

第十章 数据库恢复技术

- 本书第十章、第十一章讨论事务的处理技术；**事务处理技术主要包括数据库恢复技术和并发控制技术。**数据库恢复机制和并发控制机制是数据库管理系统的重要组成部分；
- 本章着重讨论SQL SERVER数据库的备份、恢复策略和实现技术 。

本章学习内容

- 事务的基本概念
- 备份和恢复概述
- 故障的种类
- 数据库备份
- 数据库恢复

一、事务的基本概念

- 有时，某个工作的完成要分成若干步骤，只有所有步骤都成功做完，该项工作才完成；
- 否则，其中任一步失败，该工作亦失败。针对此类工作特点，引入“事务”概念，在DBMS中，定义此类工作为事务，并保证其执行特点。

1. 什么是事务

- **事务(Transaction)** 是用户定义的一个数据库操作序列，这些操作要么全做，要么全不做，是一个不可分割的工作单位；
- 事务和程序是两个概念
 - 在关系数据库中，一个事务可以是一条SQL语句，一组SQL语句或整个程序；
 - 一个应用程序通常包含多个事务；
- **事务**是恢复和并发控制的基本单位；

2. 如何定义事务

- 显式定义方式

BEGIN TRANSACTION

SQL 语句1

SQL 语句2

◦ ◦ ◦ ◦ ◦

COMMIT

BEGIN TRANSACTION

SQL 语句1

SQL 语句2

◦ ◦ ◦ ◦ ◦

ROLLBACK

- 隐式方式

当用户没有显式地定义事务时

DBMS按缺省规定自动划分事务

事务结束

COMMIT

事务正常结束

提交事务的所有操作（读+更新）

事务中所有对数据库的更新永久生效

ROLLBACK

事务异常终止

- 事务运行的过程中发生了故障，不能继续执行

回滚事务的所有更新操作

- 事务滚回到开始时的状态

3.事务的特性(ACID特性)

事务的ACID特性：

- 原子性 (Atomicity)
- 一致性 (Consistency)
- 隔离性 (Isolation)
- 持续性 (Durability)

1) 原子性

- 事务是数据库的逻辑工作单位
- 事务中包括的诸操作要么都做，要么都不做

2) 一致性

- 事务执行的结果必须是使数据库从一个一致性状态变到另一个一致性状态
 - **一致性状态**：数据库中只包含成功事务提交的结果
 - **不一致状态**：数据库中包含失败事务的结果

原子性、一致性示例：

银行转帐：从帐号A中取出一万元，存入帐号B。

- 定义一个事务，该事务包括两个操作
- 这两个操作要么全做，要么全不做
 - 全做或者全不做，数据库都处于一致性状态。
 - 如果只做一个操作，数据库就处于不一致性状态。

A	B
$A=A-1$	$B=B+1$

3) 隔离性

- 一个事务的执行不能被其他事务干扰
 - 一个事务内部的操作及使用的数据对其他并发事务是隔离的
 - 并发执行的各个事务之间不能互相干扰

4) 持续性

- 持续性也称永久性（Permanence）
 - 一个事务一旦提交，它对数据库中数据的改变就应该是永久性的。
 - 接下来的其他操作或故障不应该对其执行结果有任何影响。

- 保证事务ACID特性是事务处理的任务
- 破坏事务ACID特性的因素：
 - 多个事务并行运行时，不同事务的操作交叉执行
 - 事务在运行过程中被强行停止

二、备份和恢复概述

- 尽管SQL SERVER 系统采取了多种措施来保证数据库的安全性和完整性，但**硬件故障、软件错误、病毒、误操作或故意破坏**仍可能发生，这些故障轻则造成运行事务非正常中断，影响数据正确性，重则破坏数据库，使数据库中的数据部分或全部丢失。
- 因此，为了避免因系统本身的故障而造成的数据的破坏或丢失，数据库管理系统提供了把数据库从错误状态恢复到某一正确状态的功能，这种功能称为**恢复**，数据库的恢复是以**备份**为基础的。

1. 恢复

- 故障是不可避免的
 - 系统故障：计算机软、硬件故障
 - 介质故障：存储设备故障
 - 人为故障：操作员的失误、恶意的破坏等。

数据库中的数据丢失或被破坏可能原因：

- (1) **计算机硬件故障**。由于使用不当或产品质量等原因，计算机硬件可能会出现故障，不能使用。如硬盘损坏会使得存储于其上的数据丢失。
- (2) **软件故障**。由于软件设计上的失误或用户使用的不当，软件系统可能会误操作数据引起数据破坏。
- (3) **病毒**。破坏性病毒会破坏系统软件、硬件和数据。
- (4) **误操作**。如用户误使用了诸如DELETE、UPDATE等命令而引起数据丢失或被破坏。
- (5) **自然灾害**。如火灾、洪水或地震等，它们会造成极大的破坏，会毁坏计算机系统及其数据。
- (6) **盗窃**。一些重要数据可能会遭窃。

数据库的恢复

- 把数据库从错误状态恢复到某一已知的正确状态(亦称为一致状态或完整状态)；
- 数据库恢复就是当数据库出现故障时，将备份的数据库加载到系统，从而使数据库恢复到备份时的正确状态。
- 恢复技术是衡量系统优劣的重要指标；
- 系统进行恢复操作时，先执行一些系统安全性的检查，包括检查所要恢复的数据库是否存在、数据库是否变化以及数据库文件是否兼容等，然后根据所采用的数据库备份类型采取相应的恢复措施。

