，不改变表的定义和结构

【增·记录】

INSERT INTO table\_name field VALUES value;

【多个列同时插入】

INSERT INTO table\_name (field1,field2) VALUES (value1,value2);

【批量插入】

INSERT INTO table\_name (field1,field2)

VALUES

(value1,value2),

(value1,value2),

(value1,value2);

【删·记录】

DELETE FROM table\_name [WHERE CONDITION]

不加WHERE会把所有记录删除

【多表删除】

例：delete a,b from emp a,dept b where a.deptno=b.deptno and a.deptno=3;

此例当中有别名

【改·记录】

UPDATE table\_name SET field1=value1, field2=value2 [WHERE CONDITION]

【多表更新】

UPDATE t1,t2 SET t1.field=value,t2.field=value [WHERE CONDITION]

多表更新的语法更多地用在了根据一个表的字段来动态地更新另外一个表的字段。

例：UPDATE t1,t2 SET t1.field=t2.field\*10 WHERE t1.name=t2.name;

【查·记录】

SELECT \* FROM table\_name [WHERE CONDITION]

# 查询的子句WHERE

【排序】子句（clause）

ORDER BY

【降序排序】DESC (descend)

【升序排序（默认）】ASC (ascend)

SELECT \* FROM tablename [WHERE CONDITION]

[ORDER BY field1[DESC|ASC]，field2 [DESC|ASC],…,fieldn [DESC|ASC]]

field1中值相同则按field2排序，以此类推

【去除重复】

DISTINCT

【指定行数】LIMIT

SELECT … [LIMIT offset\_start,row\_count]

offset\_start:起始偏移量（默认为0）

row\_count: 显示的行数

常与ORDER BY配合使用过实现记录的分页

【过滤数据】子句 确定一个列是否包含特定值

WHERE

BETWEEN \*\*\* AND \*\*\* 始末区间

IS BULL 列中部包含值

AND

OR

IN

NOT

# 权限——DCL

【授予权限】GRANT

GRANT select,insert ON db\_name.\* TO 'z1'@'localhost' IDENTIFIED BY '123';

创建用户并授予权限

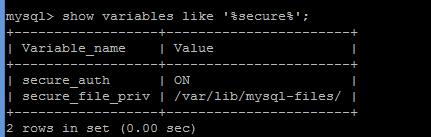
grant 权限 on 数据库对象 to '用户名'@'允许登录的主机' identified [by](https://www.baidu.com/s?wd=by&tn=SE_PcZhidaonwhc_ngpagmjz&rsv_dl=gh_pc_zhidao) '密码'

【回收权限】REVOKE

REVOKE insert ON db\_name.\* FROM 'z1'@'localhost';

# 从文件输入数据（CSV）

【输入权限】show variables like ’%secure%’;



secure\_file\_priv显示的是允许上传的文件夹，默认null不允许上传下载

可以在my.ini中修改：

secure\_file\_priv = 路径名

【CSV】

LOAD DATA LOCAL INFILE 'C:\\Users\\knight\\Desktop\\articles.csv'

INTO TABLE table\_name

FIELDS TERMINATED BY ','

ENCLOSED BY '"'

LINES TERMINATED BY '\n'

IGNORE 1 ROWS;

本地文件名

表名

字段分隔符

忽略双引号

忽略第一行

LOAD DATA LOCAL INFILE 'C:\\Users\\knight\\Desktop\\test2.csv'

INTO TABLE articles

FIELDS TERMINATED BY ','

ENCLOSED BY '"""'

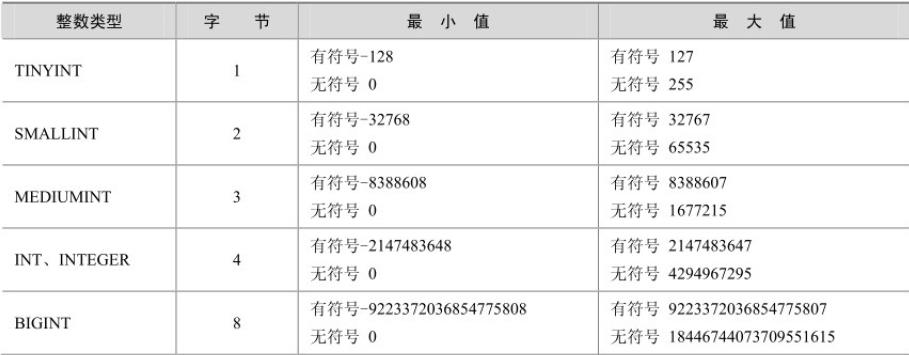
LINES TERMINATED BY '\n'

