复习：

MySQL中的列类型：

(1)数值类型

TINYINT/SMALLINT/INT/BIGINT

FLOAT/DOUBLE/DECIMAL(M,D)

BOOL(TRUE/FALSE)

(2)日期时间类型

DATE/TIME/DATETIME

(3)字符串类型

CHAR(M)/VARCHAR(M)/TEXT(M)

MySQL中的列约束：

(1)主键约束：非空、唯一、排序

uid INT PRIMARY KEY,

(2)唯一约束：不能重复

uname VARCHAR(32) UNIQUE,

(3)非空约束：不能为NULL

upwd VARCHAR(32) NOT NULL,

(4)检查约束：

(5)默认值约束：

avatar VARCHAR(128) DEFAULT 'img/d.png'

(6)外键约束：

deptId INT,

FOREIGN KEY(deptId) REFERENCES dept(did)

|  |
| --- |
| 上述约束中推荐使用的“主键约束”——会对主键列排序，提高查找速度。  有的项目中不推荐使用唯一约束、非空约束、外键约束、检查约束——会影响插入和更新的效率。 |

今日目标：

(1)简单查询 —— 重点掌握

(2)复杂查询 —— 难点，允许分两天掌握

1.MySQL专用关键字——AUTO\_INCREMENT

不是SQL标准关键字！

auto\_increment，自动增长的列，用于INT型的主键列上，可以实现新插入的记录在已有的最大值基础上+1。自增的列可以手工赋值，也可以使用NULL实现自增值。

2.真实项目中数据库如何记录日期和时间

(1)使用CHAR/VARCHAR类型

不足：不方便进行日期大小比较 '2015-10-20' 和 '3-10-20'

(2)使用DATE/TIME/DATETIME类型

不足：不便于实现**i18n**（internationalization，国际化），

中国人： 2017-10-25 22:38:55

美国人： 10-25-2017 22:38:55

欧洲人： 25/10/2017 22:38:55

(3)使用BIGINT类型

存储方便，便于实现国际化

|  |
| --- |
| 计算机如何记录时间：一个很大很大的数字，表示从计算机元年(1970-1-1 00:00:00 GMT)开始到指定的时间经过了多少“毫秒”；所有的编程语言都可以把一个整数转换为日期时间，反之亦然。 |

3.简单查询——查询特定的列

示例：查询所有员工的姓名和工资

SELECT ename, salary

FROM emp;

练习：查询所用员工的部门编号、入职日期、姓名

SELECT deptId, hireDate, ename

FROM emp;

4.简单查询——查询所有的列

示例：查询员工的所有信息

SELECT eid,ename,salary,hireDate,deptId

FROM emp;

可以简写为：

SELECT \* FROM emp;

提示：SQL语句中的 \* 指代的“所有的列名”

5.简单查询——给列取别名

示例：查询员工的编号、姓名和工资，列名用中文呈现

SELECT eid AS 工号, ename 姓名, salary '月 薪'

FROM emp;

提示：给列取别名时AS关键字可以省略；若别名中有空白字符必须用英文引号括起来

练习：查询出部门的编号和名称，列名用中文呈现

SELECT did AS 编号, dname 名称

FROM dept;

练习：查询出部门的编号和名称，列名用i和n呈现

SELECT did AS i, dname AS n

FROM dept;

6.简单查询——只显示不同的值

示例：查询出哪些部门有员工

思路：查询出员工表中的部门编号即可，重复的值只显示一次

SELECT DISTINCT deptId

FROM emp;

说明：distinct，翻译为“不同的”

7.简单查询——在查询时执行运算

示例：使用SQL计算1+2的和

SELECT 1+2 AS 求和;

示例：查询所有员工的姓名及其年薪（月薪x12）

SELECT ename, salary\*12

FROM emp;

练习：假设老板想看看给每人每月加薪500，年终再给每人5000奖金后每人的年终总薪水，编写需要的SQL语句

SELECT ename, (salary+500)\*12+5000 AS 总薪水

FROM emp ;

|  |
| --- |
| order: (1)顺序/排序 (2)订单 |

8.简单查询——查询结果的排序

示例：查询出所有员工的姓名和工资，按照工资由低到高排列

SELECT eid, ename, salary

FROM emp

ORDER BY salary ;

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

SELECT eid, ename, salary

FROM emp

ORDER BY salary ASC; #ascendant 升序

示例：查询出所有员工的姓名和工资，按照工资由高到低排列

SELECT eid, ename, salary

FROM emp

ORDER BY salary DESC;

|  |
| --- |
| DESC emp; #描述一下emp表的结构 describe |
| ORDER BY salary DESC; #descendant，降序 |

练习：查询所有员工的姓名和入职日期，先入职的先显示

SELECT ename, hireDate

FROM emp

ORDER BY hireDate;

练习：查询所有员工的姓名和入职日期，先入职的后显示

SELECT ename, hireDate

FROM emp

ORDER BY hireDate DESC;

示例：查询所有员工的姓名和工资，按照工资由低到高排序，工资相同时再按姓名由低到高排

SELECT ename, salary

FROM emp

ORDER BY salary, ename;

|  |
| --- |
| 数据库中的排序可以按照数字、日期、字符串相关的列来排序。 |

9.简单查询——分页查询 —— 重点&难点

若结果集中的记录数很多，一般都需要“分页”呈现。

分页查询在不同的数据库中所用的语句不同！MySQL中的分页查询是最简单的：

语法： SELECT ... FROM .... WHERE .... ORDER BY ...

