数据库系统原理

事务的概念 事务并发控制 故障与恢复

故障类型

◆事务故障

- ✓如运算溢出、如转帐时发现帐面金额不足

故障类型

- 事务故障
  - ✓如运算溢出、如转帐时发现帐面金额不足
- **系统故障**
  - ✓内存信息丢失，但未破坏外存中数据如CPU故障、突然停电，DBMS，OS

故障类型

- 事务故障
  - ✓如运算溢出、如转帐时发现帐面金额不足
- 系统故障
  - ✓内存信息丢失，但未破坏外存中数据如CPU故障、突然停电，DBMS，OS
- **磁盘故障**
  - ✓如磁盘的磁头碰撞、瞬时的强磁场干扰

恢复

- 在正常处理事务时采取措施，保证有足够的信息可用于故障
  - ✓备份+日志
- 故障后采取措施，将数据库恢复到一个一致性状态

备份

- 将数据库复制到磁带或另一个磁盘上保存起来的过程。这些备用的数据称为后备（后援）副本

日志

- 日志是用来记录数据库的每一次更新活动的文件，由系统自动记录
- 日志项内容包括：元组名、旧元组值、新元组值、事务标识符、操作标识符等
  - ✓事务τ_i开始时，写日志项：$<\tau_i$ **start**>
  - ✓事务τ_i执行更新前，写日志项：$<\tau_i, X, V_{old}, V_{new}>$，$V_{old}$是X更新前的值，$V_{new}$是X更新后的值
  - ✓事务τ_i结束后，写日志项：$<\tau_i$ **commit**>
- 假定每一个日志项创建后立即写入稳定存储器上日志的尾部

日志举例

τ_0: **read** (x)
x=x - 50
write (*x*)
read (*y*)
y=y + 50
write (*y*)

τ_1 : **read** (*z*)
z=z-100
write (*z*)

<τ_0 start>
<τ_0, *x*, 1000, 950>
<τ_0, *y*, 2000, 2050>

(a)

<τ_0 start>
<τ_0, *x*, 1000, 950>
<τ_0, *y*, 2000, 2050>
<τ_0 commit>
<τ_1 start>
<τ_1, *z*, 700, 600>

(b)

<τ_0 start>
<τ_0, *x*, 1000, 950>
<τ_0, *y*, 2000, 2050>
<τ_0 commit>
<τ_1 start>
<τ_1, *z*, 700, 600>
<τ_1 commit>

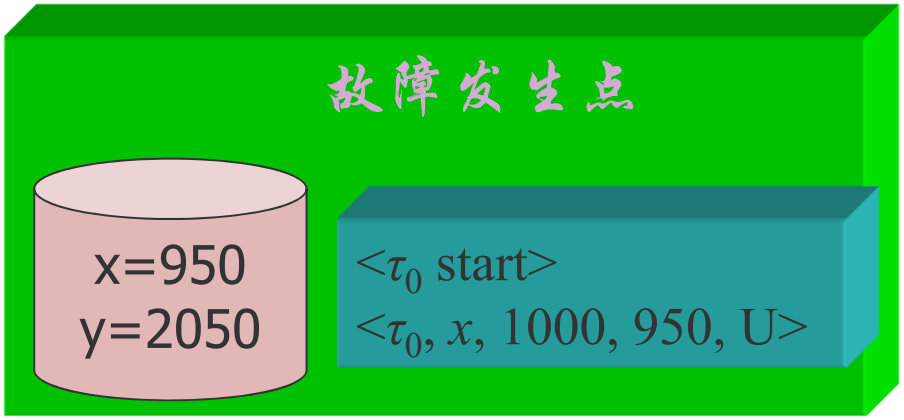
(c)

