**一、mysql数据库安装**

1、xampp安装（数据库安装）

2、sqlyog安装

3、sqlyog连接mysql数据库

我的mysql主机地址：mysql数据库所在电脑的IP地址；本机：localhost

用户名：root超级管理员，具有最高权限，可以管理普通用户（增加、修改、删除），给其他用户授权；

xampp默认密码：为空

端口：默认为mysql数据库的默认端口：3306

数据/库：连接mysql成功以后，可以看到的数据库，如果连接时指定具体库名，只可以该库，如果没有指定，可以看到全部数据库。

SQLyog简介：是一个第三方myql数据库连接工具，可以用来管理mysql数据库，可以创建数据库、表等对数据库对象，可以对表中的数据进行增加、修改、删除、查询等操作。

**二、数据库基础**

数据库：database，简称为DB

数据：data

base：基础，基地的意思

数据库：可以将数据库，看成一个小区，mysql可以看成小区的管理处，这个管理处独占一栋楼（管理处），

管理处中居住物业人员（物业经理root，普通物业人员普通用户），管理处所在楼（mysql）有双重属性，既是一个数据库，也是一个数据库管理系统，

类似物业可以管理整个小区（包括业主居住的楼（其他数据库））；

一栋楼：由不同的房间构成，一个个的房间就类似于该数据库的一个个表，表中存放数据，这些数据类似于房间中居住的人。

数据库系统：database system，简称DBS

数据库系统组成：硬件、操作系统、数据库、数据库管理员（DBA）

refresh：刷新

数据库管理员：database administrator，简称DBA，主要工作职责是开发、测试、运营、管理和维护数据库，保证数据库可以正常运行。

数据库管理系统：Database Management System，简称为DBMS，mysql就是一个数据库管理系统。

**三、cmd命令行模式连接mysql数据库以及数据库对象的查看**

1、cmd命令行模式下连接mysql数据库（适用于linux模式下）

语法：

mysql - u 用户名 -p

说明：

mysql.exe:是mysql客户端应用程序，存在mysql安装目录中的子目录bin中：C:\xampp\mysql\bin

mysqld.exe:是mysql的服务端程序，如果想连接mysql数据库，必须启动mysqld.exe 程序

- u：u为user的简写，该参数后可以跟用户名

-p：u为password的简写，该参数后可以跟具体密码

举例：root用户连接mysql数据库

mysql -u root -p

-------------------------介绍几个dos命令---------------------------------

dir:查看当前目录下有哪些子目录（后面带<dir>）和文件；

dir /w:横屏查看 当前目录下有哪些子目录（后面带<dir>）和文件；

dir /p：分页查看当前目录下有哪些子目录（后面带<dir>）和文件；

cd ..:退回到上级目录

-------------------------------------------------------------------------

2、 【连接上mysql数据库以后】，可以做如下事情：

2.1 查看有哪些数据库

show databases;

解释：show 为查看的意思

2.2 选择或者打开一个数据库

语法：use 数据库名；

举例：打开mysql数据库

use mysql；

-- 系统提示Database changed即表示sql语句执行成功

2.3 查看打开的数据库中有哪些表

语法：show tables;

解释：table为表的意思

举例：查看【打开的】mysql数据库中有哪些表

Show tables;

2.4 查看【打开的数据库中一个表的】表结构

语法1：

Desc 表名；

方法2：

Describe 表名；

-- 解释：describe为描述的意思，desc为简写形式

举例：查看mysql数据库中user表的表结构

Desc user;

Describe user;

------------------------怎么样写SQL语句才可以少出错-------------------

1)单词要写对；

2)单词都用小写；

3)语法要正确；

4)一个sql语句写完，需要以分号;结尾；

5)保证单词和符号都以英文模式下输入；

6)对于IP地址、日期时间、字符串（如各种名称）等，在insert语句中，这些数据需要用单引号或者双引号引起来；

--------------------------------------------------------------------------------------

2.5 查询user表中host（主机IP）,user（用户名）两个字段的值

Select host,user from mysql.user;

-- 解释：

Select:查询的意思

Host、user：字段名

From：后面跟库名. 表名，说明从哪个库的哪个表中查询数据；

From的意思是数据来自哪里。

2.6 退出数据库

语句1：quit（退出）

语句2：exit（退出）

1. **数据库操作**

创建数据库

语法：

Create database 库名 default character set 字符集名称 collate 排序规则；

举例：创建lianxi库，设置数据库的默认字符集为utf8,使该数据库可以支持中文，输入的字符不区分大小写；

Create database lianxi default character set utf8 collate utf8\_general\_ci;-- 创建库 指定名称，指定默认字符集再指定排序规则

