## Day\_04MySQL多表&事务

### 0.1学习目标

* 理解多表查询，并掌握内外连接
* 掌握多表查询的子查询
* 理解事务相关概念，记忆事务的四大特征与隔离级别
* 能够掌握数据库用户的crud操作

### 0.2 学习指南

* 多表查询（重要）
* 事务（重要）
* DCL管理用户（重要）

### 第一堂课

本节知识点

1.1今日内容

1.2 多表查询\_概述

1.3多表查询\_ 内连接

1.4多表查询\_外连接

本节目标

掌握多表查询

#### 今日内容

##### 1.1.1知识概述

今天的主要内容为：多表查询，事务，DDL

##### 1.1.2视频详情



##### 1.1.3总结与补充

无

##### 1.1.4课堂提问与练习

无

##### 1.1.5习题答案

无

##### 1.1.6视频缺陷

无

##### 1.1.7视频扩展

无

#### 1.2多表查询\_概述

##### 1.2.1知识概述

一次查询多张数据库表，分类：内连接查询，外链接查询，子查询

多表查询语法

查询语法：  
        select  
            列名列表  
        from  
            表名列表  
        where....

##### 1.2.2视频详情



##### 1.2.3总结与补充

在实际应用中尽量避免使用多表查询，数据量大的情况下影响效率

##### 1.2.4课堂提问与练习

多表查询有哪几种？

##### 1.2.5习题答案

内连接，外连接，子查询

##### 1.2.6视频缺陷

无

##### 1.2.7视频扩展

无

#### 1.3 多表查询\_内连接

##### 1.3.1知识概述

内连接是从结果表中删除与其他被[连接表](https://baike.baidu.com/item/%E8%BF%9E%E6%8E%A5%E8%A1%A8" \t "_blank)中没有匹配行的所有行，所以内连接可能会丢失信息。

\* 查询语法：  
        select  
            列名列表  
        from  
            表名列表  
        where....  
    \* 准备sql  
        # 创建部门表  
        CREATE TABLE dept(  
            id INT PRIMARY KEY AUTO\_INCREMENT,  
            NAME VARCHAR(20)  
        );  
        INSERT INTO dept (NAME) VALUES ('开发部'),('市场部'),('财务部');  
        # 创建员工表  
        CREATE TABLE emp (  
            id INT PRIMARY KEY AUTO\_INCREMENT,  
            NAME VARCHAR(10),  
            gender CHAR(1), -- 性别  
            salary DOUBLE, -- 工资  
            join\_date DATE, -- 入职日期  
            dept\_id INT,  
            FOREIGN KEY (dept\_id) REFERENCES dept(id) -- 外键，关联部门表(部门表的主键)  
        );  
        INSERT INTO emp(NAME,gender,salary,join\_date,dept\_id) VALUES('孙悟空','男',7200,'2013-02-24',1);  
        INSERT INTO emp(NAME,gender,salary,join\_date,dept\_id) VALUES('猪八戒','男',3600,'2010-12-02',2);  
        INSERT INTO emp(NAME,gender,salary,join\_date,dept\_id) VALUES('唐僧','男',9000,'2008-08-08',2);  
        INSERT INTO emp(NAME,gender,salary,join\_date,dept\_id) VALUES('白骨精','女',5000,'2015-10-07',3);  
        INSERT INTO emp(NAME,gender,salary,join\_date,dept\_id) VALUES('蜘蛛精','女',4500,'2011-03-14',1);  
    \* 笛卡尔积：  
        \* 有两个集合A,B .取这两个集合的所有组成情况。  
        \* 要完成多表查询，需要消除无用的数据

2. 显式内连接：  
                \* 语法： select 字段列表 from 表名1 [inner] join 表名2 on 条件  
                \* 例如：  
                    \* SELECT \* FROM emp INNER JOIN dept ON emp.`dept\_id` = dept.`id`;      
                    \* SELECT \* FROM emp JOIN dept ON emp.`dept\_id` = dept.`id`;      
  
            3. 内连接查询：  
                1. 从哪些表中查询数据  
                2. 条件是什么  
                3. 查询哪些字段

