

数据库技术

Database System Technology

郭捷

(guojie@sjtu.edu.cn)

饮水思源 · 爱国荣校



1

事务的基本概念

2

故障的种类

3

恢复的实现技术

4

恢复策略

5

具有检查点的恢复技术

6

数据库镜像

■ 故障是不可避免的

- 计算机硬件故障
- 系统软件和应用软件的错误
- 操作员的失误
- 恶意的破坏

■ 故障的影响

- 运行事务非正常中断（影响数据正确性）
- 破坏数据库（数据丢失）

- 数据库管理系统对故障的对策：
 - DBMS提供恢复子系统；
 - 保证故障发生后，能把数据库从错误状态恢复到某种逻辑一致的状态（某一已知的正确状态）；
 - 保证事务ACID；
- 恢复技术是衡量系统性能优劣的重要指标；



01

什么是事务

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。



02

如何定义事务

如何定义事务?



显式定义方式:

BEGIN TRANSACTION

SQL 语句1

SQL 语句2

○ ○ ○ ○ ○

COMMIT (提交)

BEGIN TRANSACTION

SQL 语句1

SQL 语句2

○ ○ ○ ○ ○

ROLLBACK (回滚)

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 (提交)

- 提交事务的所有操作（读+更新），事务正常结束
- 将事务中所有对数据库的更新写回到磁盘上的物理数据库中
- 事务中所有对数据库的更新永久生效

+ ROLLBACK (回滚)

- 事务异常终止，事务运行的过程中发生了某种故障，不能继续执行
- 系统将事务中对数据库的所有已完成操作全部撤销，事务滚回到开始

时的状态

03

事务的特性



事务的ACID特性:

- 原子性 (**A**tomicity)
- 一致性 (**C**onsistency)
- 隔离性 (**I**solation)
- 持续性 (**D**urability)

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（不正确状态）

数据库系统运行故障，数据库中包含失败事务的结果；

一致性举例



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：

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

- 这两个操作要么全做，要么全不做：

一致性与原子性
密切相关

- 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；

3. 隔离性



■ 对并发执行而言，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；

隔离性举例



T1	T2
① 读A=16	读A=16
②	
③ $A \leftarrow A-1$, 写回A=15	$A \leftarrow A-3$, 写回A=13
④	

T1的修改被T2覆盖了！

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；

事务的特性



■ 保证事务ACID特性是事务管理的重要任务。

■ 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；

不影响原子性

➤ 事务在运行过程中被强行停止。

被强行终止的事务对数据库
和其他事务没有任何影响



1

事务的基本概念

2

故障的种类

3

恢复的实现技术

4

恢复策略

5

具有检查点的恢复技术

6

数据库镜像

故障的种类



 事务故障

 系统故障

 介质故障

 计算机病毒

一、事务内部的故障



✚ 什么是事务故障？

- ✓ 某个事务在运行过程中由于种种原因未运行至正常终止点就夭折了；

✚ 事务故障的常见原因

- ✓ 输入数据有误
- ✓ 运算溢出
- ✓ 违反了某些完整性限制被终止
- ✓ 某些应用程序出错
- ✓ 并行事务发生死锁而被选中撤销该事务
- ✓ ○ ○ ○ ○

事务内部更多的故障是**非预期**的，不能由应用程序处理的

事物内部的故障举例



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; }

/* 事务内部可能造成事务被回滚的情况*/

/* 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; }

事务内部的故障有的是可以
通过事务程序本身发现的

事务故障的恢复



- 发生事务故障时，夭折事务可能已把对数据库的部分修改写回磁盘；
- 事务故障的恢复：事务撤消（UNDO）；
- 强行回滚（ROLLBACK）该事务；
- 撤销该事务对数据库的所有修改，使得该事务像根本没有启动过。

二、系统故障



什么是系统故障？

- 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；
- 内存中数据库缓冲区的信息全部丢失；
- 影响正在运行的所有事务，但不破坏数据库；

系统故障的常见原因



- ✚ 操作系统或DBMS代码错误；
- ✚ 操作员操作失误；
- ✚ 特定类型的硬件错误（如CPU故障）；
- ✚ 突然停电；

✚ 清除尚未完成的事务对数据库的所有修改

■ 系统重新启动时，恢复程序要强行撤消（UNDO）所有未完成事务；

✚ 将缓冲区中已完成事务提交的结果写入数据库

■ 系统重新启动时，恢复程序需要重做（REDO）所有已提交的事务；





三、介质故障



- **硬件故障**使存储在外存中的数据部分丢失或全部丢失，
并影响正在存取这部分数据的所有事务；
- **介质故障**比前两类故障的可能性小得多，但破坏性最大；

介质故障的常见原因



-  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

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；
- 重做自此时开始的所有成功事务，将这些事务已提交的结果重新记入数据库；

四、计算机病毒



- ✚ 计算机病毒是一种人为的故障或破坏,是一些恶意者研制的一种计算机程序,可以像病毒一样繁殖和传播,造成对数据库的危害;
- ✚ 至今没有使计算机终生免疫的疫苗,数据库一旦被破坏仍要用恢复技术把数据库恢复;

计算机病毒的种类



- ✚ 计算机病毒的种类很多,小的病毒只有20条指令,大的病毒像一个操作系统,由上万条指令组成;
- ✚ 有的计算机病毒传播很快,有的病毒有较长潜伏期;
- ✚ 有的病毒感染系统所有程序和数据,有的只对某些特定程序和数据感兴趣;

■ 恢复操作的基本原理：冗余

- 利用存储在系统其它地方的冗余数据来重建数据库中已被破坏或不正确的那部分数据；

■ 恢复的实现技术：复杂

- 一个大型数据库产品，恢复子系统的代码要占全部代码的10%以上；



1

事务的基本概念

2

故障的种类

3

恢复的实现技术

4

恢复策略

5

具有检查点的恢复技术

6

数据库镜像

✚ 恢复机制涉及的关键问题

■ 如何建立冗余数据

- ✓ 数据转储 (backup)
- ✓ 登录日志文件 (logging)

