**第7课 存储过程、触发器的创建和使用**

|  |  |  |
| --- | --- | --- |
| **课 题** | 存储过程、触发器的创建和使用 | |
| **课 时** | 8课时（360 min）。 | |
| **教学目标** | **知识技能目标：**  1．了解存储过程创建及应用。  2．通过学习与练习掌握创建触发器。  **思政育人目标：**  让学生通过学习存储过程、触发器的创建和使用，了解存储过程、触发器的创建和使用的相关知识。 | |
| **教学重难点** | **教学重点：**存储过程创建及应用  **教学难点：**创建触发器 | |
| **教学方法** | 讲授法、问答法、讨论法 | |
| **教学用具** | 电脑、投影仪、多媒体课件、教材 | |
| **教学设计** | 第1节课：考勤（2min）--知识讲解（40min）--作业布置（3min）  第2节课：知识讲解（40min）--课堂小结（3min）--作业布置（2min）  第3节课：知识讲解（40min）--课堂小结（3min）--作业布置（2min）  第4节课：知识讲解（40min）--课堂小结（3min）--作业布置（2min）  第5节课：知识讲解（40min）--课堂小结（3min）--作业布置（2min）  第6节课：知识讲解（40min）--课堂小结（3min）--作业布置（2min）  第7节课：知识讲解（40min）--课堂小结（3min）--作业布置（2min）  第8节课：知识讲解（40min）--课堂小结（3min）--作业布置（2min） | |
| **教学过程** | **主 要 教 学 内 容 及 步 骤** | **设计意图** |
| **考勤**  **（2min）** | ■【教师】清点上课人数，记录好考勤  ■【学生】班干部报请假人员及原因 | 培养学生的组织纪律性,掌握学生的出勤情况 |
| **知识讲解**  （40min） | **【教师】**展示存储过程创建及应用（一）  **任务描述**  在对学籍管理系统进行开发过程中，用户根据需要经常会反复执行一些数据的插入、更新、查询等操作，为了提高开发和管理效率，项目开发组通过创建存储过程来把相关功能提前定义好，在需要的时候直接调用即可，避免了代码重复，本任务主要学习如何根据需要创建不同的存储过程来管理学籍管理系统数据库。  **任务目标**  （1）能够使用 SQL Server Management Studio 创建存储过程。  （2）能够使用 T-SQL 语句创建存储过程。  （3）能够调用存储过程管理数据库。  （4）理解带参数和不带参数存储过程的区别。  （5）理解存储过程的概念及使用场景。  （6）培养分析解决问题的能力，自信自强、守正创新。  **任务分析**  要创建存储过程，用户首先连接到 SQL Server 实例，然后根据需要选择使用 SQL Server Management Studio 或者 T-SQL 语句来创建相应的存储过程，然后进行应用。  **任务实施**  **一、使用模板创建存储过程**  需求：创建存储过程 myProce\_xh\_cj，显示 xjglxt 数据中存放的学号、姓名、课程  名称和成绩。  实施步骤如下：  （1）启动 SQL Server Management Studio，  在“对象资源管理器”中展开数据库节点，顺  序展开“xjglxt”和“可编程性”节点。  （2）选中“存储过程”节点，右击在弹出  的快捷菜单中单击“新建”，选择“存储过程  （S）…”命令选项。  （3）右边窗格显示存储过程的创建模板。  （4）根据需要修改编辑模板，加入如下代码，完成编辑后，单击“执行”按钮，出现“命令已成功完成”，在“对象资源管理器”可以刷新查看新建的存储过程。  CREATE PROCEDURE myProce\_xh\_cj  AS  BEGIN  SELECT xsxxb.xh AS 学号 ,xsxxb.xm AS 姓名 ,kcxxb.kcmc AS 课程名称 ,cjb.  cj AS 成绩  FROM xsxxb,kcxxb,cjb  W H E R E x s x x b . x h =  cjb.xh AND cjb.kcbh = kcxxb.  kcbh  END  GO  在 SQL Server Management  Studio 中执行存储过程，只需要  在“对象资源管理器”中找到目  标存储过程 myProce\_xh\_cj，单  击鼠标右键，选择“执行存储过  程（E）…”菜单命令，在弹出  的“执行过程”对话框中单击“确  定” 按 钮， 即 可 在 SQL Server  Management Studio 右侧窗格看  到执行结果。  **二、使用 T-SQL 创建不带参数的存储过程**  需求：查看“数据库技术”课程考试平均成绩以及所有课程未通过考试的学生信息。  实施步骤如下：  （1）在查询编辑器中输入如下代码：  USE xjglxt  GO  /\* 检测是否存在存储过程 proc\_stu, 如果存在则删除 \*/  IF EXISTS (SELECT \* FROM sysobjects WHERE name = 'proc\_stu')  DROP PROCEDURE proc\_stu  GO  /\*--- 创建存储过程 ----\*/  CREATE PROCEDURE proc\_stu  AS  DECLARE @Avg fl oat – 存放平均成绩变量  SELECT @Avg = avg(cj) FROM cjb  WHERE kcbh = (SELECT kcbh FROM kcxxb WHERE kcmc = ' 数据库技术 ')  IF (@Avg >= 80)  SELECT 考试情况 = ' 本次考试比较理想 '  ELSE  SELECT 考试情况 = ' 本次考试成绩一般 '  SELECT ' 本次考试没有通过的学生信息为： '  SELECT xm 姓名 ,kcmc 课程名 ,cj 成绩 FROM xsxxb  INNER JOIN cjb ON xsxxb.xh = cjb.Xh  INNER JOIN kcxxb ON cjb.kcbh = kcxxb.kcbh  WHERE cjb.cj < 60 AND kcxxb.kcmc = ' 数据库技术 '  GO  /\*--- 调用存储过程 ----\*/  EXEC proc\_stu  （2）单击工具栏的 按钮或者按 F5 键。  **三、使用 T-SQL 创建带参数的存储过程**  需求：新建存储过程，查看某门课程考试平均分及该课程不及格学员信息。  