10

第十章　数据库恢复技术

Principles of Database Systems

第十章 数据库恢复技术

10.1 事务的基本概念

- 1. 事务（Transaction）
- 定义：
  - 事务是用户定义的一组数据库操作，这组操作要么都做，要么都不做，不可分割。
  - 恢复和并发控制的基本单位
- 事务和程序比较：
  - 关系数据库事务：一个/一组SQL语句，或一段程序。
  - 一个程序通常包含多个事务

10.1 事务的基本概念

- 定义事务
- 显式定义方式

```
BEGIN TRANSACTION
    SQL 语句1
    SQL 语句1
    ……
 COMMIT WORK / ROLLBACK WORK
```

- 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。

10.1 事务的基本概念

例：银行转帐事务：从A帐户过户50￥到B帐户

```
begin transaction
        read(A);
        A := A – 50;
        write(A);
        if (A<0) then rollback;
        read(B);
        B := B + 50;
        write(B);
commit;
```

read(X)：从数据库传送数据项X到事务的工作区中

write(X)：从事务的工作区中将数据项X写回数据库

10.1 事务的基本概念

- 事务的特性（ACID特性）：
- 原子性（Atomicity）

  事务中包含的所有操作要么全做，要么全不做。
- 一致性（Consistency）

  事务的执行必须是将数据库从一个一致性状态转换到另一个一致性状态。

  一致性状态指数据库中只包含成功事务提交的结果，没有夭折事务残留的修改。
- 隔离性（Isolation）

  事务的执行不受其它并发执行事务的影响。

  对任何一对事务T1，T2，在T1看来，T2要么在T1开始之前已经结束，要么在T1完成之后再开始执行。
- 持久性（Durability）

  一个事务一旦提交之后，它对数据库的影响必须是永久的。

10.1 事务的基本概念

- **隔离性例：**

Begin transaction
R(A)
U(A):A=A-1
Commit;
Begin transaction
R(A)
U(A):A=A-3
Commit

T1	T2
(1) R: A=16	
(2)	R: A=16
(3) A=A-1 写回 A＝15	
(4)	A=A-3 写回A＝13

```
事务1:
begin tran
   declare @sl  int
   select @sl=a from sales where id='A0001'
   waitfor delay '00:00:30.000'
   update sales set a=@sl-1 where id ='A0001'
comit tran
```

10.2 数据库恢复概述

- 故障是不可避免的
  - 系统故障：计算机软、硬件故障
  - 人为故障：操作员的失误、恶意的破坏等，事务在运行过程中被强行停止。
- **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)。保证事务ACID特性。
- DBMS恢复子系统占整个系统代码的10%以上，是衡量系统优劣的重要指标。

10.3 故障的种类

故障种类：事务内部的故障、系统故障、介质故障、计算机病毒。

1. 事务内部故障
   - 有的可以通过事务程序本身发现的。如：转账事务、非法输入、找不到数据等。
   - 有的是非预期的，不能由应用程序处理。如：运算溢出、死锁、违反完整性限制等
- 特点：
  - 非预期事务故障：事务没有达到预期的终点（COMMIT或ROLLBACK），数据库可能处于不正确状态。
  - DBMS仍在正常运行
- 事务故障恢复：撤销事务（UNDO），DBMS自动完成

10.3 故障的种类

例：事务内部故障

```
Create table t1 (
   a int,
   b  int check (b>2)
)
begin tran
  insert into t1 values(1,5)
   insert into t1 values(2,0)
commit
```