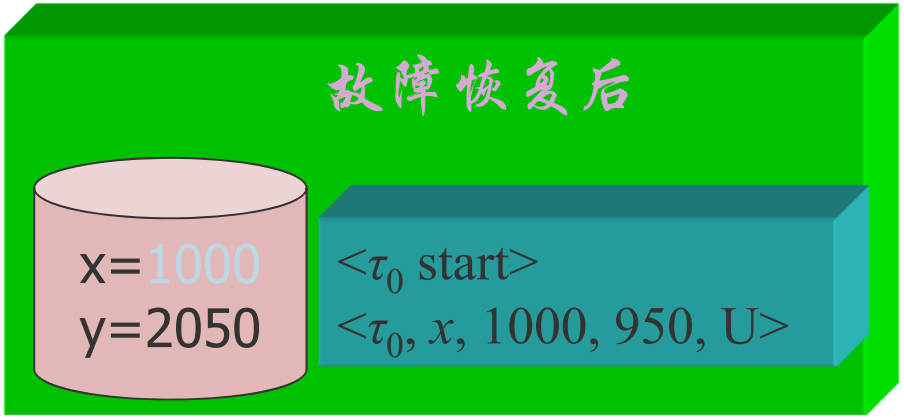
先写日志的原则（WAL）

- 对于尚未提交的事务，在将DB缓冲区写到外存之前，必须先将日志缓冲区内容写到外存去
- 日志记录将要发生何种修改
- 写入DB表示实际发生何种修改
- 如果先写DB，则可能在写的中途发生系统崩溃，导致内存缓冲区内容丢失，而外存DB处于不一致状态，由于日志缓冲区内容已破坏，导致无法对DB恢复

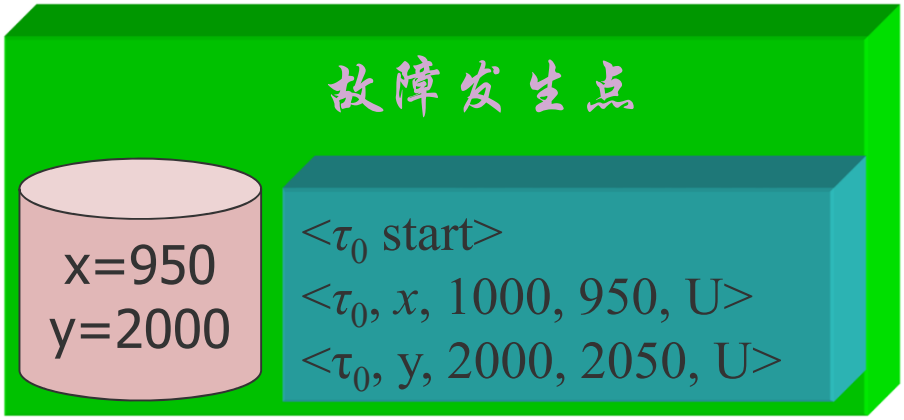
先写DB VS. 先写日志

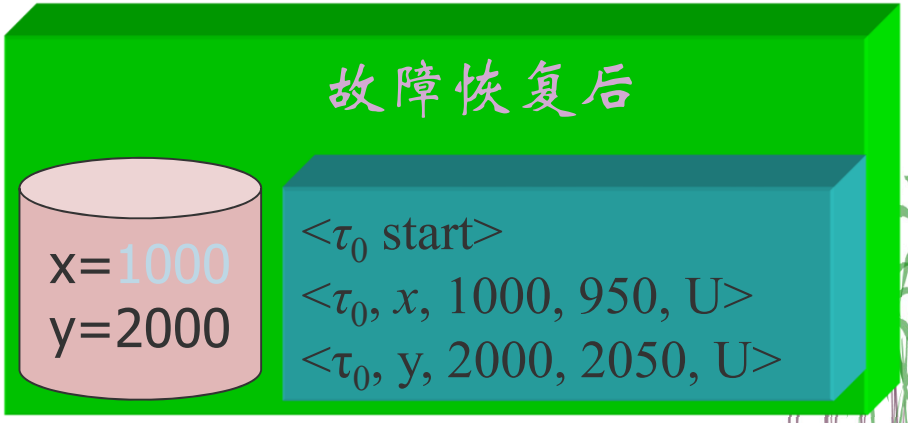
先写DB





先写日志





故障时的事务

- 圆满事务
  - 日志文件中记录了事务的commit标识
- 天折事务
  - 日志文件中只有事务的Begin transaction标识，无commit

<τ_0 start>
<τ_0, x, 1000, 950>
<τ_0, y, 2000, 2050>

τ_0天折事务

<τ_0 start>
<τ_0, x, 1000, 950 >
<τ_0, y, 2000, 2050 >
<τ_0 commit>
<τ_1 start>
<τ_1, z, 700, 600 >

