启动mysql

net start mysql80

停止mysql

net stop mysql80

# mysql客户端连接

1、安装mysql成功后，自带的mysql命令行客户端

2、直接打开windows的命令行客户端键入指令：(需要配置环境变量PATH)

mysql [-h 127.0.0.1] [-P 3306] -u root -p //[-h 127.0.0.1] [-P 3306]可省略默认就是这个配置

mysql -u root -p

# DDL-数据库操作

查询

查询所有数据库

SHOW DATABASES;

查询当前数据库

SELECT DATABASE();

创建(mysql数据库中的字符集不建议使用utf8，因为utf8存储长度只有三个字节，可以用utf8mb4支持四个字节的)

CREATE DATABASE [IF NOT EXISTS] 数据库名 [DEFAULT CHARSET 字符编码] [COLLATE 排序规则];

删除

DROP DATABASE [IF EXISTS] 数据库名

使用

USE 数据库名

# DDL-表操作-查询

查询当前数据库所有表

SHOW TABLES;

查询表结果

DESC 表名；

查询指定的建表语句

SHOW CREATE TABLE 表名

# DDL-表操作-创建

CREATE TABLE 表名(

字段1 字段1类型 [COMMENT 字段1注释],

字段2 字段2类型 [COMMENT 字段2注释],

字段3 字段3类型 [COMMENT 字段3注释],

……

字段n 字段n类型 [COMMENT 字段n注释]

)[COMMENT 表注释];

# DDL-表操作-数据类型

mysql数据类型主要分为三类：数值类型、字符串类型、日期时间类型。

100.35 精度为5(整个数值长度) 标度为 2(小数位数)

例声明年龄字段  
 age tinyint unsigned 取值范围0-255

例声明分数字段,（分数保留1位小数） 最大100.0 精度4 标度1

score double(4,1)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 分类 | 类型 | 大小 | 有符号（SIGNED） 范围 | 无符号（UNSIGNED） 范围 | 描述 |
| 数值类型 | TINYINT | 1 byte | -128,127 | 0,255 | 小整数值 |
| SMALLINT | 2 bytes | -32768,32767 | 0,65535 | 大整数值 |
| MEDIUMINT | 3 bytes | -8388608,8388607 | 0,16777215 | 大整数值 |
| INT/INTEGER | 4 bytes | -21.47亿,21.47亿 | 0,4294967295 | 大整数值 |
| BIGINT | 8 bytes | … | … | 极大整数值 |
| FLOAT | 4 bytes | … | … | 单精度浮点数值 |
| DOUBLE | 8 bytes | … | … | 双精度浮点数值 |
| DECIMAL |  | 依赖于M(精度)和D(标度)的值 | 依赖于M(精度)和D(标度)的值 | 小数值(精确定点数) |

带blob是用来描述二进制数据(视频、音频、文件)的，不建议性能不高，存储在文件服务器

带text是用来描述文本数据的

char(10) 最大长度为10（存储的只有1个字符，占用空间也是10个字符，后面空格补全）

varchar(10) 最大长度为10（存储的只有1个字符，占用空间也是1个字符）

char 相对于 varchar性能更好

|  |  |  |  |
| --- | --- | --- | --- |
| 分类 | 类型 | 大小 | 描述 |
| 字符串类型 | 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个字符的二进制数据 |
| tinytext | 0-255 bytes | 短文本字符串 |
| blob | 0-65535bytes | 二进制形式的长文本数据 |
| text | 0-65535 bytes | 长文本数据 |
| mediumblob | 0-16777215 bytes | 二进制形式的中等长度文本数据 |
| mediumtext | 0-16777215 bytes | 中等长度文本数据 |
| longblob | 0-4294967295 bytes | 二进制形式的极大文本数据 |
| longtext | 0-4294967295 bytes | 极大文本数据 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 分类 | 类型 | 大小 | 范围 | 格式 | 描述 |
| 日期时间类型 | date | 3 | 1000-01-01 至9999-12-31 | YYYY-MM-DD | 日期值 |
| time | 3 | -838:59:59 至838:59:59 | HH:MM:SS | 时间值或持续时间 |
| year | 1 | 1901至2155 | YYYY | 年份值 |
| datetime | 8 | 1000-01-01 00:00:00至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 00:00:01至2038-01-19 03:14:07 | YYYY-MM-DD HH:MM:SS | 混合日期和时间值、时间戳 |

# DDL-表操作-修改

添加字段

alter table 表名 add 字段名 类型(长度) [comment 注释] [约束];

修改数据类型

alter table 表名 modify 字段名 新数据类型(长度);

修改字段名和字段类型

alter table 表名 change 旧字段名 新字段名 类型(长度) [comment 注释] [约束];

删除字段

alter table 表名 drop 字段名;

