复习：

列类型：

数值类型： 20 '20'

tinyint / smallint / int / bigint

float / double / decimal(m,d)

bool (TRUE-1/FALSE-0)

日期时间类型： '1990-1-1'

date / time / datetime

字符串类型： 'TOM'

char(m) / varchar(m) / text(m)

列约束： CREATE TABLE stu( age TINYINT 约束)

(1)唯一约束： unique

(2)非空约束： not null

(3)主键约束： primary key = 唯一+非空+排序

练习： 注意使用合理的列类型和列约束

编写1\_xz.sql，丢弃并重建数据库xz，进入该库。

创建笔记本型号表laptop\_family：

fid, fname, laptopCount

10 联想E480 3

20 Mac Book 2

30 戴尔燃700 2

创建笔记本信息表，并插入7行数据laptop：

lid(编号), pic(图片路径), title(标题), price(价格), spec(规格), marketDate(上市日期), isRecommended(是否为首页推荐), familyId(所属型号的编号)

今日目标：

(1)补充：列约束

(2)SQL中的查询

1.补充：列约束

CREATE TABLE stu( age TINYINT 约束)

(1)唯一约束： unique

(2)非空约束： not null

**(3)主键约束：** primary key = 唯一+非空+排序

(4)默认值约束： default

可以为某个列声明一个默认值，若插入数据时未指定具体值，就可以使用默认值。

CREATE TABLE user(

avatar VARCHAR(128) DEFAULT 'img/logo.gif'

);

INSERT INTO user VALUES( 'img/lwh.jpg' ); #不使用默认值

INSERT INTO user VALUES( DEFAULT ); #使用默认值

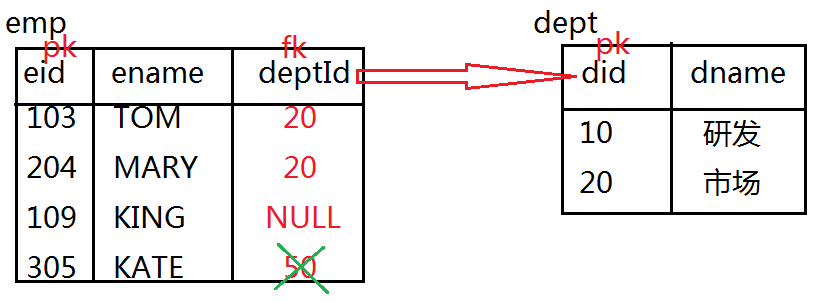
(5)检查约束：check

检查约束可以对新插入的数据值的范围进行检查，满足条件的允许插入，否则插入失败！ —— MySQL不支持，因为检查约束会降低所有插入和更新操作的执行效率。

CREATE TABLE stu ( age INT CHECK( age>=6 AND age<=15) )

(6)外键约束：foreign key

声明了外键约束的列，可以插入重复值或NULL值，但所有插入的值必须在另外一个表的主键列上出现过。因为检查约束会降低所有插入和更新操作的执行效率，慎重使用。



CREATE TABLE dept( did INT PRIMARY KEY, dname VARCHAR(32));

CREATE TABLE emp(

eid INT PRIMARY KEY,

ename VARCHAR(32),

deptId INT ,

**FOREIGN KEY(deptId) REFERENCES dept(did)**

);

2.项目中如何保存日期和时间

方式1：使用VARCHAR，如'1990-1-12'，不便于对齐，不便于比较大小

方式2：使用DATE，如'1990-1-1'，

好处：会自动补齐，便于比较大小

不足：不便于实现国际化应用，不同系统日期时间表示范围不同

方式3：使用BIGINT，如1520478184225，表示指定的时间距离计算机元年多少毫秒

|  |
| --- |
| i18n： internationalization，国际化，一个实现国际化的项目可以做到“对中国人现实汉语及对应的习惯，对美国人显示英语及对应的习惯”  中国人： 1997-7-25  美国人： 7-25-1997， Jul 25, 1997  欧洲人： 25/7/1997 |
| 计算机元年：1970-1-1 0:0:0 GMT |
| 计算机中如何存储系统时间？ —— 转换为一个大整数，表示指定的时间距离计算机元年经过了多少毫秒  0 1970-1-1 0:0:0  1000 1970-1-1 0:0:1  1000\*60 1970-1-1 0:1:0  1000\*3600 1970-1-1 1:0:0  1000\*3600\*24 1970-1-2 0:0:0  1000\*3600\*24\*365 1971-1-1 0:0:0  1000\*3600\*24\*365\*48 大约就是今天的时间 |

3.扩展小知识：MySQL中如何实现自增列

MySQL专门提供了一个 AUTO\_INCREMENT 关键字，只能用于INT 型的 PRIMARY KEY 列上，用于实现主键上整数的自动增长。

自增列上允许手工赋值，后续的插入只要插入NULL即可，就会在当前最大值基础上+1.

