Day1

1.访问服务器

1. 通过**域名或IP地址**
2. 每一个服务都有对应的**端口号**
3. 提供访问该服务所使用的**协议**

2.访问自己电脑创建的web服务器

启动web服务器 /webserver/start.cmd

<http://127.0.0.1:80> (本地服务器)

<http://localhost:80>

http:// 176.226.107.37(IP地址) 回车

Day2

1.保存数据的几种形式

内存 临时保存数据

文件 不便于管理

第三方机构 腾讯云、百度云……………..

独立数据库服务器

2.数据库概述

Database:数据库，以特定的结构批量存储数据的软件

数据库的发展史

网状数据库->树状数据库->**关系型数据库**->非关系型数据库

关系型数据库中数据的逻辑结构

Server->Database->Table->Row->Column

3.MySQL

Xampp是一个服务器套装，包括多款服务器软件。

Mysqld.exe 启动MySQL服务器

4.MySQL数据库系统

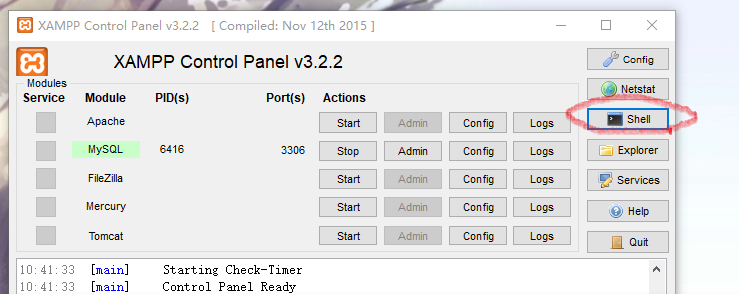
(1)服务器端

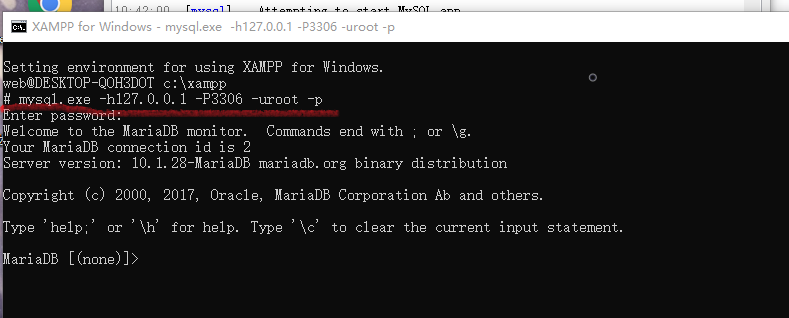
负责存储数据

(2)客户端

负责向服务器发起增删改查的命令

1. 链接MySQL数据库服务器





-h 服务器的IP地址/域名 可以使用localhost

-P Port 端口 MySQL服务使用3306端口

-u 用户名

-p 密码

简写



(5)MySQL管理命令

quit; 退出服务器链接

show databases; 显示数据库服务器上当前所有的数据服务

use 数据库名; 进入指定的数据库

show tables; 显示当前数据库中所有的数据表

desc 表名; 描述表中有哪些列

6.MySQL常用的SQL命令

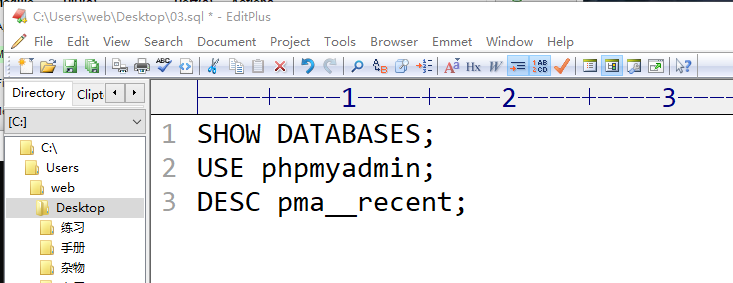
SQL:Sructured Qery Lnguage，结构化查询语言，用于操作关系型数据库服务器中的数据——增删改差

SQL语言最早是有IBM提出，后交给ISO，成为行业内标准语言。分为多个版本：SQL-87、SQL-92、SQL-99.。当前标准的SQL语言背绝大多数的关系型数据库所支持。