2. 备份

- 我们希望永远不进行恢复数据库的操作，但是数据库的备份操作是必须定期进行的；数据库必须适时地进行备份，以防意外事件的发生而造成数据的损失。
- 数据库备份需要根据实际情况，制定不同的备份策略，一方面可以保证数据的安全性，另一方面又要避免不必要浪费。
- 备份策略：确定备份的**内容**、确定备份**介质**、确定备份**方式**、确定备份**频率**。

1) 确定备份的内容

- 数据库中数据的重要程度决定了数据恢复的必要与重要性，也就决定了数据是否需要备份。数据库需备份的内容可分为系统数据库和用户数据库两部分。
- 系统数据库包括master、msdb、model数据库，他们是确保DBMS正常运行的重要依据，因此**系统数据库必须被完全备份**。
- 用户数据库是存储用户数据的存储空间集。**备份时要取决于数据重要程度**，主要依据实际的应用领域。

2) 确定备份介质

SQL Server 支持3种类型的备份介质：

- **硬盘**：本地磁盘或网络中磁盘，是最常用的备份介质，但费用较高；
- **磁带**：是大容量的备份介质，磁带仅可用于备份本地文件。价格较为便宜，存储容量大，便于保存和携带；
- **命名管道**（Named Pipe）：主要用于第三方备份软件，SQL Server提供其他软件公司所开发的数据库备份和恢复软件，提供特殊的数据库备份和恢复方法。

3) 确定备份方式

- **数据库备份**
 - **完整数据库备份**：将整个数据库全部备份下来；
 - **差异数据库备份**：指在一次完整备份数据库后，只备份以后对数据库的修改内容；
 - **事务日志备份**：仅备份用户对数据库操作的记载；
- **文件、文件组备份**：仅备份特定的数据库文件或文件组，对于在多个文件中的大型数据库，可以使用这种方法进行备份；
 - **完整、差异、事务日志数据库备份**

4) 备份的频率

- 确定数据库备份频率是一件很困难的事情；备份太频繁既浪费时间，又浪费设备；备份间隔时间过长，就有可能造成部分数据的损失；
- 要考虑两个因素：一是存储介质出现故障或其他故障可能导致数据损失而需要恢复被损失数据的工作量的大小；二是数据库的事务数量；
- 更应该考虑用户自己的系统环境；

5) 何时备份？

对于系统数据库和用户数据库，其备份时机是不同的。

- 当系统数据库master、msdb、model中任何一个被修改以后，都要将其备份。
- 当创建数据库或修改、加载数据库时，应备份数据库。

6) 谁来作备份？

具有以下角色的成员可以作备份操作：

- 固定的服务器角色sysadmin（系统管理员）；
- 固定的数据库角色db_owner（数据库所有者）；
- 固定的数据库角色db_backupoperator（允许进行数据库备份的用户）。

说明：

- 备份一个数据库所需的时间主要取决于物理设备的速度，如磁盘设备的速度通常比磁带设备快；
- 通常备份到多个物理设备比备份到一个物理设备要快；
- 系统的并发活动对数据库的备份有影响，因此在备份数据库时，应减少并发活动，以减少数据库备份所需的时间。

三、故障的种类

- 事务内部的故障
- 系统故障
- 介质故障
- 计算机病毒

1. 事务内部的故障

- 事务内部的故障
 - 有的是可以通过事务程序本身发现的(见转账事务的例子);
 - 有的是非预期的;

例：银行转账事务，这个事务把一笔金额从一个账户甲转给另一个账户乙

```
BEGIN TRANSACTION                                /*读账户甲的余额BALANCE
BALANCE=BALANCE-AMOUNT;                          /*AMOUNT 为转账金额
IF (BALANCE < 0 ) THEN
    { 打印 ‘金额不足，不能转账’ ;   ROLLBACK; /*撤销刚才的修改，恢复事务 }
ELSE
    { 读账户乙的余额BALANCE1;   BALANCE1=BALANCE1+AMOUNT;
      写回BALANCE1;   COMMIT;  }
```

- 这个例子所包括的两个更新操作要么全部完成要么全部不做。否则就会使数据库处于不一致状态，例如只把账户甲的余额减少了而没有把账户乙的余额增加。
- 在这段程序中若产生账户甲余额不足的情况，应用程序可以发现并让事务滚回，撤销已作的修改，恢复数据库到正确状态。

- 事务内部更多的故障是非预期的，是不能由应用程序处理的：
 - 运算溢出
 - 并发事务发生死锁而被选中撤销该事务
 - 违反了某些完整性限制等

以后，事务故障仅指这类非预期的故障

- 事务故障的恢复：撤销事务（UNDO）

2. 系统故障

- 系统故障：称为软故障，是指造成系统停止运转的任何事件，使得系统要重新启动；
 - 整个系统的正常运行突然被破坏；
 - 所有正在运行的事务都非正常终止；
 - 不破坏数据库；
 - 内存中数据库缓冲区的信息全部丢失；

系统故障的常见原因

- 特定类型的硬件错误（如CPU故障）
- 操作系统故障
- DBMS代码错误
- 系统断电

系统故障的恢复

- 发生系统故障时，事务未提交
 - **恢复策略：强行撤消（UNDO）所有未完成事务**
- 发生系统故障时，事务已提交，但缓冲区中的信息尚未完全写回到磁盘上。
 - **恢复策略：重做（REDO）所有已提交的事务**

