复习

解决中文乱码

脚本文件(.sql)，客户端 set names utf8，

服务器端 create database xuezi charset=utf8;

列类型

创建数据表的时候，给列设置的数据类型

数值类型：

tinyint smallint int bigint float double decimal(4,2) bool(true/false 1/0)

日期时间类型

date '2018-8-3' time '9:08:30' datetime

字符串类型

VARCHAR() 变长字符串，操作速度相对慢，空间节省了65535

CHAR(11) 定长字符串，操作速度更快，空间浪费 255

TEXT 2G

drop database if exists xuezi;

create database xuezi;

use xuezi;

create table t1(

tid int...

);

insert into t1 values();

delete from t1 where tid=1;

update t1 set 列名称='值'... where tid=3;

select \* from t1;

SQL语句的分类

|  |
| --- |
| (1)DDL: Data Define Language 定义数据  CREATE / DROP / ALERT(修改) / TRUNCATE(截断)  (2)DML: Data Manipulate Language 操作数据  INSERT / DELETE / UPDATE  (3)DQL: Data Query Language 查询数据  SELECT  (4)DCL: Data Control Language 控制用户权限  GRANT(授权) / REVOKE(收权) |

学习目标

列约束

简单查询

复杂查询

1.列约束

(1)主键约束 —— PRIMARY KEY

见day3笔记

(2)唯一约束 —— UNIQUE

声明了唯一约束的列不能插入重复的值，但允许插入NULL值，而且允许插入多个NULL值。

(3)默认值约束 —— DEFAULT

可以使用默认值DEFAULT来为指定的列设置默认值；

使用方法：

INSERT INTO user VALUES(1,'tom',**DEFAULT**);

INSERT INTO user (uid,uname) VALUES(2,'jerry');

(4)非空约束 —— NOT NULL

所插入的值不允许为空

(5)检查约束 —— CHECK

检查约束是对输入的数据范围进行检验

CREATE TABLE student(

age TINYINT CHECK(age>=0 AND age<=100)

)

注意：MySQL不支持检查约束

(6)外键约束 —— FOREIGN KEY

声明了外键的列，取值必须在另一个表的主键列上出现过。

允许使用NULL作为外键的值

foreign key(列名称) references 另一个表(主键)

2.MySQL的自增列

AUTO\_INCREMENT：自动增长，如果设置一个列为自增列，无需手动赋值，赋值为NULL，就会获取当前的最大值，新纪录在当前的基础之上加1

注意：只适用于整数型的主键上，同时允许手动赋值。

练习：列约束的使用

创建文件02\_tedu.sql，创建数据库tedu，使用UTF8编码，进入该数据库。

创建部门表dept，包含编号(did)，名称(dname)；

10 研发部 20 运营部 30 市场部 40 测试部

创建员工表empt，包含编号(eid)，姓名(ename)，性别(gender)，生日(birthday)，工资(salary)，隶属的部门编号(deptId)

插入14行记录，分别隶属部门10,20,30，其中14号员工不在任何部门(NULL)

3.项目中如何存储时间

DATE #2018-08-03

03/08/2018 08/03/2018 2018年08月03日

时间存的是，距离1970年1月1日距离当前的毫秒数。

48\*365\*24\*60\*60\*1000 = 1513728000000

将来存储时间列类型 BIGINT

存储性别 BOOL 1/0 男/女 male/female man/woman...

1.简单查询 —— 只查询特定的列

示例：查询员工的姓名，工资，生日

SELECT ename,salary,birthday FROM emp;

示例：查询员工的编号，姓名，性别，部门编号

SELECT eid,ename,sex,deptId FROM emp;

2.简单查询 —— 查询所有的列

SELECT \* FROM emp;

3.简单查询 —— 给列取别名

示例：查询员工的姓名，生日，收入，取别名

SELECT ename AS 姓名,birthday AS 生日,salary AS 收入 FROM emp;

SELECT ename 姓名,birthday 生日,salary 收入 FROM emp;

示例:查询员工的编号，姓名，性别，起英文别名

SELECT eid a,ename b,sex c FROM emp;

4.简单查询 —— 只显示不同的记录

示例：查询哪些部门编号下有员工

SELECT DISTINCT deptId FROM emp;

示例：查询公司有哪些性别的员工

SELECT DISTINCT sex FROM emp;

5.简单查询 —— 在查询时执行计算

示例：计算2+3\*5/10

SELECT 2+3\*5/10 结果;

示例：查询出所有员工的姓名和年薪

SELECT ename 姓名,salary\*12 年薪 FROM emp;

示例：假设每个员工的工资+500，年终每个人再给5000的年终奖，查询每个员工一年的收入

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

6.简单查询 —— 查询结果集的排序

示例：查询所有的员工，结果按照工资由低到高

SELECT \* FROM emp ORDER BY salary ASC; #**asc**endant升序

示例：查询所有的员工，结果按照工资由高到低

SELECT \* FROM emp ORDER BY salary DESC;#**desc**endant降序

示例：查询所有的员工，结果按照年龄从大到小排序

SELECT \* FROM emp ORDER BY birthday ASC;

示例：查询所有的员工，结果集按照姓名排序

SELECT \* FROM emp ORDER BY ename ASC;