SQL命令的两种执行方式

1. 交互模式：客户端连接MySQL，输入一行，回车，服务器执行一行。适用于临时性的查看数据
2. 脚本模式：客户端把要执行的多行命令卸载一个脚本文件中，一次性的提交给服务器执行。适用于批量的操作数据。mysql –uroot <C:/xampp/…/02.sql

SQL语法规范

1. 每条SQL语句必须以英文分号作为结尾，一条语句可以跨越多行，见到分号认为结束。
2. 若第n条SQL语句有语法错误，则后续所有的语句不会再执行。
3. 在SQL语句中不区分大小写。习惯上数据库关键字用大写非关键字用小写
4. SQL语句中还可以使用单行注释：#......或多行注释/\*…..\*/，注释的内容不会被服务器所执行。

日常开发中常用的SQL命令

1. 丢弃数据库，如果存在的话

DROP DATABASE IF EXISTS jd;

1. 创建新的数据库

CREATE DATABASE jd;

1. 进入刚刚创建的数据库

USE jd;

1. 创建数据表

CTEATE TABLE student(

sid INT,

name VARCHAR(6),

sex VARCHAR(1),

score INT

);

1. 向表中插入记录

INSERT INTO student VALUES

(‘10’,’Tom’,’M’,’89’),(‘11’,’Kate’,’F’,’92’)………..

1. 查询数据所有记录

SELECT \* FROM student;

Day3

1.MySQL中的常用命令

增删改查

更改记录

UPDATE user SET;

删除记录

DELETE FROM user WHERE uid=’3’;

2.数据库中的中文乱码

计算机中如何存储英文字符

ASCII:总共128个，对所有的英文字符及符号进行了编码。

Latin-1:总共有256个，兼容ASCII，同事对欧洲符号进行了编码。

计算机中如何存储中文字符

GB2312:对6千多常用的汉子进行了编码，兼容ASCII码。

GBK:对2万多汉字进行了编码，兼容GB2312。

BIG5:台湾繁体字编码，不兼容GB2312和GBK，兼容ASCII码。

Unicode:对世界上主要的语言的字符进行编码，兼容ASCII码，不兼容GBK,GB2312,BIG5等。具体分为UTF-8,UTF-16,UTF-32三种存储方案。

中文乱码产生的根源

MySQL默认使用Latin-1编码，而这个编码不能存储中文的。

解决MySQL存储中文乱码

1. sql脚本文件另存为的编码
2. 客户端连接服务器使用的编码
3. 服务器端数据库存储数据使用的编码

都统一使用UTF8

设置editplus的默认编码：

Tools->preferences->file->Default encoding

3.MySQL中列类型

列类型：创建表的时候，指定列可以存储的数据类型

CREATE TABLE book(bid 列类型);

1. 数值类型

TINYINT：微整型，占用1个字节

SMALLINT：小整型，占用2个字节

INT：整型，占用4个字节

BIGINT：大整型，占用8个字节

FLOAT：单精度浮点数，占用4个字节，范围比INT大的多，但是精度不如INT，可能产生误差

DOUBLE：双精度浮点型，占用8个字节，范围比BIGINT大得多，但是精度不如BIGINT，可能产生误差

DECIMAL(M,D)：定点小数，不会产生计算误差舍入，M代表总的有效位数（不包括小数点），D代表小数点后有效位数

BOOL：布尔型，取值只能是TRUE/1、FALSE/0，数据库在储存的时候使用TINYINT，TRUE和FALSE不能加引号

1. 字符串类型

VARCHAR(M)：变长字符串，不会产生空间浪费，操作速度相对慢

CHAR(M)：定长字符串，可能产生空间浪费，操作速度相对快

TEXT(M)：大型变长字符串

1. 日期时间类型

DATETIME：日期时间类型

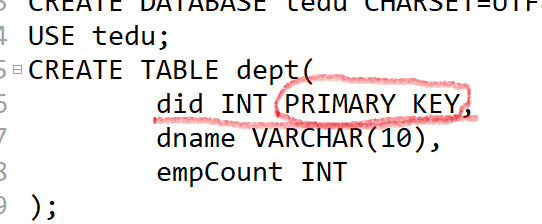
DATE：日期类型

TIME：时间类型

4.列约束

MySQL可以对插入的数据进行特定的检验，只有满足条件的数据才能插入成功，否则会提示“非法插入”，“禁止插入”