##### 1.3.2视频详情



##### 1.3.3总结与补充

**1.1.等值连接：**在连接条件中使用等于号(=)运算符比较被连接列的列值，其查询结果中列出被连接表中的所有列，包括其中的重复列。

**1.2.不等值连接：**在连接条件使用除等于运算符以外的其它比较运算符比较被连接的列的列值。这些运算符包括>、>=、<=、<、!>、!<和<>。

**1.3.自然连接：**在连接条件中使用等于(=)运算符比较被连接列的列值，但它使用选择列表指出查询结果集合中所包括的列，并删除连接表中的重复列。

##### 1.3.4课堂提问与练习

练习内连接操作

##### 1.3.5习题答案

无

##### 1.3.6视频缺陷

无

##### 1.3.7视频扩展

无

#### 1.4 多表查询\_外连接

##### 1.4.1知识概述

外连接分为：**左外连接，右外连接**

在左外连接和右外连接时都会以一张表为[基表](https://baike.baidu.com/item/%E5%9F%BA%E8%A1%A8" \t "_blank)，该表的内容会全部显示，然后加上两张表匹配的内容。如果基表的数据在另一张表没有记录。那么在相关联的[结果集](https://baike.baidu.com/item/%E7%BB%93%E6%9E%9C%E9%9B%86)行中列显示为[空值](https://baike.baidu.com/item/%E7%A9%BA%E5%80%BC)（NULL）。

1. 左外连接：  
                \* 语法：select 字段列表 from 表1 left [outer] join 表2 on 条件；  
                \* 查询的是左表所有数据以及其交集部分。  
                \* 例子：  
                    -- 查询所有员工信息，如果员工有部门，则查询部门名称，没有部门，则不显示部门名称  
                    SELECT     t1.\*,t2.`name` FROM emp t1 LEFT JOIN dept t2 ON t1.`dept\_id` = t2.`id`;  
            2. 右外连接：  
                \* 语法：select 字段列表 from 表1 right [outer] join 表2 on 条件；  
                \* 查询的是右表所有数据以及其交集部分。  
                \* 例子：  
                    SELECT     \* FROM dept t2 RIGHT JOIN emp t1 ON t1.`dept\_id` = t2.`id`;

##### 1.4.2视频详情



##### 1.4.3总结与补充

1）LEFT  JOIN或LEFT OUTER JOIN       
左向外联接的结果集包括  LEFT OUTER子句中指定的左表的所有行，而不仅仅是联接列所匹配的行。如果左表的某行在右表中没有匹配行，则在相关联的结果集行中右表的所有选择列表列均为空值。         
2）RIGHT  JOIN 或 RIGHT  OUTER  JOIN       
右向外联接是左向外联接的反向联接。将返回右表的所有行。如果右表的某行在左表中没有匹配行，则将为左表返回空值。

##### 1.4.4课堂提问与练习

练习外连接操作

##### 1.4.5习题答案

无

##### 1.4.6视频缺陷

无

##### 1.4.7视频扩展

无

### 第二堂课

本节知识点：

1. 多表查询子查询概述
2. 多表查询子查询情况1&情况2
3. 多表查询子查询情况3
4. 多表查询\_练习1

本节目标 ：

掌握多表查询子查询

#### 2.1 多表查询子查询概述

##### 2.1.1知识概述

查询中嵌套查询，称嵌套查询为子查询

##### 2.1.2视频详情



##### 2.1.3总结与补充

在SQL语言中，一个SELECT-FROM-WHERE语句称为一个查询块。当获得一个查询的答案需要多个步骤的操作，首先必须创建一个查询来确定用户不知道但包含在数据库中的值，将一个查询块嵌套在另一个查询块的WHERE字句或HAVING短语的条件中查询块称为子查询或内层查询。

##### 2.1.4课堂提问与练习

无

##### 2.1.5习题答案

无

##### 2.1.6视频缺陷

无

##### 2.1.7视频扩展

无

#### 2.2多表查询子查询情况1&情况2

##### 2.2.1知识概述

1.子查询的结果是单行单列的：  
                    \* 子查询可以作为条件，使用运算符去判断。 运算符： > >= < <= =  
                    \*   
                    -- 查询员工工资小于平均工资的人  
                    SELECT \* FROM emp WHERE emp.salary < (SELECT AVG(salary) FROM emp);

 2.子查询的结果是多行单列的：  
                    \* 子查询可以作为条件，使用运算符in来判断  
                    -- 查询'财务部'和'市场部'所有的员工信息  
                    SELECT id FROM dept WHERE NAME = '财务部' OR NAME = '市场部';  
                    SELECT \* FROM emp WHERE dept\_id = 3 OR dept\_id = 2;  
                    -- 子查询  
                    SELECT \* FROM emp WHERE dept\_id IN (SELECT id FROM dept WHERE NAME = '财务部' OR NAME = '市场部');

##### 2.2.2视频详情



##### 2.2.3总结与补充

多表查询要找到两张表的对应字段。并根据实际情况来使用运算符。

##### 2.2.4课堂提问与练习

无

##### 2.2.5习题答案

无

##### 2.2.6视频缺陷

无

##### 2.2.7视频扩展

无

#### 2.3 多表查询子查询情况3

##### 2.3.1知识概述

子查询的结果是多行多列的：  
                    \* 子查询可以作为一张虚拟表参与查询  
                    -- 查询员工入职日期是2011-11-11日之后的员工信息和部门信息  
                    -- 子查询  
                    SELECT \* FROM dept t1 ,(SELECT \* FROM emp WHERE emp.`join\_date` > '2011-11-11') t2  
                    WHERE t1.id = t2.dept\_id;

