

# 课程回顾

```
1  1.update语句
2  update 表名 set 列名=新的值,列名=新的值,...,列名=新的值 where 条件;
3
4  条件中运算符:
5  算术运算符: + - * / %
6  关系运算符: = != <> > >= < <=
7  逻辑运算符: and or not
8
9  模糊匹配:
10 between 最小值 and 最大值
11
12 in:
13 not in:
14
15 is null:
16 is not null:
17
18 like '% _'
19
20 2.delete语句
21 -- 清空一张表所有的数据
22 delete from 表名; -- 推荐优先使用delete
23
24 -- 截断表
25 truncate table 表名; -- 非法操作方式
26
27 -- 带条件删除
28 delete from 表名 where 条件
29
30 3. 约束
31 主键约束: primary key
32 唯一约束: unique
33 非空约束: not null
34 默认值约束: default
35 自增长约束:auto_increment
36
37 检查约束: mysql不支持!!!
38 外键约束:
```

```
Query : insert test values (default,null)
Error Code : 1048
Column 'tName' cannot be null 该列具有非空约束
Execution Time : 00:00:00:000
Transfer Time : 00:00:00:000
Total Time : 00:00:00:000
```

## 课程目标

### 1 数据库备份和还原

### 2 单表查询 ===== 重点

### 3 两表连接查询 ===== 重点

### 4 外键约束和实体映射关系 ===== 理解

### 5 子查询 ===== 重点

## 课程实施

### 1 数据库备份和还原

#### 数据库备份意义：

方便数据的共享和移动

数据因为失误丢失，可以通过备份快速恢复数据

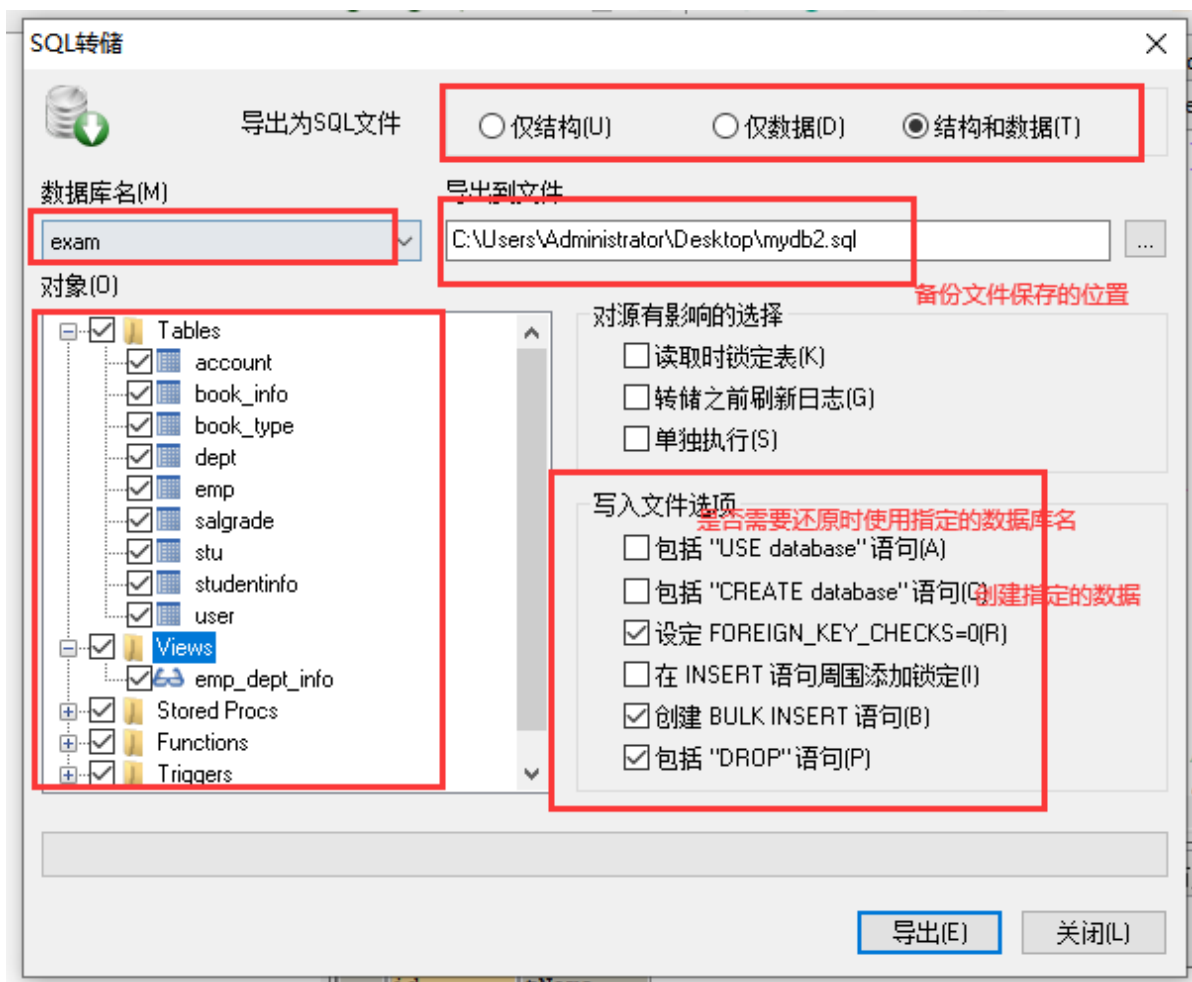
#### 备份实现方式：

mysql启动客户端

mysqld安装服务器

mysqldump数据备份

```
1 | mysqldump -u用户名 -p密码 数据库名>生成的脚本文件路径
```



## 2 单表查询

查询语句：不会改变数据表中的数据。获取数据。

### 2-1 DQL语法

- 1 `select` 列名, ..., 列名
- 2 `from` 表名
- 3 `where` 条件
- 4 `group by` 分组条件