SET XACT_ABORT { ON | OFF }

- 当为OFF（默认）时：
  只回滚产生错误的SQL语句，而事务将继续进行处理；

- 当为ON时，
  如果SQL语句运行产生错误，整个事务将终止并回滚。

10.3 故障的种类

2. 系统故障：软故障（Soft Crash）

- 造成系统停止运转的任何事件，使得系统要重新启动。
  - 软件故障（DBMS，OS，APS）；
  - 操作失误；
  - 特定类型的硬件错误（CPU故障等）；
  - 停掉电。
- 特点：
  - 整个系统的正常运行突然被破坏，DBMS不能正常运行；
  - 所有正在运行的事务都非正常终止；
  - 内存数据丢失；外存数据不受影响；
  - DB处于不正确或不一致状态：一些尚未完成事务的结果可能已送入DB；已完成事务的结果可能部分还未送入DB；已完成事务的结果全部未送入DB（未及提交）。

10.3 故障的种类

- 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

10.3 故障的种类

3. 介质故障：硬故障，外存发生故障。

- **分类：**

  磁盘损坏、磁头碰撞、强磁场干扰等

- **特点：**
  - **数据库遭到破坏，存在外存的数据部分丢失或全部丢失，正在执行的事务中断。**
  - **发生可能性小**
  - **破坏性最大**
- **介质故障恢复：**
  - **装入数据库发生介质故障前某个时刻的数据副本**
  - **重做REDO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**

10.3 数据恢复技术

- 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
  - 数据库本身被破坏，或者DB正常，但数据可能不正确

10.4 恢复的实现技术

- 数据恢复技术的基本原理：冗余

  利用冗余，重建数据库，使其达到一致的状态。

- 恢复技术涉及的关键问题：

  1. 如何建立数据的冗余数据?
     - 数据转储（backup）
     - 登录日志文件（logging）
  2. 如何利用这些冗余数据实施数据库恢复?

10.4.1 数据转储（备份）

DBA定期地将整个数据库复制到磁带或另一个磁盘上保存起来的过程。备用的数据称为后备副本或后援副本。

- **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
- **转存方法：**
  1. **静态**转储与**动态**转储
  2. **海量**转储与**增量**转储

1. 静态转储和动态转储

- **静态转储**：
  - 在系统中无运行事务时进行的转储操作
  - 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
  - 得到的一定是一个数据一致性的副本
- 优点：实现简单
- 缺点：降低了数据库的可用性
  - 转储必须等待正运行的用户事务结束
  - 新的事务必须等转储结束

1. 静态转储和动态转储

- **动态转储**：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
- **例**：在转储期间的某个时刻*Tc*，系统把数据A=100转储到磁带上，而在下一时刻*Td*，某一事务将A改为200。转储结束后，后备副本上的A已是过时的数据了。
- **解决**：
  - 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
  - **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

2. 海量转储与增量转储

- **海量转储**：定期或不定期将DB全部数据转储。

  优点：简单

  缺点：重复转储，转储量大。

- **增量转储**（incremental clumping）：只转储上次转储后更新过的数据

  优点：备份量小

  缺点：恢复过程较复杂

- 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

第十章 数据库恢复技术

ACID特性

故障种类：

- 事务内部的故障、
- 系统故障、
- 介质故障、
- 计算机病毒。

10.4 恢复的实现技术

- 数据恢复技术的基本原理：冗余

  利用冗余，重建数据库，使其达到一致的状态。

- 恢复技术涉及的关键问题：

  1. 如何建立数据的冗余数据?
     - 数据转储（backup）：静态转储 / 动态转储

       海量转储 / 增量转储
     - 登录日志文件（logging）
  2. 如何利用这些冗余数据实施数据库恢复?

10.4.2 登记日志文件

1. 日志文件的格式和内容

- **日志log：记录事务对数据库的更新操作的文件。**
- **日志文件类型**
  - **1）记录为单位的日志文件；**
  - **2）数据块为单位的日志文件。**
- **记录为单位日志文件内容:**
  - **1）事务开始(BEGIN TRANSACTION)标记（一个日志记录）；**
  - **2）事务结束(COMMIT或ROLLBACK)标记（一个日志记录）；**
  - **3）每个事务的所有更新操作（每个操作一个日志记录）。**