1. 主键约束——PRIMARY KEY



声明了主键约束的列上不能插入重复的值，如果主键的数据是数值会自动按照由小到大的顺序排列——加快查找速度。注意：一个表中最多有一个主键。

MySQL中的NULL含义：空的，空缺的；指应该有某个数据但暂时还不能确定具体的值。

Day4

1. 唯一约束——UNIQUE

声明了唯一约束的列上不能插入重复的值，但是可以插入NULL，而且可以插入多个NULL

1. 非空约束——NOT NULL

声明了非空约束的列上不能插入NULL值，如果插入的是NULL也不会报错，会显示空内容

1. 默认值约束——DEFAULT

可以使用DEFAULT关键字为列声明默认值；有以下两种用法：

INSERT INTO xz\_laptop VALUES(‘1’,’小米air’,’**DEFAULT**’…….);

INSERT INTO xz\_laptop(lid,title)VALUES(‘2’,’苹果pro’);

1. 检查约束

检查约束可以对插入的数据范围进行验证

CREATE TABLE student(

age TINYINT **CHECK(age>18 AND age<60)**,

);

MySQL不支持，会降低插入和更新等操作效率

1. 外键约束——FOREIGN KEY

声明了外键约束的列，取值必须在另一个表的主键上出现过，取值可以是NULL

#**familyId**取值范围必须在**xz\_laptop\_family**表中**fid**出现过的值

FOREIGN KEY(**familyId**) REFERENCES **xz\_laptop\_family**(**fid**);

注意：外键的列类型要和另外一个表主键的列类型必须一致。

所有的列约束中，主键约束推荐使用—可以提高查询的效率；其他的列约束师傅使用根据项目而定，可以不用—会影响数据的插入和更新效率。

MySQL中的自增列

AUTO\_INCREMENT:自动增加，假如一个列声明了自增列，无需手动赋值，直接指定为NULL，会自动获取当前的最大值，新记录会执行+1然后插入。

注意：

允许手动赋值

只用于整数型的主键列上

3.项目中如何存储日期时间数据

存储的是距离计算机元年的毫秒数。使用BIGINT类型。

计算机元年 1970-1-1 00:00:00

1秒钟=1000毫秒

4.简单查询

(1)查询所有的列

**SELECT \* FROM** emp;

(2)查询特定的列

SELECT **ename,birthday,salary** FROM emp;

(3)给列取别名

SELECT ename **AS** 姓名,salary **AS** 工资 FROM emp;

AS关键字可以省略

(4)只显示不同的值/合并相同的项

SELECT **DISTINCT** sex 性别 FROM emp;

(5)查询时执行计算

示例：计算5+3-7/2\*9

SELECT 5+3-7/2\*9;

(6)查询的结果排序

示例：查询所有员工的信息，结果按照工资由低到高排序

SELECT \* FROM emp **ORDER BY** salary **ASC**; ASC=ascend

示例：查询所有员工的信息，结果按照工资由高到低排序

SELECT \* FROM emp **ORDER BY** salary **DESC**; DESC=descend

如果不加排序规则，默认是按照升序排列(asc)

ORDER BY 可以按照**数值、日期、字符串**（字符的编码）来排序

(7)条件查询

示例：查询编号为5的员工工资

SELECT \* FROM emp **WHERE** eid=5;

SQL语句中支持的比较运算符：=、!=、>、<、>=、<=。

注意：NULL不能和任何值进行等于/不等于判定，包括NULL自己。可以使用IS NULL和IS NOT NULL。

**AND**(并且) **OR**(或者)

**BETWEEN** … **AND** … >=第一个值 <=第二个值

**NOT BETWEEN** … **AND** …

SELECT \* FROM emp WHERE deptid **IN**(10,20);

SELECT \* FROM emp WHERE deptid **NOT** **IN**(10,20);