■ 如何利用这些冗余数据实施数据库恢复



01

数据转储

一、什么是转储

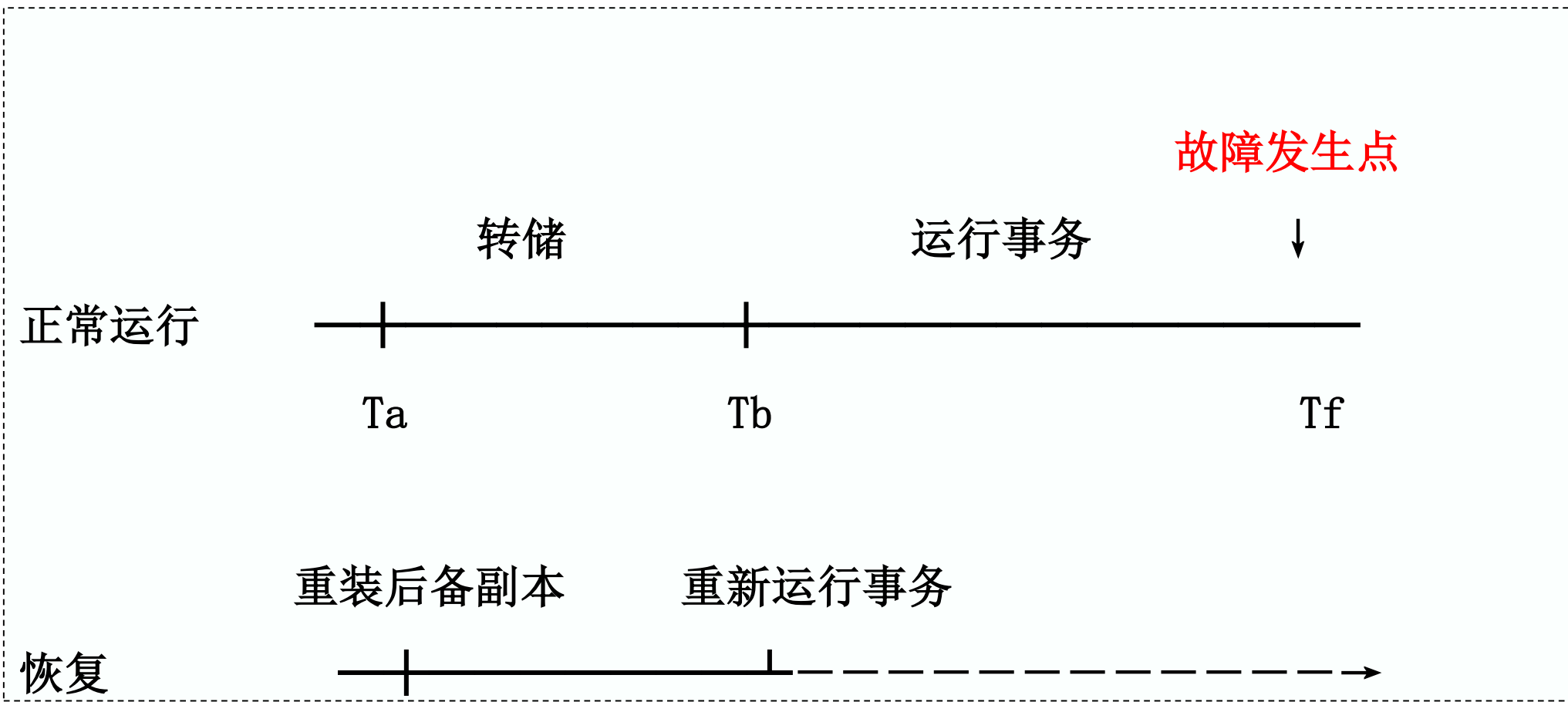


✚ 转储是指DBA定期将整个数据库复制到磁带、磁盘或其他存储介质上保存起来的过程。

✚ 这些备用的数据文本称为**后备副本**或**后援副本**。

✚ 数据库遭到破坏后可以将**后备副本**重新装入，但**重装后备副本**只能将数据库恢复到转储时的状态；

转储



二、转储方法



■ 静态转储与动态转储

■ 海量转储与增量转储

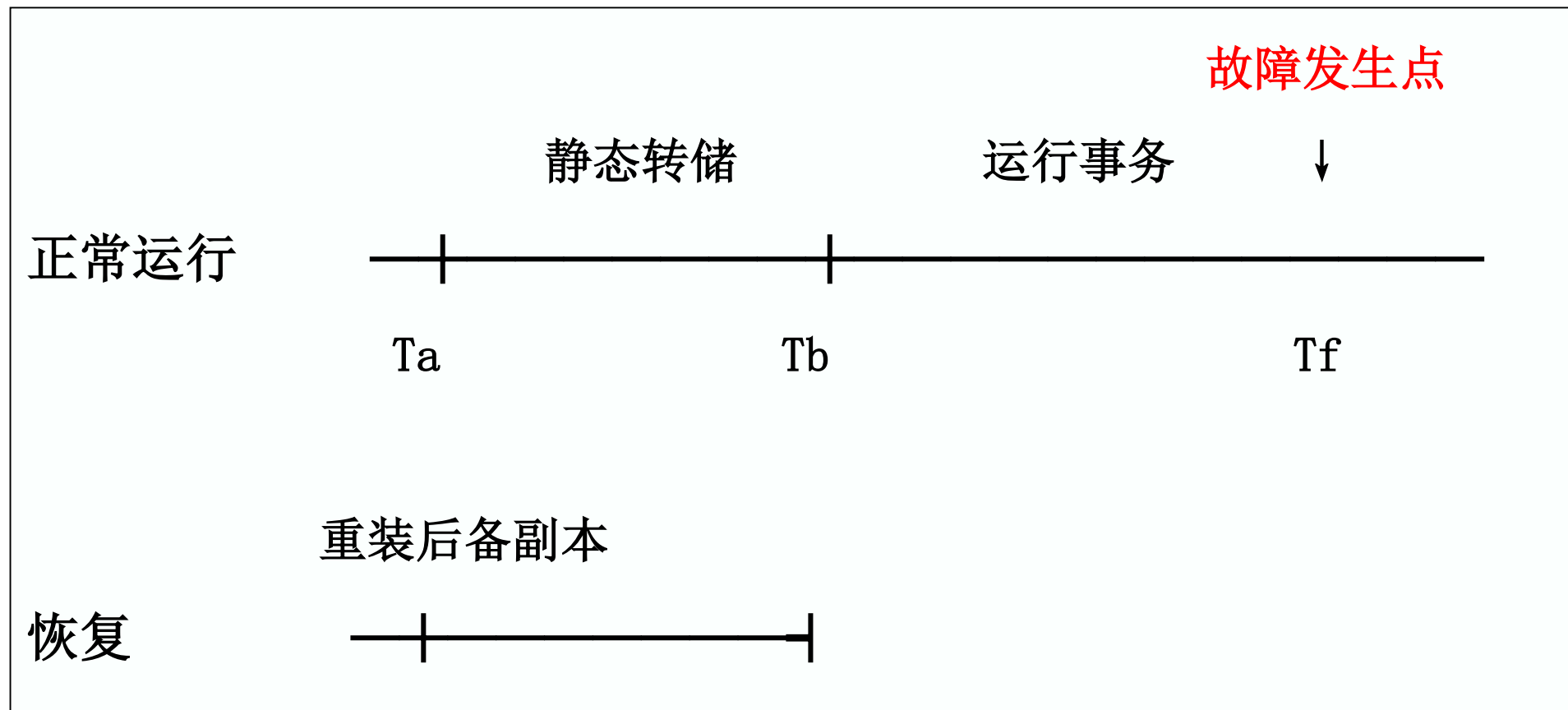
■ 转储方法小结

1、静态转储



- ✚ 在系统中无运行事务时进行转储；
- ✚ 转储开始时数据库处于一致性状态；
- ✚ 转储期间不允许对数据库的任何存取、修改活动；
- ✚ 优点：实现简单
- ✚ 缺点：降低了数据库的可用性
 - 转储必须等正在运行的用户事务结束；
 - 新的事务必须等转储结束才能执行；