10.4.2 日志（Logging）

- **每个日志记录内容**
  1）事务标识
  2）操作类型（插入、删除或修改）
  3）操作对象
  4）前像（BI）：更新前数据旧值
  5）后像（AI）：更新后数据新值

2. 以数据块为单位日志文件内容

1）数据块（整块）更新前内容
2）数据块更新后内容

3. 日志的作用

1）事务故障恢复和系统故障恢复
2）协助后备副本进行介质故障恢复

10.4.2 日志（Logging）

- 例：登记日志文件

```
begin tran
    insert into sales values('A0002',0,10);
    update sales set b=b-3 where id = 'A0001'
commit tran
```

- 以记录为单位的日志文件：

```
<T0 start>
<T0, I, sales, null, ('A0002',0,10)>
<T0, U, sales, b(id='A0001'),30,27 >
<T0 commit>
```

10.4.2 日志（Logging）

3. 记录日志的方式

- 在内存中开辟的临时保存日志记录的区域（日志缓冲区）
- 根据需要一次将一个或多个缓冲块写入磁盘，从而减少写磁盘的次数。日志必须存储在稳定存储器上
- 登记次序严格按并发事务执行的时间次序；写到磁盘中的日志记录顺序必须与写入日志缓冲区的次序完全一致
- 先写日志后写数据库规则（why? ）

  - 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DB正确性

10.5 恢复策略

1．事务故障的恢复

事务故障：事务在运行至正常终点前被终止。

目标：维护事务的原子性

例：

```
begin tran
   insert into sales values('A0002',0,10);
   update sales set b=b-3 where id = 'A0001'
   update sales set b=b-3 where id = 'A0001'
commit tran
```

发生故障，事务终止

- **日志文件：**

```
<T0 start>
<T0, I, sales, null, ('A0002',0,10)>
<T0, U, sales, b(id='A0001'),30,27 >
```

10.5 恢复策略

1. 事务故障的恢复方法（续）

恢复方法：利用log UNDO此事务对DB的修改。

步骤：UNDO操作

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。

③继续①→②。

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。

特点：由DBMS自动完成

10.5 恢复策略

2. 系统故障的恢复

目标：撤销故障发生时未完成的事务，重做已完成的事务。

步骤：UNDO未完成事务，REDO已完成事务

①正向扫描日志文件，找出在故障发生前已经提交的事务，将其事务标识记入重做(REDO)队列；同时找出故障发生时尚未完成的事务，将其事务标识记入撤销(UNDO)队列。

