1、标准SQL命令分类

DDL：Data Define Language 定义数据结构

CREATE/DROP/ALTER(修改表的结构)

DML:Data Manipulate Language 操作数据

INSERT/UPDATE/DELETE

DQL:Data Query Language 查询数据

SELECT

DCL:Data Control Lagnguage 控制用户权限

GRANT(授权)/REVOKE(收权)

2、计算机中存储字符

（1）如何存储英文字符

ASCII :码总共有128个，对所有英文字母和符合进行编码。

abc 9798999

Latin-1(拉丁)：总共有256，兼容ASCII码，同时对欧洲符号进行编码，mysql默认使用这种编码。

（2）如何存储中文字符

GB2312: (国标)对常用6千多汉字进行编码，兼容ASCII码

GBK: 对两万多汉字进行编码，兼容GB2312

BIG5：台湾繁体字编码，兼容ASCII码

Unicode:对世界上主流国家常用的语言进行了编码，兼容ASCII码，不兼容GB2312、GBK、BIG5.具体分为UTF-8、UTF-16、UTF-32存储方案。

（3）、解决mysql存储中文乱码

使用UTF-8编码形式

SET NAMES UTF8;

sql脚本文件另存为的编码

客户端连接服务器的编码（SET NAMES UTF8）;

服务器端创建数据库使用的编码（CHARSET=UTF8）;

编写脚本文件01-dang.sql，创建数据库dang,设置编码格式Utf8；创建保存图书的数据库Book，包含bid,title标题，author作者，price价格，publish出版社，pubtime出版时间,插入四条数据

2、mysql中的列类型

创建数据表的时候，指定的列可以存储的数据类型

CREAT TABLE book(bid 列类型)；

（1）数值类型（引号可加，可不加）

TINYINT (微整形，占一个字节,范围-128-127)

SMALLINT(小整形,占两个字节，范围-32768-32767)

INT (整形，占四个字节,范围-2147483647-2147483647)

BIGINT（大整形，占八个字节）

FLOAT（M,D） (单精度浮点型，占四个字节，范围3.4\*10^38,比INT大的多，可能产生计算误差)

DOUBLE(M,D)(双精度浮点型，占八个字节，范围比BLGINT大得多.代表总的有效位数不包含小数点，D代表小数点后的有效位数)

DECIMAL（M,D）(定点小数，不会产生计算误差)

BOOL(布尔型，存储只有两个结果TRUE、FALSE（不能加引号）,真正存储数据的时候，会自动变成1和0，也可以直接使用1和0。数据库中的列类型自动变成TINYINT .)

（2）日期时间类型（必须加引号）

DATE 日期型‘2018-12-31’

TIME 时间型 ‘14:37:30’

DATETIME 日期时间型 ‘2018-12-31 14:37:30’

（3）字符串类型（必须加引号）

VARCHAR（M） （变长字符串，需要设置长度，不会产生空间浪费，操作速度相对较慢，M最大值65535）

CHAR(M) (定长字符串字符长度设定多少，则字符就为设定值，不会产生空间浪费，操作速度较快，常用于存储手机号，省份证等固长度字符，M最大值255)

TEXT(M) (大型变长字符串，M最多存2G)

1TB->1024GB->1024MB->1024KB->1024BYTE(字节)->8BIT(位)

1TB=1024GB 1GB=1024MB 1MB=1024KB

1KB=1024BYTE 1BYTE=8BIT

编写脚本文件03\_tedu.sql传建数据库tedu，设置编码为Utf8，进入该数据库，创建保存部门的数据表dept,包含did,dname部门名称，empcount部门员工数量，插入以下数据10 研发部 3,20 市场部 2,30 运营部 2

创建保存员工数据的表emp,包含eid,ename,sex,brirthday生日，salary工资，deptld所属部门编号

3、列约束

MYSQL 可以对插入的数据进行特定的验证，只有满足条件才允许插入到数据表中，否则认为是非法的插入。

列入一个人的性别只能是男或女，一个人的年龄0-100

（1）主键约束——PRIMARY KEY（主键约束的列上不能出现重复的值，放在设定值后面，他会按照从小到大顺序自动排列，加快查找速度，通常主键添加到编号列中，不允许存在多个主键，只能给某一列用）主键上不允许插入NULL.