利用静态转储副本进行恢复



- ✚ 转储操作与用户事务并发进行
- ✚ 转储期间允许对数据库进行存取或修改
- ✚ 优点
 - 不用等待正在运行的用户事务结束
 - 不会影响新事务的运行
- ✚ 缺点
 - 不能保证副本中的数据正确有效

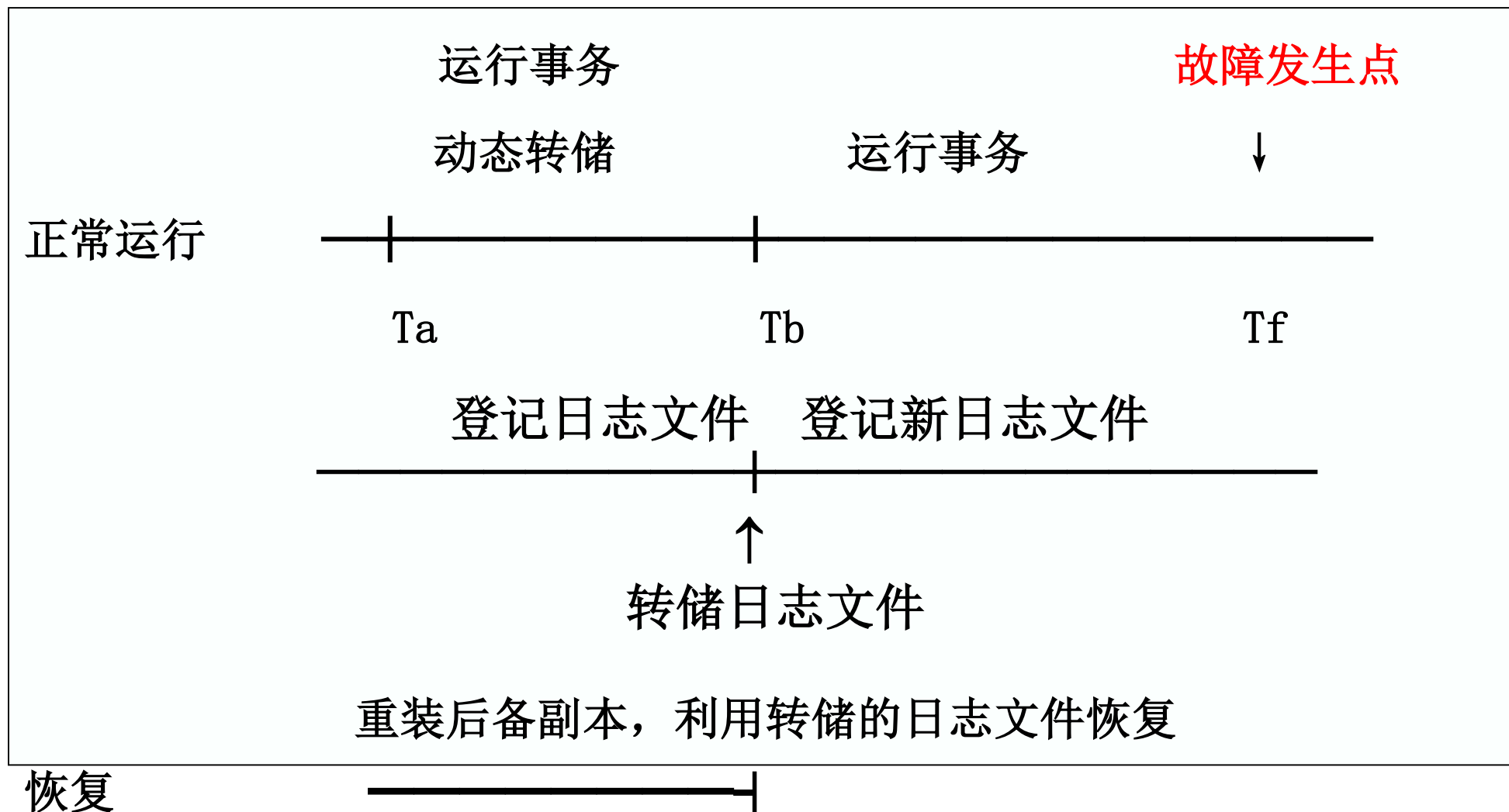
■ 利用动态转储得到的副本进行故障恢复

■ 需要把动态转储期间各事务对数据库的修改活动登记下来，建立 **日**

志文件 (log file) ;

■ **后备副本**加上**日志文件**才能把数据库恢复到某一时刻的正确状态;

利用动态转储副本进行恢复



2、海量转储与增量转储



- 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；
- 海量转储与增量转储比较：
 - 从恢复角度看，使用海量转储得到的后备副本进行恢复往往更方便；
 - 但如果数据库很大，事务处理又十分频繁，则增量转储方式更实用更有效；

3. 转储方法小结



转储方法分类；

		转储状态	
		动态转储	静态转储
转储方式	海量转储	动态海量转储	静态海量转储
	增量转储	动态增量转储	静态增量转储

- ✚ 应定期进行数据转储，制作后备副本。
- ✚ 但转储又是十分耗费时间和资源的，不能频繁进行。
- ✚ DBA应该根据数据库使用情况确定适当的转储周期和转储方法：
 - 每天晚上进行动态增量转储；
 - 每周进行一次动态海量转储；
 - 每月进行一次静态海量转储；

三、Oracle的恢复技术：转储



■ 转储后备副本的方法：

- 文件拷贝
- EXPORT实用程序
- 用SQL命令SP00L
- 自己编程实现

Oracle的恢复技术：转储



■ 重装后备副本的方法：

- 文件拷贝
- IMPORT实用程序
- SQL*LOADER实用程序
- 自己编程实现



02

登记日志文件

一、日志文件的内容



■ 什么是日志文件

■ 日志文件(log)是用来记录事务对数据库的更新操作的文件;

■ 日志文件的格式

■ 以记录为单位的日志文件

■ 以数据块为单位的日志文件

■ 日志文件内容

- 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
- 与事务有关的内部更新操作

■ 均为日志文件中的一个日志记录 (log record)

■ 每条日志记录的内容

- 事务标识（标明是哪个事务）
- 操作类型（插入、删除或修改）
- 操作对象（记录ID、Block NO.）
- 更新前数据的旧值（对插入操作而言，此项为空值）
- 更新后数据的新值（对删除操作而言，此项为空值）