IGNORE 1 ROWS;

# 数据类型

## 【整数】

tinyint smallint mediumint integer bigint



int(M)括号和zerofill配合使用，指定显示宽度，不足补0，默认11。

ALTER TABLE table\_name MODIFY COLUMN column\_name int(11) zerofill;

只改变显示，实际的精度还是由数据类型决定。整数位超过M时指定位数失去意义。

## 【小数】

float double decimal

浮点数和定点数都可以后面加括号(M,D)限定精度（是真正的精度，按四舍五入计算）

共M位数字（整数+小数），其中D位小数。

浮点数后面没有(M,D)默认实际精度

定点数后面没有(M,D)默认(10,0)

【浮点型】float double



【定点型】decimal（十进制的、小数的）

定点数在MySQL中以字符串形式存放，比浮点数更加精确。

## 【位类型】

bit



(M)指定位数，默认1位

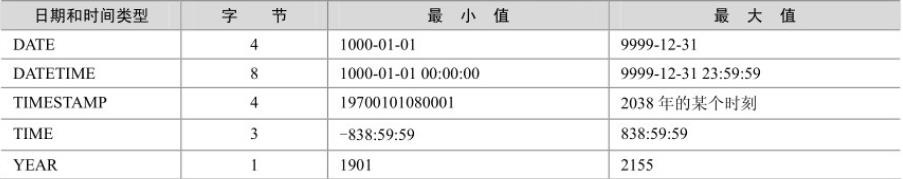
## 【时间】

年月日 date

时分秒 time

年月日时分秒 datetime

时间戳 timestamp



用now()函数插入当前日期

CREATE TABLE table\_name (d DATE,t TIME,dt DATETIME,ts TIMESTAMP);

INSERT INTO table\_name VALUES (now(),now(),now());

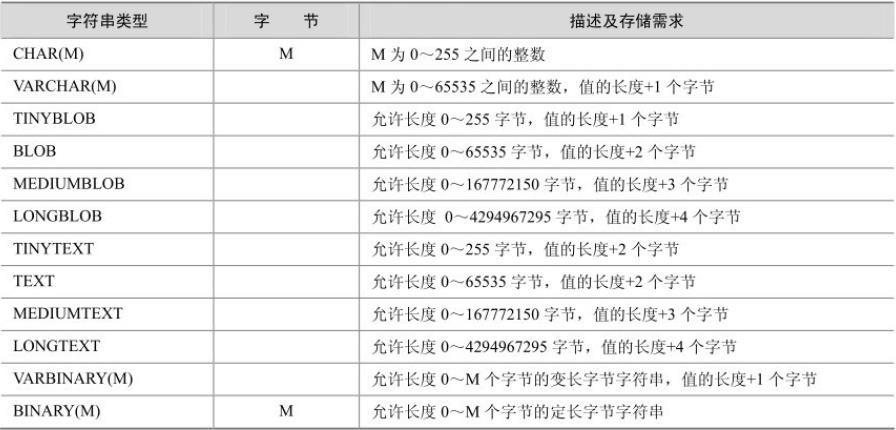
TIMPSTAMP会显示年月日时分秒格式，如果想显示14位数字格式则后面+0。

select ts+0 from table\_name;

