第六章　数据库保护

主要内容

- 事务
- 数据库恢复
- 并发控制
- 数据库安全性
- 数据库完整性

6.1 事务

- **事务**：是一个不可分割的操作序列，该操作序列要么全做，要么全不做。

事务和程序是两个概念。一个程序中可以包含多个事务。

一、事务的概念

- **隐式控制：**

  由DBMS按缺省规定自动划分。

- **显式控制：**

  BEGIN TRANSACTION

  [事务开始]

  COMMIT

  [事务提交，重新改写数据库]

  ROLLBACK

  [事务提交，发生错误撤消]

隐式控制

```
BEGIN   TRANSACTION
   INSERT
   INTO  S(S#, Sname, Sage, Sdept)
   VALUES('10002', '李娜' , 18,'计算机' )
COMMIT
```

显式控制

```
BEGIN  TRANSACTION
 读账户甲的余额Balance；
Balance=Balance-Amount
If( Balance<0) Then
   { 打印’ 金额不足，不能转账’;
     Rollback；
Else
  {  读账户乙的余额Balance1;
      Balance1=Balance1+Amount;
      写回Balance1;
      commint;}
```

二、事务的特性（ACID）

- **原子性（Atomicity）**

  事务是不可分割的工作单位

- **一致性（Consistency）**

  事务提交后，数据库从一个一致性状态变到另一个一致性状态。

- **隔离性（Isolation）**

  在事务完成之前，它对数据库产生的结果不能被其它事务引用。

- **持续性（Durability）**

  一旦事务执行成功(提交)，其对数据产生的效果永久有效。

破坏ACID特性的因素

（1）多个事务并行运行时，不同事务交叉执行

（2）事务在运行过程中被强行停止。

6.2 数据库恢复

➢ **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。

6.2.1 故障的类型

1. 事务故障
2. 系统故障
3. 介质故障
4. 计算机病毒

软故障

硬故障

➢ 1．事务故障

是事务内部的故障（不需要重新启动系统）。

- 预期故障：

  通过在程序中加判断条件来实现

- 非预期的故障：

  如由于死锁而被迫撤销的事务等

2. 系统故障 （需要系统重新启动）

- 故障类型：硬件错误、操作系统故障、系统死机、DBMS代码错误、突然停电等。

- 特点：故障影响正在运行的所有事务，但不破坏数据库。可能会造成数据库中数据的不一致性。其原因：
  - 故障发生时，尚未完成的事务的结果可能已送入到物理数据库。
  - 故障发生时，有些已完成的事务所做的数据更改还在缓冲区中，尚未写到物理数据库中。