②对撤销队列中的各个事务进行撤销(UNDO)处理：反向扫描日志文件，对每个UNDO事务执行UNDO操作；

③对重做队列中的各个事务进行重做(REDO)处理：正向扫描日志文件，对每个REDO事务重新执行登记的操作。即将日志记录中“更新后的值”写入数据库。

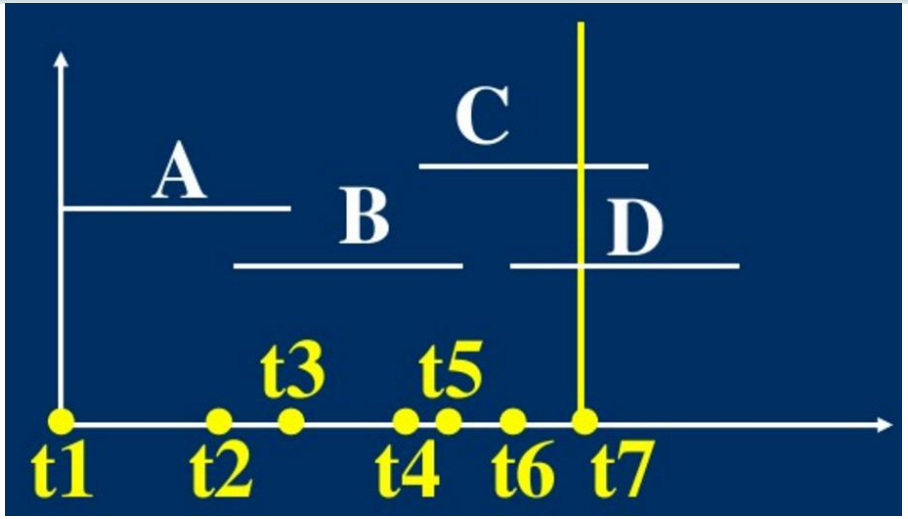
特点：由DBMS在重启时自动完成

10.5 恢复策略

2. 系统故障的恢复

日志文件的内容：

<A start>
<…>
<B start>
<…>
<A commit>
<…>
<C start>
<…>
<B commit>
<…>
<D start>
<…>



1. 正向扫描log，生成Redo-List={A,B}，
   Undo-List={C,D}
2. 反向扫描，对D，C写入“更新前的值”；
3. 正向扫描，对A，B写入“更新后的值”