##### 2.3.2视频详情



##### 2.3.3总结与补充

根据业务需要，要判定虚拟表需要查询出那些字段。

##### 2.3.4课堂提问与练习

1、练习多表查询子查询（10分钟）

##### 2.3.5习题答案

详情见笔记

##### 2.3.6视频缺陷

无

##### 2.3.7视频扩展

无

#### 2.4 多表查询练习1

##### 2.4.1知识概述

-- 部门表  
                CREATE TABLE dept (  
                  id INT PRIMARY KEY PRIMARY KEY, -- 部门id  
                  dname VARCHAR(50), -- 部门名称  
                  loc VARCHAR(50) -- 部门所在地  
                );  
                  
                -- 添加4个部门  
                INSERT INTO dept(id,dname,loc) VALUES   
                (10,'教研部','北京'),  
                (20,'学工部','上海'),  
                (30,'销售部','广州'),  
                (40,'财务部','深圳');  
                  
                  
                  
                -- 职务表，职务名称，职务描述  
                CREATE TABLE job (  
                  id INT PRIMARY KEY,  
                  jname VARCHAR(20),  
                  description VARCHAR(50)  
                );  
                  
                -- 添加4个职务  
                INSERT INTO job (id, jname, description) VALUES  
                (1, '董事长', '管理整个公司，接单'),  
                (2, '经理', '管理部门员工'),  
                (3, '销售员', '向客人推销产品'),  
                (4, '文员', '使用办公软件');  
                  
                  
                  
                -- 员工表  
                CREATE TABLE emp (  
                  id INT PRIMARY KEY, -- 员工id  
                  ename VARCHAR(50), -- 员工姓名  
                  job\_id INT, -- 职务id  
                  mgr INT , -- 上级领导  
                  joindate DATE, -- 入职日期  
                  salary DECIMAL(7,2), -- 工资  
                  bonus DECIMAL(7,2), -- 奖金  
                  dept\_id INT, -- 所在部门编号  
                  CONSTRAINT emp\_jobid\_ref\_job\_id\_fk FOREIGN KEY (job\_id) REFERENCES job (id),  
                  CONSTRAINT emp\_deptid\_ref\_dept\_id\_fk FOREIGN KEY (dept\_id) REFERENCES dept (id)  
                );  
                  
                -- 添加员工  
                INSERT INTO emp(id,ename,job\_id,mgr,joindate,salary,bonus,dept\_id) VALUES   
                (1001,'孙悟空',4,1004,'2000-12-17','8000.00',NULL,20),  
                (1002,'卢俊义',3,1006,'2001-02-20','16000.00','3000.00',30),  
                (1003,'林冲',3,1006,'2001-02-22','12500.00','5000.00',30),  
                (1004,'唐僧',2,1009,'2001-04-02','29750.00',NULL,20),  
                (1005,'李逵',4,1006,'2001-09-28','12500.00','14000.00',30),  
                (1006,'宋江',2,1009,'2001-05-01','28500.00',NULL,30),  
                (1007,'刘备',2,1009,'2001-09-01','24500.00',NULL,10),  
                (1008,'猪八戒',4,1004,'2007-04-19','30000.00',NULL,20),  
                (1009,'罗贯中',1,NULL,'2001-11-17','50000.00',NULL,10),  
                (1010,'吴用',3,1006,'2001-09-08','15000.00','0.00',30),  
                (1011,'沙僧',4,1004,'2007-05-23','11000.00',NULL,20),  
                (1012,'李逵',4,1006,'2001-12-03','9500.00',NULL,30),  
                (1013,'小白龙',4,1004,'2001-12-03','30000.00',NULL,20),  
                (1014,'关羽',4,1007,'2002-01-23','13000.00',NULL,10);  
                  
                  
                  
                -- 工资等级表  
                CREATE TABLE salarygrade (  
                  grade INT PRIMARY KEY,   -- 级别  
                  losalary INT,  -- 最低工资  
                  hisalary INT -- 最高工资  
                );  
                  
                -- 添加5个工资等级  
                INSERT INTO salarygrade(grade,losalary,hisalary) VALUES   
                (1,7000,12000),  
                (2,12010,14000),  
                (3,14010,20000),  
                (4,20010,30000),  
                (5,30010,99990);  
                  
                -- 需求：  
                  
                -- 1.查询所有员工信息。查询员工编号，员工姓名，工资，职务名称，职务描述  
                /\*  
                    分析：  
                        1.员工编号，员工姓名，工资，需要查询emp表  职务名称，职务描述 需要查询job表  
                        2.查询条件 emp.job\_id = job.id  
                  