实施步骤如下：  （1）在查询编辑器中输入如下代码：  USE xjglxt  GO  /\* 检测是否存在存储过程 proc\_withPara( 存储过程存放在系统表 sysobjects 中 )\*/  IF EXISTS (SELECT \* FROM sysobjects WHERE name = 'proc\_withPara')  DROP PROCEDURE proc\_withPara  GO  /\*--- 创建存储过程 ----\*/  CREATE PROCEDURE proc\_withPara  @subject varchar(10), /\* 课程名称参数 \*/  @studentName varchar(10) /\* 学生参数 \*/  AS  DECLARE @Avg fl oat /\* 平均分变量 \*/  SELECT @Avg = avg(cj) FROM cjb  WHERE kcbh = (SELECT kcbh FROM kcxxb WHERE kcmc = @subject)  IF (@Avg >= 80)  SELECT ' 本次考试比较理想 '  ELSE  SELECT ' 本次考试成绩一般 '  SELECT xm 姓名 ,kcmc 课程名 ,cj 成绩 FROM xsxxb  JOIN cjb ON xsxxb.xh = cjb.xh  JOIN kcxxb ON cjb.kcbh = kcxxb.kcbh  WHERE cjb.cj < 60 AND kcxxb.kcmc = @subject  GO  /\*--- 调用存储过程 ----\*/  EXEC proc\_withPara ' 数据库技术 ', ' 赵敏 '  --或者EXEC proc\_withPara @subject='数据库技术',@studentName = '赵敏'  --或者EXEC proc\_withPara @studentName='赵敏',@subject = '数据库技术'  （2）单击工具栏的 按钮或者按 F5 键，运行结果。  **【学生】**思考、讨论。 | **展示存储过程创建及应用（一），让学生更加仔细的阅读，从而激发学生的学习欲望。** |
| **作业布置**（3min） | **【教师】**布置课后作业  简述使用 T-SQL 创建带参数的存储过程。 | 通过课后练习，使学生巩固所学新知识 |
| **知识讲解**  （40min） | **【教师】**展示存储过程创建及应用（二）  **四、使用 T-SQL 创建带参数的存储过程，把输入参数设置默认值**  需求：新建存储过程，查看某门课程考试平均分及该课程指定学员成绩。  实施步骤如下：  （1）在查询编辑器中输入如下代码：  USE xjglxt  GO  IF EXISTS (SELECT \* FROM sysobjects WHERE name = 'proc\_  withDefaultPara')  DROP PROCEDURE proc\_withDefaultPara  GO  /\*--- 创建存储过程 ----\*/  CREATE PROCEDURE proc\_withDefaultPara  @subject varchar(20) = ' 数据库技术 ',  @studentName varchar(10) = ' 赵敏 '  AS  DECLARE @Avg fl oat  SELECT @Avg = avg(cj) FROM cjb WHERE kcbh = (SELECT kcbh FROM kcxxb WHERE kcmc = @subject)  IF(@Avg >= 80)  SELECT ' 本次考试比较理想 '  ELSE  SELECT ' 本次考试成绩一般 '  SELECT xm 姓名 ,kcmc 课程名 ,cj 成绩 FROM xsxxb  JOIN cjb ON xsxxb.xh = cjb.xh  JOIN kcxxb ON cjb.kcbh = kcxxb.kcbh  WHERE xsxxb.xm = @studentName AND kcxxb.kcmc = @subject  GO  /\*--- 调用存储过程 ----\*/  EXEC proc\_withDefaultPara 'Linux 基础 ',' 张三丰 ' -- 课程名称和学生名都  不用默认值  EXEC proc\_withDefaultPara ' 云计算技术 ' -- 学生名用默认值  /\*-- 课程名称用默认值 , 注意带形参名 --\*/  EXEC proc\_withDefaultPara @studentName = ' 殷丽 '  EXEC proc\_withDefaultPara -- 都采用默认值  （2）单击工具栏的 按钮或者按 F5 键，运行结果。  **五、使用 T-SQL 创建带输出参数的存储过程**  要求：调用存储过程后，返回未通过“数据库技术”考试的学生人数，以供其他  T-SQL 语句访问。  实施步骤如下：  （1）在查询编辑器中输入如下代码：  USE xjglxt  GO  /\*--- 检测是否存在 proc\_withOutputPara：存储过程存放在系统表 sysobjects  中 ---\*/  IF EXISTS (SELECT \* FROM sysobjects WHERE name = 'proc\_  withOutputPara')  DROP PROCEDURE proc\_withOutputPara  GO  /\*--- 创建存储过程 ----\*/  CREATE PROCEDURE proc\_withOutputPara  @subject varchar(10) = ' 数据库技术 ',  @count int output  AS  SELECT @count = count(\*) FROM cjb,kcxxb  WHERE cjb.kcbh = kcxxb.kcbh AND kcxxb.kcmc = @subject AND  cjb.cj < 60  GO  /\*--- 调用存储过程 ----\*/  DECLARE @count int,@sub varchar(10)  SET @sub = ' 数据库技术 '  EXEC proc\_withOutputPara @sub, @count = @count output  PRINT @sub + ' 考试不及格的共有 ' + convert(varchar(2),@count) + ' 人 '  GO  （2）单击工具栏的 按钮或者按 F5 键，运行结果。  **【学生】**思考、讨论。 | **通过教师讲解，了解存储过程创建及应用（二）的基本理论知识。** |
| **课堂小结**  （3min） | 【**教师**】**回顾和总结本节课的知识点。**  **这节课我们一起学习了存储过程创建及应用（二），让学生培养分析解决问题的能力，自信自强、守正创新。** | 通过对所学知识的回顾，培养学生的归纳总结能力 |
| **作业布置**（2min） | **【教师】**布置课后作业  简述使用 T-SQL 创建带输出参数的存储过程。 | 通过课后练习，使学生巩固所学新知识 |