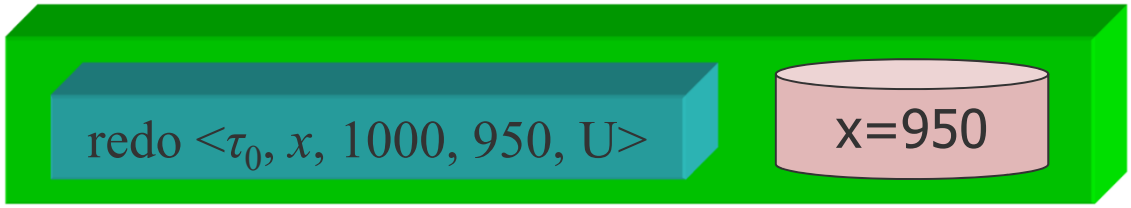
τ_0圆满事务，τ_1天折事务

<τ_0 start>
<τ_0, x, 1000, 950>
<τ_0, y, 2000, 2050>
<τ_0 commit>
<τ_1 start>
<τ_1, z, 700, 600>
<τ_1 commit>

τ_0圆满事务，τ_1圆满事务

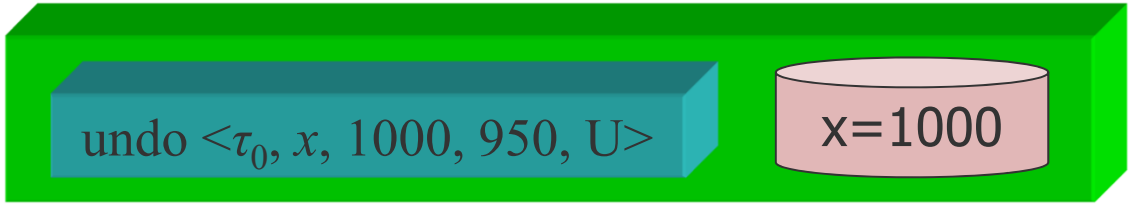
基本的恢复操作

- 对圆满事务所做过的修改操作应执行redo操作，即重新执行该操作，修改对象被赋予新记录值



redo=redo^2

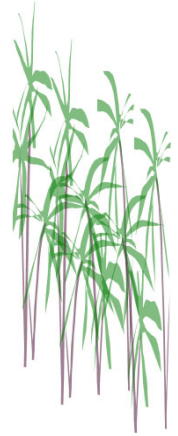
- 对夭折事务所做过的修改操作应执行undo操作，即撤消该操作，修改对象被赋予旧记录值