(8)模糊条件查询

示例：查询出姓名中含有字母e的员工信息

SELECT \* FROM emp WHERE ename **LIKE** ‘**%**e**%**’; **%** 任意个数的字符 >=0

SELECT \* FROM emp WHERE ename **LIKE** ‘**%**e’; 倒数第一个字符是e

SELECT \* FROM emp WHERE ename **LIKE** ‘**%**e\_’; 倒数第二个字符是e

**\_** 任意一个字符 =1

上述两个符号不能和=连用，必须使用LIKE

(9)分页查询

分页显示：假如查询的结果集中有太多的记录，一次显示不完，可以分多页显示。

开始=(页码-1)\*每页大小

SELECT \* FROM emp **LIMIT** start,count

start 开始的值

count 查询的数量

示例：每页大小是5，查询第1页的数据

SELECT \* FROM emp **LIMIT** 0,5;

注意：LIMIT后边的两个数值不能添加引号

Day5

标准SQL语句分类

(1)**DDL:Data Define Language 定义数据**

CREATE/DROP/ALTER(修改)

(2)**DML:Data Manipulate Language 操作数据**

INSERT/DELETE/UPDATE

(3)**DQL:Data Query Languange 查询数据**

SELECT

(4)**DCL:Data Control Languange 控制用户权限**

GRANT(授权)/REVOKE(收权)

1.复杂查询

(1)聚合查询

示例：查询出所有员工的数量

SELECT **COUNT**(eid) FROM emp;

SELECT COUNT(\*) FROM emp; 推荐写法 可以防止NULL

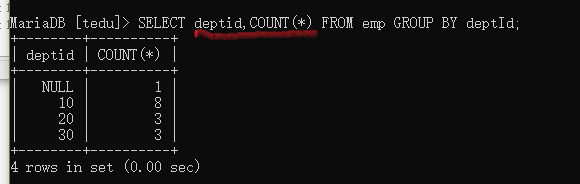
聚合函数

函数就是一个功能体，提供数据，产出结果。

**COUNT(...)/SUM(...)/AVG(...)（平均数）/MAX(...)/MIN(...)**

(2)分组查询

分组条件只能查询分组条件和聚合函数



示例：查询出每个部门员工数量

SELECT COUNT(\*) FROM emp **GROUP BY** deptId

函数补充：

YEAR(...) 获取日期中的年份

MONTH(...) 获取日期中的月份

DAY(...) 获取日期中的日

示例：查询出1991年出生员工的所有信息

SELECT \* FROM emp WHERE YEAR(birthday)=1991

(2)子查询

**把一个SQL语句的查询结果作为另外一个SQL语句的查询条件**

示例：查询出研发部所有的员工信息

步骤1：查询出研发部的部门编号——10

SELECT did FROM dept WHERE dname='研发部';

步骤2：根据研发部的部门编号查询员工信息

SELECT \* FROM emp WHERE deptid=10;

综合：

SELECT \* FROM emp WHERE deptid=(

SELECT did FROM dept WHERE dname='研发部'

);

(3)多表查询

示例：查询出所有员工的姓名及其部门名称

SELECT ename,dname FROM emp,dept;

错误：产生笛卡尔积！

多表查询如何避免产生笛卡尔积，**添加查询条件**即可

SELECT ename,dname FROM emp,dept **WHERE deptid=did**;

上述多表查询语法是SQL-92中，无法查询出没有部门的员工，也无法查询出没有员工的部门

SQL-99中提出了新的多表查询方法

(1)内连接 INNER JOIN...ON... 和SQL92结果一致

SELECT ename,dname FROM emp **INNER JOIN** dept **ON** deptid=did;

(2)左外连接LEFT OUTER JOIN...ON...

SELECT ename,dname FROM emp **LEFT OUTER JOIN** dept **ON** deptid=did;

查询结果是左侧所有的记录都显示，OUTER可以省略

(3)右外连接RIGHT OUTER JOIN...ON...

SELECT ename,dname FROM emp **RIGHT OUTER JOIN** dept **ON** deptid=did;

查询结果是显示右侧所有的记录都显示，OUTER可以省略

(4)全连接FULL JOIN

显示左侧和右侧所有的记录——MySQL不支持

UNION 合并相同的项

UNION ALL 不合并相同的项

(SELECT ename,dname FROM emp LEFT OUTER JOIN dept ON deptid=did)**UNION**(SELECT ename,dname FROM emp RIGHT OUTER JOIN dept ON deptid=did);