修改表名

alter table 表名 rename to 表名;

# DDL-表操作-删除

删除表

drop table [if exists] 表名;

删除指定表，并重新传建该表（清空表数据用）

truncate table 表名;

# MySql图形化界面工具

Sqlyog

Navicat

DataGrip

# DML

对数据的增删改查操作

添加数据 insert

修改数据 update

删除数据 delete

# DML-添加数据

给指定字段添加数据

insert into 表名(字段名1,字段名2,…) values (值1,值2,…);

给全部字段添加数据

insert into 表名 values(值1,值2,…);

批量添加数据

insert into 表名(字段名1,字段名2,…) values (值1,值2,…), (值1,值2,…), (值1,值2,…);

insert into 表名 values(值1,值2,…), (值1,值2,…), (值1,值2,…);

插入数据字符串和日期类型数据要用引号包裹

插入数据的大小要在规定范围内

# DML-修改数据

update 表名 set 字段名1=值1，字段名2=值2,… [where 条件];

修改语句可以不加where条件，不加的话就是修改表内所有数据

# DML-删除数据

delete from 表名 [where 条件];

delete语句可以不加where条件，不加的话就是删除表内所有数据

delete语句不可以删除某一个字段的值，可以使用update语句将某个字段设为null

# DQL-查询数据

select

字段列表

from

表名列表

where

条件列表

group by

分组字段列表

having

分组后条件列表

order by

排序字段列表

limit

分页参数

# DQL-基本查询

查询多个字段

select 字段1,字段2,… from 表名

select \* from 表名;

设置别名（可为中文） as可以省略

select字段1[as 别名1], 字段2[as 别名2],…from 表名;

去除重复记录

select distinct 字段列表 from 表名;

# DQL-条件查询

select 字段列表 form 表名 where 条件列表

条件：



sleect \* from user where name like ‘\_\_’; 查询姓名为两个字符的数据

sleect \* from user where name like ‘%三’;查询姓名最后一个字是三的数据



# DQL-聚合函数

select 聚合函数(字段) from 表名;

将一列数据作为一个整体进行纵向计算

常见聚合函数(null值不参与所有聚合函数的运算)



# DQL-分组查询

select 字段列表 form 表名 [where 条件] group by 分组字段名 [having 分组后过滤条件];

where与having区别

1. 执行时机不同

where是分组之前进行过滤，不满足where条件，不参与分组；

having是分组之后对结果进行过滤

1. 判读条件不同

where不能对聚合函数进行判断

having可以

执行顺序

where > 聚合函数 > having

分组之后，查询的字段一般为聚合函数和分组字段，查询其它字段无任何意义

# DQL-排序查询

select 字段列表 from 表名 order by 字段1 排序方式1,字段2 排序方式2;

排序方式

asc 升序 默认

desc 降序

如果是多字段排序，当地一个字段值相同时，才会根据第二个字段进行排序

# DQL-分页查询

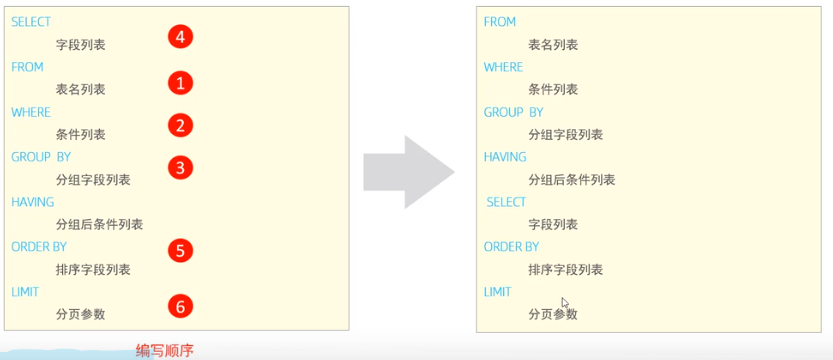
select 字段列表 form 表名 limit 起始索引,查询记录数;

起始索引从0开始，起始索引=(查询页面-1)\*每页显示记录数

分页查询是数据库的方言，不同数据库有不同的实现，mysql是limit

如果查询的是第一页数据，起始索引可以省略，直接简写为 limit10

# DQL-执行顺序



# DCL-管理用户



# DCL-权限控制



查询权限  
show grants for ‘用户名’@’主机名’;

授予权限

grant 权限列表 on 数据库名.表名 to ‘用户名’@’主机名’;

撤销权限

revoke权限列表 on 数据库名.表名 from ‘用户名’@’主机名’;

# sql查询拼接

SELECT concat('update xahj\_app\_module\_copy5 set event\_id ="',event\_id,'" where id =',id) FROM xahj\_app\_module\_copy5 where event\_id is not null;