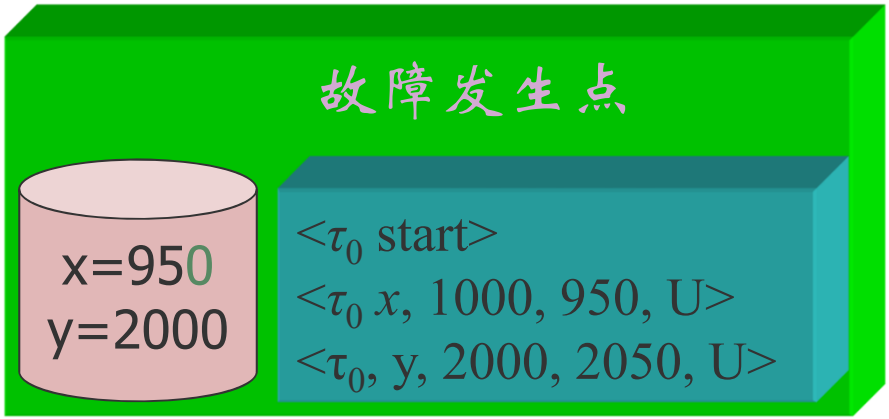
undo=undo^2

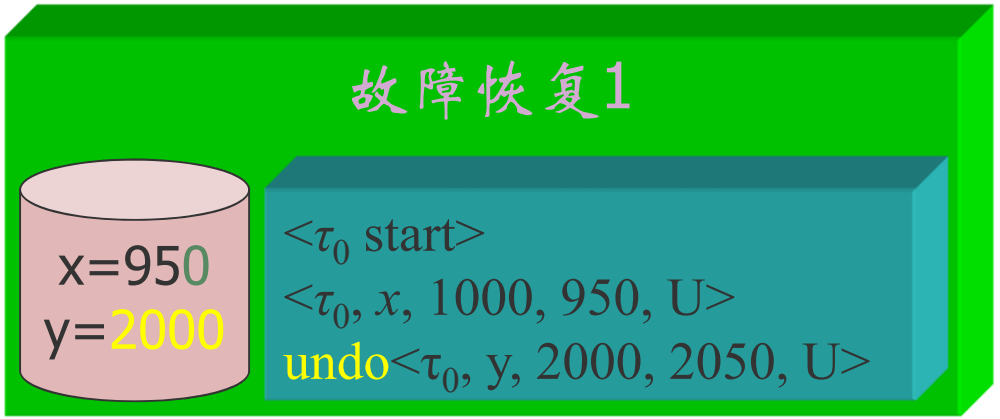
事务故障恢复

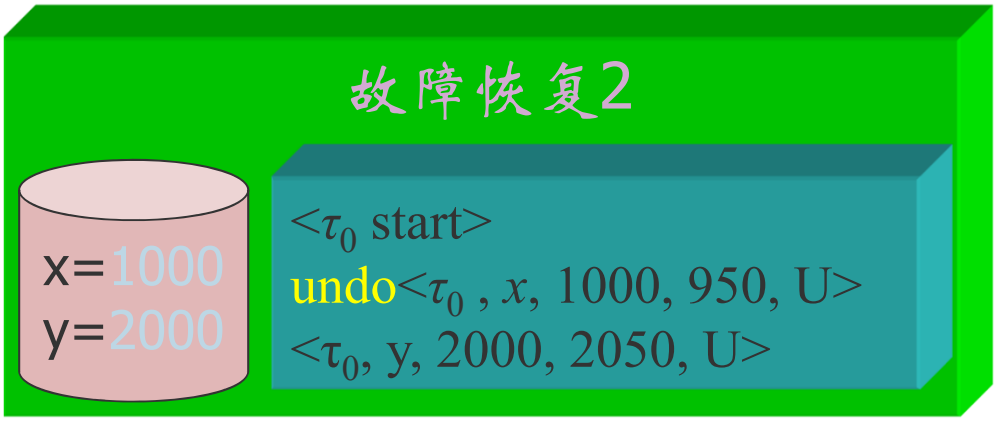
- 撤消事务已对数据库所做的修改
- 措施
  - 反向扫描日志文件，查找该事务的更新操作
  - 对该事务的更新操作执行逆操作，即将事务更新前的旧值写入数据库
  - 继续反向扫描日志文件，查找该事务的其他更新操作，并做同样处理
  - 如此处理下去，直至读到此事务的开始标识，事务的故障恢复就完成了



事务故障恢复


故障发生点
x=950
y=2000
<τ0 start>
<τ0 x, 1000, 950, U>
<τ0, y, 2000, 2050, U>


故障恢复1
x=950
y=2000
<τ0 start>
<τ0, x, 1000, 950, U>
undo<τ0, y, 2000, 2050, U>


故障恢复2
x=1000
y=2000
<τ0 start>
undo<τ0 , x, 1000, 950, U>
<τ0, y, 2000, 2050, U>


故障恢复结束
x=1000
y=2000
<τ0 start>
<τ0, x, 1000, 950, U>
<τ0, y, 2000, 2050, U>

系统故障恢复

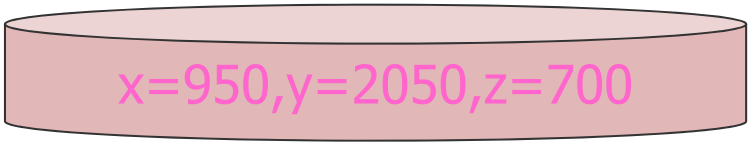
- 不一致状态原因
  - 未完成事务对数据库的更新已写入数据库
  - 已提交事务对数据库的更新未写入数据库
- 措施
  - 正向扫描日志文件，找出圆满事务，记入重做队列；找出夭折事务，记入撤消队列
  - 反向扫描日志，对撤消队列中事务τi的每一个日志记录执行undo操作
  - 正向扫描日志文件，对重做队列中事务τi的每一个日志记录执行redo操作

系统故障恢复举例

<τ_0 start>
<τ_0, x, 1000, 950, U>
<τ_0, y, 2000, 2050, U>
<τ_0 commit>
<τ_1 start>
<τ_1, z, 700, 600, U>