NULL(表示空，再插入数据时，无法确定要保存的数值。列入:无法确定员工的工资，姓名…)

练习

编写脚本文件xz.sql，创建数据库xz，设置编码格式utf8，创建保存笔记本家族分类的表laptop\_family(fid,fname分类名称，laptopcount笔记本数量)10 联想 2 ，20 戴尔 2 ，30 小米 3

创建笔记本数据表laptop(lid,title,price,spec规格，detail详情，shelftime上架时间，isonsale是否在售，familyld所属分类编号)插入若干条数据

预习mysql中的

列约束

列约束：在服务器端对数据验证，会使服务器数据压力增大；

在浏览器端中用JS 验证

(1)主键约束——PRIMARY KEY（必用）

（主键约束的列上不能出现重复的值，放在设定值后面，他会按照从小到大顺序自动排列，加快查找速度，通常主键添加到编号列中，不允许存在多个主键，只能给某一列用）主键上不允许插入NULL.

(2)非空约束——NOT NULL声明非空约数的列上不允许插入NULL值，（不允许为空）

(3)唯一约束——UNIQUE声明唯一约束的列上不能插入重复的值，允许插入，甚至多个NULL，一个表中可出现多个约束（NULL和任何职不等,甚至包括自身都不等）

(4)检查约束——CHECK

检查约束可以对插入的数据进行自定义验证

CREATE TABLE student(score TINYINT CHECK(score>=0 AND score<=100)；

mysql中不支持检查约束，否则会降低数据的插入速度。

（5）默认值约束——DEFAULT(default)

可以使用DEFAULE关键字声明为默认值，有两种方式可以使用默认值

INSERT INTO laptop\_family VALUES(50,'华硕',DEFAULT);

INSERT INTO laptop\_family (fid,fname)VALUES(60,'神州’)；

（6）外键约束——FOREIGN KEY

foreign key(familyld) references laptop\_family(fid));

声明了外键约束的列上，取值必须在另一个表中的主键列上出现过，两者的列类型要保持一致，允许使用NULL

FOREIGN KEY (外键列) REFERENCES 另一数据表（主键列）

mysql中的自增列

AUTO\_INCREMENT:自动增长假如一个列声明了自增列，无需手动赋值，直接赋值

简单查询

（1）查询特定的列

查询所有员工的姓名、生日

SELECT ename,birthday FROM emp;

查询所有员工的编号、姓名、性别、工资

SELECT eid,ename,sex,salary FROM emp;

(2)查询所有的列

SELECT\*FROM emp;

SELECT eid,ename,sex,borthday,salary,deptid FROM emp;

(3)给列起别名

查询所有与员工的姓名和工资，使用中文别名

SELECT ename AS 姓名,salary AS 工资 FROM emp;

查询所有员工的编号，姓名，性别，生日，使用中文别名

SELECT eid AS 编号,ename AS 编号,sex AS 性别,borthday AS 生日 FROM emp;

查询所有员工的编号和姓名，使用一个字母为别名

SELECT eid a,ename b FROM emp;

再起别名的时候，AS关键字可以省略

（4）显示不同的记录/合并相同的记录

查询出都有哪些性别的员工

SELECT DISTINCT sex FROM emp;

查询出员工都在哪些部门

SELECT DISTINCT deptid FROM emp;

(5)查询时执行计算

计算2+3-4+5\*6/3

SELECT 2+3-4+5\*6/3;

查询所有员工的姓名及其年新

SELECT ename,salary\*12 FROM emp;

假如每个员工的工资加500，年终奖加5000，查询所有员工的姓名及其年薪，给列其中文别名

SELECT (salary+500)\*12+5000 年薪,ename 姓名 FROM emp;