-- 解释：

Create database：创建数据库

Lianxi：库名

default character set：默认字符集

Collate：校对

utf8\_general\_ci：排序规则，utf8可以支持中文的字符集，general通用的意思

Ci：character insensitive：大小写不敏感，不区分大小写

Cs：区分大小写

**重要提醒：**

**为数据库命名时**，名称可以使用大写字母A-Z，小写字母a-z,数字0-9，下划线\_,#、$号，字符长度不超过30个字符；可以使用中文，但不建议使用中文，不能使用纯数字；

---------------------------------------------------------------------------------------------------------------------

-- sql语句执行结果提示信息解释：

1 queries executed：1个查询北执行

Queries：为单词query的单词第三人称

Executed：是单词Execute的过去分词

1 success：1个（查询）成功

0 errors：0个错误

0 warnings：0个警告

---------------------------------------------------------------------------------------------------------------------

2、修改数据库

语法：

ALTER DATABASE 库名 CHARACTER SET 字符集名称 COLLATE 排序规则;

-- 举例：修改数据库lianxi的字符集为gb2312，排序规则为gb2312\_chinese\_ci,使该数据库可以支持中文，输入的sql语句不区分大小写

ALTER DATABASE lianxi CHARACTER SET gb2312 COLLATE gb2312\_chinese\_ci;

3、删除数据库

语法:DROP DATABASE 库名

举例：删除数据库lianxi1

DROP DATABASE lianxi1;