■ 每条日志记录的内容

- 事务标识（标明是哪个事务）
- 更新前数据的旧值（对插入操作而言，此项为空值）
- 更新后数据的新值（对删除操作而言，此项为空值）

取消了操作类型和操作对象！

二、日志文件的用途



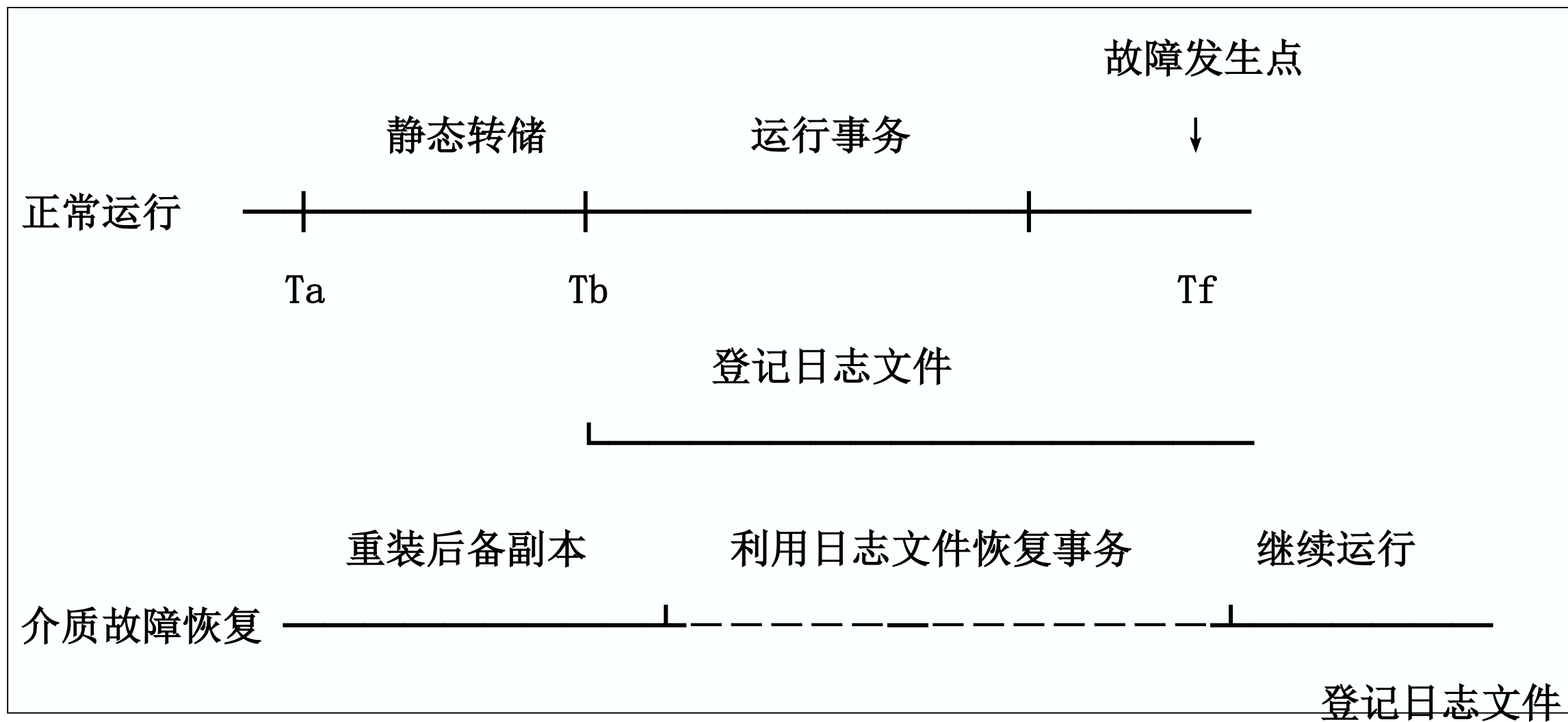
■ 用途

- 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；

■ 与静态转储后备副本配合进行介质故障恢复

- 静态转储的数据已是一致性的数据；
- 如果静态转储完成后，仍能定期转储日志文件，则在出现介质故障重装数据副本后，可利用日志文件副本对已完成的事务进行重做处理，对尚未完成的事务进行撤销处理；
- 这样不必重新运行那些已完成的事务程序就可把数据库恢复到故障前某一时刻的正确状态；

日志文件的用途



- 动态转储介质故障恢复：LOG FILE + 动态转储后备副本
 - 动态转储数据库：同时转储同一时点的日志文件；
 - 后备副本与该日志文件结合起来才能将数据库恢复到一致性状态；
 - 利用这些日志文件副本进一步恢复事务，避免重新运行事务程序。

三、登记日志文件的原则



■ 为保证数据库是可恢复的，登记日志文件时必须遵循两条原则：

■ 登记的次序严格按并行事务执行的时间次序；

■ 必须先写日志文件，后写数据库；

✓ 写日志文件操作：把表示这个修改的日志记录写到日志文件；

✓ 写数据库操作：把对数据的修改写到数据库中；

登记日志文件的原则



为什么要先写日志文件？

- 写数据库和写日志文件是两个不同的操作；
- 在这两个操作之间可能发生故障；
- 如果先写了数据库修改，而在日志文件中没有登记下这个修改，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，按日志文件恢复时只不过是多执行一次不必要的UNDO操作，并不会影响数据库的正确性；

四、Oracle登记日志文件



- ✚ ORACLE V.5: 日志文件以数据块为单位，恢复操作不是基于操作，而是基于数据块；
- ✚ 将更新前的旧值与更新后的新值分别放在两个不同的日志文件中；
 - 记录数据库更新前旧值的日志文件称为数据库前像文件（Before Image，简称BI文件）；
 - 记录数据库更新后新值的日志文件称为数据库的后像文件（After Image，简称AI文件）；
 - BI文件是必须的，AI文件是任选的；
 - 没有AI文件：只能执行UNDO处理，不能执行REDO处理；

■ ORACLE V. 7: REDO日志 + 回滚段

- REDO日志文件：更新数据的前像和后像；
- 回滚段(Rollback Segment)：记录尚未完成的更新事务的更新数据的前像；
- 事务故障恢复：根据回滚段中的数据，撤消该事务的操作；



1

事务的基本概念

2

故障的种类

3

恢复的实现技术

4

恢复策略

5

具有检查点的恢复技术

6

数据库镜像



01

事务故障的恢复

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：

■ 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，不需要用户干预；

事务故障的恢复步骤



1. 反向扫描日志文件（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”（Before Image, BI）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，则用BI 代替 AI（After Image）；

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。



02

系统故障的恢复

系统故障的恢复



■ 系统故障造成数据库不一致状态的原因：

- 一些未完成事务对数据库的更新已写入数据库；
- 一些已提交事务对数据库的更新还留在缓冲区还没来得及写入数据库；

■ 恢复方法：

- Undo 故障发生时未完成的事务；
- 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）

■ REDO-LIST重做队列：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

■ UNDO-LIST撤销队列：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Undo撤销队列事务进行UNDO处理