(6)查询的结果集排序

查询所有部门表数据，结果集按照编号从小到大排序

SELECT\*FROM dept ORDER BY did ASC ;(ASC:ascendant)

查询所有部门表数据，结果集按照编号从大到小排序

SELECT\*FROM dept ORDER BY did DESC;(descendant降序排列）

查询所有员工的列，结果集按照工资的降序排列

SELECT\*FROM emp ORDER BY salary DESC;

查询员工所有的列，结果集从小到大排列

SELECT\*FROM emp ORDER BY birthday DESC;

查询员工所有的列，结果集按照升序排列

SELECT\*FROM emp ORDER BY ename ASC;（ASC可省略，会按照字母顺序排列）

查询员工所有的列，结果按照降序排列，如果工资相同，按照升序排列

SELECT\*FROM emp ORDER BY salary desc,ename;

查询员工所有的列，要求所有的女员工在前，如果性别相同按照生日的降序排列

SELECT\*FROM emp ORDER BY sex ASC,birthday DESC;（DESC在这里为降序排列）

ORDER BY可以按照数值，字符串，日期时间排序，默认是按照升序排列（ASC）

(7)条件查询

查询编号为五的员工的所有列

SELECT\*FROM emp WHERE eid=5;

查询出姓名为King的员工的编号，工资姓名，生日

SELECT eid,salary,ename,birthday from emp where ename="king";

查询出20号部门下员工所有列

SELECT\*FROM emp WHERE deptid=20;

查询出工资在6000以上员工所有列

SELECT\*FROM emp WHERE salary>=6000;

比较运算符：< > = ！=(不等于) >= <=

查询出1993-1-1日后出生的员工的所有列

SELECT\*FROM emp WHERE birthday>"1993-01-01";

查询出不在10号部门的员工的所有列

SELECT\*FROM emp WHERE deptid!=10;

查询出没有明确部门的员工所有列

SELECT\*FROM emp WHERE deptid IS NULL;

查询出有明确部门的员工的所有列

SELECT\*FROM emp WHERE deptid IS NOT NULL;

查询出工资在六千以上的男员工所有列

SELECT\*FROM emp WHERE salary>=6000 AND sex=1;

查询出工资在7000-10000之间员工所有咧

SELECT\*FROM emp WHERE salary>=7000 AND salary<=10000;

或SELECT\*FROM emp WHERE salary BETWEEN 7000 AND 10000;

BETWEEN……AND(在两者之间，必须)

查询出工资不在7000-10000之间员工所有列

SELECT\*FROM emp WHERE salary NOT BETWEEN 7000 AND 10000;（NOT BETWEEN不在两者之间）

或SELECT\*FROM emp WHERE salary <7000 OR salary>10000;

(OR或)

查询出在1990年之前和1995年之后出生的员工的所有列

SELECT\*FROM emp WHERE birthday<"1990-01-01" OR birthday>"1995-12-12";

查询出20号和30号部门员工所有的列

SELECT\*FROM emp WHERE deptid=20 OR deptid=30;

或SELECT\*FROM emp WHERE dePtid IN(20,30);

查询出不在20和30部门员工所有的列

SELECT\*FROM emp WHERE deptid not in(20,30);

IS NULL/IS NOT NULL

AND/OR

BETWEEN …AND…/NOT BETWEEN …AND…

IN()/NOT IN()

（8）分页查询

假如查询的结果集有太多的数据一次显示不完，可以使分页显示

需要两个条件：当前的页码，每页的数据量。

每页的开始=(当前的页码-1)\*每页数量

SELECT\*FROM emp LIMIT start,count;(LIMIT 限制呢，start每页的开始,count每页数量)

假如每页显示五条数据

SELECT\*FROM emp 0,5；

SELECT\*FROM emp 5,5;

假设每页显示6条数据，查询前三页

SELECT\*FROM emp limit 0,6;

SELECT\*FROM emp limit 6,6;

SELECT\*FROM emp limit 12,6;