3. 介质故障

- 介质故障：称为硬故障（Hard Crash），指外存故障；
 - 磁盘损坏，磁头碰撞
 - 操作系统的某种潜在错误
 - 瞬时强磁场干扰
- 硬件故障使存储在外存中的数据部分丢失或全部丢失；
- 介质故障比前两类故障的可能性小得多，但破坏性大得多；

介质故障的恢复

- 装入数据库发生介质故障前某个时刻的数据**副本**；
- **重做**自此时始的所有**成功事务**，将这些事务已提交的结果重新记入数据库；

4. 计算机病毒

- 计算机病毒
 - 一种人为的故障或破坏，是一些恶作剧者研制的一种计算机程序
 - 可以繁殖和传播
- 危害
 - 破坏、盗窃系统中的数据
 - 破坏系统文件

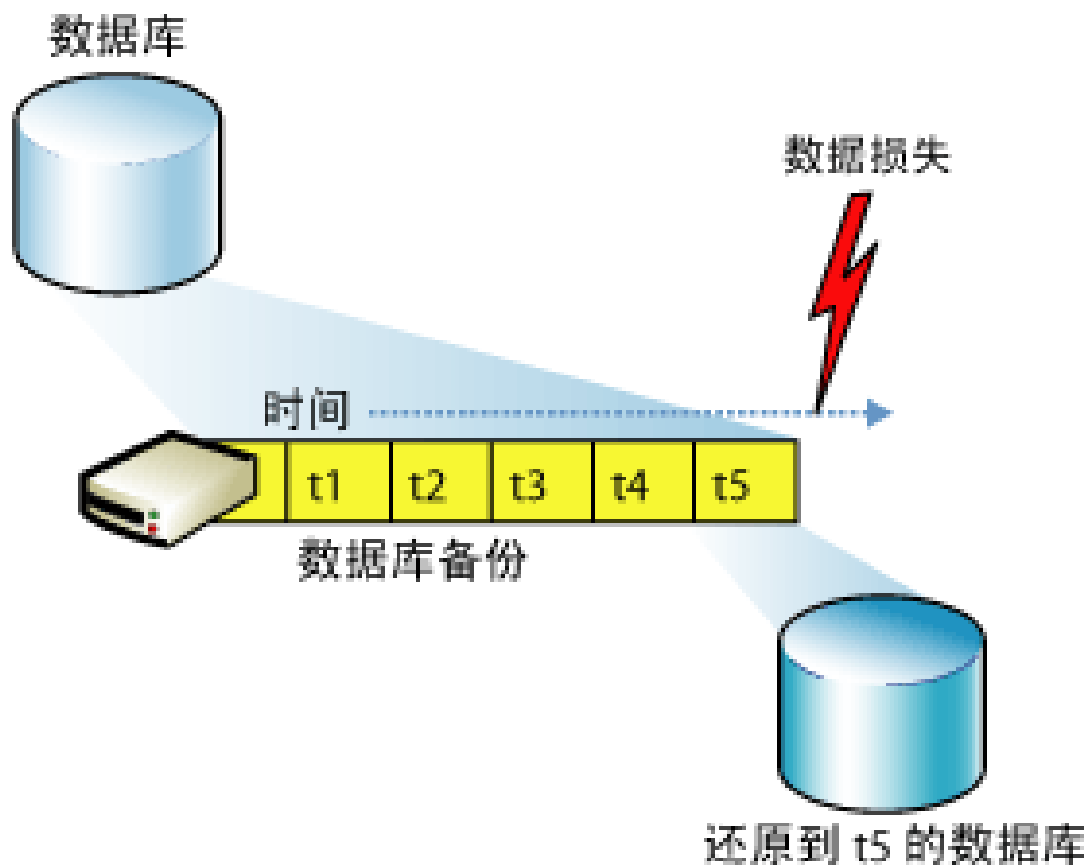
故障小结

各类故障，对数据库的影响有两种可能性：

- 一是数据库本身被破坏；
- 二是数据库没有被破坏，但数据可能不正确，这是由于事务的运行被非正常终止造成的。

四、数据库备份

- 故障会引起数据库数据的丢失或不一致，作为DBA，就要采取措施恢复丢失的数据，而恢复数据最直接最常用的手段就是“备份”（Backup），也就是采取“冗余”方法；
- Microsoft SQL Server 提供了高性能的备份和还原功能。SQL Server 备份和还原组件提供了重要的保护手段，以保护存储在 SQL Server 数据库中的关键数据。



- “备份”是数据的副本，用于在系统发生故障后还原和恢复数据。备份使您能够在发生故障后还原数据。通过适当的备份，可以从多种故障中恢复；
- SQL Server 数据库备份及文件、文件组备份类型：
 - 完整数据库备份
 - 差异数据库备份
 - 事务日志数据库备份