| **知识讲解**（45min） | **【教师】**展示创建触发器（一）  **任务描述**  在对学籍管理系统进行开发过程中，避免了代码重复，除了使用存储过程来把经常反复执行的插入、更新、查询等操作中的业务规则提前定义好，在需要的时候直接调用外，对于数据库中一些复杂的约束，主要通过触发器来实现。本任务主要学习如何根据需要创建不同的触发器管理数据库。  **任务目标**  （1）能够使用 SQL Server Management Studio 创建触发器。  （2）能够使用 T-SQL 语句创建 INSERT 触发器管理数据库。  （3）能够使用 T-SQL 语句创建 DELETE 触发器管理数据库。  （4）能够使用 T-SQL 语句创建 UPDATE 触发器管理数据库。  （5）理解触发器的概念、工作原理及使用场景。  （6）理解触发器与存储过程的区别。  （7）引导学生自觉弘扬和践行社会主义核心价值观，理解科教兴国战略，培养爱国情怀。  **任务分析**  要创建触发器管理 xjglxt 数据库，用户首先连接到 SQL Server 实例，然后根据需要选择使用 SQL Server Management Studio 或者 T-SQL 语句来创建相应的触发器，然后进行应用。  **任务实施**  **一、使用 SQL Server Management Studio 模板创建触发器**  实施步骤如下：  （1）启动 SQL Server Management Studio，在“对象资源管理器”中展开数据库节点，  顺序展开“xjglxt”和“表”节点。  （2）选择要创建触发器的 xsxxb 表并展开表目录，右击“触发器”节点，单击“新  建触发器（N）…”命令，系统将打开查询编辑器，并按照触发器的格式显示编码。  （3）在查询编辑器中用户根据需要修改触发器名称，添加触发器内容，完成如下  触发器的编码，再单击“执行”按钮，在出现“命令已成功完成”的提示后，即完成创建。  CREATE TRIGGER myTrigger  ON dbo.xsxxb  AFTER INSERT,DELETE,UPDATE  PRINT ' 哦 , 管理员修改记录了！！ '  GO  **二、使用 T-SQL 创建 INAERT 触发器**  需求：解决银行取款问题，在 bankDB 数据库中，当用户向交易信息表（information）中插入一条交易信息时，应自动更新对应账户余额。  分析：应该在交易信息表上创建 INSERT 触发器，根据交易类型（transType）字段的值是“存入”还是“支取”进行“增加”还是“减少”对应账户的余额。如何获取插入数据行的交易类型以及卡号呢？可以检查系统自动创建的临时表 Inserted 表，该表保存了插入数据行的副本。  实施步骤如下：  （1）在查询编辑器中输入如下代码：  USE bankDB  GO  /\*---- 检测是否存在：触发器存放在系统表 sysobjects 中 --------\*/  IF EXISTS (SELECT name FROM sysobjects WHERE name = 'trig\_  information')  DROP TRIGGER trig\_information  GO  /\*---- 创建INSERT 触发器：在交易信息表information 上创建INSERT 触发器----\*/  CREATE TRIGGER trig\_information  ON information  AFTER INSERT  AS  BEGIN  /\*--- 定义变量：用于临时存放插入的卡号、交易类型、交易金额等 ----\*/  DECLARE @type CHAR(4),@outMoney MONEY, @myCardID char(20),@  balance MONEY  /\*--- 从 Inserted 临时表中获取插入的记录行信息：包括交易类型、卡号、交易金  额 --\*/  SELECT @type = transType,@outMoney = transMoney,@myCardID  =cardID FROM Inserted  /\*---根据交易类型是支取/存入,减少或增加账户表(bank)中对应卡号的余额--\*/  IF (@type = ' 支取 ')  UPDATE bank SET currentMoney =currentMoney-@outMoney WHERE  cardID=@myCardID  ELSE  UPDATE bank SET currentMoney=currentMoney+@outMoney WHERE  cardID=@myCardID  /\*--- 显示交易金额及余额 ---\*/  IF @@ERROR <> 0  BEGIN  PRINT ' 交易失败！ '  ROLLBACK TRANSACTION  RETURN  END  PRINT ' 交易成功！交易金额：' + CONVERT(varchar(20),@outMoney)  SELECT @balance = currentMoney FROM bank WHERE cardID = @myCardID  PRINT ' 卡 号 '+ @myCardID + ' 余 额：'+ CONVERT (varchar(20),@  balance)  END  GO  /\*-- 测试触发器插入测试数据：张三取钱 100, 李四存钱 5000 ---\*/  DELETE FROM information -- 删除历史数据  SET NOCOUNT ON -- 不显示 T － SQL 语句影响的记录行数  INSERT INTO information(cardID,transType,transMoney,transDate)  VALUES('10010001', ' 支取 ',100, '2020-03-08 12:10:30')  INSERT INTO information(cardID,transType,transMoney,transDate)  VALUES('10010002', ' 存入 ',5000, '2020-03-10 14:30:10')  -- 查看结果  SELECT \* FROM bank  SELECT \* FROM information  （2）单击工具栏的 按钮或者按 F5 键。  （3）执行后的结果。当张三取钱 100 元时（在交易表中插入交易记录），将触发交易表上的 INSERT 触发器，自动根据交易的类型，修改账户表中对应卡号的余额信息，并打印显示本次交易信息。  **【学生】**思考、讨论。 | **教师通过创建触发器（一）展示，让学生了解创建触发器（一）的基本理论知识。** |
| **课堂小结**  （3min） | 【**教师**】**回顾和总结本节课的知识点。**  **这节课我们一起学习了创建触发器（一），引导学生自觉弘扬和践行社会主义核心价值观，理解科教兴国战略，培养爱国情怀。** | 通过对所学知识的回顾，培养学生的归纳总结能力 |
| **作业布置**（2min） | 【**教师**】**布置课后作业**  **简述使用 T-SQL 创建 INAERT 触发器。** | 通过课后练习，使学生巩固所学新知识 |
| **知识讲解**（45min） | **【教师】**展示创建触发器（二）  **三、使用 T-SQL 创建 DELETE（删除）触发器**  需求：当删除交易信息表时，自动备份被删除的数据到表 backupTable 中。  分析：应在交易信息表上创建 DELETE 触发器，则被删除的数据可以从 deleted 表  中获取。  