10.5 恢复策略

3. 介质故障的恢复

目标： 维护事务的持久性

步骤：

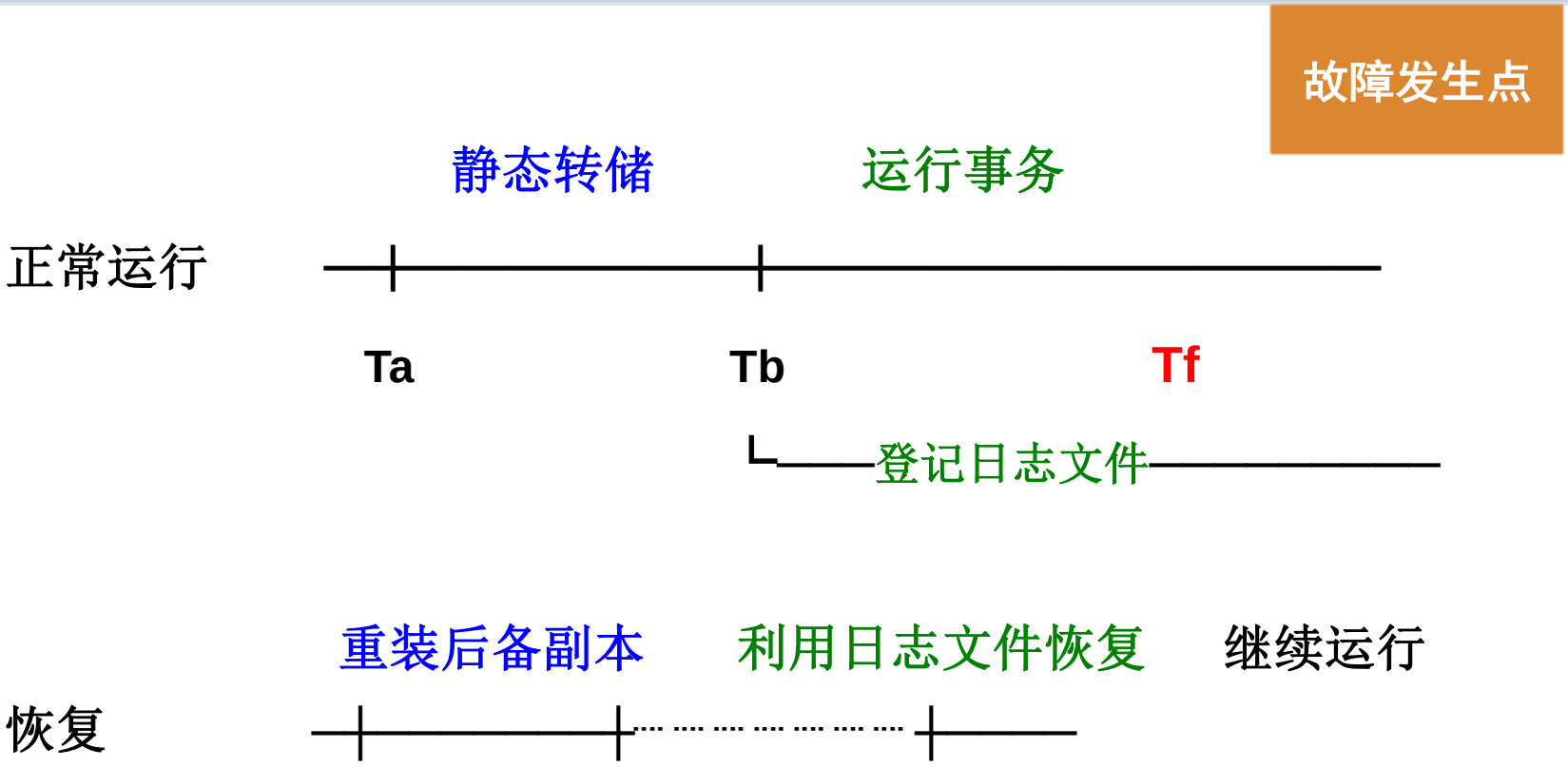
①装入最新的后备数据库副本，使数据库恢复到最近一次转储时的一致性状态。

- 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，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。

②装入有关的日志文件副本，重做已完成的事务。

- **特点：** 需要DBA干预。DBA负责：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

介质故障的恢复



10.6 CheckPoint技术

- **问题的提出：**
  - **日志文件的缺点：耗费大量时间；重复执行；耗费大量空间**
  - REDO操作，重新执行，浪费。
- **基本策略**

  周期性地对日志做检查点，保存DB状态，避免故障恢复时检查整个日志。
- **方法**

  在日志文件中增加一类新的记录——**检查点(Checkpoint)**记录，并增加一个“**重新开始文件**”。周期性执行如下步骤：

  ① 将日志缓冲区中的日志记录写入磁盘

  ② 在日志文件中写一个检查点记录

  ③ 将数据缓冲区中的数据写入磁盘DB中

  ④ 将检查点记录在日志文件中的地址写入重新开始文件

10.6 CheckPoint技术

- **检查点记录的内容**
  - checkpoint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