- 反向扫描日志文件，对每个UNDO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；

3. 对Redo重做队列事务进行Redo处理

- 正向扫描日志文件，对每个Redo事务重新执行日志文件登记的操作，即将日志记录中“更新后的值”写入数据库；

✚ 系统故障恢复

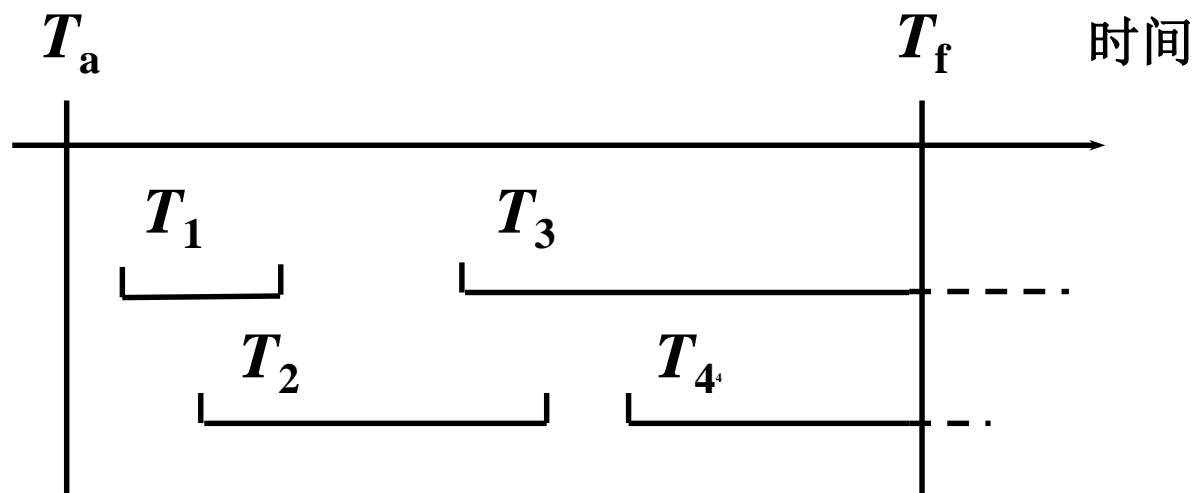
- 首先扫描REDO日志文件，重做所有操作，并对更新操作建立回滚段数据。当遇到提交记录，取消相应回滚段中数据。
- 再根据回滚段中的数据，撤消未正常提交的事务的操作；