- 5 `having` 筛选条件
- 6 `order by` 排序条件
- 7 `limit` 限制查询结果显示的行数

### 2-2 课堂案例

#### 查看一张表所有的数据

- 1 `select` 列名, ..., 列名 `from` 表名;
- 2
- 3 简写:
- 4 如果查询一张表所有的列的数据, 可以使用\*表示所有列

49 -- 查询所有的员工工资和  
50 -- 数值类型+null==>null  
51 SELECT \*,sal+IFNULL(comm,0) FROM emp;  
52  
53  
54  
55

物表表 Columns  
empno, int  
ename, var  
job, varcha  
mgr, int(11)  
hiredate, d  
sal, decima  
COMM, de  
deptno, int

1 结果 2 分析 3 信息 4 表数据 5 资料 6 历史 虚拟表: select语句查询结果集

empno	ename	job	mgr	hiredate	sal	COMM	deptno	sal+ifnull(comm,0)	列名
1001	甘宁	文员	1013	2000-12-17	8000.00	(NULL)	20	8000.00	
1002	蔡锦丝	销售员	1006	2001-02-20	16000.00	3000.00	30	19000.00	
1003	殷天正	销售员	1006	2001-02-22	12500.00	5000.00	30	17500.00	
1004	刘备	经理	1009	2001-04-02	29750.00	(NULL)	20	29750.00	
1005	谢逊	销售员	1006	2001-09-28	12500.00	14000.00	30	26500.00	
1006	关羽	经理	1009	2001-05-01	28500.00	(NULL)	30	28500.00	
1007	张飞	经理	1009	2001-09-01	24500.00	(NULL)	10	24500.00	
1008	诸葛亮	分析师	1004	2007-04-19	30000.00	(NULL)	20	30000.00	
1009	曾阿牛	董事长	(NULL)	2001-11-17	50000.00	(NULL)	10	50000.00	
1010	韦一笑	销售员	1006	2001-09-08	15000.00	0.00	30	15000.00	
1011	周泰	文员	1008	2007-05-23	11000.00	(NULL)	20	11000.00	
1012	檀香	文员	1006	2001-12-03	9500.00	(NULL)	30	9500.00	