-- 解释：

DROP DATABASE:删除数据库

DROP:删除的意思

**五、数据库中表的操作**

**1、表创建**

**1.1 表的创建（重点）**

**语法：create TABLE 库名.表名(字段名1 数据类型1(数据长度1)**

**,字段名2 数据类型2(数据长度2))；**

**举例：在lianxi库中，创建学生表stu，**

**字段名包括：学号sno,数据类型为int,长度为5，姓名sname，数据类型为char,长度为20**

**CREATE TABLE lianxi.stu(sno INT(5),sname CHAR(20));**

**举例：在lianxi库中，创建表stu1，字段名包括：学号sno,数据类型为int,长度为5，姓名sname，数据类型为char,长度为20**

**将字段sno设置为主键**

**CREATE TABLE lianxi.stu1(sno INT(5) PRIMARY KEY,sname CHAR(20));**

**-- 解释：**

**PRIMARY KEY 为主键的意思，意思是强制使该表中的数据不能重复。**

**PRIMARY：主要的意思，key关键字的意思，简称键**

重要提醒：

**为表命名时**，名称可以使用大写字母A-Z，小写字母a-z,数字0-9，下划线\_,#、$号，字符长度不超过30个字符；可以使用中文，但不建议使用中文，表名不能使用纯数字；

**举例：在lianxi库中，创建班级表class，字段名包括：班号cno,数据类型为int,长度为6，**

**班级名cname，数据类型为char,长度为15，将字段cno设置为主键**

**CREATE TABLE lianxi.class(cno INT(6) PRIMARY KEY,cname CHAR(15));**

**举例：在lianxi库中，创建表stu2，字段名包括：学号sno,数据类型为int,长度为5，**

**姓名sname，数据类型为char,长度为20，班号cno,数据类型为int,长度为6；将字段sno设置为主键，将字段cno**

**设置为外键**

**CREATE TABLE lianxi.`stu2`**

**(sno INT(5) PRIMARY KEY,**

**sname CHAR(20),**

**cno INT(6),FOREIGN KEY(cno) REFERENCES lianxi.class(cno));**

**-- 说明：stu2表的外键字段cno的数据，来源于class表的主键字段cno的数据**

**-- FOREIGN KEY(cno)：将cno字段设置为外键**

**-- FOREIGN：外部的**

**-- REFERENCES：参考，引用的意思**

**-- REFERENCES lianxi.class(cno)：数据引用（来源于）lianxi库class中cno字段的数据**

**CREATE TABLE employees(hire\_date DATE); -- 创建表**

**INSERT INTO lianxi.`employees`(hire\_date) VALUES("2018-11-1"); -- 插入时间数据**

**CREATE TABLE employees1(sex ENUM("M","F")); -- 创建表**

**INSERT INTO lianxi.`employees1` VALUES("女"); -- 错误写法**

**INSERT INTO lianxi.`employees1` VALUES("F"); -- 插入性别数据**

**SHOW WARNINGS; -- 查看告警信息**

**INSERT INTO lianxi.`employees1` VALUES("M"); -- 插入性别数据**

**1.2 表的约束**

**1.2.1 唯一键 （unique）：**

**将表中的某个字段定义为唯一键，意味着该字段中的数据不允许重复**

**1.2.1 主键（primary key 简称pk）**

**表中北定义为主键的字段，对应数据不允许重复（具有唯一性），不允许为空（not null）；一个表中只能有一个主键，主键可以是由一个字段构成，也可以是由两个或者两个以上字段构成；**

**1.2.2 外键（foreign key，简称fk）**

**一个表的外键，用来关联其他表，一个表的外键通常关联其他表的主键或者唯一键，一个表中的外键可以有多个，外键字段的数据可以为空；**

**1.2.3 查看一个表的键值**

**Show keys from 库名.表名；**

**举例：**

**查看hrdb库员工表（employees）表的外键有哪些**

**Show keys from hrdb.employees;**

**1.3 表的数据类型**

**1.4表的修改**

**1.4.1 表的名称修改**

**举例：修改hr库中sales表名称为sale?**

**ALTER TABLE hr.`sales`**

**RENAME TO sale;**

**-- 语法：**

**ALTER TABLE 库名.表名**

**RENAME TO 新表名;**

**1.4.2 修改表的字段或者约束**

**1.4.2.1 为表添加主键约束**

**举例： 为表sales 增加主键约束sales\_customer\_no\_pk**

**ALTER TABLE hr.`sales`**

**ADD CONSTRAINT sales\_customer\_no\_pk PRIMARY KEY(customer\_no);**

**-- 解释：**

**-- ALTER TABLE：修改表的意思，ALTER为修改或者改变的意思**

**-- ADD CONSTRAINT：添加主键约束，CONSTRAINT为约束的意思**

**-- sales\_customer\_no\_pk：主键约束名称，构成：表名+字段名+键类型**

**-- PRIMARY KEY(customer\_no)：意思是将字段customer\_no设为主键**

**-- 语法：**

**ALTER TABLE 库名.表名**

**ADD CONSTRAINT 主键名 PRIMARY KEY(字段名);**

**举例：修改sales表约束，使姓名customer\_no不允许为空**

**ALTER TABLE hr.sale**

**MODIFY customer\_name CHAR(25) NOT NULL;**

**-- 语法：**

**ALTER TABLE 库名.表名**

**MODIFY 字段名 数据类型(数据长度) NOT NULL;**

**-- 解释： MODIFY为修改的意思**

**举例：修改表sale，将字段`pro\_name`名称修改为province\_name**

**ALTER TABLE hr.`sale`**

**CHANGE pro\_name province\_name CHAR(40);**

**-- 语法：**

**ALTER TABLE 库名.