                \*/  
                SELECT   
                    t1.`id`, -- 员工编号  
                    t1.`ename`, -- 员工姓名  
                    t1.`salary`,-- 工资  
                    t2.`jname`, -- 职务名称  
                    t2.`description` -- 职务描述  
                FROM   
                    emp t1, job t2  
                WHERE   
                    t1.`job\_id` = t2.`id`;  
                  
                  
                  
                -- 2.查询员工编号，员工姓名，工资，职务名称，职务描述，部门名称，部门位置  
                /\*  
                    分析：  
                        1. 员工编号，员工姓名，工资 emp  职务名称，职务描述 job  部门名称，部门位置 dept  
                        2. 条件： emp.job\_id = job.id and emp.dept\_id = dept.id  
                \*/  
                  
                SELECT   
                    t1.`id`, -- 员工编号  
                    t1.`ename`, -- 员工姓名  
                    t1.`salary`,-- 工资  
                    t2.`jname`, -- 职务名称  
                    t2.`description`, -- 职务描述  
                    t3.`dname`, -- 部门名称  
                    t3.`loc` -- 部门位置  
                FROM   
                    emp t1, job t2,dept t3  
                WHERE   
                    t1.`job\_id` = t2.`id` AND t1.`dept\_id` = t3.`id`;

##### 2.4.2视频详情



##### 2.4.3总结与补充

首先要分析清除业务需求，分析清楚用多少张表并且需要什么数据来作为子查询表以及对应字段的匹配。

##### 2.4.4课堂提问与练习

1、练习多表查询练习1（10分钟）

##### 2.4.5习题答案

详情见笔记

##### 2.4.6视频缺陷

无

##### 2.4.7视频扩展

无

### 第三堂课

本节知识点

3.1 多表查询练习2

3.2 多表查询练习3

3.3 事务基本演示

本节目标

掌握多表查询，理解事务

#### 3.1多表查询\_练习2

##### 3.1.1知识概述

-- 3.查询员工姓名，工资，工资等级  
                /\*  
                    分析：  
                        1.员工姓名，工资 emp  工资等级 salarygrade  
                        2.条件 emp.salary >= salarygrade.losalary and emp.salary <= salarygrade.hisalary  
                            emp.salary BETWEEN salarygrade.losalary and salarygrade.hisalary  
                \*/  
                SELECT   
                    t1.ename ,  
                    t1.`salary`,  
                    t2.\*  
                FROM emp t1, salarygrade t2  
                WHERE t1.`salary` BETWEEN t2.`losalary` AND t2.`hisalary`;  
                  
                  
                  
                -- 4.查询员工姓名，工资，职务名称，职务描述，部门名称，部门位置，工资等级  
                /\*  
                    分析：  
                        1. 员工姓名，工资 emp ， 职务名称，职务描述 job 部门名称，部门位置，dept  工资等级 salarygrade  
                        2. 条件： emp.job\_id = job.id and emp.dept\_id = dept.id and emp.salary BETWEEN salarygrade.losalary and salarygrade.hisalary  
                              
                \*/  
                SELECT   
                    t1.`ename`,  
                    t1.`salary`,  
                    t2.`jname`,  
                    t2.`description`,  
                    t3.`dname`,  
                    t3.`loc`,  
                    t4.`grade`  
                FROM   
                    emp t1,job t2,dept t3,salarygrade t4  
                WHERE   
                    t1.`job\_id` = t2.`id`   
                    AND t1.`dept\_id` = t3.`id`  
                    AND t1.`salary` BETWEEN t4.`losalary` AND t4.`hisalary`;

##### 3.1.2视频详情



##### 3.1.3总结与补充

分析表与表之间字段对应关系，利用对应字段查询出想要的结果数据。

##### 3.1.4课堂提问与练习

* 完成多表查询练习2

##### 3.1.5习题答案

参考课堂笔记

##### 3.1.6视频缺陷

无

##### 3.1.7视频扩展

无

#### 3.2多表查询\_练习3

##### 3.2.1知识概述

-- 5.查询出部门编号、部门名称、部门位置、部门人数  
                  
                /\*  
                    分析：  
                        1.部门编号、部门名称、部门位置 dept 表。 部门人数 emp表  
                        2.使用分组查询。按照emp.dept\_id完成分组，查询count(id)  
                        3.使用子查询将第2步的查询结果和dept表进行关联查询  
                          
                \*/  
                SELECT   
                    t1.`id`,t1.`dname`,t1.`loc` , t2.total  
                FROM   
                    dept t1,  
                    (SELECT  
                        dept\_id,COUNT(id) total  
                    FROM   
                        emp  
                    GROUP BY dept\_id) t2  
                WHERE t1.`id` = t2.dept\_id;  
                  