3．介质故障

是指存储数据库的磁盘发生故障。

原因：可能是磁盘损坏、磁头碰撞、瞬时强磁场干扰等。

特点：使数据库受到破坏。虽然可能性小，但破坏性最大。

4．计算机病毒

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。

6.2.2 数据库恢复的实现技术

冗余技术

- **两个关键问题：**

  1 如何建立冗余数据

  2 如何利用这些冗余数据实施数据库恢复

1. 通过数据转储建立冗余

转储：定期将DB复制到其它外存保存（副本）

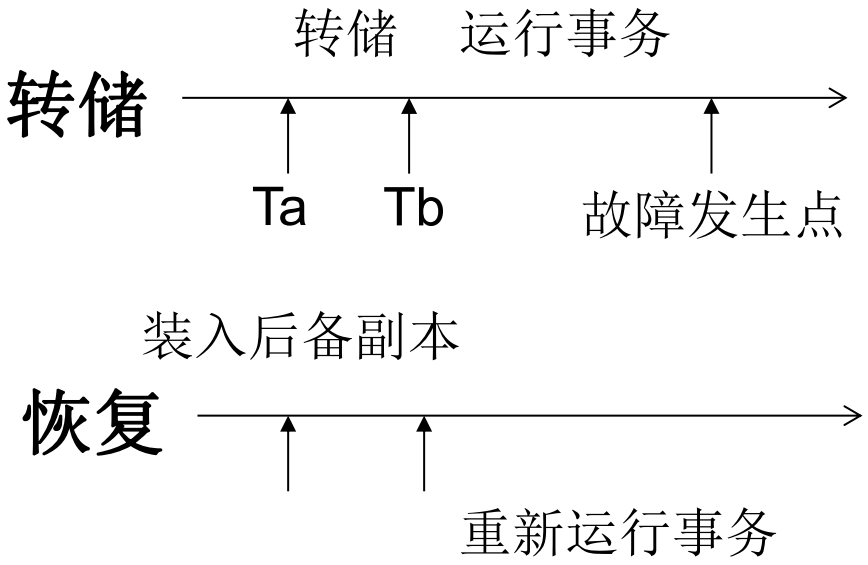
转储类型：

- 静态转储

能保证副本与数据库的一致性；但是效率太低。

- 动态转储

效率高，但不能保证副本与数据库的一致性。



转储方式

- 海量转储：每次转储数据库中的全部数据
- 增量转储：每次转储上一次转储后更新过的数据

		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

2. 通过日志文件建立冗余

日志文件记录对数据库每次更新活动的文件。每次更新活动的内容作为一个“记录”写入日志文件。主要内容包括：

- 事务标识（标明是哪个事务）
- 操作类型及对象（插入、删除、修改，记录内部标识）
- 更新前后的值

日志文件的格式

事务标识	操作类型	对象标识	前像	后像

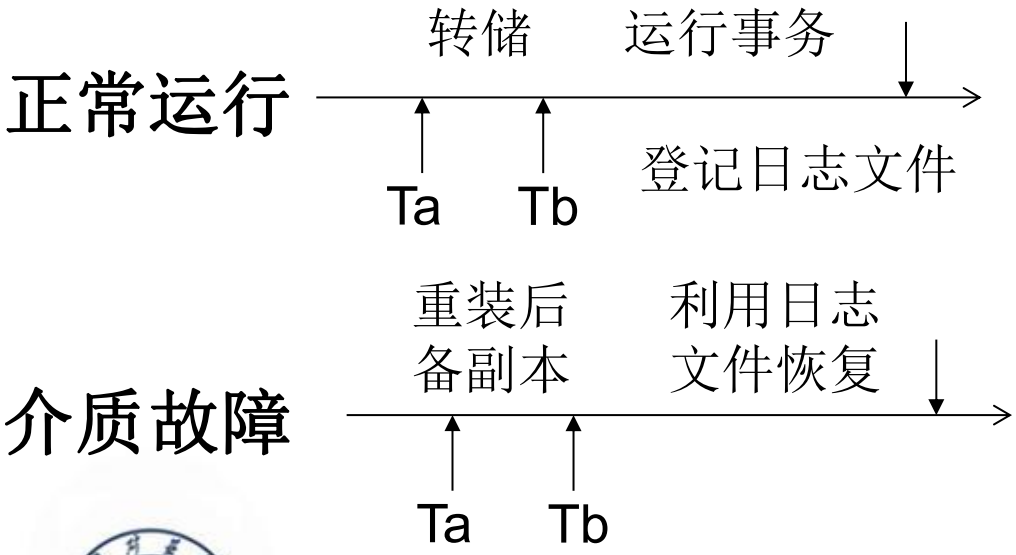
- 事务T开始，日志记录为（T，START…）
- 事务T修改对象A，日志记录为（T， UPDATE ， A ，前像， 后像）
- 事务T插入对象A，日志记录为（T，INSERT，A，后像）
- 事务T删除对象A，日志记录为（T，DELETE，A，前像）
- 事务T提交， 日志记录为（T，COMMIT…）
- 事务T回滚， 日志记录为（T，ROLLBACK…）