表名**

**CHANGE 原字段名 新字段名 字段类型(数据长度);**

**-- 解释：change为更换的意思，province为省份的意思**

**举例：为表sale添加字段product\_name,字段类型为varchar，长度为30（重点）**

**ALTER TABLE hr.`sale`**

**ADD COLUMN product\_name VARCHAR(30);**

**-- 语法：**

**ALTER TABLE 库名.表名**

**ADD COLUMN 字段名 字段类型(字段长度);**

**-- 解释：**

**ADD COLUMN：增加列的意思，column为列，字段的意思**

**举例：修改sale表中字段`province\_name`的字段类型为varchar(30)**

**ALTER TABLE hr.`sale`**

**MODIFY province\_name VARCHAR(30);**

**-- 语法：**

**ALTER TABLE 库名.表名**

**MODIFY `字段名` 新数据类型(新数据长度);**

**举例：删除表sale中的字段product\_name （重点）**

**ALTER TABLE hr.`sale`**

**DROP COLUMN product\_name;**

**-- 解释：DROP COLUMN product\_name，意思是删除字段product\_name**

**-- DROP：删除的意思**

**-- COLUMN：字段，列的意思**

**举例：为部门表添加外键，先创建两个表：**

**-- 在hr库中，创建学生表 student(学生编号sno int(5),该字段定义为主键,**

**-- 学生名称sname char(30),班级编号classno int（6）)**

**CREATE TABLE hr.student**

**(**

**sno INT(5) PRIMARY KEY,**

**sname CHAR(30),**

**classno INT(6) -- 班级编号**

**);**

**-- 举例：创建表班级表class（班号cno int(6),班级名称 cname char(25),系名depno int（4） ）**

**CREATE TABLE hr.class**

**(**

**cno INT(6), -- 班级编号**

**cname CHAR(25),**

**depno INT(4)**

**);**

**举例：将class表的cno字段修改为主键**

**ALTER TABLE hr.`class`**

**ADD CONSTRAINT class\_cno\_pk PRIMARY KEY(cno); -- 将字段cno修改为主键**

**举例：为hr库中表student的字段classno添加外键，该外键字段的数据引用自表class的主键字段cno**

**ALTER TABLE hr.`student`**

**ADD CONSTRAINT student\_classno\_fk FOREIGN KEY(classno)**

**REFERENCES hr.`class`(cno);**

**-- 解释：**

**-- ADD CONSTRAINT：添加约束的意思**

**-- student\_classno\_fk：外键约束的名称**

**-- FOREIGN KEY(classno)：将字段classno设置为外键**

**-- REFERENCES hr.`class`(cno)：(数据）引用自hr库的class表的主键cno字段**

**-- REFERENCES：引用，参考的意思**

**举例：删除hr库student表的外键 student\_classno\_fk**

**说明：删除外键，分为两步：**

**步骤一：删除外键**

**ALTER TABLE hr.`student`**

**DROP FOREIGN KEY student\_classno\_fk;**

**-- 解释：DROP FOREIGN KEY 删除外键，drop是删除的意思，FOREIGN KEY外键的意思**

**-- student\_classno\_fk :外键约束的名称**

**步骤二：删除外键索引**

**DROP INDEX student\_classno\_fk ON hr.`student`;**

**-- 解释：**

**-- DROP INDEX：删除索引**

**-- student\_classno\_fk：索引名称（外键名称）**

**-- ON hr.`student`：通过关键字ON指明是删除hr库中学生表的索引表的删除**

**1.5 表结构的删除**

**语法：**

**Drop table 库名.表名；**

**举例：**

**删除hr库中的表sale**

**Drop table hr.sale;**

**六、表中数据的操作(重点)**

**1、往表中插入数据(重点)**

**-- 语法：**

**INSERT INTO 库名.表名(字段名1，字段名2) VALUES(字段值1，字段值2);**

**举例：往表stu1中插入一条数据，编号为01，姓名为张三**

**INSERT INTO lianxi.`stu1`(sno,sname) VALUES(01,"张三");**

**INSERT INTO lianxi.`stu1`(sno,sname) VALUES(02,"李四");**

**-- INSERT插入，添加的意思**

**-- INTO：往什么里面**

**-- VALUES：值的意思**

**2、修改表中的数据（重点）**

**语法：**

**UPDATE 库名.表名**

**SET 字段名1=新字段值1**

**WHERE 字段名2=字段名2；**

**-- 解释：**

**update修改的意思，SET设置的意思，where条件的意思**

**举例：修改姓名为李四的人的编号由2修改为3**

**UPDATE lianxi.`stu1`**

**SET sno=3**

**WHERE sname="李四";**

**3、删除表中的数据(重点)**

**-- 方法一：**

**语法：delete FROM 库名.表名 WHERE 字段名1=字段值1；**

**-- 说明：**

**-- delete 删除的意思**

**-- FROM 库名.表名：删除哪个表中的数据**

**-- WHERE 字段名1=字段值1：删除字段名1的值为字段值1对应的数据（对应的行）**

**举例：删除表stu1中编号为1的数据**

**DELETE FROM lianxi.stu1**

**WHERE sno=1;**

**举例：创建一个表，该表的表结构和数据均复制于表employees**

**CREATE TABLE hrdb.emp1 SELECT \* FROM hrdb.`employees`; -- 复制表(复制表结构+数据)**

**举例：删除表emp中的所有数据**

**DELETE FROM hrdb.emp; -- 删除数据时，如果不指定条件，默认删除表中的所有数据；**

**方法二：**

**语法：**

**TRUNCATE 库名.