                  
                -- 6.查询所有员工的姓名及其直接上级的姓名,没有领导的员工也需要查询  
                  
                /\*  
                    分析：  
                        1.姓名 emp， 直接上级的姓名 emp  
                            \* emp表的id 和 mgr 是自关联  
                        2.条件 emp.id = emp.mgr  
                        3.查询左表的所有数据，和 交集数据  
                            \* 使用左外连接查询  
                      
                \*/  
                /\*  
                select  
                    t1.ename,  
                    t1.mgr,  
                    t2.`id`,  
                    t2.ename  
                from emp t1, emp t2  
                where t1.mgr = t2.`id`;  
                  
                \*/  
                  
                SELECT   
                    t1.ename,  
                    t1.mgr,  
                    t2.`id`,  
                    t2.`ename`  
                FROM emp t1  
                LEFT JOIN emp t2  
                ON t1.`mgr` = t2.`id`;

##### 3.2.2视频详情



##### 3.2.3总结与补充

对外连接的进一步应用。关键字为left(right) join on，以左侧为主表，主表数据全部展示。

##### 3.2.4课堂提问与练习

练习多表查询练习3

##### 3.2.5习题答案

参考笔记

##### 3.2.6视频缺陷

无

##### 3.2.7视频扩展

无

#### 3.3事务基本演示

##### 3.3.1知识概述

如果一个包含多个步骤的业务操作，被事务管理，那么这些操作要么同时成功，要么同时失败。

[数据库事务](https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%BA%93%E4%BA%8B%E5%8A%A1)(Database Transaction) ，是指作为单个逻辑工作单元执行的一系列[操作](https://baike.baidu.com/item/%E6%93%8D%E4%BD%9C/33052)，要么完全地执行，要么完全地不执行。 事务处理可以确保除非事务性单元内的所有操作都成功完成，否则不会永久更新面向数据的资源。通过将一组相关操作组合为一个要么全部成功要么全部失败的单元，可以简化错误恢复并使应用程序更加可靠。一个逻辑工作单元要成为事务，必须满足所谓的ACID（原子性、一致性、隔离性和持久性）属性。事务是数据库运行中的逻辑工作单位，由DBMS中的事务管理子系统负责事务的处理。

##### 3.3.2视频详情



##### 3.3.3总结与补充

事务提供了一种机制，可用来将一系列数据库更改归入一个逻辑操作。更改数据库后，所做的更改可以作为一个单元进行提交或取消。事务可确保遵循原子性、一致性、隔离性和持续性（ACID）这几种属性，以使数据能够正确地提交到数据库中。  
使用事务机制的好处非常明显，例如银行转账之类的交易操作中，事务有着重要的作用。事务的成功取决于事务单元帐户相互依赖的操作行为是否能全部执行成功，只要有一个操作行为失败，整个事务将失败。例如：客户A和客户B的银行账户金额都是10000元人民币，客户A需要把自己帐户中的5000元人民币转到客户B的账户上。这个过程看似简单，实际上涉及了一系列的数据库操作，可以简单地视为两步基本操作，即从客户A帐户的金额中扣除5000元人民币，以及将客户B帐户中金额添加5000元人民币。假设第1步数据库操作成功，而第二步失败的话，将导致整个操作失败，并且客户A帐户金额将被扣除5000元人民币。事务机制可以避免此类情况，以保证整个操作的完成，如果某步操作出错，之前所作的数据库操作将全部失效。

##### 3.3.4课堂提问与练习

什么是事务？

##### 3.3.5习题答案

[数据库事务](https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%BA%93%E4%BA%8B%E5%8A%A1)(Database Transaction) ，是指作为单个逻辑工作单元执行的一系列[操作](https://baike.baidu.com/item/%E6%93%8D%E4%BD%9C/33052)，要么完全地执行，要么完全地不执行。 事务处理可以确保除非事务性单元内的所有操作都成功完成，否则不会永久更新面向数据的资源。通过将一组相关操作组合为一个要么全部成功要么全部失败的单元，可以简化错误恢复并使应用程序更加可靠。一个逻辑工作单元要成为事务，必须满足所谓的ACID（原子性、一致性、隔离性和持久性）属性。事务是数据库运行中的逻辑工作单位，由DBMS中的事务管理子系统负责事务的处理。

##### 3.3.6视频缺陷

无

##### 3.3.7视频扩展

无

### 第四堂课

本节知识点：

4.1 事务\_默认自动提交&手动提交

4.2 事务\_事务的四大特征

4.3 事务\_事务的隔离级别介绍

4.4 事务\_事务隔离级别演示1

本节目标 ：

完成事务提交方式修改

记忆事务四大特征

了解事务的隔离级别

完成事务隔离级别演示

#### 事务\_默认自动提交&手动提交

##### 4.1.1知识概述

事务提交的两种方式：  
                \* 自动提交：  
                    \* mysql就是自动提交的  
                    \* 一条DML(增删改)语句会自动提交一次事务。  
                \* 手动提交：  
                    \* Oracle 数据库默认是手动提交事务  
                    \* 需要先开启事务，再提交  
            \* 修改事务的默认提交方式：  
                \* 查看事务的默认提交方式：SELECT @@autocommit; -- 1 代表自动提交  0 代表手动提交  
                \* 修改默认提交方式： set @@autocommit = 0;