select \*,sal+ifnull(comm,0) from emp LIMIT 0, 1000

批量查询成功完成 执行: 00:00:00:000 总计: 00:00:00:000 20 行 行 52 列 1 连接: 1

## 案例1：查询指定列、列起别名以及过滤列的重复值

```

1  USE k2502;
2  CREATE TABLE test(
3      -- 一张表必须有主键，列名可以没有意义
4      id INT PRIMARY KEY AUTO_INCREMENT,
5      tName VARCHAR(20) NOT NULL
6  );
7
8  -- insert 1000
9  INSERT test VALUES(NULL, 'aa'), (NULL, 'bb'), (DEFAULT, 'cc')
10
11 INSERT test VALUES(NULL, 'dd')
12 -- select
13 SELECT * FROM test;
14
15 -- delete
16 -- 自增长：删除的id，自动作废了
17 DELETE FROM test WHERE id=1002;
18
19 -- 表清空，重头开始
20 DELETE FROM test;
21
22 -- 截断表：自增长值，重置开始
23 -- 谨慎使用：底层不走数据库日志，数据删除“黑客”操作 无法还原
24 -- 底层实现：先删除表：drop 再创建：create
25 TRUNCATE TABLE test;
26
27
28 USE exam;
29 -- 查看所有的员工
30 -- *影响sql查询的性能，一般实际开发中，不推荐使用*
31 SELECT * FROM emp;
32 -- 查询所有
33 SELECT empno, ename, emp.job, emp.sal, emp.COMM, emp.mgr,
34 emp.hiredate, emp.deptno
35 FROM emp
36
37 -- 查看所有的部门
38 SELECT * FROM dept;
39

```

```

40 -- 查看所有的薪资等级
41 SELECT * FROM salgrade;
42
43
44 -- 查看所有的员工姓名和岗位
45 SELECT emp.ename,emp.job
46 FROM emp;
47
48 -- 起别名
49 -- 查询所有的员工工资和
50 -- 数值类型+null==>null
51 -- as 别名
52 SELECT *,sal+IFNULL(comm,0) AS 薪资和 FROM emp;
53 -- 简化方式 as省略
54 SELECT *,sal+IFNULL(comm,0) 'aaa' FROM emp;
55
56 -- 查询emp有哪些工作岗位
57 -- DISTINCT 过滤指定列的重复值
58 SELECT DISTINCT job FROM emp;

```

## 案例2：多条件查询

```

1  USE exam;
2
3  -- 查询底薪sal大于10000 小于15000
4  SELECT * FROM emp WHERE sal>10000 AND sal<15000;
5
6  -- 查询没有奖金的员工信息
7  SELECT * FROM emp WHERE comm IS NULL;
8
9  -- 查询有奖金的员工信息
10 SELECT * FROM emp WHERE comm IS NOT NULL;
11
12 -- 查询哪些员工是文员或经理
13 SELECT * FROM emp WHERE job='文员' OR job='经理'
14 -- 变形
15 SELECT * FROM emp WHERE job IN('文员','经理')
16
17 -- 查询员工信息，条件员工的姓名第二字'飞'
18 SELECT * FROM emp WHERE ename LIKE '_飞%';
19
20 -- 查询入职时间晚于2001年的员工信息
21 SELECT * FROM emp WHERE emp.hiredate > '2001-12-31'
22 -- year(列名)获取列值中年份部分值
23 SELECT * FROM emp WHERE YEAR(hiredate)> 2001;

```

## 案例3：分组查询

概念：将查询结果集按照指定列，列值相同的数据为一组，实现数据统计

分组应用场景：结合聚合函数

```

1  -- 统计：各个部门有几个员工
2  -- group by 归类，分析，聚合
3  -- group by 结合聚合函数使用，跟单独列放在一起查询，查询结果有歧义
4  -- 分组编写：select 聚合函数，分组依据 from
5  SELECT deptno,COUNT(*),MAX(sal),MIN(sal),AVG(sal)

```

```

6 FROM emp GROUP BY deptno
7
8 -- 查询公司各个岗位人数
9 SELECT job,COUNT(*)
10 FROM emp
11 GROUP BY job
12
13 -- 查询每年入职的员工人数
14 SELECT YEAR(hiredate),COUNT(*)
15 FROM emp
16 GROUP BY YEAR(hiredate)