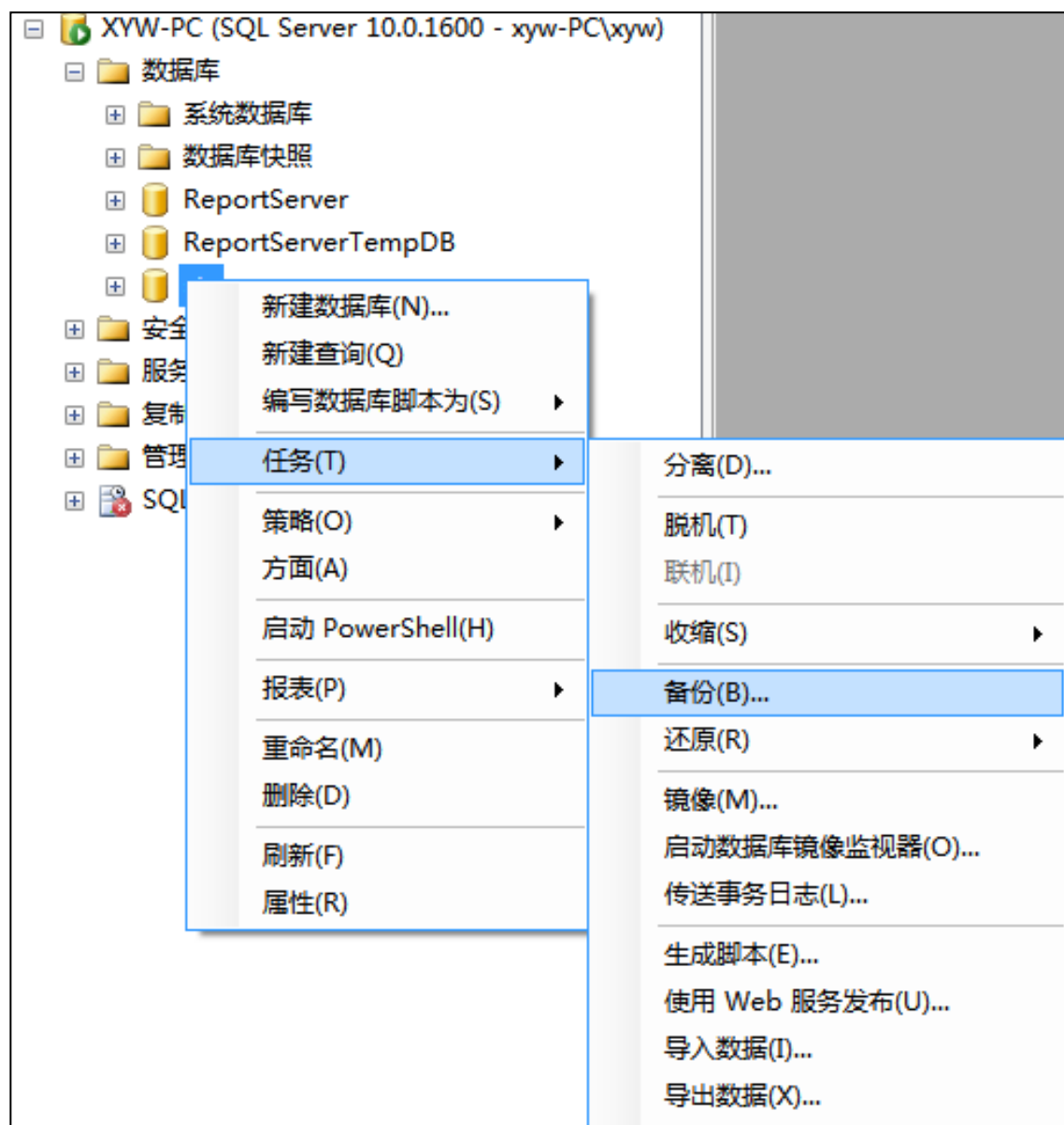
1、创建完整数据库备份

- 完整备份（以前称为数据库备份）将备份整个数据库，包括事务日志部分（以便可以恢复整个备份）。创建完整备份是单一操作，通常会安排该操作定期发生。
- 每个完整备份使用的存储空间比其他差异备份使用的存储空间要大。因此，完成完整备份需要更多的时间，因而创建完整备份的频率通常要比创建差异备份的频率低。

例1：将数据库stu完整备份到本地磁盘D，备份文件名称为stu ；

操作步骤为：

- 启动SQL Server Management Studio →对象资源管理器→数据库
→stu ；
- 右击stu →任务 →备份；
- 在“备份数据库窗口”中，选择源数据库stu ，选择备份类型“完整”，选择备份组件“数据库”，为备份集起名称“stu”，选择备份目标，备份到“磁盘→添加目标路径→本地磁盘D”，单击“确定”即可完成对数据库stu的完全备份；



选择页

- 常规
- 选项

连接

服务器:
XYW-PC

连接:
xyw-PC\xyw

 查看连接属性

进度



就绪

脚本 帮助

源

数据库 (T):

stu

恢复模式 (M):

完整

备份类型 (K):

完整

仅复制备份 (Y)

备份组件:

数据库 (B)

文件和文件组 (G):

备份集

名称 (N):

stu-完整 数据库 备份

说明 (S):

备份集过期时间:

晚于 (E):

0

天

在 (O):

2014-06-04

目标

备份到:

磁盘 (I)

磁带 (P)