##### 4.1.2视频详情



##### 4.1.3总结与补充

事务有两种提交方式：自动提交与手动提交

手动提交需要开启事务START TRANSACTION; ，操作完成后，还必须提交事务COMMIT;，修改后的数据才会生效。

##### 4.1.4课堂提问与练习

* 练习查询及修改事务提交方式

##### 4.1.5习题答案

 查看事务的默认提交方式：SELECT @@autocommit; -- 1 代表自动提交  0 代表手动提交  
                \* 修改默认提交方式： set @@autocommit = 0;

##### 4.1.6视频缺陷

无

##### 4.1.7视频扩展

无

#### 事务\_事务的四大特征

##### 4.2.1知识概述

1. 原子性：是不可分割的最小操作单位，要么同时成功，要么同时失败。  
        2. 持久性：当事务提交或回滚后，数据库会持久化的保存数据。  
        3. 隔离性：多个事务之间。相互独立。  
        4. 一致性：事务操作前后，数据总量不变

##### 4.2.2视频详情



##### 4.2.3总结与补充

　　1．原子性是指事务包含的所有操作要么全部成功，要么全部失败回滚，这和前面两篇博客介绍事务的功能是一样的概念，因此事务的操作如果成功就必须要完全应用到数据库，如果操作失败则不能对数据库有任何影响。

　　一致性是指事务必须使数据库从一个一致性状态变换到另一个一致性状态，也就是说一个事务执行之前和执行之后都必须处于一致性状态。

　　2.拿转账来说，假设用户A和用户B两者的钱加起来一共是5000，那么不管A和B之间如何转账，转几次账，事务结束后两个用户的钱相加起来应该还得是5000，这就是事务的一致性。

　　3.隔离性是当多个用户并发访问数据库时，比如操作同一张表时，数据库为每一个用户开启的事务，不能被其他事务的操作所干扰，多个并发事务之间要相互隔离。

　　即要达到这么一种效果：对于任意两个并发的事务T1和T2，在事务T1看来，T2要么在T1开始之前就已经结束，要么在T1结束之后才开始，这样每个事务都感觉不到有其他事务在并发地执行。

　　关于事务的隔离性数据库提供了多种隔离级别，稍后会介绍到。

　　4.持久性是指一个事务一旦被提交了，那么对数据库中的数据的改变就是永久性的，即便是在数据库系统遇到故障的情况下也不会丢失提交事务的操作。

　　例如我们在使用JDBC操作数据库时，在提交事务方法后，提示用户事务操作完成，当我们程序执行完成直到看到提示后，就可以认定事务以及正确提交，即使这时候数据库出现了问题，也必须要将我们的事务完全执行完成，否则就会造成我们看到提示事务处理完毕，但是数据库因为故障而没有执行事务的重大错误。

##### 4.2.4课堂提问与练习

**事务的四大特征**

。

##### 4.2.5习题答案

参考笔记

##### 4.2.6视频缺陷

无

##### 4.2.7视频扩展

无

#### 4.3事务\_事务的隔离级别介绍

##### 4.3.1知识概述

概念：多个事务之间隔离的，相互独立的。但是如果多个事务操作同一批数据，则会引发一些问题，设置不同的隔离级别就可以解决这些问题。

\* 存在问题：

1. 脏读：一个事务，读取到另一个事务中没有提交的数据

2. 不可重复读(虚读)：在同一个事务中，两次读取到的数据不一样。

3. 幻读：一个事务操作(DML)数据表中所有记录，另一个事务添加了一条数据，则第一个事务查询不到自己的修改。

\* 隔离级别：

1. read uncommitted：读未提交

\* 产生的问题：脏读、不可重复读、幻读

2. read committed：读已提交 （Oracle）

\* 产生的问题：不可重复读、幻读

3. repeatable read：可重复读 （MySQL默认）

\* 产生的问题：幻读

4. serializable：串行化

\* 可以解决所有的问题

\* 注意：隔离级别从小到大安全性越来越高，但是效率越来越低

\* 数据库查询隔离级别：

\* select @@tx\_isolation;

\* 数据库设置隔离级别：

\* set global transaction isolation level 级别字符串;

\* 演示：

set global transaction isolation level read uncommitted;

start transaction;

-- 转账操作

update account set balance = balance - 500 where id = 1;

update account set balance = balance + 500 where id = 2;