表名; -- 清空指定表中数据，TRUNCATE不能和from、where配合使用**

**举例：删除表emp1中的所有数据**

**TRUNCATE hrdb.emp1;**

**4、查询表中的数据(重中之重)**

**4.1 查询单个表中的数据**

**语法：**

**SELECT 查询列表 FROM 库名.表名**

**WHERE 字段名1 比较运算符 字段值1 逻辑运算符 字段名2 比较运算符 字段值2；**

**-- 解释：**

**-- 查询列表包括：**

**-- \* 代表查询所有字段的值**

**-- 单个字段：查询所有行对应该字段的值**

**-- 逗号隔开的多个字段：查询所有行对应的相应字段的值**

**-- 比较运算符：等于号=，大于号>,小于号<,不等于号!=、<>,大于等于>= 、小于等于<=**

**-- 逻辑运算符：not 非，and 与，or 或**

**-- not 表达式：取表达式计算结果相反的数据**

**条件1 AND 条件2：返回两个条件同时成立的数据**

**-- 条件1 or 条件2：返回符合任意一个条件的数据**

**举例：查询员工表（employees）中所有人员的所有信息**

**SELECT \* FROM hrdb.`employees`;**

**举例：查询employees表中所有人员的姓、名**

**SELECT first\_name,last\_name FROM hrdb.`employees`;**

**举例：员工表中员工编号为101的员工的员工编号、工资**

**SELECT employee\_id,salary**

**FROM hrdb.`employees`**

**WHERE employee\_id=101;**

**### 大于号>、小于号<、不等于号!=\<>、大于等于号>=、小于等于号<=的用法举例**

**举例：查询 工资大于9000的人员的姓名、工资**

**SELECT first\_name,last\_name,salary**

**FROM HRDB.`employees`**

**WHERE salary>9000;**

**举例：查询工资大于等于9000的人员的工资、姓、部门编号**

**SELECT salary,last\_name,department\_id**

**FROM hrdb.`employees`**

**WHERE salary>=9000;**

**举例：查询工资不等于9000的人员的工资、姓、部门编号**

**方法一：**

**SELECT salary,last\_name,department\_id**

**FROM hrdb.`employees`**

**WHERE salary!=9000;**

**方法二：**

**SELECT salary,last\_name,department\_id**

**FROM hrdb.`employees`**

**WHERE salary<>9000;**

**#### not（非）、and（与）、or（或）的用法**

**方法三：**

**SELECT salary,last\_name,department\_id**

**FROM hrdb.`employees`**

**WHERE NOT salary=9000; -- not是取反的意思**

**举例：查询工资大于9000，部门编号小于102的人员的员工编号、入职日期、工资、部门编号**

**SELECT employee\_id,hire\_date,salary,department\_id**

**FROM hrdb.`employees`**

**WHERE salary>9000 AND department\_id<102; -- and 连接两个查询条件**

**举例：查询员工表中部门编号分别为60 和90 的人员的工资、部门编号、员工编号**

**SELECT salary,department\_id,employee\_id**

**FROM hrdb.`employees`**

**WHERE department\_id=60 OR department\_id=90;**

**-- or 连接两个查询条件，符合任何一个条件的数据都将被显示在查询结果中**

**#### 关键字in在查询语句中的用法**

**举例：查询工资分别为9000、10000、11000、12000的员工编号、工资**

**SELECT employee\_id,salary**

**FROM hrdb.`employees`**

**WHERE salary IN(9000,10000,11000,12000);**

**-- in后跟数据列表，匹配其中任何一条数据的记录都将显示在查询结果中**

**####关键字between...and...的用法**

**-- between 值1 and 值2，意思是在值1和值2之间**

**举例：查询工资在3000-6000之间的人员的工资、员工编号、职位编号**

**SELECT salary,employee\_id,job\_id**

**FROM hrdb.`employees`**

**WHERE salary BETWEEN 3000 AND 6000;**

**#### null关键字在查询语句中的用法**

**-- 说明：is null代表为空的意思， is not null为非空的意思**

**举例：查询部门表中经理编号为空的部门的编号、部门名称、经理编号**

**SELECT department\_id,department\_name,manager\_id**

**FROM hrdb.`departments`**

**WHERE manager\_id IS NULL;**

**错误写法：**

**SELECT department\_id,department\_name,manager\_id**

**FROM hrdb.`departments`**

**WHERE manager\_id = NULL;**

**举例：查询部门表中经理编号不为空的部门的编号、部门名称、经理编号**

**SELECT department\_id,department\_name,manager\_id**

**FROM hrdb.`departments`**

**WHERE manager\_id IS NOT NULL;**

**####聚合函数的用法**

**-- 说明：分组函数主要包括：求最大值max()函数、最小值min()函数、**

**-- 平均值avg()函数、求和sum()函数**

**-- 统计条数count()函数。**

**举例：查询员工表中的最高工资、最低工资、平均工资、工资总和**

**SELECT MAX(salary) 最高工资,MIN(salary) 最低工资,AVG(salary) 平均工资,SUM(salary) 和**

**FROM hrdb.`employees`;**

**-- 说明：最高工资、最低工资、平均工资、和为对应函数的别名**

**举例：查询员工表中的所有人员的工资之和以及人员的数量**

**SELECT SUM(salary) 总工资,COUNT(\*) 总人数 FROM hrdb.`employees`;**

**-- \*号代表所有记录**