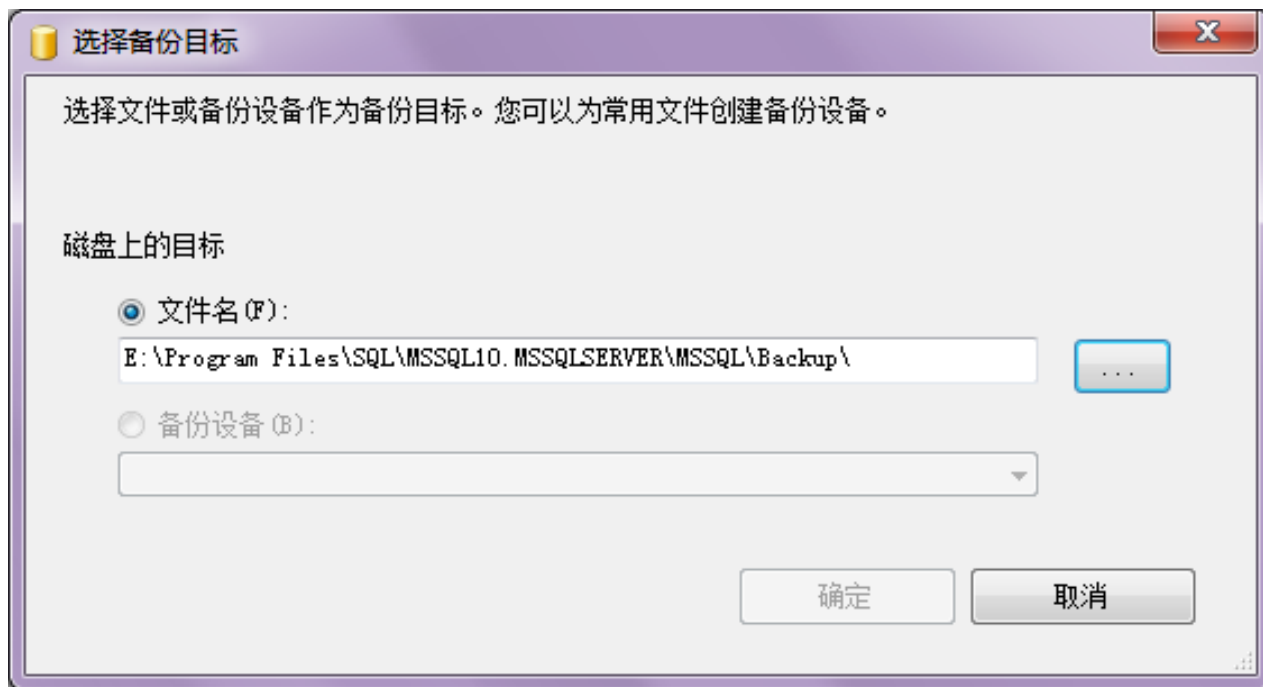
添加 (A)...

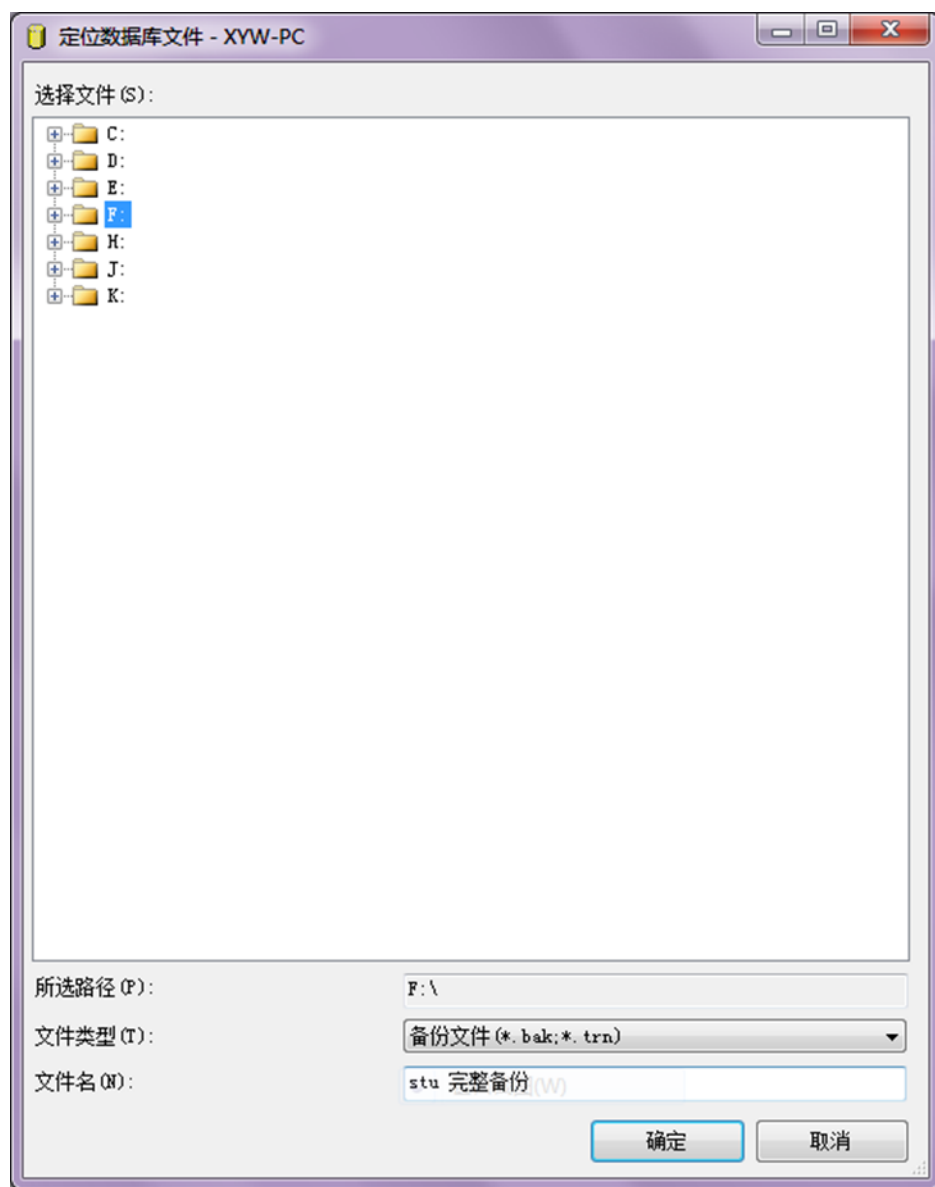
删除 (R)