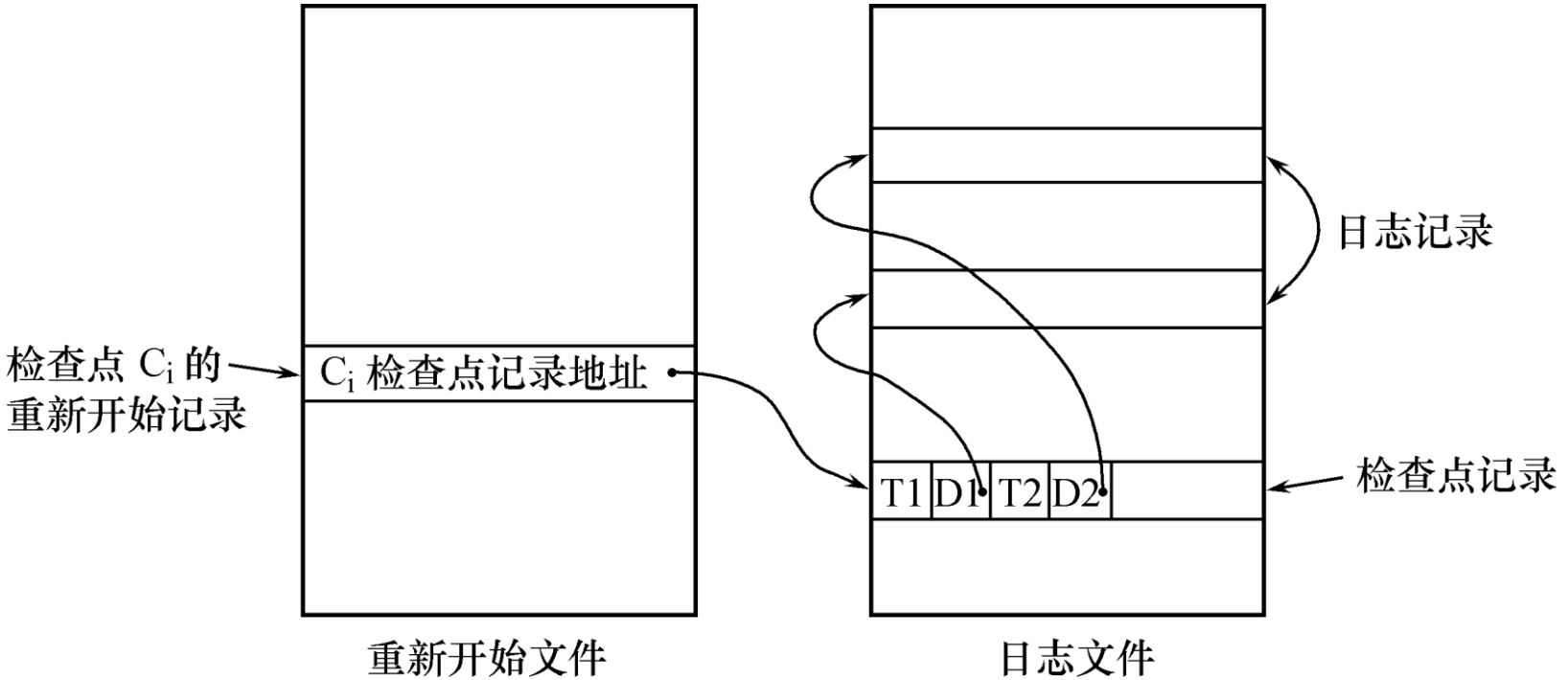


图 具有检查点的日志文件和重新开始文件

CheckPoint技术

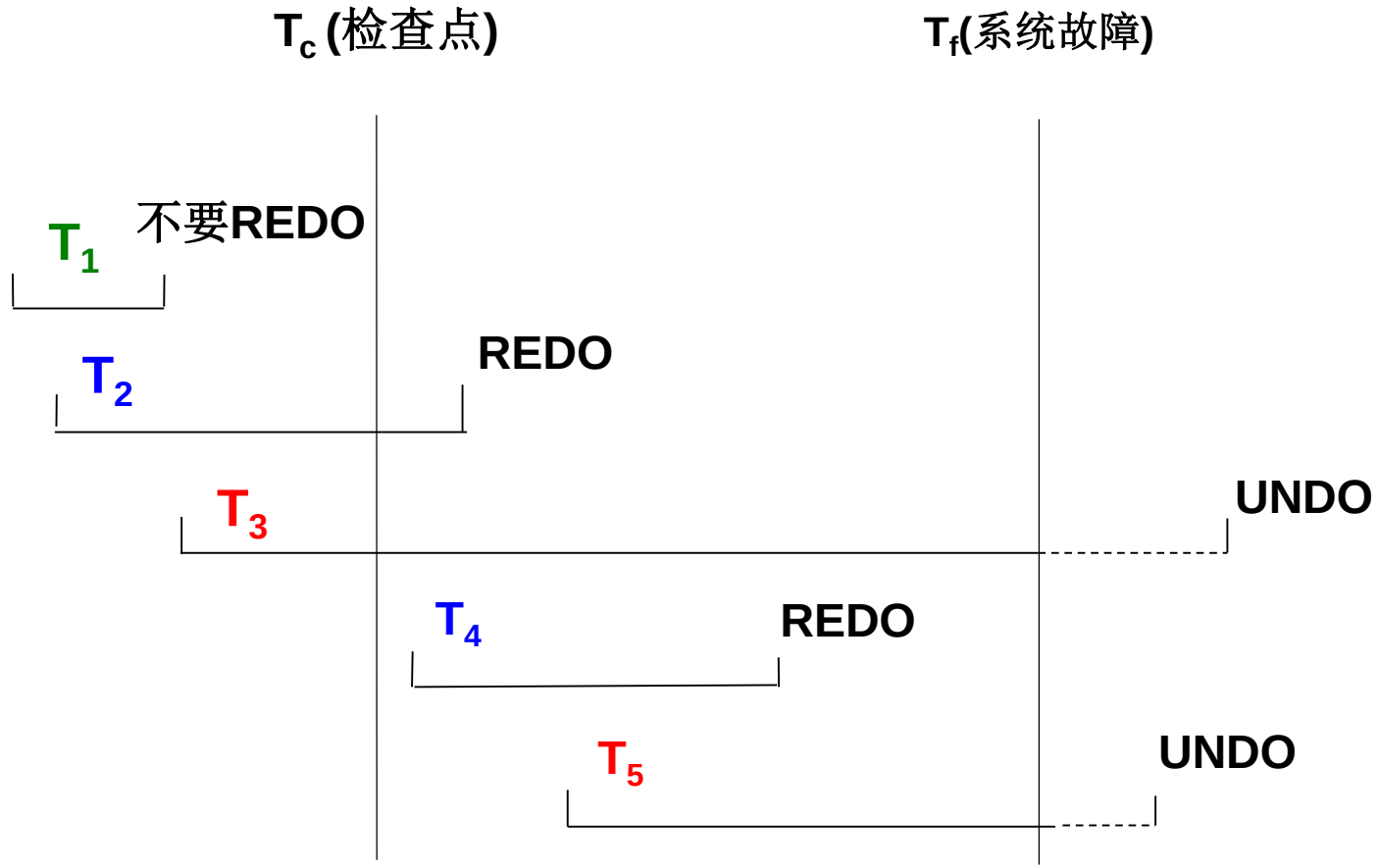
- 恢复子系统可以定期或不定期地建立检查点,保存数据库状态
  - 定期
    - 按照预定的一个时间间隔，如每隔一小时建立一个检查点
  - 不定期
    - 按照某种规则，如日志文件已写满一半建立一个检查点

利用检查点的恢复策略

- 使用检查点方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
    - 写入时间是在这个检查点建立之前或在这个检查点建立之时
    - 在进行恢复处理时，没有必要对事务T执行REDO操作

利用检查点的恢复策略（续）

- 系统出现故障时，恢复子系统将根据事务的不同状态采取不同的恢复策略



CheckPoint技术

- **利用Checkpoint进行恢复的步骤：**
  1. **根据重新开始文件中最后一个检查点记录的地址，在日志文件中找到最后一个检查点记录；**
  2. **由检查点记录得到该检查点记录时刻的事务清单ACTIVE-LIST；**
     - **建立两个事务队列: UNDO-LIST，REDO-LIST**
     - **把ACTIVE-LIST暂时放入UNDO-LIST队列，REDO队列暂为空。**
  3. **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**
  4. **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