实施步骤如下：  （1）在查询编辑器中输入如下代码：  USE bankDB  GO  /\* 检测是否存在：触发器存放在系统表 sysobjects 中 \*/  IF EXISTS (SELECT name FROM sysobjects WHERE name = 'trig\_delete\_  information')  DROP TRIGGER trig\_delete\_information  GO  /\* 创建 DELETE 触发器：在交易信息表 information 上创建删除触发器 \*/  CREATE TRIGGER trig\_delete\_information  ON information  AFTER DELETE  AS  print ' 开始备份数据 , 请稍后 . . . . . . '  IF NOT EXISTS(SELECT \* FROM sysobjects WHERE name='backupTable')  SELECT \* INTO backupTable FROM deleted -- 从 deleted 表中获取被删  除的数据  ELSE  INSERT INTO backupTable SELECT \* FROM deleted  PRINT ' 备份数据成功 , 备份表中的数据为 : '  SELECT \* FROM backupTable  GO  /\*-- 测试触发器：删除数据 ---\*/  SET NOCOUNT ON -- 不显示 T － SQL 语句影响的记录行数  DELETE information  -- 查看结果  print ‘交易信息表中的数据：’  SELECT \* FROM information  （2）单击工具栏的 按  钮或者按 F5 键。  （3）执行后的输出结果。交易信息表中的数据被删空，备份到表 backupTable 中。  **四、使用 T-SQL 创建 UPDATE（更新）触发器**  需求：跟踪用户的交易，交易金额超过 20000 元，则取消交易，并给出错误提示。  分析：可以从交易信息表中直接获取用户每次的交易金额，也可以根据账户信息表中余额的变化来获取，交易的方式较多，用户可能用卡消费也可能用存折消费，存折的交易信息与卡略有不同，可能不会将交易信息存放在交易信息表中，而保存在其他表中。但存折和卡对应的账号余额只有一个，所以更安全的方案是，根据账户信息表（bank）中余额的变化来获取交易金额。为了获取余额的变化，应该在账户信息表上创建 UPDATE 触发器。  更新操作可以视为两步操作：  ● 删除更改前原有的数据行：删除的数据转移到 deleted 表中。  ● 再插入更改后的新行：插入的数据同时也保存在 inserted 表中。  既然更改前的原有数据保存在 deleted 表中，更改后的数据保存在 inserted 表中，只  需要将更改前后的余额进行比较，就可以知道交易金额的数量是否超过 20000 元了。  实施步骤如下：  （1）在查询编辑器中输入如下代码：  USE bankDB  GO  /\*---- 检测是否存在：触发器存放在系统表 sysobjects 中 --------\*/  IF EXISTS (SELECT name FROM sysobjects WHERE name = 'trig\_UPDATE\_  bank')  DROP TRIGGER trig\_UPDATE\_bank  GO  /\*---- 创建 UPDATE 触发器：在表 information 上创建更新触发器 ----\*/  CREATE TRIGGER trig\_UPDATE\_bank  ON bank  AFTER UPDATE  AS  DECLARE @beforeMoney MONEY,@afterMoney MONEY  SELECT @beforeMoney=currentMoney FROM deleted -- 获取交易前的余额  SELECT @afterMoney=currentMoney FROM Inserted -- 获取交易后的余额  IF ABS(@afterMoney-@beforeMoney) > 20000  BEGIN  print ' 交易金额 : '+ convert(varchar(8),ABS(@afterMoney - @  beforeMoney))  RAISERROR (' 每笔交易不能超过 2 万元 , 交易失败 ',16,1)  ROLLBACK TRANSACTION -- 回滚事务 , 撤销交易  END  GO  /\*-- 测试触发器：删除数据 ---\*/  SET NOCOUNT ON -- 不显示 T － SQL 语句影响的记录行数  UPDATE bank SET currentMoney = currentMoney + 25000 WHERE cardID  ='10010001'  GO  INSERT INTO information(cardID,transType,transMoney,transDate)  VALUES('10010002', ' 支取 ',30000, '2020-03-12 16:30:20')  GO  INSERT INTO information(cardID,transType,transMoney,transDate)  VALUES('10010002', ' 存入 ',5000, '2020-03-12 16:40:50')  GO  -- 查看结果  PRINT ' 账户信息表中的数据：'  SELECT \* FROM bank  PRINT ' 交易信息表中的数据：'  SELECT \* FROM information  （2）单击工具栏的 按钮或者按 F5 键。  （3）执行后的结果。  凭存折存钱时，执行 UPDATE 语句，触发 UPDATE 触发器。当凭卡取钱时，虽然  执行 INSERT INTO information…语句，但该语句触发了 information 表上的 INSERT 触发器，在该触发器中，执行了如下的更新语句：  UPDATE bank set currentMoney = currentMoney + @outMoney  WHERE cardID = myCardID  从而触发了 bank 表上的 UPDATE 触发器。所以凭卡取钱超过 20000 元，也会报告错误。当取钱没超过 20000 元时将顺利通过检查，如凭卡存入 5000 元。  **【学生】**思考、讨论。 | **教师通过创建触发器（二）展示，让学生了解创建触发器（二）的基本理论知识。** |
| **课堂小结**  （3min） | 【**教师**】**回顾和总结本节课的知识点。**  **这节课我们一起学习了创建触发器（二），让学生了解使用 T-SQL 创建 UPDATE（更新）触发器。** | 通过对所学知识的回顾，培养学生的归纳总结能力 |
| **作业布置**（2min） | 【**教师**】**布置课后作业**  **简述使用 T-SQL 创建 UPDATE（更新）触发器的方法。** | 通过课后练习，使学生巩固所学新知识 |
| **知识讲解**（45min） | **【教师】**展示知识链接  **一、存储过程概述**  存储过程（Procedure）类似于 C 语言中的函数，它是 SQL 语句和控制流语句的预编译集合。存储过程存储在数据库内，可由应用程序通过一个调用执行，而且允许用户声明变量、逻辑控制语句以及其他强大的编程功能。  存储过程可包含逻辑控制语句和数据操纵语句，它可以接收参数、输出参数以及返回状态值。存储过程可以只包含一条 SELECT 语句，也可以包含一系列使用控制流的SQL 语句，存储过程可以包含个别或全部的控制流结构。  存储过程在创建时即在服务器上进行编译，所以执行起来比单个 SQL 语句快，可以使用 EXECUTE 语句来运行存储过程。  SQL Server 提供了一些预编译的存储过程，用以管理 SQL Server 和显示有关数据库和用户信息。这些存储过程称为“系统存储过程”。  1. 使用存储过程的优点  （1）允许模块化程序设计，实现代码重用。  只需创建一次存储过程并将其存储在数据库中，以后即可在程序中调用该过程任意次。存储过程可由数据库编程方面有专长的人员创建，并可独立于程序源代码而单独修改。  （2）封装复杂操作。  当对数据库进行复杂操作时，可以用存储过程将复杂操作封装起来和数据库提供的事务处理结合一起使用。  （3）加快系统运行速度。  如果某操作需要大量的 T-SQL 代码或需要重复执行，存储过程将比 T-SQL 批代码的执行要快。将在创建存储过程时对其进行分析和优化，并可在首次执行该过程后使用该过程的内存中的版本。但如果使用 T-SQL 批代码，每次运行 T-SQL 语句时，都要从  客户端重复发送，并且在 SQL Server 每次执行这些语句时，都要对其进行编译和优化。  （4）减少网络流量。  创建使用存储过程后，一个需要数百行 T-SQL 代码的操作由一条执行过程代码的单独语句就可实现，而不需要在网络中发送数百行代码。  （5）可作为安全机制使用。  即使对于没有直接执行存储过程中语句的权限用户，也可授予他们执行该存储过程的权限。  2. 存储过程的分类  ● 系统存储过程（类似 C 语言中的系统函数）。  ● 扩展存储过程。  ● 用户定义的存储过程（类似 C 语言中的自定义函数）。  3. 常用系统存储过程  统存储过程是一组预编译的 T-SQL 语句，提供了管理数据库和更新的表机制，充当从系统表中检索信息的快捷方式。  通过配置 SQL Server，可以生成对象、用户和权限的信息和定义，这些信息和定义存储在系统表中。每个数据库都分别有一个包含配置信息的系统表集，用户数据库的系统表是在创建数据库时自动创建的。用户可以通过系统存储过程访问和更新系统表。  所有系统存储过程的名称都以“sp\_”开头。系统存储过程位于 master 数据库和msdb 数据库中。系统管理员拥有这些过程。可以在任何数据库中运行系统存储过程，但执行的结果会反映在当前数据库中。  4. 扩展存储过程  扩展存储过程是 SQL Server 实例可以动态加载和运行的 DLL。扩展存储过程是使用 SQL Server 扩展存储过程 API 编写的，可直接在 SQL Server 实例的地址空间中运行。扩展存储过程一般以“\_xp”为前缀。  5. 用户自定义的存储过程  用户自定义存储过程是由用户为完成某一特定任务而编写的存储过程，用户自定义存储过程存储在当前数据库中。在SQL Server 2016中用户自定义存储过程分为两种类型：T-SQL 和 CLR。  ● T-SQL 存储过程：是指保存的 T-SQL 语句集，可以接受和返回用户提供参数的存储过程。T-SQL 存储过程在 SQL Server 中多用于数据库业务逻辑的处理。  ● CLR 存储过程：是指对 Microsoft .NET Framework 公共语言运行时（CLR）方法引用，CLR 存储过程可以接受和返回用户提供参数的程序。它们在 .NET Framework 程序集中是作为类的公共静态方法实现的，可以通过 SQL Server 2016 数据库引擎直接运行。CLR 存储过程多用于实现基于数据的数值计算，WEB 电子商务等。  创建存储过程使用 CREATE PROCEDURE 定义，几乎所有可以写成批处理的 SQL语句都可以用来创建存储过程。  **二、触发器概述**  1. 触发器的功能  触发器的主要作用是，实现由主键和外键所不能保证的复杂的参照完整性和数据的一致性，除此之外触发器还有如下功能。  （1）强化约束。  触发器能够实现比 CHECK 语句更为复杂的约束。  约束和触发器在特殊情况下各有优势。触发器的主要好处在于它们可以包含使用T-SQL 代码的复杂处理逻辑。因此，触发器可以支持约束的所有功能，但它在所给出的功能上并不总是最好的方法。  （2）跟踪变化。  触发器可以侦测数据库内的操作从而不允许数据库中未经许可的指定更新和变化。  （3）级联运行。  触发器可以侦测数据库内操作，并自动地级联影响整个数据库的各项内容。实例某个表上的触发器中包含对另外一个表的数据操作，如删除、更新、插入。而该操作又导致该表上触发器被触发。触发器可通过数据库的相关表实现级联更改；不过，通过级联引用完整性约束可以更有效地执行这些更改。  2. 触发器的分类  SQL Server 2016 数据库提供两大类触发器：DDL 触发器和 DML 触发器。  （1）DDL 触发器。  当服务器和数据库中发生数据定义语言（DDL）事件时将调用 DDL 触发器。DDL触发器是 SQL Server 2016 的新增功能，一般用于在数据库中执行管理任务，在此不作重点学习。  （2）DML 触发器。  DML 触发器是当数据库服务器中发生数据操作语言（DML）事件时要执行的操作。  DML 触发器在以下方面非常有用：  ● DML 触发器可通过数据库中的相关表实现级联更改。  ● DML 触发器可以防止恶意或错误的 INSERT、UPDATE 以及 DELETE 操作，并强制执行比 CHECK 约束定义的限制更为复杂的其他限制。  ● DML 触发器可以评估数据修改前后表的状态，并根据该差异采取措施。  ● 一个表中的多个同类 DML 触发器（INSERT、UPDATE 或 DELETE）允许采取  多个不同的操作来响应同一个修改语句。  3. 触发器的工作原理  当对某一表进行修改，诸如 UPDATE、INSERT、DELETE 这些操作时，SQL Server就会自动执行触发器所定义的 SQL 语句，从而确保对数据的处理必须符合由这些 SQL语句所定义的规则。DML 触发器可分为以下几种：  ● INSERT 触发器：当向表中插入数据时触发，自动执行触发器所定义的 SQL 语句。  ● UPDATE 触发器：当更新表中某列、多列时触发，自动执行触发器所定义的SQL 语句。  ● DELETE 触发器：当删除表中记录时触发，自动执行触发器所定义的 SQL 语句。  每个触发器有两个特殊的表：插入表（inserted 表）和删除表（deleted 表）。这两个表是逻辑表，并且这两个表是由系统管理的，存储在内存中，不是存储在数据库中，因此，不允许用户直接对其修改。  这两个表的结构与被该触发器作用的表有相同的表结构。这两个表是动态驻留在内存中的，当触发器工作完成，它们也被删除。这两个表主要保存因用户操作而被影响到的原数据值或新数据值。另外这两个表是只读的，即用户不能向其写入内容，但可以引用表中的数据。  ● deleted 表： 用 于 存 储 DELETED 和 UPDATE 语 句 所 影 响 的 行 的 副 本， 即在 deleted 表中临时保存了被删除或被更新前的记录行。在执行 DELETE 或UPDATE 语句时，行从触发器表中删除，并传输到 deleted 表中。  ● inserted 表： 用 于 存 储 INSERT 和 UPDATE 语 句 所 影 响 的 行 的 副 本， 即 在inserted 表中临时保存了被插入或被更新后的刻录行。在执行 INSERT 或UPDATE 语句时，新加行被同时添加到 inserted 表和触发器表中。  更新（UPDATE）语句类似于在删除之后执行插入：首先旧行被复制到 deleted 表中，然后新行被复制到触发器表和 inserted 表中。  综上所述，inserted 表和 deleted 表用于临时存放对表中数据行的修改信息，它们在具体的增加、删除、更新操作时的情况触发器可以禁止或回滚违反引用完整性的更改，从而取消所尝试的数据修改。当新值与主键不匹配时，此类触发器就可能发生作用。如果触发器表上存在约束，则执行INSERT、UPDATE 及 DELETE 触发器执行前检查这些约束。如果不满足约束，则不执行 INSERT、UPDATE 及 DELETE 触发器。  提示：INSERT、UPDATE、DELETE 触发器在数据行已修改完成后，对修改的数据进行必要的善后处理。若发现有错误，则用事务回滚（ROLLBACK TRANSACTION）撤销本次操作，所以 INSERT、UPDATE、DELETE 触发器在约束检查之后才执行。  **【学生】**思考、讨论。 | **教师通过知识链接展示，让学生了解知识链接的基本理论知识。** |