登记日志文件要遵循两条原则：

- 登记的次序必须严格按照并发事务执行的次序
- 必须先写日志，后写数据库，并且日志文件和数据库文件不能和数据库文件放在同一个磁盘上。

日志文件的作用

- 事务故障恢复和系统故障恢复必须用日志文件
- 在动态转储方式中必须建立日志文件，后备副本和日志文件结合才能有效恢复数据库
- 在静态转储方式中，也可以建立日志文件



3. 故障恢复

- 1）. 事务故障的恢复

恢复策略：反向扫描日志文件，将日志中更新前的数据写回到数据库中，直至事务的开始标志。

- 事务T修改对象A，日志记录为（T， UPDATE ， A ，前像， 后像）
- 事务T插入对象A，日志记录为（T，INSERT，A，后像）
- 事务T删除对象A，日志记录为（T，DELETE，A，前像

3. 故障恢复

2）、系统故障的恢复

恢复策略：撤销故障发生时未完成的事务，重做已完成的事务。

方法：

（1）正向扫描日志文件；找出故障发生前已经提交的事务，将该事务放入REDO队列；找出故障发生前未提交的事务，将其放入UNDO队列。

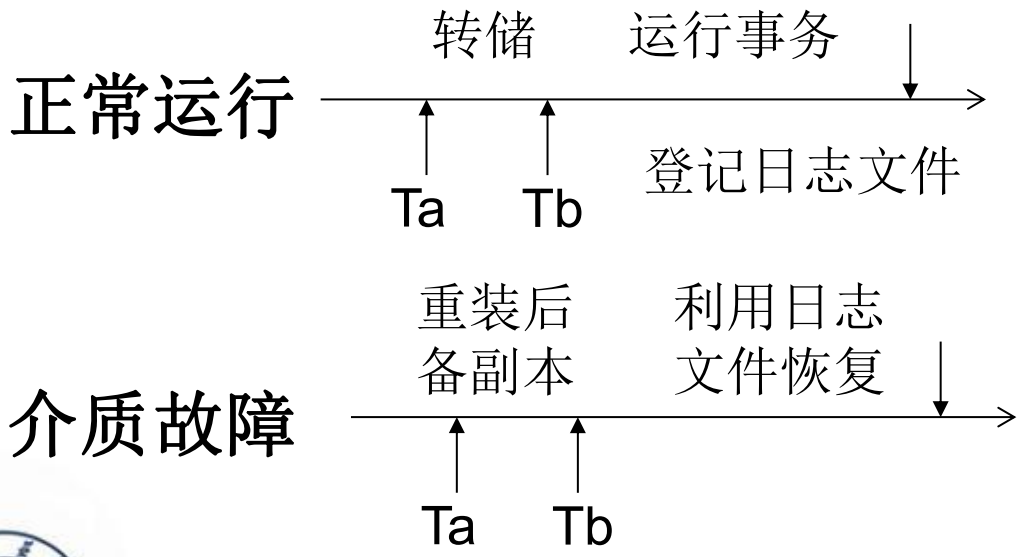
（2）对UNDO队列做撤销操作

（3）对REDO队列做重做操作

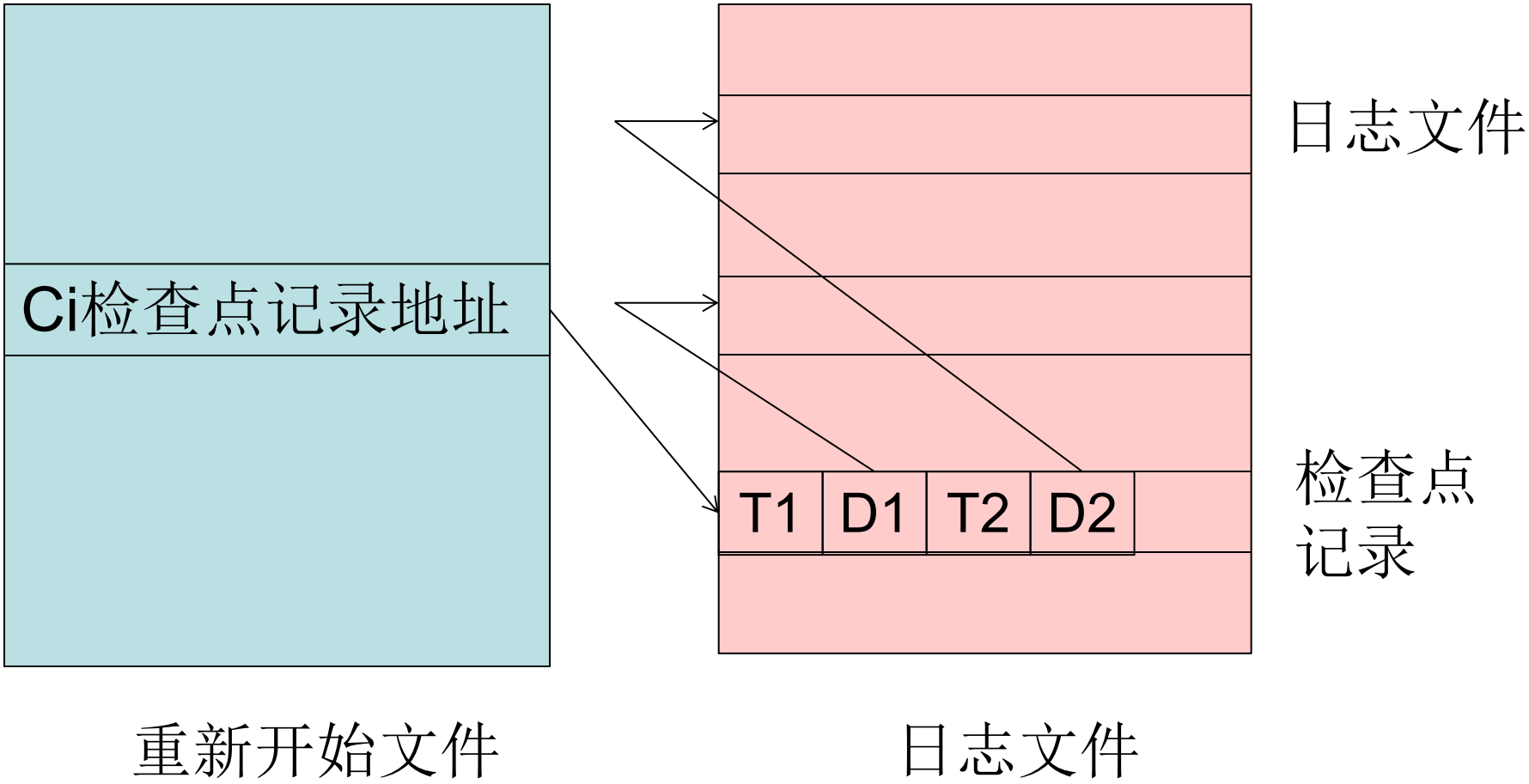
3. 故障恢复

3）、介质故障的恢复

恢复策略：利用数据库副本和日志文件副本进行恢复。（需要DBA介入）

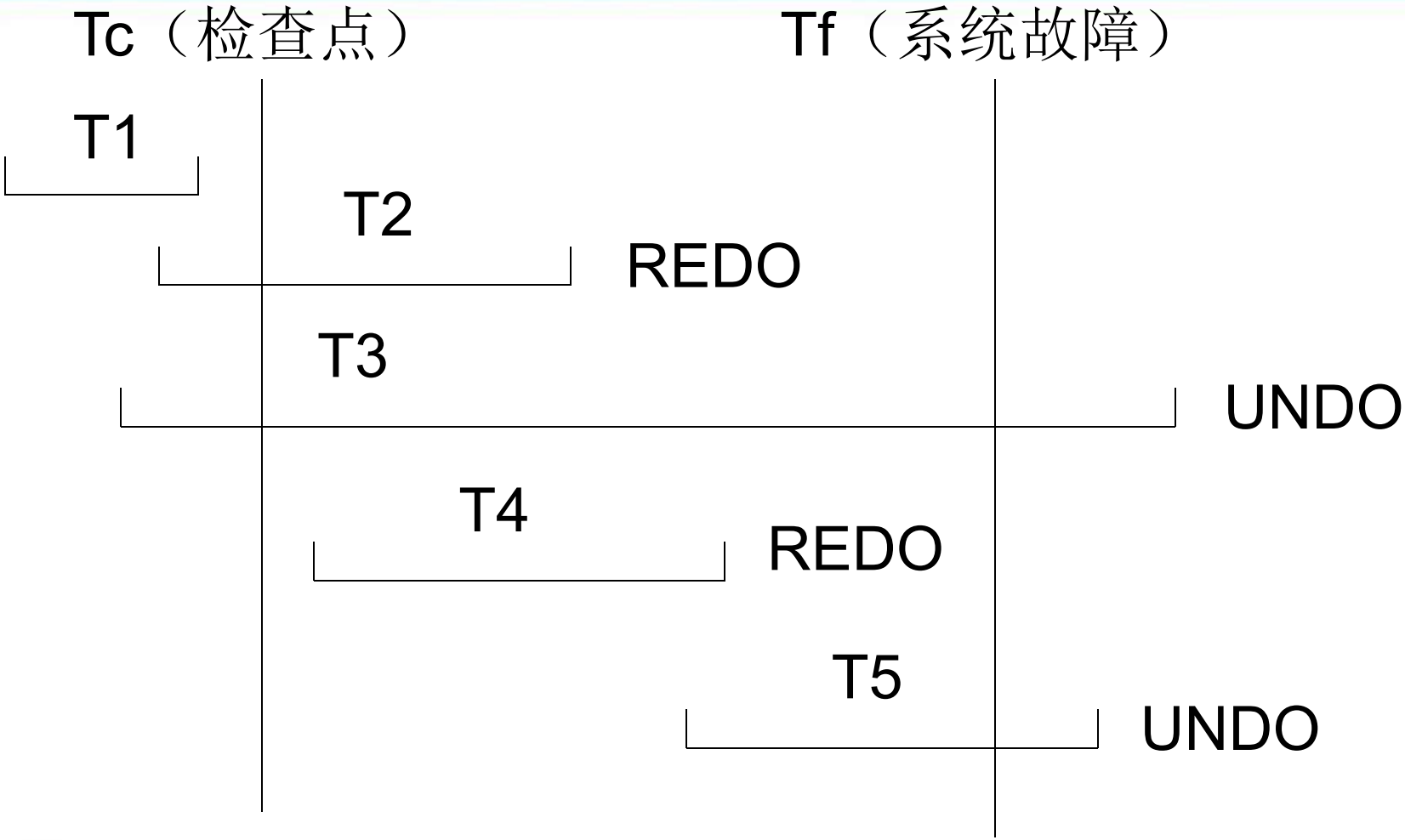


4）具有检查点的恢复策略



具有检查点时，动态维护数据库日志的方法

- 将当前日志缓冲区中的日志记录写入磁盘的日志文件
- 在日志文件中加入一个检查点记录
- 将当前数据缓冲区中的数据记录写入磁盘的数据库
- 将检查点记录在日志文件中的地址写入重新开始文件


Tc（检查点）
Tf（系统故障）
T1
T2
REDO
T3
UNDO
T4
REDO
T5
UNDO

系统使用检查点进行恢复的步骤

- 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，找到最后一个检查点
- 由该检查点找到检查点建立时刻所有的动态事务清单
- 从检查点正向扫描日志文件
  - 如有新开始的事务，放入UNDO队列
  - 如有提交的事务，将该事务从UNDO放入REDO队列
- 执行UNDO处理（反向扫描日志文件，前像代替后像）
- 执行REDO处理（反向扫描日志文件，后像代替前像）