✚ 优点：只需要扫描日志文件一遍。

Oracle V.7 的系统故障恢复



Oracle的恢复过程

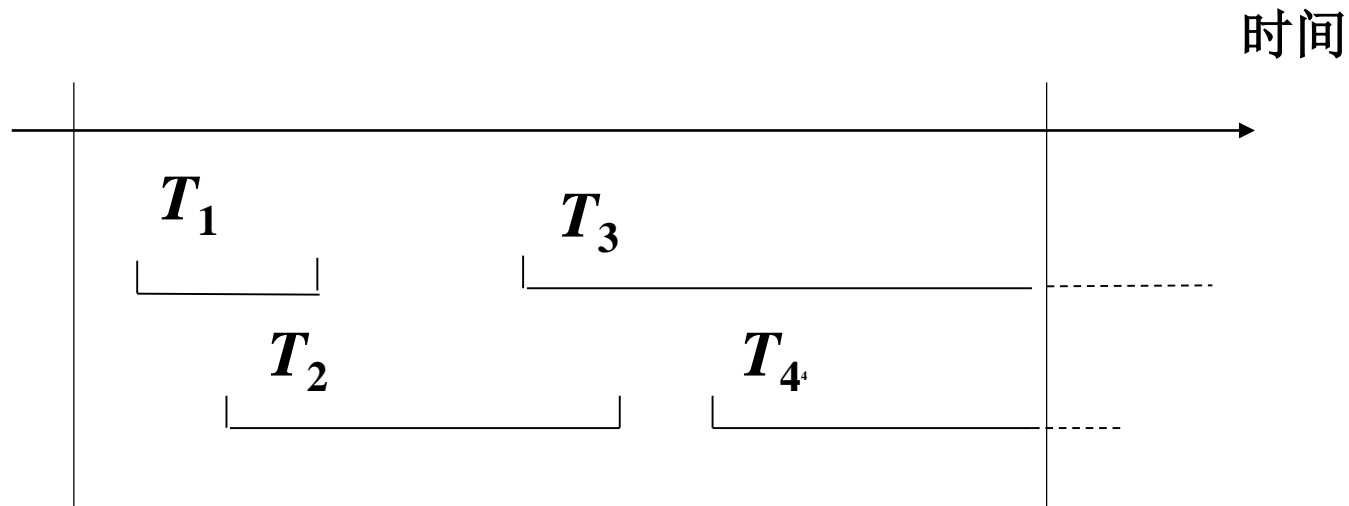


(a) 发生故障，事务非正常终止

Oracle V.7 的系统故障恢复



Oracle的恢复过程

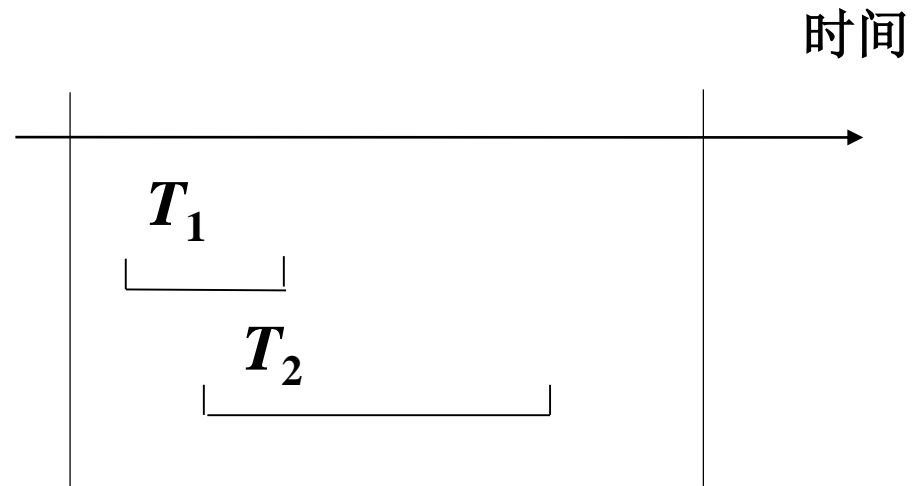


(b) 利用**REDO**文件，重做所有操作

Oracle V.7 的系统故障恢复



Oracle的恢复过程



(c) 利用回滚段撤消未提交的事务数据库恢复到一致性状态

03

介质故障的恢复





重装数据库

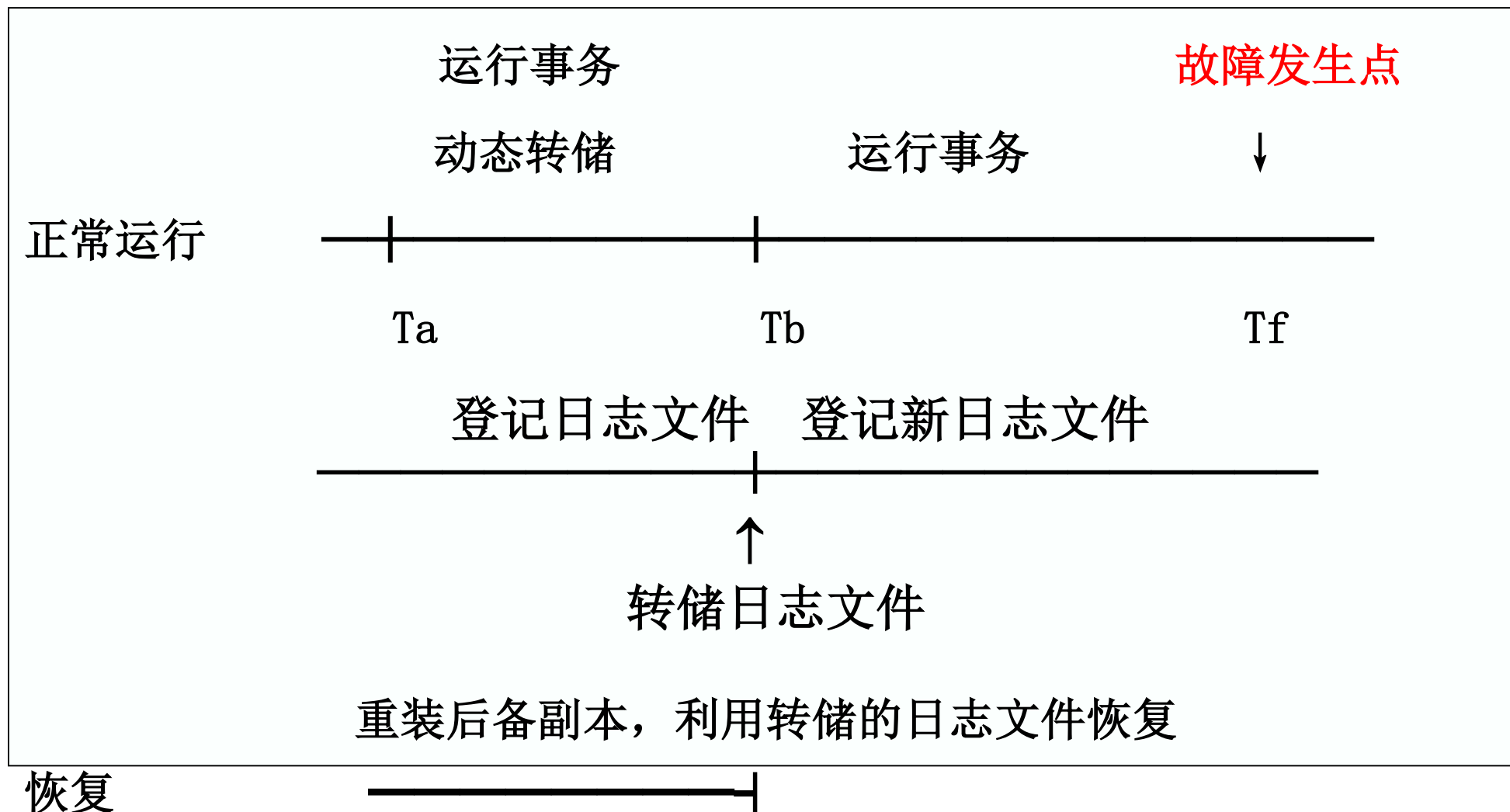
- 使数据库恢复到一致性状态；

重做已完成的事务

1、装入最新的后备数据库副本，使数据库恢复到最近一次转储时的一致性状态。

- 对于静态转储的数据库副本，装入后数据库即处于一致性状态；
- 对于动态转储的数据库副本，还须同时装入转储时刻的日志文件副本，利用与恢复系统故障相同的方法（即REDO+UNDO），才能将数据库恢复到一致性状态。

利用动态转储副本将数据库恢复到一致性状态



2. 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。

- 首先扫描日志文件, 找出故障发生时已提交的事务的标识, 将其记入重做队列。
- 然后正向扫描日志文件, 对重做队列中的所有事务进行重做处理。即将日志记录中“更新后的值”写入数据库。

介质故障的恢复



- ✚ 介质故障的恢复需要DBA介入;
- ✚ DBA的工作:
 - 重装最近转储的数据库副本和有关的各日志文件副本;
 - 执行系统提供的恢复命令;
- ✚ 具体的恢复操作仍由DBMS完成;

ORACLE V.7的介质故障恢复



■ 介质故障恢复:

- 重装数据库后备副本文件，恢复到转储时的数据库一致性状态；
- 利用在此之后转储的REDO日志文件副本将数据库恢复到最近点(类似于系统故障恢复)；



1

事务的基本概念

2

故障的种类

3

恢复的实现技术

4

恢复策略

5

具有检查点的恢复技术

6

数据库镜像



一、问题的提出

二、检查点技术

三、利用检查点的恢复策略

一、问题的提出



✚ 两个问题:

- 搜索整个日志将耗费大量的时间;
- REDO处理: 重新执行, 浪费了大量时间;