T_0圆满事务，T_1夭折事务

<τ_0 start>
redo<τ_0, x, 1000, 950, U>
redo<τ_0, y, 2000, 2050, U>
<τ_0 commit>
<τ_1 start>
undo<τ_1, z, 700, 600， U>



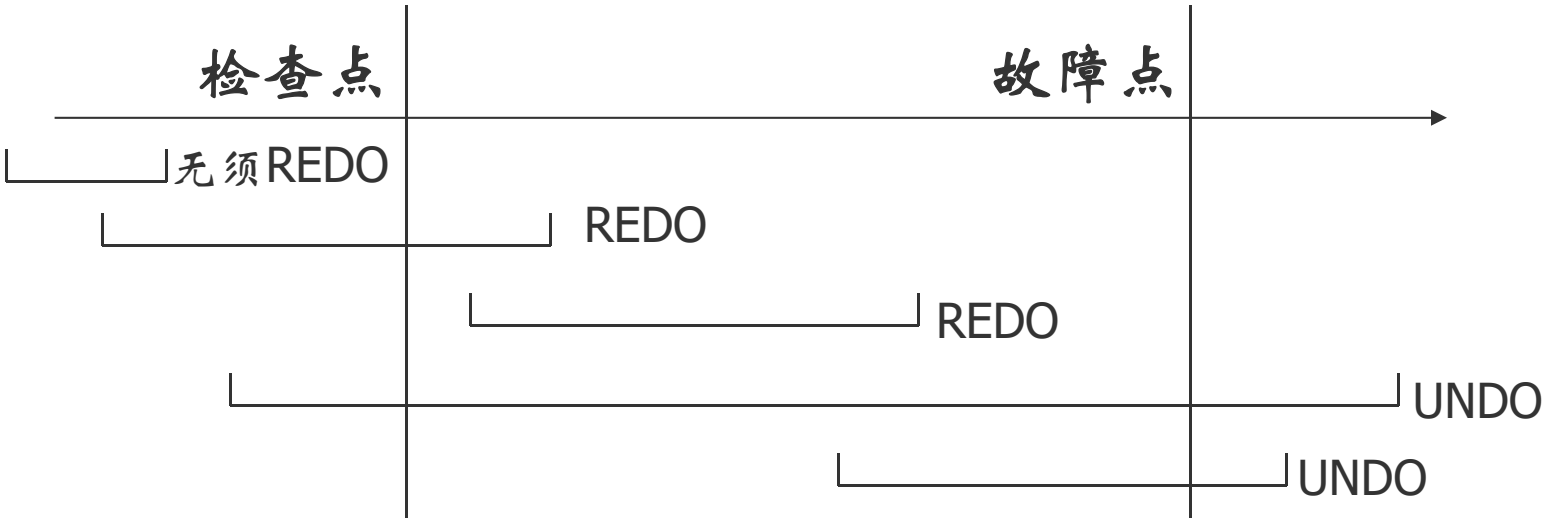
检查点(Checkpoint)

- 问题
  - ✓ 当系统故障发生时，我们必须搜索整个日志，以决定哪些事务需要redo，哪些需要undo
  - ✓ 大多数需要被重做的事务其更新已经写入了数据库中（redo^2）。尽管对它们重做不会造成不良后果，但会使恢复过程变得更长
- 检查点原理

保证在检查点时刻日志与数据库内容是一致的

检查点(Checkpoint)

- 带有检查点记录的日志生成
  - ✓ 将当前日志缓冲区的所有日志记录写入稳存中
  - ✓ 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稳存中
  - ✓ 输出检查点时活跃事务的列表L