TIMESTAMP

表中第一个TIMESTAMP会自动设置为系统时间。插入NULL会变为系统当前时间，插入或更新一个记录单不明确指定TIMESTAMP时也会自动更新为当前系统时间。

## 【字符串】



【text类型】

|  |  |  |
| --- | --- | --- |
| TINYTEXT | 256 bytes |  |
| TEXT | 65,535 bytes | ~64kb |
| MEDIUMTEXT | 16,777,215 bytes | ~16MB |
| LONGTEXT | 4,294,967,295 bytes | ~4GB |

char 长度固定为声明列时的长度

varchar 可变长字符串

text 只能用来保存字符数据

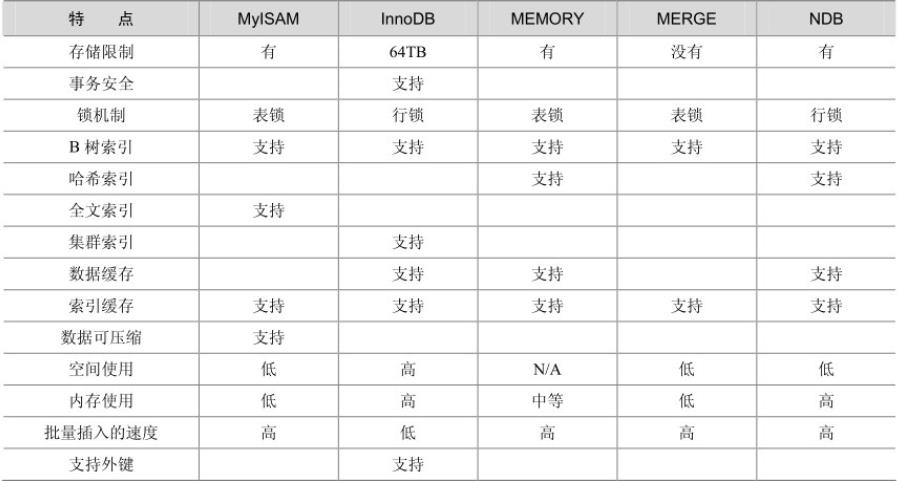
blob 能用来保存二进制数据（如图片）

OPTIMIZE TABLE进行碎片整理

BLOB和TEXT值会引起⼀些性能问题，特别是在执⾏了⼤量的删除操作时。

删除操作会在数据表中留下很⼤的“空洞”，以后填⼊这些“空洞”的记录在插⼊的性能上会有影响。为了提⾼性能，建议定期使⽤OPTIMIZE TABLE功能对这类表进⾏碎⽚整理，避免因为“空洞”导致性能问题。

# 存储引擎



【默认引擎】

MySQL5.5后默认为InnoDB

【查看数据库支持的引擎】

SHOW ENGINES\G;

【**定义表的存储引擎**】和字符集

CREATE TABLE table\_name ( … )

ENGINE=InnoDB CHARSET=gbk;

【修改表的存储引擎】

ALTER TABLE table\_name ENGINE=innodb;

## 【MyISAM】

不支持事务，不支持外键

优点是访问速度快，对事务没有完整性要求或者以SELECT、INSERT为主的应用可以选择MyISAM。

## 【InnoDB】

具有提交、回滚和崩溃恢复能力的事务安全。但比MyISAM引擎效率差，并且会占用更多的磁盘空间保留数据和引索。

自动增长列 AUTO\_INCREMENT

外键约束 foreign key

存储方式 共享表空间存储 多表空间存储

# 字符集

## 【常见字符集】

【字符集与校对规则】character 与 collation

字符集是字符串的编码方式，校对规则用于比较字符集是否相等。

字符集与校对规则是一对多的关系，每个字符集对应至少一种校对规则：

\_ci（大小写不敏感）

\_cs（大小写敏感）

\_bin（比较基于字符编码的值）

【字符集的设置】四个级别

没有指定时,默认使用上一级别的字符集和校对规则。

## 【查看MySQL可用的字符集】

SHOW CHARACTER SET;

SHOW CHARSET;

## 【查看某个字符集的校对规则】如gbk

SHOW COLLATION LIKE ‘gbk%’;

## 【查看字符集和校对规则】

服务器级和数据库级

SHOW VARIABLES LIKE ‘%character%’;

SHOW VARIABLES LIKE ‘collation%’;