```

## 案例4：排序查询

```

1 order by 排序列名 排序规则（ASC-升序 DESC-降序），如果没有指定排序规则，默认ASC
2
3 -- 3.6 查看雇员信息以及员工的月薪与佣金之和，并根据薪资和降序排列
4 SELECT *,sal+IFNULL(comm,0) total FROM emp ORDER BY total DESC
5
6 -- order by 列名 排序规则,列名 排序规则,....,列名 排序规则
7 -- order by 对查询结果的虚拟表列进行排序。
8 -- order by后面可以有多个排序列，但是排序原则：优先第一个列，第一个列相同的情况，
9 -- 才会使用第二个列进行排序，以此类推
10 -- 优先按照薪资降序排，如果薪资相同，再按照入职时间升序排序
11 SELECT *,sal+IFNULL(comm,0) total FROM emp ORDER BY total DESC,hiredate
    ASC,ename ASC

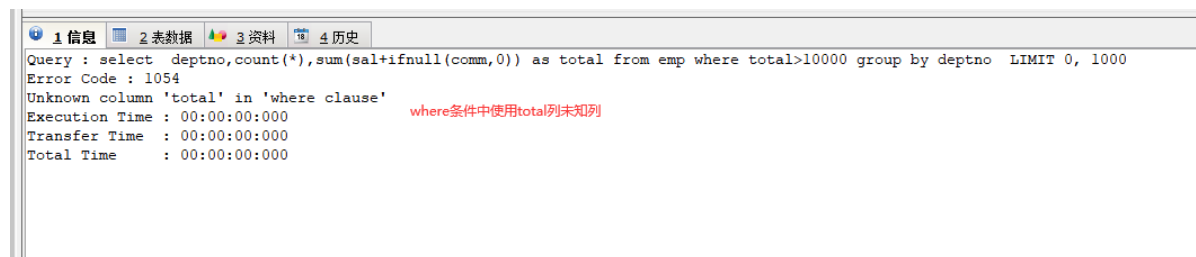
```

## 案例5：having二次筛选

### having和where有什么区别

where: 对分组前的列进行条件筛选，通常都是单列（没有被聚合函数包含的列）

having: 对分组后的查询结果中的列条件二次筛选。having通常跟聚合函数生成的列



```

1 -- 查询每个部门的部门编号以及每个部门工资和大于10000的部门人数
2 -- 1. 根据部门分组，查询各个部门部门人数、工资和
3 SELECT
4 deptno,COUNT(*) cnt,SUM(sal+IFNULL(comm,0)) AS total
5 FROM emp
6 -- where total>10000
7 GROUP BY deptno
8 HAVING total>10000
9 -- 排序：部门总人数降序
10 ORDER BY cnt DESC
11 -- 限制，只要一条
12 LIMIT 0,1

```

## 案例6：限制查询结果行数

```
1  -- mysql特有的关键字，分页核心实现方式
2  limit  开始显示的行对应下标,显示几行
3  -- 下标起始值从0开始，从第一行开始显示 对应下标0，以此类推
4
5  -- 查看所有的员工信息
6  -- 查询结果按照指定的行数显示
7  -- 从第一行开始显示，一共显示3行记录
8  SELECT * FROM emp LIMIT 0,3;
9
10 -- 第二页 3,3
11 SELECT * -- ③
12 FROM emp -- ①
13 WHERE -- ②
14 ORDER BY -- ④
15 LIMIT 3,3 -- ⑤
16
17 -- 第N页， 从第几行开始显示,3
18 SELECT * FROM emp LIMIT (N-1)*3,3
19
20 SELECT * FROM EMP LIMIT 6,3
21
22 SELECT * FROM EMP LIMIT 9,3
```

## DQL关键字的执行顺序

select和from必备关键字，其他关键字酌情出现

```
1  SELECT  普通列,COUNT() -- 查询 ⑤出普通列的值 ⑥出聚合结果的值
2  FROM    -- 来自 ①
3  WHERE   -- 条件 ②
4  GROUP BY -- 分组 ④
5  HAVING  -- 二次筛选 ⑥
6  ORDER BY -- 排序 ⑦
7  LIMIT   -- 限制 ⑧
```

## 3 聚合函数

作用：对select查询结果集进行聚合分析。

```
1  count():统计查询结果集的总行数
2  sum(列名):统计查询结果指定列的值求和
3  avg(列名):统计查询结果指定列的值平均值
4  max(列名):统计查询结果指定列的值最大值
5  min(列名):统计查询结果指定列的值最小值
```