示例：查询所有的员工，按照生日由小到大排序，如果生日相同，再按照姓名由小到大排序

SELECT \* FROM emp ORDER BY birthday ASC,ename ASC;

示例：查询所有的员工，按照工资由小到大，如果工资相同，女员工显示在男员工的前边。

SELECT \* FROM emp ORDER BY salary ASC,sex ASC;

说明：排序默认是按照升序排列。

7.简单查询 —— 条件查询

示例：查询出编号为5的员工所有信息

SELECT \* FROM emp WHERE eid=5;

练习：查询出姓名为KING的员工的编号、工资、生日

SELECT eid,salary,birthday FROM emp WHERE ename='king';

练习：查询出20号部门下所有员工的信息

SELECT \* FROM emp WHERE deptId=20;

练习：查询出所有的男员工信息

SELECT \* FROM emp WHERE sex=1;

示例：查询出工资大于等于5000的员工所有信息

SELECT \* FROM emp WHERE salary>=5000;

练习：查询出1991-1-1之后出生的员工所有信息

SELECT \* FROM emp WHERE birthday>'1991-1-1';

|  |
| --- |
| > >= < <= = != |

练习：查询出不在10号部门的所有员工信息

SELECT \* FROM emp WHERE deptId!=10;

练习：查询出暂时没有部门的员工信息

SELECT \* FROM emp WHERE deptId IS NULL;

|  |
| --- |
| 注意：NULL和任何值都不能进行相等/不等判定，可以使用IS NULL 和 IS NOT NULL |

练习：查询出有明确部门的员工信息

SELECT \*FROM emp WHERE deptId IS NOT NULL;

练习：查询出工资大于6000的女员工所有信息

SELECT \* FROM emp WHERE salary>6000 AND sex=0;

练习：查询出工资在5000~5999之间的员工信息

SELECT \* FROM emp WHERE salary>=5000 AND salary<=5999;

SELECT \* FROM emp WHERE salary BETWEEN 5000 AND 5999;

|  |
| --- |
| between 5000 and 5999; >=5000 and <=5999 |

练习：查询出工资小于4000和大于8000的员工信息

SELECT \* FROM emp WHERE salary<4000 OR salary>8000;

|  |
| --- |
| 多条件拼接：  AND 并且 两个都要满足  OR 或者 两个满足其一 |

练习：查询出在1991年出生的员工信息

SELECT \* FROM emp WHERE birthday>='1991-1-1' AND birthday<='1991-12-31';

SELECT \* FROM emp WHERE birthday BETWEEN '1991-1-1' AND '1991-12-31';

练习：查询出在1990年之前和1993之后出生的员工信息

SELECT \* FROM emp WHERE birthday<'1990-1-1' OR birthday>'1993-12-31';

练习：查询出在10和30号部门的员工信息

SELECT \* FROM emp WHERE deptId=10 OR deptId=30;

SELECT \* FROM emp WHERE deptId IN(10,30);

练习：查询出在10、30、50、80号部门的员工信息

SELECT \* FROM emp WHERE deptId=10 OR deptId=30 OR deptId=50 OR deptId=80;

SELECT \* FROM emp WHERE deptId IN(10,30,50,80);

练习：查询出不在10和30号部门的员工信息

SELECT \* FROM emp WHERE deptId NOT IN(10,30);

8.简单查询 —— 模糊条件查询

示例：查询出姓名中包含字母E的员工所有信息

SELECT \* FROM emp WHERE ename LIKE '%E%';

练习：查询出姓名以E结尾的员工所有信息

SELECT \* FROM emp WHERE ename LIKE '%E';

练习：查询出姓名中倒数第二个字符是E的员工所有信息

SELECT \* FROM emp WHERE ename LIKE '%E\_';

|  |
| --- |
| SQL中提供了两个模糊查询的匹配符  % 可以匹配任意多个字符 >= 0  \_ 可以匹配任意一个字符 =1  上述两个符号不能和=连用，只能使用LIKE连用。 |

9.简单查询 —— 分页查询

查询出所有的男员工，对查询的结果集，按照工资升序排序。

SELECT \* FROM emp WHERE sex=1 ORDER BY salary ASC;

分页显示：查询的结果集记录太多，一次显示不完，可以一页一页的显示。分页查询语句在不同的数据库中实现方法不一样，MySQL是最简单的。

SELECT \* FROM emp LIMIT start, count;

start: 从结果集中开始读取的值

count: 指定读取的最多行数

第1页： SELECT \* FROM emp LIMIT 0,5; #(页码-1)\*每页数量

第2页： SELECT \* FROM emp LIMIT 5,5;

第3页： SELECT \* FROM emp LIMIT 10,5;

...

第n页： SELECT \* FROM emp LIMIT (n-1)\*5,5

start = (n-1)\*count

练习：

分页查询所有员工信息，假设每页最多显示6条记录，写出查询前4页的sql语句。

SELECT \* FROM emp LIMIT 0,6; #第1页

SELECT \* FROM emp LIMIT 6,6; #第2页

SELECT \* FROM emp LIMIT 12,6; #第3页

SELECT \* FROM emp LIMIT 18,6; #第4页

**注意：LIMIT后边的两个值不能添加引号。**

课后练习：

(1)删除所有的演示代码，根据注释重新编写SQL语句。

(2)创建产品表三个。