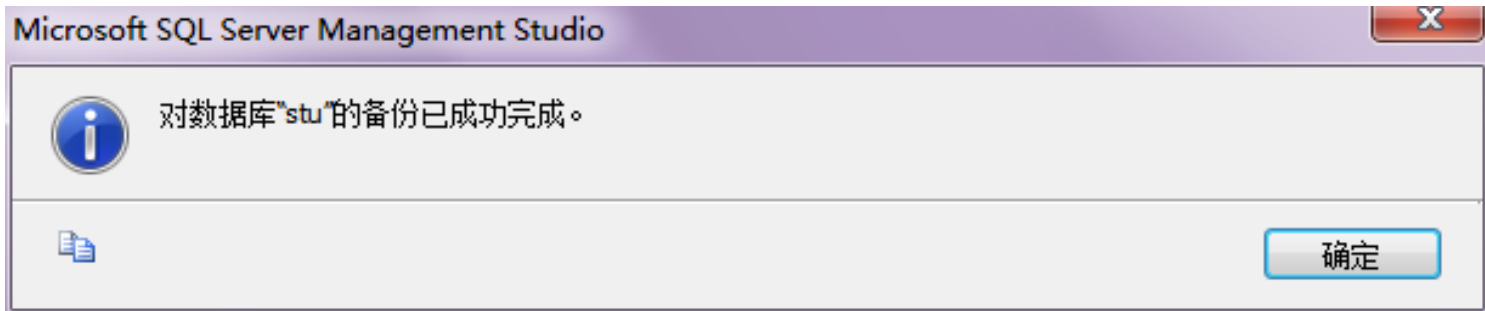
内容 (C)

确定

取消







2、创建差异数据库备份

- **差异备份**：仅记录自上次数据库完整备份后**更改过的数据**。比完全备份工作量小而且备份速度快，对正在运行的系统影响也较小，可以简化频繁的备份操作，减少数据丢失的风险；
- **在进行差异备份之前，必须进行一次完整数据库备份**，所以也称为**创建完整差异备份**；
- 一般来说，可在创建数据库后，或在数据库中已经拥有了大量数据以后，进行一次数据库完整备份，然后再定期进行差异备份；

例2：在数据库stu进行部分操作后，对stu进行差异备份；

操作步骤为：

- 启动SQL Server Management Studio →对象资源管理器→数据库
→stu ；
- 右击stu →任务 →备份；
- 在“备份数据库窗口”中，选择源数据库stu ，选择备份类型“差异”，选择备份组件“数据库”，为备份集起名称“stu”，单击“确定”即可完成对数据库stu的差异备份；（如图所示）

选择页

- 常规
- 选项

连接

服务器:
XYW-PC

连接:
xyw-PC\xyw

 [查看连接属性](#)

进度



就绪

 脚本  帮助

源

数据库 (T):

stu

恢复模式 (M):

完整

备份类型 (K):

差异

仅复制备份 (Y)

备份组件:

数据库 (B)

文件和文件组 (G):

备份集

名称 (N):

stu-差异 数据库 备份

说明 (S):

备份集过期时间:

晚于 (E):

0

天

在 (O):

2014-06-04

目标

备份到:

磁盘 (I)

磁带 (T)

F:\stu 差异备份

添加 (A)...

删除 (R)

内容 (C)

确定

取消

3、创建事务日志数据库备份

- 仅备份用户对数据库操作的记载；将事务日志中从前一次成功备份结束位置开始到当前事务日志的结尾处的内容进行备份；
- 在完整恢复模式和大容量日志恢复模式下，执行常规事务日志备份对于恢复数据至关重要；使用事务日志备份，可以将数据库恢复到故障点或特定的时间点；
- 要保证事务日志备份的连续性；
- 操作方法同差异数据库备份！

4、数据库文件或文件组备份

- 仅备份特定的数据库文件或文件组，同时还要定期备份事务日志，这样在恢复时可以只还原已损坏的文件，加快了恢复速度；
- 对于在多个文件中的大型数据库，可以使用这种方法进行备份；
- 例如：如果数据库有几个在物理上位于不同磁盘上的文件组成，当其中一个磁盘发生故障时，只需还原发生了故障的磁盘上的文件。
- 文件或文件组备份和还原操作必须与事务日志备份一起使用，以确保恢复后的文件与数据库的其他部分是一致的。

选择页

- 常规
- 选项

脚本 帮助

源

数据库 (T):

stu

恢复模式 (M):

完整

备份类型 (K):

仅复制备份 (Y)

备份组件:

- 数据库 (B)
- 文件和文件组 (G):

备份集

名称 (N):

说明 (S):

备份集过期时间:

- 晚于 (E):
- 在 (O):

目标

备份到:

F:\stu 差异备份

连接

服务器:
XYW-PC

连接:
xyw-PC\xyw

查看连接属性

进度

就绪

选择文件和文件组



全选 (S)

全部清除 (C)

确定

取消

确定

取消

五、数据库恢复的实现技术

主要内容：

- 数据库恢复实现技术
 - 数据转储
 - 日志文件
- 数据库恢复策略
- 数据库恢复模型
- 案例分析

1、数据库恢复实现技术

1) 数据转储

- 转储是指DBA将整个数据库复制到磁盘或另一个磁带上保存起来的过程，备用的数据称为后备**副本**或后援副本；
- 如何使用
 - 数据库遭到破坏后可以将后备副本重新装入；
 - 重装后备副本只能将数据库恢复到转储时的状态；