补充:

```
1  now():获取当前时间
2  year():获取列值的年份
3  ifnull():如果指定列是null，设置默认值
```

## 课堂案例

```
1  -- 统计emp表有几个员工
2  -- 统计一个结果集有几行，使用count()
3  -- count(*)统计select * from emp查询结果的行数
4  -- count(10)统计select 10 from emp查询结果的行数
5  SELECT COUNT(*) FROM emp;
6
7  -- 统计这个公司一共有几位经理
8  -- 1.查询这个公司所有的经理的信息
9  -- 2.结果集上使用count()
10 SELECT COUNT(*) FROM emp WHERE job='经理'
11
12 -- 查询有提成的员工总数
13 -- SELECT * FROM emp WHERE comm IS NOT NULL;
14 SELECT COUNT(*) FROM emp WHERE comm IS NOT NULL;
15 -- 简化形式
16 -- count(列名):统计时忽略NULL值行
17 SELECT COUNT(comm) FROM emp;
18 SELECT comm FROM emp;
19
20 -- sum(列名): 列所有的值求和
21 -- 1.查询所有的员工，薪资sal和
22 SELECT sal FROM emp;
23
24 SELECT SUM(sal) FROM emp;
25
26 -- 2.提成
27 -- sum():求和忽略NULL值
28 SELECT SUM(sal) 底薪和,SUM(comm) 提成和 FROM emp;
29
30 -- 3.查询这个公司所有员工平均薪资
31 SELECT SUM(sal)/COUNT(sal) FROM emp;
32
33 -- 求平均值运行
34 SELECT AVG(sal) FROM emp;
35
36 -- 最低工资 最高工资，经理里面
37 SELECT MAX(sal),MIN(sal) FROM emp WHERE job='经理'
```

## 4 外键约束

作用：外键保证实体引用完整性

分析外键，引入E(entity 实体)-R(Relationship 关系)图例

### 外键概述

主表：主键所在的表就是主表

从表（子表）：外键所在的表

外键的特点：

- 1、外键列的值必须引用自主表的主键值
- 2、外键列的值可以为空



### 3、外键值可以重复

主键是构成表与表关联的唯一途径，有了这个途径才能实现多表的连接查询！

注意：外键其实就是另一张表的主键！例如员工表与部门表之间就存在关联关系，其中员工表中的部门编号字段就是外键，是相对部门表的外键。

## 演示案例

```
1 CREATE DATABASE testDb;
2 USE testDb;
3 -- 主表：一般是1的一方
4 CREATE TABLE grade(
5     gId INT PRIMARY KEY AUTO_INCREMENT,
6     gName VARCHAR(20)
7 );
8
9 -- 外键所在的表，称为子表 从表
10 CREATE TABLE student(
11     sId INT PRIMARY KEY AUTO_INCREMENT,
12     -- gid表示学生所属的班级编号，班级编号必须在班级表中实际存在
13     gId INT,
14     -- 约束 stu_fk给外键起的名称 外键(外键名称) 引用自 主表(主键)
15     CONSTRAINT stu_fk FOREIGN KEY(gid) REFERENCES grade(gid),
16     sname VARCHAR(10) NOT NULL
17 );
18 -- DDL修改表
19 ALTER TABLE student ADD CONSTRAINT stu_fk FOREIGN KEY(gid) REFERENCES
    grade(gid);
20
21 -- 外键的作用：保证数据引用完整性（理解：正确的数据）
22 -- 添加数据添加顺序的要求：先有父 再有子表
23 INSERT grade VALUES(1, 'k2502'), (2, 'k2503')
24 -- 外键约束：学生添加数据库中时，所在的班级编号是实际存在的，
25 INSERT student VALUES(NULL, 10, '张健');
```

## 5 实体映射关系

emp: 保存员工对象的表

dept: 保存部门对象的表

### 一对一

丈夫信息和妻子信息：一个丈夫只能有一个合法妻子，一个妻子只能有一个合法的丈夫

国家和最高领导人：

```
1 | 两张表的主键互为对方的外键
```

### 一对多\*

员工信息和部门信息：一对多关系（解读：一个部门中可以有多多个员工）

班级信息和学生信息：一对多关系（一个班级可以有多个学生）

- 1 | 外键创建在多的一方表中

## 多对多

学生信息和科目信息

- 1 | 必须创建一个独立表，管理学生和科目之间的外键关系

## 课程总结

---

### DQL内容

---

- 1 | 1. 查询所有
- 2 | 2. 起别名
- 3 | 3. 过滤指定列重复值 `distinct`
- 4 | 4. 多条件查询 `where`
- 5 | 5. 排序 `order by`
- 6 | 6. 限制 `limit`
- 7 | 7. 聚合函数 `count()` `sum()` `avg()` `max()` `min()`
- 8 | 8. 分组 `group by`
- 9 | 9. 二次筛选 `having`

### 理解外键的作用

---

### 了解：实体关系（1对多）

---