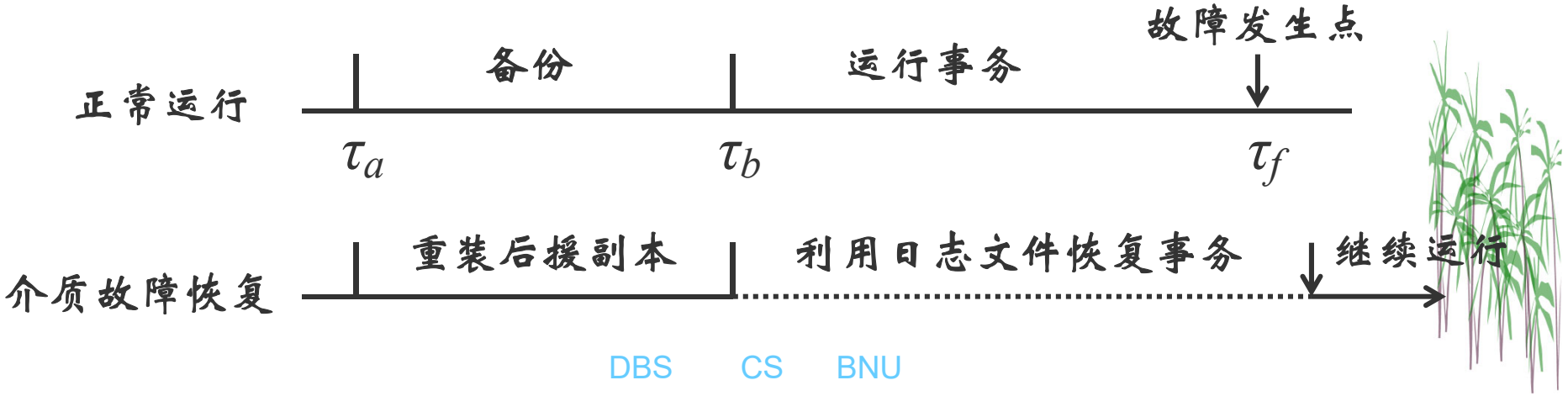


利用检查点的恢复策略

- 需要考虑的事务
  - 检查点时刻仍然活跃的事务
  - 检查点时刻后新启动的事务
- 措施
  - 正向扫描日志文件，找出圆满事务，记入重做队列；找出夭折事务，记入撤消队列
  - 反向扫描日志，对撤消队列中事务τi的每一个日志记录执行undo操作
  - 正向扫描日志文件，对重做队列中事务τi的每一个日志记录执行redo操作

磁盘故障恢复

- 磁盘上数据文件和日志文件遭到破坏
- 措施
  - 装入最新的数据库后备副本，使数据库恢复到最近一次备份时的一致性状态
  - 装入相应的日志文件副本，重做已完成的事务



事务的概念　　事务并发控制　　故障与恢复

- 1、事务日志记录包含_____。
- a)事务名称、数据项名称、数据项的旧值和新值
- b)事务名称、数据项名称、数据项的旧值
- c)事务名称、数据项名称、数据项的新值
- d)事务名称、数据项名称

- 15、应对介质(或磁盘)故障时，必要的操作是_____。
- a)保留数据库的一份冗余备份　b)从不中断事务
- c)在单用户环境下执行事务　d)以上都是

- 2、检查点技术用来限制_____。
- a)日志信息的容量　　b)搜索的数量
- c)需要在事务日志文件上实施的后续处理
- d)以上都是
- 3、引起故障的原因可能是_____。
- a)由硬件或软件错误导致的系统故障
- b)介质故障，如磁头碰撞
- c)应用程序中的软件错误，如正在处理数据库的程序发生的逻辑错误
- d)以上都是
- 4、数据库备份存储在安全的地方，通常是_____。
- a)放在不同的建筑物里　b)受到保护以免遭受火灾、盗窃和洪水等危险
- c)免受其他潜在的灾害　d)以上都是

选择题

- 5、下述_____是事务性质。
  - a)隔离性　b)持久性　C)原子性　d)以上都是

- 6、下述_____确保了事务的持久性。
  - a)应用程序员　b)并发控制　c)恢复管理　d)以上都是

- 7、下述_____确保了事务的原子性。
- a)应用程序员　　b)并发控制
- c)恢复管理　　d)以上都是

下面的事务日志表给出了发生系统故障时不同情况下的日志状态。请问系统会对这些情况分别采取什么样的恢复活动，为什么？指出恢复活动完成后给定属性的值？

表：　事务日志

情况 1	情况 2
$<T_1,BEGIN>$	$<T_1,BEGIN>$
$<T_1,A,500,395>$	$<T_1,A,500,395>$
$<T_1,B,800,950>$	$<T_2,BEGIN>$
	$<T_1,B,800,950>$
	$<T_2,C,320,419>$
	$< T_2,COMMIT>$
	$<T_1,COMMIT>$

Any Question ?

Thank you !!!