- 转储方法

- 静态转储：在系统中无运行事务时进行的转储操作；
- 动态转储：转储期间允许对数据库进行存取或修改；

- 转储模式

- 海量转储：每次转储全部数据库；
- 增量转储：只转储上次转储后更新过的数据；

2) 登记日志文件(Logging)

- 日志文件(log)是用来记录事务对数据库的更新操作的文件;
- 日志文件包括内容
 - 各个事务的开始标记(BEGIN TRANSACTION)
 - 各个事务的结束标记(COMMIT或ROLLBACK)
 - 各个事务的所有更新操作
- 日志文件的作用
 - 进行事务故障恢复
 - 进行系统故障恢复
 - 协助后备副本进行介质故障恢复

问题1：先写日志文件还是先写数据文件？

- 由于日志文件好比数据库的“黑匣子”，所以其记录的用户对数据操作的重要性远远甚于当前数据的重要性，且其完整性也远远甚于数据文件的完整性，DBA应尽可能地保证日志文件的安全；
- 如果先写了数据库修改，而在运行过程中没有登记，则以后就无法恢复这个修改了；如果先写日志，但没有修改数据库，按日志文件恢复只不过是多执行一次不必要的UNDO操作，不会影响数据库的正确性；
- 故为了安全，一定要先写日志文件，然后再写对数据库的修改。

2、数据库恢复策略

- 当系统运行过程中发生故障，利用数据库后备副本和日志文件就可以将数据库恢复到故障前的某个一致性状态；
- 不同故障其恢复策略和方法也不一样
 - 事务故障的恢复
 - 系统故障的恢复
 - 介质故障的恢复

1) 事务故障的恢复

- 事务故障：事务在运行至正常终止点前被终止
- 恢复方法
 - 由恢复子系统应利用日志文件撤消（UNDO）此事务已对数据库进行的修改
- 事务故障的恢复由系统自动完成，对用户是透明的，不需要用户干预。

事务故障的恢复步骤：

1. **反向**扫描文件日志（即从最后向前扫描日志文件），查找该事务的更新操作。
2. 对该事务的更新操作**执行逆操作**。即将日志记录中“更新前的值”写入数据库。
 - 插入操作，“更新前的值”为空，则相当于做删除操作
 - 删除操作，“更新后的值”为空，则相当于做插入操作
 - 若是修改操作，则相当于用修改前值代替修改后值
3. 继续反向扫描日志文件，查找该事务的其他更新操作，并做同样处理。
4. 如此处理下去，**直至读到此事务的开始标记**，事务故障恢复就完成了。

2) 系统故障的恢复

- 系统故障造成数据库不一致状态的原因
 - 未完成事务对数据库的更新已写入数据库
 - 已提交事务对数据库的更新还留在缓冲区来不及写入数据库
- 恢复方法
 - 1. **Undo** 故障发生时未完成的事务
 - 2. **Redo** 已完成的事务
- 系统故障的恢复由**系统在重新启动时自动完成**，不需要用户干预。

- 系统故障的恢复步骤：
 - 1. **正向扫描日志文件**（即从头扫描日志文件）
 - 重做 (REDO) 队列：在故障发生前已经提交的事务；这些事务既有 BEGIN TRANSACTION记录，也有COMMIT记录；
 - 撤销 (Undo) 队列：故障发生时尚未完成的事务；这些事务只有 BEGIN TRANSACTION记录，无相应的COMMIT记录；
 - 2. 对撤销 (Undo) 队列事务进行**撤销 (UNDO)** 处理；
 - 3. 对重做 (Redo) 队列事务进行**重做 (REDO)** 处理

3) 介质故障的恢复

- 发生介质故障后，磁盘上的物理数据和日志文件被破坏，这是最严重的一种故障，恢复的方法是**重装数据库**，然后**重做已完成的事务**；
- 恢复步骤：
 - 装入最新的后备数据库副本（离故障发生时刻最近的转储副本），使数据库恢复到最近一次转储时的一致性状态；
 - 装入有关的日志文件副本（转储结束时刻的日志文件副本），重做已完成的事务。

3、数据库恢复模型

- **完整恢复模型：**

- 对于特别重要的数据库，如银行、电信系统，任何日志都不能缺少，在发生故障时要求能恢复到历史上的某个时刻，一旦发生故障要求数据不丢失，这样的数据库就必须工作在完整恢复模型下。
- 此模型下，必须**定期进行数据备份或者事务日志备份**，确保日志空间被定期回收；
- 设置：数据库→stu →属性→选项→恢复模式选择“完整”（如图所示）；

数据库属性 - stu

选择页

- 常规
- 文件
- 文件组
- 选项
- 更改跟踪
- 权限
- 扩展属性

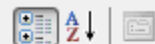
脚本 帮助

排序规则 (C): Chinese_PRC_CI_AS

恢复模式 (M): 完整

兼容级别 (L): SQL Server 2008 (100)

其他选项 (O):



Service Broker

Broker 已启用	False
Service Broker 标识符	ed3d9b2a-6147-4b3f-a440-eb49b90d4cfe
服从 Broker 优先级	False

恢复

页验证	CHECKSUM
-----	----------

- **简单恢复模型：**

- 对于特别数据库**数据安全性要求不高**，而对**性能要求很高**，这样的数据库可以工作在简单恢复模型下。
- 简单恢复模型的数据库**可能会导致无法恢复到历史上某个时刻的情况**；
- 设置：数据库→stu →属性→选项→恢复模式选择“简单” ；