■ 具有检查点（checkpoint）的恢复技术：

- 在日志文件中增加检查点记录（checkpoint）；
- 增加重新开始文件；
- 恢复子系统在登录日志文件期间动态地维护日志；

二、检查点技术



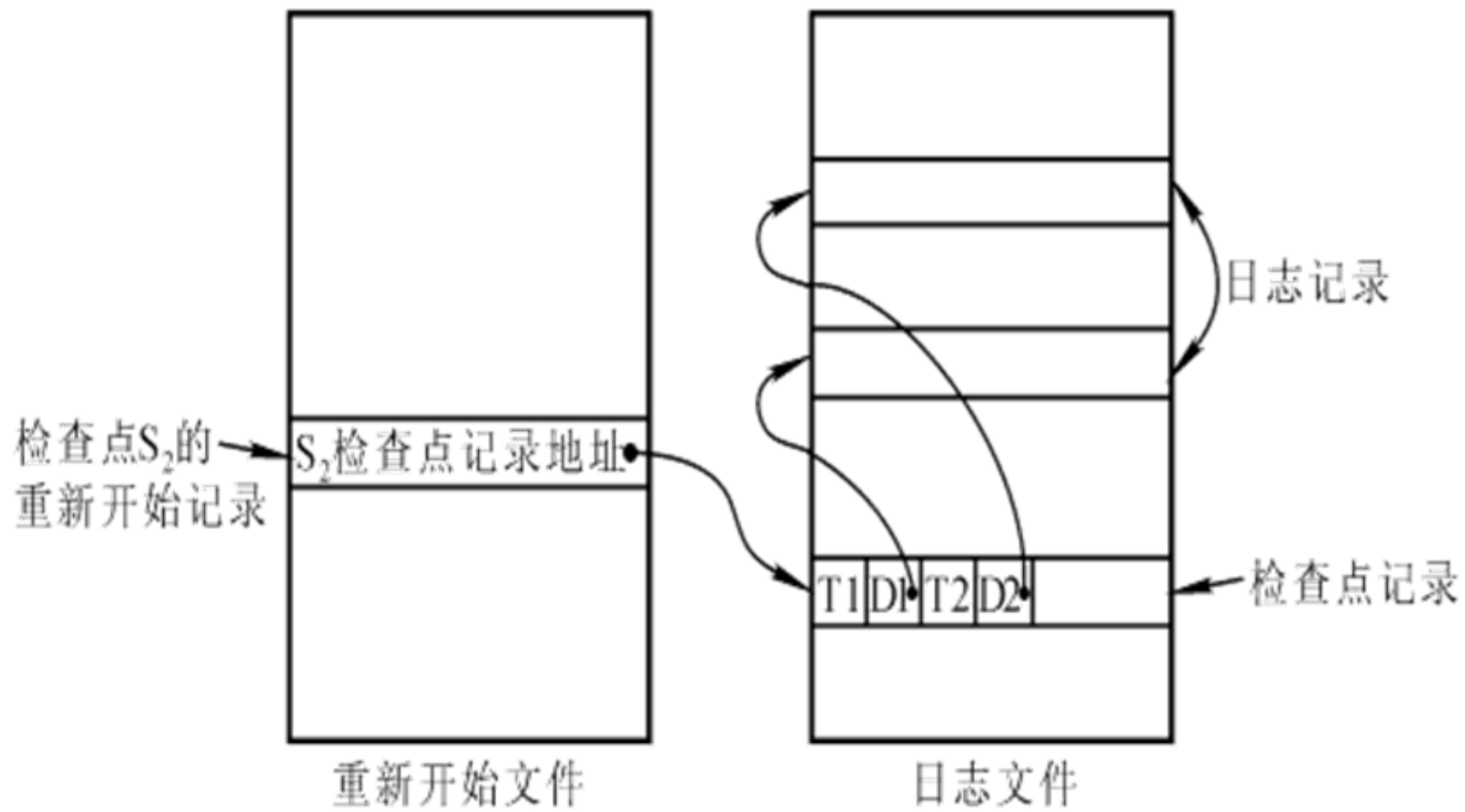
✚ 检查点记录的内容:

- 建立检查点时刻所有正在执行的事务清单;
- 这些事务最近一个日志记录的地址;

✚ 重新开始文件的内容:

- 记录各个检查点记录在日志文件中的地址;

检查点记录的内容



动态维护日志文件的方法



- 周期性地执行**建立检查点、保存数据库状态**的操作：
 - 将当前日志缓冲区中的所有日志记录写入磁盘的日志文件上；
 - 在日志文件中写入一个检查点记录；
 - 将当前数据缓冲区的所有数据记录写入磁盘的数据库中；
 - 把检查点记录在日志文件中的地址写入一个重新开始文件；

建立检查点



✚ 定期:

- 按照预定的一个时间间隔，如每隔1小时建立1个检查点；

✚ 不定期:

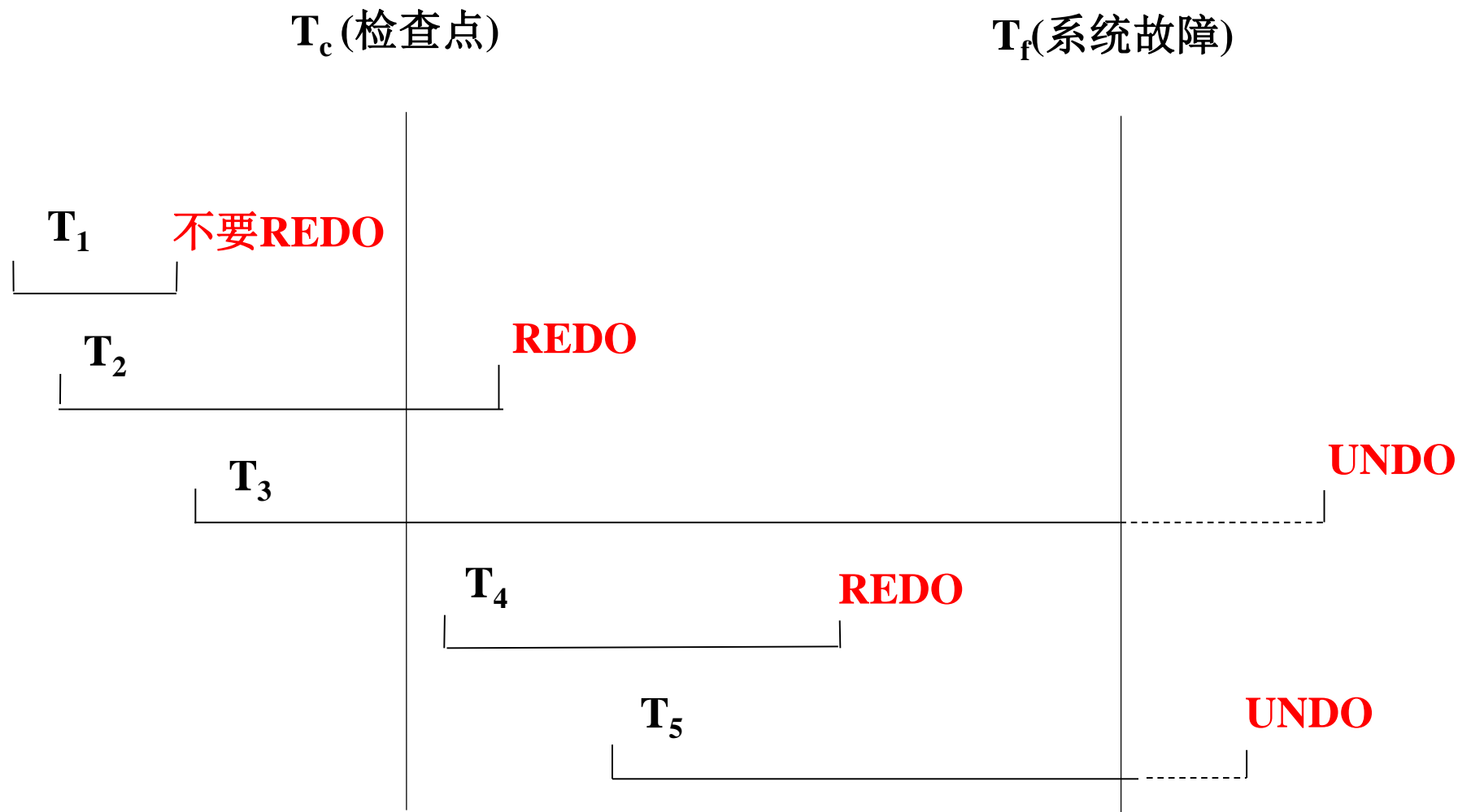
- 按照某种规则，如日志文件已写满一半建立一个检查点；

三、利用检查点的恢复策略



- ✚ 使用检查点法，可以改善恢复效率；
- ✚ 当事务T在一个检查点之前提交，T对数据库所做的修改已写入数据库；
- ✚ 在进行恢复处理时，没有必要对事务T执行REDO操作；