| **课堂小结**  （3min） | 【**教师**】**回顾和总结本节课的知识点。**  **这节课我们一起学习了知识链接，让学生了解更多的相关知识。** | 通过对所学知识的回顾，培养学生的归纳总结能力 |
| **作业布置**（2min） | 【**教师**】**布置课后作业**  **简述触发器的功能。** | 通过课后练习，使学生巩固所学新知识 |
| **知识讲解**（45min） | **【教师】**展示知识扩展  **一、修改存储过程**  存储过程是一段 T-SQL 代码，在使用过程中，如果业务不满足需求，要对存储过程进行修改。修改存储过程的方法有两种：通过 SQL Server Management Studio 和 T-SQL语句修改。  1. 通过 SQL Server Management Studio 修改存储过程  使用 SQL Server Management Studio 修改存储过程的步骤和创建过程类似，首先在对象资源管理器中选中要修改的存储过程，单击右键从弹出的快捷菜单中执行“修改（Y）”选项，在打开的代码中按需求修改，最后单击工具栏的执行按钮 就完成了对存储过程的修改。  2. 通过 T-SQL 语句修改存储过程  打开要修改的存储过程 SQL 文件，然后按需求输入语句，最后单击工具栏的执行  按钮 就完成了对存储过程的修改。  **二、删除存储过程**  如果存储过程不再使用，则需要将其删除，SQL Server 2016 提供两种方法删除存储过程，分别是通过 SQL Server Management Studio 和 T-SQL 语句。  1. 通过 SQL Server Management Studio 删除存储过程  首先在对象资源管理器中选中要删除的存储过程，单击右键从弹出的快捷菜单中执  行“删除（D）”命令，在打开的 “删除对象”对话框中单击“删除”按钮即可。  2. 通过 T-SQL 语句删除存储过程  使用 T-SQL 语句删除存储过程的语法代码如下：  DROP { PROC | PROCEDURE } { [ schema\_name. ] procedure } [ ,...n ]  参数含义：  schema\_name：过程所属架构的名称。不能指定服务器名称或数据库名称。  procedure：要删除的存储过程或存储过程组的名称。  例如：删除存储过程 proc\_stu 代码如下：  DROP PROCEDURE proc\_stu  **三、删除 DML 触发器**  当不再需要某个触发器时，可将其删除，删除触发器有两种方法：一种是通过 SQL Server Management Studio，步骤与禁用触发器类似，只是在快捷菜单中选择“删除”选项，打开“删除对象”窗口，单击“确定”按钮即可。  另一种是通过 T-SQL 语句，使用 DROP TRIGGER 命令，语法代码如下。  DROP TRIGGER schema\_name.trigger\_name [ ,...n ] [ ; ]  参数含义：  ● trigger\_name：要删除的触发器名称。  例如：删除触发器 mytrigger。  DROP TRIGGER mytrigger  **【学生】**思考、讨论。 | **教师通过知识扩展展示，让学生了解知识扩展的基本理论知识。** |
| **课堂小结**  （3min） | 【**教师**】**回顾和总结本节课的知识点。**  **这节课我们一起学习了知识扩展，让学生了解更多的相关拓展知识。** | 通过对所学知识的回顾，培养学生的归纳总结能力 |
| **作业布置**（2min） | 【**教师**】**布置课后作业**  **简述删除 DML 触发器的操作过程。** | 通过课后练习，使学生巩固所学新知识 |
| **知识讲解**（45min） | **【教师】**展示思考与训练  **一、选择题**  1. 有关存储过程的参数默认值，说法正确的是（　　）。  A. 输入参数必须有默认值  B. 带默认值的输入参数，方便用户调用  C. 带默认值的输入参数，用户不能再传入参数，只能采用默认值  D. 输出参数也可以带默认值  2. 有关存储过程的说法，（　　）是错误的。  A. 它可作为一个独立的数据库对象并作为一个单元供用户在应用程序中调用  B. 存储过程可以传入和返回（输出）参数值  C. 存储过程必须带参数，要么是输入参数，要么是输出参数  D. 存储过程提高了执行效率  3. 下列有关触发器的说法，错误的是（　　）。  A. 触发器是一种特殊的存储过程，它可以包含 if、while、case 等复杂的 T-SQL语句  B. 使用触发器需要两步：先创建触发器，然后调用触发器  C. 如果检测到修改的数据不满足业务规则，触发器可以回滚撤销操作  D. 使用触发器可以创建比 CHECK 约束更复杂的高级约束  4. 当执行 UPDATE 语句时，系统将自动创建（　　）逻辑表。（选两项）  A. temp B. deleted  C. hold D. inserted  5. UPDATE 触发器能够对下列（　　）修改进行检查。（选两项）  A. 修改数据库名称 B. 修改表中的某行数据  C. 修改表中的某列数据 D. 修改表的结构  **二、简答题**  1. 存储过程和触发器有什么区别？  2. 触发器有哪几种？  **【学生】**思考、讨论。 | **教师通过思考与训练展示，让学生了解思考与训练的基本理论知识。** |
| **课堂小结**  （3min） | 【**教师**】**回顾和总结本节课的知识点。**  **这节课我们一起学习了思考与训练，让学生能够巩固所学知识。** | 通过对所学知识的回顾，培养学生的归纳总结能力 |
| **作业布置**（2min） | 【**教师**】**布置课后作业**  **存储过程和触发器有什么区别？** | 通过课后练习，使学生巩固所学新知识 |
| **知识讲解**（45min） | **【教师】**展示项目实战  **【项目实战背景】**  智游集团网上书店管理系统开发小组在系统开发过程中，需要创建存储过程和触发器，具体数据操作要求如下：  （1）通过存储过程对于过时的图书进行降价处理，降价的幅度是这批图书的平均价为每本不低于 20 元。  （2）存储过程来实现查询指定出版社的图书出版情况。  （3）查看出版社的某本图书本月订单情况。  （4）通过存储过程进行常规用户管理。  （5）通过触发器对图书进行添加、修改、删除管理。  **【项目实战内容】**  任务 1：为网上书店数据库创建不带参数的存储过程进行常规图书管理。  任务 2：为网上书店数据库创建带输入参数的存储过程进行常规出版社管理。  任务 3：为网上书店数据库创建带输出参数的存储过程进行常规订单管理。  任务 4：为网上书店数据库创建输入参数带默认值的存储过程进行常规用户管理。  任务 5：为网上书店数据库创建添加图书触发器。  任务 6：为网上书店数据库创建修改图书记录的触发器。  任务 7：为网上书店数据库创建删除图书触发器。  **【项目实战步骤】**  1. 