LIMIT start, count; #对查询结果进行限制

start：表示从哪一行开始读取数据，下标从0开始！

count：表示一次最多可以读取的行数

假设：每页最多显示5行:

第1页： SELECT ....FROM ... LIMIT 0, 5; #0~4

第2页： SELECT ....FROM ... LIMIT 5, 5; #5~9

第3页： SELECT ....FROM ... LIMIT 10, 5; #10~14

第4页： SELECT ....FROM ... LIMIT 15, 5; #15~19

....

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

练习：查询所有员工的姓名和工资，按照工资排序，显示第1页(每页最多显示5行)

SELECT ename, salary FROM emp

ORDER BY salary LIMIT 0, 5 ;

练习：查询所有员工的姓名和工资，按照工资排序，显示第2页(每页最多显示5行)

SELECT ename, salary FROM emp

ORDER BY salary LIMIT 5, 5 ;

练习：查询所有员工的姓名和工资，按照工资排序，显示第3页(每页最多显示5行)

SELECT ename, salary FROM emp

ORDER BY salary LIMIT 10, 5 ;

练习：查询所有员工的姓名和工资，按照工资排序，显示第4页(每页最多显示5行)

SELECT ename, salary FROM emp

ORDER BY salary LIMIT 15, 5 ;

|  |
| --- |
| WHERE： 在哪里，满足...条件，符合...要求 |

10.简单查询——条件查询

示例：查询出员工编号为7788的员工所有信息

SELECT \* FROM emp

WHERE eid=7788 ;

练习：查询姓名为TOM的员工所有信息

SELECT \* FROM emp

WHERE ename='TOM' ;

练习：查询10号部门的员工的所有信息

SELECT \* FROM emp

WHERE deptId=10 ;

练习：查询入职日期为2000-1-1的员工所有信息

SELECT \* FROM emp

WHERE hireDate='2000-1-1';

示例：查询工资大于等于5000的员工所有信息

SELECT \* FROM emp

WHERE salary>=5000; #> < >= <= !=

练习：查询员工编号小于7788的员工所有信息

SELECT \* FROM emp

WHERE eid<7788;

练习：查询不在10号部门的员工所有信息

SELECT \* FROM emp

WHERE deptId != 10;

练习：查询尚未确定部门的员工所有信息

SELECT \* FROM emp

WHERE deptId IS NULL; #错误写法WHERE deptId=NULL ;

SELECT \* FROM emp

WHERE deptId IS NOT NULL;

|  |
| --- |
| null表示“不确定”，一个不确定的值与任何其他值都是不能判定相等与否的（包括null）！不能用=和!=对NULL进行判定。 |

|  |
| --- |
| 可以使用 AND(并且) / OR(或者) 进行多条件组合判定 |

示例：查询出工资在6000~8000之间的员工所有信息

SELECT \* FROM emp

#错误写法：WHERE 6000<=salary<=8000 ;

WHERE salary>=6000 AND salary<=8000;

示例：查询出工资小于6000以及大于8000的员工所有信息

SELECT \* FROM emp

WHERE (salary<6000) OR (salary>8000);

练习：查询出20和30号部门的员工所有信息

SELECT \* FROM emp

WHERE deptId=20 OR deptId=30;

练习：查询出10部门工资中大于8000的员工所有信息

SELECT \* FROM emp

WHERE deptId=10 AND salary>8000 ;

练习：查询出在2000年入职的员工所有信息

SELECT \* FROM emp

WHERE hireDate>='2000-1-1' AND hireDate<='2000-12-31';

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

SELECT \* FROM emp

WHERE hireDate BETWEEN '2000-1-1' AND '2000-12-31';

练习：查询出不在2000年入职的员工所有信息

SELECT \* FROM emp

WHERE hireDate<'2000-1-1' OR hireDate>'2000-12-31';

练习：查询出部门编号为10、30、50的员工所有信息

SELECT \* FROM emp

WHERE deptId=10 OR deptId=30 OR deptId=50;

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

SELECT \* FROM emp

WHERE deptId IN ( 10, 30, 50 );

11.简单查询——条件查询——模糊条件查询

示例：查询出姓名中包含字符E的员工所有信息

SELECT \* FROM emp WHERE ename='E'; #错误

SELECT \* FROM emp WHERE ename='%E%'; #错误

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

练习：查询出姓名中以T开头的员工所有信息

SELECT \* FROM emp WHERE ename LIKE 'T%' ;

练习：查询出姓名中倒数第二个字符为E的员工所有信息

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

|  |
| --- |
| SQL中可以使用 % 匹配任意多个任意字符；  SQL中可以使用 \_ 匹配任意一个任意字符；  这两个符号必须与LIKE关键字配合使用 |

12.复杂查询 —— 分组/聚合查询

SQL中提供了如下几个分组查询函数：

**SUM()、AVG()、COUNT()、MAX()、MIN()**

示例：查询出工资大于6000的员工的数量

SELECT COUNT(salary) FROM emp

WHERE salary>6000;

示例：查询出工资大于6000的员工的最小值

SELECT MIN(salary) FROM emp

WHERE salary>6000;

示例：查询出工资大于6000的员工的平均工资

SELECT AVG(salary) FROM emp

WHERE salary>6000;

示例：查询出10号部门的员工的工资总和

SELECT SUM(salary) FROM emp

WHERE deptId=10;

课后作业：

(1)删除笔记中的示例/练习代码，根据说明文字写出SQL语句

(2)根据学子商城数据库设计说明书，编写出SQL脚本文件