##### 4.3.2视频详情



##### 4.3.3总结与补充

数据库有四种隔离级别（read uncommitted：读未提交；read committed：读已提交 （Oracle）；repeatable read：可重复读 （MySQL默认）； serializable：串行化），隔离级别越高性能越差，所以一般不用最高级别。

##### 4.3.4课堂提问与练习

* 事务有几种隔离级别

##### 4.3.5习题答案

 隔离级别：  
            1. read uncommitted：读未提交  
                \* 产生的问题：脏读、不可重复读、幻读  
            2. read committed：读已提交 （Oracle）  
                \* 产生的问题：不可重复读、幻读  
            3. repeatable read：可重复读 （MySQL默认）  
                \* 产生的问题：幻读  
            4. serializable：串行化  
                \* 可以解决所有的问题

##### 4.3.6视频缺陷

无

##### 4.3.7视频扩展

无

#### 4.4事务\_事务隔离级别演示1

##### 4.4.1知识概述

set global transaction isolation level read uncommitted;  
            start transaction;  
            -- 转账操作  
            update account set balance = balance - 500 where id = 1;  
            update account set balance = balance + 500 where id = 2;

commit

##### 4.4.2视频详情



##### 4.4.3总结与补充

每次修改完隔离级别后，都要重启mysql才可以生效。

##### 4.4.4课堂提问与练习

* 事务隔离级别练习

##### 4.4.5习题答案

参考笔记。

##### 4.4.6视频缺陷

无

##### 4.4.7视频扩展

无

### 第五堂课

本节知识点：

5.1 事务\_事务隔离级别演示2

5.2 DCL\_管理用户\_增删查

5.3 DCL\_管理用户\_修改密码

本节目标 ：

1、能够完成事务隔离级别演示

2、能够完成DCL管理用户CRUD操作

#### 事务\_事务隔离级别演示2

##### 5.1.1知识概述

修改事务的隔离级别：

set global transaction isolation level 级别字符串;

练习时一定要重连mysql并且先开启事务：

start transaction;

操作完成后需手动提交事务：

Commit；

##### 5.1.2视频详情



##### 5.1.3总结与补充

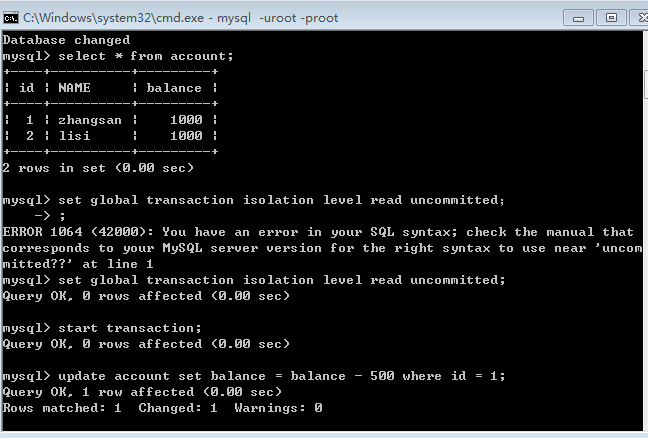
只有重新连接mysql后，隔离级别才可以生效

##### 5.1.4课堂提问与练习

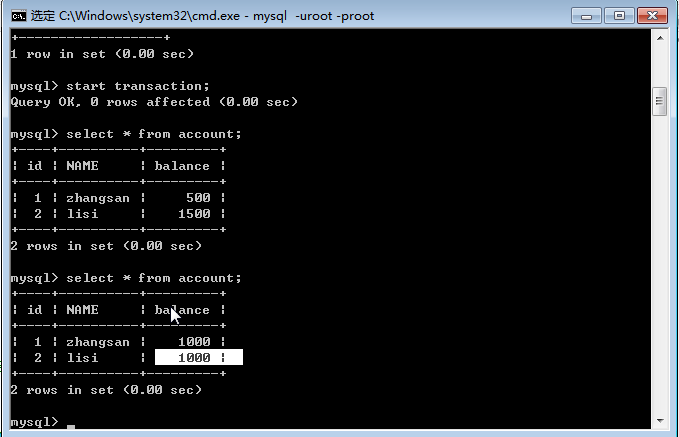
* 事务隔离级别练习

##### 5.1.5习题答案

修改数据：



查询数据（重新连接mysql）:



##### 5.1.6视频缺陷

无

##### 5.1.7视频扩展

无

#### DCL\_管理用户\_增删查

##### 5.2.1知识概述

 SQL分类：  
        1. DDL：操作数据库和表  
        2. DML：增删改表中数据  
        3. DQL：查询表中数据  
        4. DCL：管理用户，授权  
  