## 【character变量解释】

character\_set\_client

character\_set\_connection

character\_set\_database

character\_set\_filesystem

character\_set\_results

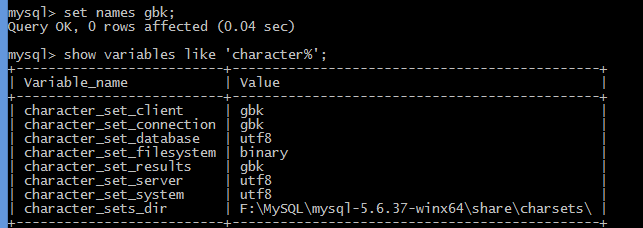
character\_set\_server

character\_set\_system

character\_sets\_dir

character\_set\_client、character\_set\_connection、character\_set\_results这3个参数值是由客户端每次连接进来设置的，和服务器端没关系。登陆进mysql后，执行 set names utf8; 实际就是同时修改上面的这3个参数值的。

当客户端连接服务器的时候，它会将自己想要的字符集名称发给mysql服务器，然后服务器就会使用这个字符集去设置这三个值。



character\_set\_server是服务器的字符集，haracter\_set\_database是当前所在的数据库字符集。如果没有use数据库，则character\_set\_database显示的和character\_set\_server一致。

character\_set\_filesystem、character\_set\_system、character\_sets\_dir三个系统变量不需要关心。

## 【服务器级】

【查看】

SHOW VARIABLES LIKE ‘character\_set\_server’;

SHOW VARIABLES LIKE ‘collation\_server’;

【修改】

在配置文件中指定：windows在my.ini中，重启MySQL

[mysqld]

character\_set\_server=utf8 #注意是下划线

[client]

default-character-set=utf8 #注意是横杆

## 【数据库级】

数据库的字符集和校对规则在创建数据库的时候指定，也可以在创建完数据库之后修改。

【查看】

SHOW VARIABLES LIKE ‘character\_set\_database’;

SHOW VARIABLES LIKE ‘collation\_database’;

SHOW CREATE DATABASE db\_name\G

\G后面不需要再加分号

【创建时设置】

CREATE DATABASE db\_name CHARACTER SET utf8;

【修改】

ALTER DATABASE db\_name default CHARACTER SET utf8;

不影响已有内容，只影响之后插入的内容（？）

## 【表级】

【查看】

SHOW CREATE TABLE table\_name\G

【创建】

推荐在创建表的时候明确指定字符集和校对规则，避免受默认值影响。

CREATE TABLE table\_name(

col\_name1 type1,

col\_name2 type2,

…)ENGINE=InnoDB CHATSET=gbk;

## 【字段级】

【查看】

SHOW FULL COLUMNS FROM table\_name\G

【修改】

ALTER TABLE table\_name CHANGE col\_name col\_name type

CHARACTER SET character\_name [COLLATE collation\_name];

# 索引

（个人理解是像单词书那样分类排列A~Z开头的单词）

索引用于快速找出一个列中有某特定值的行。

如果不使用索引，MySQL需要对整个表进行顺序查找，表越大，花费时间越多。如果表中有索引，MySQL就能快速到达一个中间位置开始搜索数据，没有必要搜索所有数据。

primary key、unique、index、fulltext在BTREE中存储。

# 视图view

一种虚拟存在的表。

视图并不在数据库中实际存在，行和列的数据来自于定义视图的查询中使用的表，并且是在使用视图时动态生成的。

【创建语句】

CREATE [OR REPLACE] [ALGORITHM = {UNDEFINED | MERGE | TEMPTABLE}]

VIEW view\_name [(column\_list)] AS

SELECT col\_name FROM table\_name WHERE condition

[WITH [CASCADED | LOCAL] CHECK OPTION]

例：

CREATE OR REPLACE VIEW staff\_list\_view AS

SELECT s.staff\_id,s.first\_name,s.last\_name,a.address

FROM staff AS s, address AS a

WHERE s.address\_id = a.address\_id ;

【修改语句】

ALTER [ALGORITHM = {UNDEFINED | MERGE | TEMPTABLE}]

VIEW view\_name [(column\_list)]

AS select\_statement

[WITH [CASCADED | LOCAL] CHECK OPTION]