恢复子系统采取的不同策略



利用检查点的恢复步骤



1. 从重新开始文件中找到最后一个检查点记录在日志文件中的地址;
2. 由该地址在日志文件中找到最后一个检查点记录;

3. 由该检查点记录得到检查点建立时刻所有正在执行的事务清单ACTIVE-LIST;

■ 建立两个事务队列

- ✓ UNDO-LIST: 需要执行UNDO操作的事务集合;
- ✓ REDO-LIST: 需要执行REDO操作的事务集合;

■ 把ACTIVE-LIST暂时放入UNDO-LIST队列, REDO队列暂为空。



4. 从检查点开始正向扫描日志文件，直到日志文件结束；
 - 如有新开始的事务 T_i ，把 T_i 暂时放入UNDO-LIST队列；
 - 如有提交的事务 T_j ，把 T_j 从UNDO-LIST队列移到REDO-LIST队列。
5. 对UNDO-LIST中的每个事务执行UNDO操作，对REDO-LIST中的每个事务执行REDO操作；



1

事务的基本概念

2

故障的种类

3

恢复的实现技术

4

恢复策略

5

具有检查点的恢复技术

6

数据库镜像

■ 介质故障是对系统影响最为严重的一种故障，严重影响数据库的可用性；

■ 介质故障恢复比较费时；

■ 为预防介质故障，DBA必须周期性地转储数据库；

■ 提高数据库可用性的解决方案：

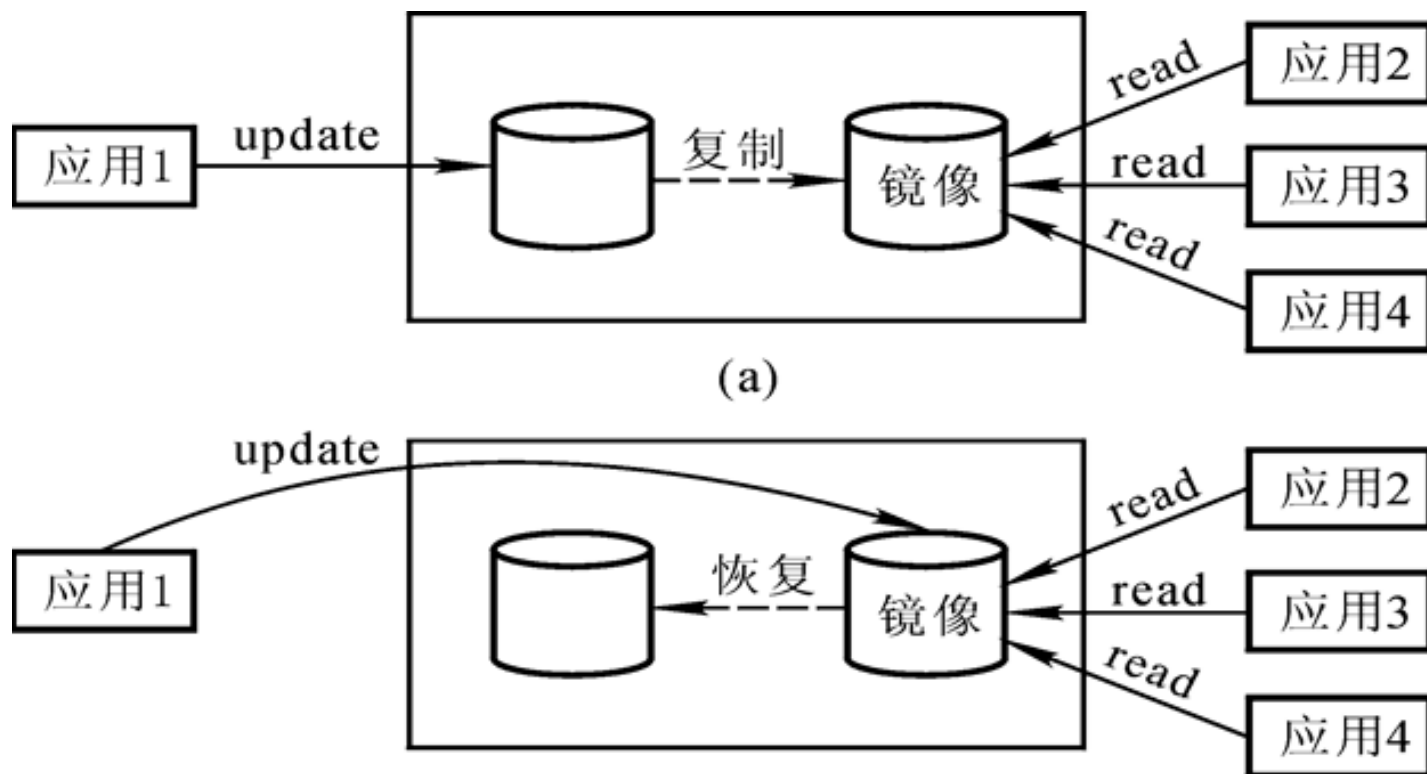
■ 数据库镜像（Mirror）；

数据库镜像



数据库镜像：

- DBMS 自动把整个数据库或其中的关键数据复制到另一个磁盘上；
- DBMS 自动保证镜像数据与主数据的一致性；





■ 出现介质故障时：

■ 镜像磁盘继续提供使用；

■ DBMS 自动利用镜像磁盘数据进行数据库的恢复，不需要关闭系统

和重装数据库副本；



■ 没有出现故障时：

- 可用于并发操作；

- 一个用户对数据加排他锁修改数据；

- 其他用户可以读镜像数据库上的数据，不必等待该用户释放锁；

- 如果数据库只包含成功事务提交的结果，就说数据库处于一致性状态。保证数据一致性是对数据库的最基本的要求。
- 事务是数据库的逻辑工作单位：
- DBMS保证系统中一切事务的原子性、一致性、隔离性和持续性；

- DBMS必须对事务故障、系统故障和介质故障进行恢复；
- 恢复中最经常使用的技术：数据库转储和登记日志文件；
- 恢复的基本原理：利用存储在后备副本、日志文件和数据库镜像中的冗余数据来重建数据库；

常用恢复技术:

■ 事务故障的恢复

✓ UNDO

■ 系统故障的恢复

✓ UNDO + REDO

■ 介质故障的恢复

✓ 重装备份并恢复到一致性状态 + REDO;

提高恢复效率的技术:

检查点技术

- ✓ 可以提高系统故障的恢复效率
- ✓ 可以在一定程度上提高利用动态转储备份进行介质故障恢复的效率

镜像技术

- ✓ 镜像技术可以改善介质故障的恢复效率



THANK YOU !

饮水思源 爱国荣校