    \* DBA：数据库管理员  
  
    \* DCL：管理用户，授权  
        1. 管理用户  
            1. 添加用户：  
                \* 语法：CREATE USER '用户名'@'主机名' IDENTIFIED BY '密码';  
            2. 删除用户：  
                \* 语法：DROP USER '用户名'@'主机名';  
            3. 修改用户密码：  
                  
                UPDATE USER SET PASSWORD = PASSWORD('新密码') WHERE USER = '用户名';  
                UPDATE USER SET PASSWORD = PASSWORD('abc') WHERE USER = 'lisi';  
                  
                SET PASSWORD FOR '用户名'@'主机名' = PASSWORD('新密码');  
                SET PASSWORD FOR 'root'@'localhost' = PASSWORD('123');  
  
                \* mysql中忘记了root用户的密码？  
                    1. cmd -- > net stop mysql 停止mysql服务  
                        \* 需要管理员运行该cmd  
  
                    2. 使用无验证方式启动mysql服务： mysqld --skip-grant-tables  
                    3. 打开新的cmd窗口,直接输入mysql命令，敲回车。就可以登录成功  
                    4. use mysql;  
                    5. update user set password = password('你的新密码') where user = 'root';  
                    6. 关闭两个窗口  
                    7. 打开任务管理器，手动结束mysqld.exe 的进程  
                    8. 启动mysql服务  
                    9. 使用新密码登录。  
            4. 查询用户：  
                -- 1. 切换到mysql数据库  
                USE myql;  
                -- 2. 查询user表  
                SELECT \* FROM USER;  
                  
                \* 通配符： % 表示可以在任意主机使用用户登录数据库

##### 5.2.2视频详情



##### 5.2.3总结与补充

添加用户的时候要设置用户的ip地址，可以有效防止密码被盗取后，其他人访问数据库，对数据库造成损坏，极大提高数据库的安全性

##### 5.2.4课堂提问与练习

* 练习数据库用户的增删查

##### 5.2.5习题答案

参考笔记

##### 5.2.6视频缺陷

无

##### 5.2.7视频扩展

无

#### DCL\_管理用户\_修改密码

##### 5.3.1识概述

修改用户密码：

UPDATE USER SET PASSWORD = PASSWORD('新密码') WHERE USER = '用户名';

UPDATE USER SET PASSWORD = PASSWORD('abc') WHERE USER = 'lisi';

SET PASSWORD FOR '用户名'@'主机名' = PASSWORD('新密码');

SET PASSWORD FOR 'root'@'localhost' = PASSWORD('123');

\* mysql中忘记了root用户的密码？

1. cmd -- > net stop mysql 停止mysql服务

\* 需要管理员运行该cmd

2. 使用无验证方式启动mysql服务： mysqld --skip-grant-tables

3. 打开新的cmd窗口,直接输入mysql命令，敲回车。就可以登录成功

4. use mysql;

5. update user set password = password('你的新密码') where user = 'root';

6. 关闭两个窗口

7. 打开任务管理器，手动结束mysqld.exe 的进程

8. 启动mysql服务

9. 使用新密码登录。

##### 5.3.2视频详情



##### 5.3.3总结与补充

用户修改密码一般需要向公司的上级申请，有可能大家都用一个账号密码来访问数据库，如果擅自改动可能会造成其他开发人员不能连接数据库。

##### 5.3.4课堂提问与练习

* 无

##### 5.3.5习题答案

无

##### 5.3.6视频缺陷

无

##### 5.3.7视频扩展

无

### 第六堂课

本节知识点：

DCL\_管理权限

本节目标 ：

1. 能够完成数据库用户权限管理

#### DCL\_管理权限

##### 6.1.1知识概述

1. 查询权限：  
                -- 查询权限  
                SHOW GRANTS FOR '用户名'@'主机名';  
                SHOW GRANTS FOR 'lisi'@'%';  
  
            2. 授予权限：  
                -- 授予权限  
                grant 权限列表 on 数据库名.表名 to '用户名'@'主机名';  
                -- 给张三用户授予所有权限，在任意数据库任意表上  
                  
                GRANT ALL ON \*.\* TO 'zhangsan'@'localhost';  
            3. 撤销权限：  
                -- 撤销权限：  
                revoke 权限列表 on 数据库名.表名 from '用户名'@'主机名';  
                REVOKE UPDATE ON db3.`account` FROM 'lisi'@'%';

##### 6.1.2视频详情



##### 6.1.3总结与补充

权限管理极大提高了数据库的安全性，有效降低了数据库数据被误删的几率。

授权的关键字为:   grant 权限列表 on 数据库名.表名 to '用户名'@'主机名';  
撤销权限的关键字为：revoke 权限列表 on 数据库名.表名 from '用户名'@'主机名';

##### 6.1.4课堂提问与练习

1、完成权限管理等操作

##### 6.1.5习题答案

参考笔记

##### 6.1.6视频缺陷

无

##### 6.1.7视频扩展

无