4.MySQL简单查询 —— 查询特定的列

示例：查询所有员工的姓名及其工资

SELECT ename, salary FROM emp ;

练习：查询出所有部门的名称及所在地

SELECT dname, location FROM dept;

练习：查询所有员工的姓名、月薪、生日、性别

SELECT ename, salary, birthday, sex

FROM emp; #列出现顺序随意

练习：查询所有员工的姓名、所在部门编号、员工编号、所在部门的编号、员工编号

SELECT ename, deptId, eid, deptId, eid

FROM emp ; #某列可以反复出现

练习：查询出员工的编号、姓名、性别、生日、工资、部门编号

SELECT eid, ename, sex, birthday, salary, deptId

FROM emp;

SELECT \* FROM emp ; #\*指代所有的列名

5.MySQL简单查询 —— 给列取别名

示例：查询所有员工的姓名和工资，列名用中文呈现

SELECT ename AS 姓名, salary 月薪

FROM emp; #AS用于给列取别名，可以省略

练习：查询所有员工的编号、姓名、所在部门编号，列名都用中文显示

SELECT eid AS 工号, ename AS 姓名, deptId 部门编号

FROM emp;

练习：查询所有员工的姓名(用n呈现)、生日(用b呈现)、工资(用s呈现)

SELECT ename AS n, birthday AS b, salary AS s

FROM emp;

6.MySQL简单查询 —— 只显示不同值

示例：查询出有员工的所有部门的编号

SELECT DISTINCT deptId

FROM emp; #distinct：不同的，相异的

练习：查询出公司有哪些性别的员工

SELECT DISTINCT sex

FROM emp;

7.MySQL简单查询 —— 查询时执行运算

示例：计算2+3的和

SELECT 2+3 ;

SELECT (123+256-378)\*119 / 110 AS 结果;

示例：查询出所有员工的姓名，月薪以及年薪

SELECT ename, salary, salary\*12

FROM emp;

练习：老板想看看如果每个员工每月加薪10%，年底再给一次性的20000年终奖，每人年底总收入是多少

SELECT ename, salary, salary\*(1+0.1)\*12 + 20000

FROM emp;

练习：假设每个员工每月需要缴纳社保比例占月薪的30%，但年终的20000年终奖无需缴税，计算每个员工每个月的平均到手收入

SELECT ename, (salary\*(1-0.3)\*12 + 20000) / 12

FROM emp;

8.MySQL简单查询 —— 单条件查询

MySQL提供比较运算符： = != > >= < <=

示例：查询出10号部门的所有信息

SELECT \* FROM dept

WHERE did=10; #where：满足...条件

示例：查询出编号大于10的部门的所有信息

SELECT \* FROM dept

WHERE did>10;

练习：查询出7788号员工的所有信息

SELECT \* FROM emp

WHERE eid=7788;

练习：查询出20号部门所有的员工信息

SELECT \* FROM emp

WHERE deptId=20;

练习：查询出所有的女员工信息

SELECT \* FROM emp

WHERE sex=0 ;

练习：查询出不在10号部门的员工信息

SELECT \* FROM emp

WHERE deptId != 10;

练习：查询出工资达到6000的所有信息

SELECT \* FROM emp

WHERE salary >= 6000;

练习：查询出不在任何部门的员工的信息

SELECT \* FROM emp

#WHERE deptId = NULL ; #NULL不等于任何数据，以及它自身

WHERE deptId IS NULL ; #NULL的判定不能用=或!=

练习：查询出在某个特定部门的员工的信息

SELECT \* FROM emp

WHERE deptId IS NOT NULL ; #此处不能用 != NULL

9.MySQL简单查询 —— 多条件查询

MySQL提供了两个逻辑运算符： AND(并且) OR(或者)

示例：查询出10号部门的女员工所有信息

SELECT \* FROM emp

WHERE deptId=10 AND sex=0 ;

练习：查询出工资在6000~8000之间的所有员工信息

SELECT \* FROM emp

#WHERE 6000<salary<8000; #错误写法！

WHERE salary>=6000 AND salary<=8000;

练习：查询出10号和30号部门的所有员工信息

SELECT \* FROM emp

WHERE deptId=10 OR deptId=30;

练习：查询出在1990年出生的所有员工信息(在1-1号和12-31间)

SELECT \* FROM emp

WHERE birthday>='1990-1-1' AND birthday<='1990-12-31';

练习：查询出工资小于5000和大于8000的员工所有信息

SELECT \* FROM emp

WHERE salary<5000 OR salary>8000;

10.MySQL简单查询 —— 模糊条件查询

MySQL中使用 **%** 代表“任意多个任意字符”； 用 **\_** 代表“任意一个字符”—— 这两个符号表示模糊查询必须与 **LIKE** 关键字配合！

示例：查询出姓名中包含字符E的所有员工信息

SELECT \*

FROM emp

#WHERE ename = 'E'; #错误

#WHERE ename = '%E%'; #错误

WHERE ename LIKE '%E%';

练习：查询出姓名中第二个字符是E的员工所有信息

SELECT \*

FROM emp

WHERE ename LIKE '\_E%';

课后练习：

(1)删除上面笔记中所有的SQL语句，根据提示写出所有语句 —— 经典练习！！

(2)根据“学子商城数据库结构设计说明书”编写出该项目所需要的.sql文件——创建出数据库，及所有的表，无需插入数据。