10.7 数据库镜像

1. 原因

介质故障：中断运行，周期备份，恢复麻烦。

2. 方法

利用自动复制技术，数据库镜像（Mirror）

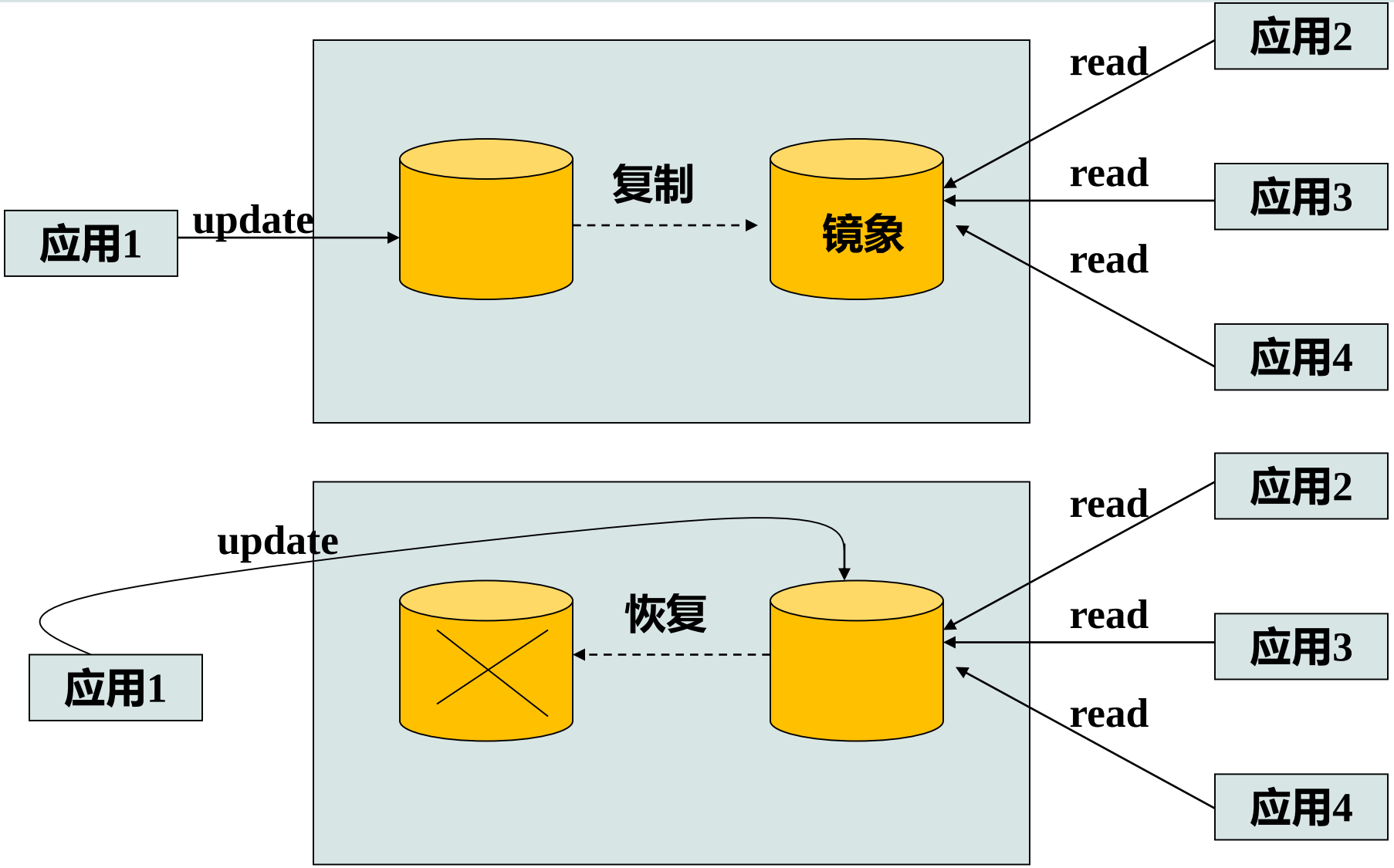
3. 策略

1）整个DB/关键数据复制到另一个介质（镜像磁盘）；

2）DB更新时，DBMS自动保证镜像数据与主数据库的一致性，自动将更新结果复制到该副本；

3）故障发生时，利用该镜像磁盘进行恢复。

镜像


应用1
update
复制
镜象
read
read
read
应用2
应用3
应用4
update
恢复
应用1
read
read
read
应用2
应用3
应用4

数据库镜像

4. 优点

1）无需关闭系统

2）无需重装付本

3）提高可用性

4）提高并发性

5. 缺点

频繁复制更新，效率下降。

小结

- 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

课堂练习

- **Q1: 为什么事务非正常结束时会影响数据库数据的正确性？请举例说明。**

- **Q2: 请简述记录日志的方式的2大原则。**