任务 1 的操作步骤：  分析：  ● 对于过时的图书书店要降价处理，降价的幅度是这批图书的平均价为每本不低于 20 元。  ● 要解决这个问题首先考虑重新改变图书单价，改变多少不能一次性确定，所以利用循环结构多次更新之道满足要求，这样应用存储过程比较方便。  ● 可以通过 Microsoft SQL Server Management Studio 生成创建存储过程的模板，也可以直接 T-SQL 编码。  [ 关键步骤 ]  （1）打开 Microsoft SQL Server Management Studio 连接数据库引擎，在“对象资源  管理器”中选中数据库 BookSaleDB，展开“可编程性”节点，选中“存储过程”单击  鼠标右键，在弹出的快捷菜单中选择“新建存储过程”。  （2）右侧模板中编辑代码如下：  use BookSaleDB  GO  ALTER PROCEDURE BookProc  AS  BEGIN  DECLARE @avgPrice numeric(18, 0)-- 过时图书的平均价格  DECLARE @Toatl int -- 降价总量  SET @Toatl=0  -- 统计两年前出版的图书  SELECT @avgPrice=AVG(UnitPrice) FROM Books  WHERE PublishDate<DATEADD(yy,-2,getdate())  IF(@avgPrice<=20)  PRINT' 这批过时图书平均价已经低于 20 元！ '  ELSE  WHILE(@avgPrice>20)  BEGIN  -- 两年前出版的图书每次降价 2 元  U P D A T E B o o k s S E T U n i t P r i c e = U n i t P r i c e - 2 W H E R E  PublishDate<DATEADD(yy,-2,getdate())  SET @Toatl=@Toatl+2  SELECT @avgPrice=AVG(UnitPrice) FROM Books  WHERE PublishDate<DATEADD(yy,-2,getdate())  END  PRINT ' 两年前出版的图书一共降价 '+convert(varchar(3),@Toatl)+' 元 '  END  GO  （3）调用存储过程 BookProc 查看两年前出版的图书降价情况。  2. 任务 2 的操作步骤：  分析：  ● 查询指定出版社的图书出版情况。  ● 使用带输入参数的存储过程来实现。  [ 关键步骤 ]  （1）打开 Microsoft SQL Server Management Studio 连接数据库引擎。  （2）在“标准工具栏”中单击“新建查询”按钮，启动查询分析器。  （3）在查询分析器的编辑区域编辑代码如下：  USE BookSaleDB  GO  /\*--- 检测是否存在：存储过程存放在系统表 sysobjects 中 ---\*/  IF EXISTS (SELECT \* FROM sysobjects WHERE name = 'PubProc')  DROP PROCEDURE PubProc  GO  /\*--- 创建存储过程 ----\*/  CREATE PROCEDURE PubProc  @PublisherName varchar(100)  AS  BEGIN  SELECT @PublisherName+' 图书情况为：'  SELECT \* FROM Books WHERE PublisherId  IN(SELECT Id FROM Publishers WHERE Name=@PublisherName)  END  （4）调用存储过程 PubProc 查看“清华大学出版社”图书情况，参考代码为：EXEC PubProc ' 清华大学出版社 '  3. 任务 3 的操作步骤：  分析：  ● 查看某出版社的某本书本月图书销售情况。  ● 使用带输入和输出参数的存储过程来实现。  [ 关键步骤 ]  （1）打开 Microsoft SQL Server Management Studio 连接数据库引擎。  （2）在标准工具栏中选中“新建查询”按钮，启动查询分析器。  （3）在查询分析器的编辑区域编辑代码如下：  USE BookSaleDB  GO  /\*--- 检测是否存在：存储过程存放在系统表 sysobjects 中 ---\*/  IF EXISTS (SELECT \* FROM sysobjects WHERE name = 'OrderProc')  DROP PROCEDURE OrderProc  GO  /\*--- 创建存储过程 ----\*/  CREATE PROCEDURE OrderProc  @PublisherName varchar(100),  @BookName varchar(100),  @OrderTotalPrice bigint output  AS  BEGIN  DECLARE @bookId int  SELECT @bookId=id FROM books  WHERE Title like '%'+@BookName+'%' and PublisherId  IN(SELECT Id FROM Publishers WHERE Name=@PublisherName)  SELECT @PublisherName+' 的 '+@BookName+' 图书销售情况为：'  SELECT \* FROM [Orders] WHERE BookId =@bookId and OrderDate>=convert(  varchar(4),Year(getdate()))+'-'+convert(varchar(2),month(getdate()))+'-01'  SELECT @OrderTotalPrice=Quantity\*Price FROM [Orders]  WHERE BookId =@bookId and OrderDate>=convert(varchar(4),Year(getda  te()))+'-'+convert(varchar(2),month(getdate()))+'-01'  END  （4）调用存储过程 PubProc 查看“清华大学出版社”图书情况，参考代码如下：  DECLARE @TotalPrice numeric(18,0)  EXEC OrderProc ' 清华大学出版社 ','Java 程序设计 ',@TotalPrice  SELECT ' 销售总价为：'=@TotalPrice  4. 任务 4 的操作步骤：  分析：  ● 查看某用户信息。  ● 使用输入参数带默认值的存储过程来实现。  [ 关键步骤 ]  （1）打开 Microsoft SQL Server Management Studio 连接数据库引擎。  （2）在标准工具栏中选中“新建查询”按钮，启动查询分析器。  （3）在查询分析器的编辑区域编辑代码如下：  USE BookSaleDB  GO  /\*--- 检测是否存在：存储过程存放在系统表 sysobjects 中 ---\*/  IF EXISTS (SELECT \* FROM sysobjects WHERE name = 'UsersProc')  DROP PROCEDURE UsersProc  GO  /\*--- 创建存储过程 ----\*/  CREATE PROCEDURE UsersProc  @userLoginId varchar(50)=' 鸽子 '  AS  BEGIN  /\* 使用子查询解决问题 \*/  SELECT LoginId,LoginPwd,Name,[Address],Phone,Mail,role=  (SELECT Name FROM UserRoles WHERE Id=Users.UserRoleId)  ,state=(SELECT Name FROM UserStates WHERE Id=Users.UserStateId)  FROM users  WHERE LoginId=@userLoginId  /\* 使用多表联合查询解决问题 \*/  --SELECT LoginId,LoginPwd,a.Name userName,[Address],Phone,  Mail,b.name userRole,c.name useState  --FROM Users a,UserRoles b,UserStates c WHERE a.UserRoleId=b.  Id and a.UserStateId=c.Id and LoginId=@userLoginId  -- 或者  --SELECT LoginId,LoginPwd,a.Name userName,[Address],Phone,  Mail,b.name userRole,c.name useState  --FROM Users a join UserRoles b on a.UserRoleId=b.Id join  UserStates c on a.UserStateId=c.Id WHERE LoginId=@userLoginId  END  （4）调用存储过程 UsersProc 查看用户信息，参考代码如下：  EXEC UsersProc -- 查询的是默认值 ' 鸽子 ' 的情况  EXEC UsersProc ' 冷云 '-- 查询的是用户 ' 冷云 ' 的情况  5. 任务 5 的操作步骤：  分析：  ● 图书表添加记录时，响应触发器告知用户添加成功，并进行查看。  ● 应该在图书表上创建 INSERT 触发器。  [ 关键步骤 ]  （1）启动 SQL Server Management Studio，在“对象资源管理器”中依次展开“数据库”、“BookSaleDB”数据库、“表”节点。  （2）展开“Books”表，右键单击“触发器”，选择“新建触发器”。  （3）在右边弹出的查询窗口显示“触发器”的模板，输入触发器文本如下：  USE BookSaleDB  GO  IF OBJECT\_ID('BooksInsertTrig', 'TR') IS NOT NULL  DROP TRIGGER BooksInsertTrig  GO  CREATE TRIGGER BooksInsertTrig  ON Books  AFTER INSERT  AS  SELECT ' 添加成功 !'  SELECT ' 图书表中新添加图书了 !'  SELECT \* FROM INSERTED  GO  （4） 执行语句，提示成功信息，则 Insert 触发器创建成功。  （5） 使用 INSERT 语句向 Books 表添加一条记录，验证触发器功能。  6. 任务 6 的操作步骤：  分析：  ● 随着时间变化，一些热销图书要进行降价处理，以便打开销路。  ● 降价促销图书，即打折扣要更新图书折扣字段值，为了了解折扣修改信息，在Books 表上响应 UPDATE 触发器。  ● UPDATE 触发器响应过程，存在两个临时表，即 INSERTED 和 DELETED，所以我们可以比较修改图书折扣前后的数据变化。  [ 关键步骤 ]  为表“Books”创建一个修改记录后显示提示信息的触发器 BooksUpdateTrig  （1）查询编辑器中输入以下文本：  USE BookSaleDB  GO  IF OBJECT\_ID('BooksUpdateTrig', 'TR') IS NOT NULL  DROP TRIGGER BooksUpdateTrig  GO  CREATE TRIGGER BooksUpdateTrig  ON Books  AFTER UPDATE  AS  SELECT ' 修改成功 !' as ' '  SELECT ' 图书被修改之前的信息：' as ' '  SELECT Title ' 书 名 ',Author ' 作 者 ',' 出版社 '=(SELECT name FROM  Publishers WHERE id=DELETED.PublisherId), 种类 =(SELECT name FROM Categories  WHERE id=DELETED.CategoryId),' 折扣 '=DISC FROM DELETED  SELECT ' 图书被修改之前的信息：' as ' '  SELECT Title ' 书 名 ',Author ' 作 者 ',' 出版社 '=(SELECT name FROM  Publishers  WHERE id=INSERTED.PublisherId), 种 类 =(SELECT name FROM Categories  WHERE id=INSERTED.CategoryId),' 折扣 '=DISC FROM INSERTED  GO  （2）执行以上代码。  （3）修改图书折扣信息，用以验证更新触发器。  7. 任务 7 的操作步骤：  分析：  ● 随着时间推移，大量图书已经不再销售，这些图书信息造成了图书表数据的冗余。  ● 图书表中无用数据记录要被删除，作为历史销售图书资料，为了满足后期查询需要，删除的记录还想自动保存到备份数据表。  ● 为图书表创建 DELETE 触发器，实现删除图书记录自动保存备份。  [ 关键步骤 ]  为 Books 表创建一个删除记录后显示提示信息的触发器 BooksDeleteTrig  （1） 查询编辑器中输入以下文本：  USE BookSaleDB  GO  IF OBJECT\_ID('BooksDeleteTrig', 'TR') IS NOT NULL  DROP TRIGGER BooksDeleteTrig  GO  CREATE TRIGGER BooksDeleteTrig  ON Books  AFTER DELETE  AS  INSERT INTO BooksBackup SELECT \* FROM DELETED  SELECT ' 记录已经被备份 , 删除成功！ '  GO  （2）执行以上代码。  （3）验证触发器，删除图书表记录后提示“记录已经被备份，删除成功！”查看备份数据表，删除的图书表记录已经添加到 BooksBackup 表。  **【学生】**思考、讨论。 | **教师通过项目实战展示，让学生了解项目实战的基本理论知识。** |
| **课堂小结**  （3min） | 【**教师**】**回顾和总结本节课的知识点。**  **这节课我们一起学习了项目实战，提高学生对于实际的操作能力。** | 通过对所学知识的回顾，培养学生的归纳总结能力 |
| **作业布置**（2min） | 【**教师**】**布置课后作业**  **尝试举出相关例子进行操作。** | 通过课后练习，使学生巩固所学新知识 |
| **教学反思** | 不断地激发学生的学习积极性与主动性，培养学生思维能力、想象力和创新精神，使每个学生的身心都能得到充分的发展。 | |