- **大容量恢复模型：**

- DBA在某些场合需要对SQL数据库**执行一些大批量的数据录入、更新或者删除操作**（如一次需导入10000条记录），这样的特殊环境可以将数据库工作在大容量恢复模型下。
- 大容量操作数据的语句就是**简化日志记录，不记录足够的细节**，这样可大大减少日志记录的数量；正由于日志记录不完全，所以一旦操作发生故障就可能会**导致不能恢复的后果**；
- 设置：数据库→stu →属性→选项→恢复模式选择“大容量日志” ；

4、案例分析

例1：完整数据库备份与恢复；

- 假设现在有3个完整数据库备份：
 - 10：00时有完整数据库备份1；
 - 11：00时有完整数据库备份2；
 - 12：00时有完整数据库备份3；
- 则恢复时只能选择任意的一个完全数据库备份进行恢复；也就是说，要么恢复到10：00，要么恢复到11：00或12：00，其他任何时刻都不可能。

例2：完整+差异数据库备份与恢复；

- 假设现在有2个完整数据库备份：
 - 10：00时有完整数据库备份1；
 - 12：00时有完整数据库备份2；
- 同时，还假设有3个差异数据库备份：
 - 10：30时有差异数据库备份1，在完整数据库备份1的基础上做；
 - 11：00时有差异数据库备份2，在完整数据库备份1的基础上做；
 - 12：30时有差异数据库备份3，在完整数据库备份2的基础上做；
- 如果需要恢复到11：00时的状态，则应为：

- **完整数据库备份1 + 差异数据库备份2**
- 能否为：完整数据库备份1 + 差异数据库备份1 + 差异数据库备份2，
答案：**不行**；
- 如果需要恢复到12:30时的状态，则应为：
 - 完整数据库备份2 + 差异数据库备份3
- 完整+差异数据库备份同样不能提供恢复到历史任意某个时刻和故障点的功能。

例3：完整+日志数据库备份与恢复；

- 假设现在有2个完整数据库备份：
 - 10：00时有完整数据库备份1；
 - 12：00时有完整数据库备份2；
- 同时，还假设有3个日志数据库备份：
 - 10：30时有日志数据库备份1，在完整数据库备份1的基础上做；
 - 11：00时有日志数据库备份2，在完整数据库备份1的基础上做；
 - 12：30时有日志数据库备份3，在完整数据库备份2的基础上做；
- 如果需要恢复到11：00时的状态，则应为：

- **完整数据库备份1 + 日志数据库备份1+日志数据库备份2**
- 日志文件一定要保证是连续的；
- 如果需要恢复到12: 30时的状态，则有两个选择：
 - 完整数据库备份2 + 日志数据库备份3
 - 完整数据库备份1 + 日志数据库备份1+日志数据库备份2+日志数据库备份3
- 如果需要恢复到10: 45时的状态呢？则为：
 - 完整数据库备份1 + 日志数据库备份1+日志数据库备份2（10: 30~11: 00之间的事务不要做完，而是指定做到10: 45，即时点恢复就可以；

- 如果需要恢复到12:45时的状态呢？则有两个选择：
 - 完整数据库备份1 + 日志数据库备份1 + 日志数据库备份2 + 日志数据库备份3 + 当前日志备份（12:30以后的日志的事务不要做完，而是指定做到12:45，即时点恢复就可以）；
 - 完整数据库备份2 + 日志数据库备份3 + 当前日志备份（12:30以后的日志的事务不要做完，而是指定做到12:45，即时点恢复就可以）；

例4：完整+差异+日志数据库备份与恢复；

- 假设现在有6个数据库备份内容：
 - ① 10：00时有完整数据库备份1；
 - ② 10：30时有事务日志数据库备份1；
 - ③ 11：00时有事务日志数据库备份2；
 - ④ 11：30时有差异数据库备份1；
 - ⑤ 12：00时有事务日志数据库备份3；
 - ⑥ 12：30时有差异数据库备份2；

- 备份步骤：

- 10：00时完整数据库备份1，备份名称为stu，位置为D盘；
- 10：30时有事务日志数据库备份1，备份名称为stu，位置为D盘；
- 11：00时有事务日志数据库备份2，备份名称为stu，位置为D盘；
- 11：30时有差异数据库备份1，备份名称为stu，位置为D盘；
- 12：00时有事务日志数据库备份3，备份名称为stu，位置为D盘；
- 12：30时有差异数据库备份2，备份名称为stu，位置为D盘；

恢复数据库的多种选择：

- 恢复完整数据库备份：①
- 恢复完整数据库备份+日志备份1：①+②
- 恢复完整数据库备份+日志备份1+日志备份2：①+②+③
- 恢复完整数据库备份+差异备份1+日志备份3：①+④+⑤

- 问题1：为何无法执行差异备份？
 - 创建差异备份必须至少一次的完整数据库备份为基础，所以如果还没有对数据库执行完整数据库备份，则无法执行差异数据库备份；
- 问题2：为什么无法选择事务日志备份、文件和文件组备份？
 - 当数据库工作在简单恢复模型时这两项是无法选择的；
- 问题3：检查点有什么用？
 - SQL每次恢复过程都需要从头到尾扫描日志文件，若日志内容很大，扫描将耗费大量的资源，而且有些事务已经写入数据库，Redo显得多余；检查点机制可减少恢复时前滚的恢复量；