第十章 数据库恢复技术

1. 答:

事务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。

事务具有四个特性：原子性（Atomicity）、一致性（Consistency）、隔离性（Isolation）和持续性（Durability）。这个四个特性也简称为 ACID 特性。

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。

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。

隔离性：一个事务的执行不能被其他事务干扰。即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。

持续性：持续性也称永久性（Permanence），指一个事务一旦提交，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。

为了保证事务的原子性、一致性与持续性，DBMS 必须对事务故障、系统故障和介质故障进行恢复；为了保证事务的隔离性和一致性，DBMS 需要对并发操作进行控制。

2. 答:

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。如果数据库系统运行中发生故障，有些事务尚未完成就被迫中断，这些未完成事务对数据库所做的修改有一部分已写入物理数据库，这时数据库就处于一种不正确的状态，或者说是不一致的状态。

例如某工厂的库存管理系统中，要把数量为 Q 的某种零件从仓库 1 移到仓库 2 存放。则可以定义一个事务 T，T 包括两个操作；Q1=Q1-Q，Q2=Q2+Q。如果 T 非正常终止时只做了第一个操作，则数据库就处于不一致性状态，库存量无缘无故少了 Q。

3. 答:

因为计算机系统中硬件的故障、软件的错误、操作员的失误以及恶意的破坏是不可避免的，这些故障轻则造成运行事务非正常中断，影响数据库中数据的正确性，重则破坏数据库，使数据库中全部或部分数据丢失，因此必须要有恢复子系统。

恢复子系统的功能是：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。

4. 答：数据库系统中可能发生各种各样的故障，大致可以分以下几类:

（1） 事务内部的故障;

（2） 系统故障;

（3） 介质故障;

（4） 计算机病毒。

事务故障、系统故障和介质故障影响事务的正常执行；介质故障和计算机病毒破坏数据库数据。

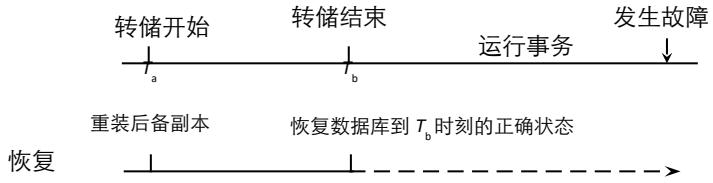
5．答:

数据转储和登录日志文件是数据库恢复的基本技术。

当系统运行过程中发生故障，利用转储的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。

6. 答:

数据转储是数据库恢复中采用的基本技术。所谓转储即 DBA 定期地将数据库复制到磁带或另一个磁盘上保存起来的过程。当数据库遭到破坏后可以将后备副本重新装入，将数据库恢复到转储时的状态。



静态转储：在系统中无运行事务时进行的转储操作。如上图所示。静态转储简单，但必须等待正运行的用户事务结束才能进行。同样，新的事务必须等待转储结束才能执行。显然，这会降低数据库的可用性。

动态转储：指转储期间允许对数据库进行存取或修改。动态转储可克服静态转储的缺点，它不用等待正在运行的用户事务结束，也不会影响新事务的运行。但是，转储结束时后援副本上的数据并不能保证正确有效。因为转储期间运行的事务可能修改了某些数据，使得后援副本上的数据不是数据库的一致版本。

为此，必须把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）。这样，后援副本加上日志文件就能得到数据库某一时刻的正确状态。

转储还可以分为**海量转储**和**增量转储**两种方式。

海量转储是指每次转储全部数据库。增量转储则指每次只转储上一次转储后更新过的数据。从恢复角度看，使用海量转储得到的后备副本进行恢复一般说来更简单些。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。

7. 答:

（1） 日志文件是用来记录事务对数据库的更新操作的文件。

（2） 设立日志文件的目的是： 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。

有关日志文件的具体作用，参考《概论》书上 10.4.2。

8. 答:

把对数据的修改写到数据库中和把表示这个修改的日志记录写到日志文件中是两个不同的操作。有可能在这两个操作之间发生故障，即这两个写操作只完成了一个。

如果先写了数据库修改，而在运行记录中没有登记这个修改，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，在恢复时只不过是多执行一次 UNDO 操作，并不会影响数据库的正确性。所以一定要先写日志文件，即首先把日志记录写到日志文件中，然后写数据库的修改。

9. 答:

事务故障的恢复:

事务故障的恢复是由 DBMS 自动完成的，对用户是透明的。

DBMS 执行恢复步骤是：

（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）继续反向扫描日志文件，做同样处理。

（4）如此处理下去，直至读到此事务的开始标记，该事务故障的恢复就完成了。

系统故障的恢复：

系统故障可能会造成数据库处于不一致状态：

一是未完成事务对数据库的更新可能已写入数据库；

二是已提交事务对数据库的更新可能还留在缓冲区，没来得及写入数据库。

因此恢复操作就是要撤销(UNDO)故障发生时未完成的事务，重做(REDO)已完成的事务。

系统的恢复步骤是：

（1）正向扫描日志文件，找出在故障发生前已经提交的事务队列（REDO 队列）和未完成的事务队列（UNDO 队列）。

（2）对撤销队列中的各个事务进行 UNDO 处理。

进行 UNDO 处理的方法是，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”（Before Image）写入数据库。

（3）对重做队列中的各个事务进行 REDO 处理。

进行 REDO 处理的方法是：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”（After Image）写入数据库。

解析：

在第（1）步中如何找出 REDO 队列和 UNDO 队列？请大家思考一下。

下面给出一个算法：

1） 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；

两个事务队列初始均为空。

2） 从日志文件头开始，正向扫描日志文件

如有新开始（遇到 Begin Transaction）的事务 T_i，把 T_i 暂时放入 UNDO-LIST 队列；

如有提交的事务（遇到 End Transaction）T_j，把 T_j 从 UNDO-LIST 队列移到 REDO-LIST 队列；

直到日志文件结束

介质故障的恢复：

介质故障是最严重的一种故障。

恢复方法是重装数据库，然后重做已完成的事务。具体过程是：

（1） DBA 装入最新的数据库后备副本（离故障发生时刻最近的转储副本），使数据库恢复到转储时的一致性状态。

（2） DBA 装入转储结束时刻的日志文件副本

（3） DBA 启动系统恢复命令，由 DBMS 完成恢复功能，即重做已完成的事务。

解析

1）我们假定采用的是静态转储，因此第（1）步装入数据库后备副本便可以了。

2）如果采用的是动态转储，第（1）步装入数据库后备副本还不够，还需同时装入转储开始时刻的日志文件副本，经过处理后才能得到正确的数据库后备副本。

3）第（3）步重做已完成的事务的算法是：

a. 正向扫描日志文件，找出故障发生前已提交的事务的标识，将其记入重做队列

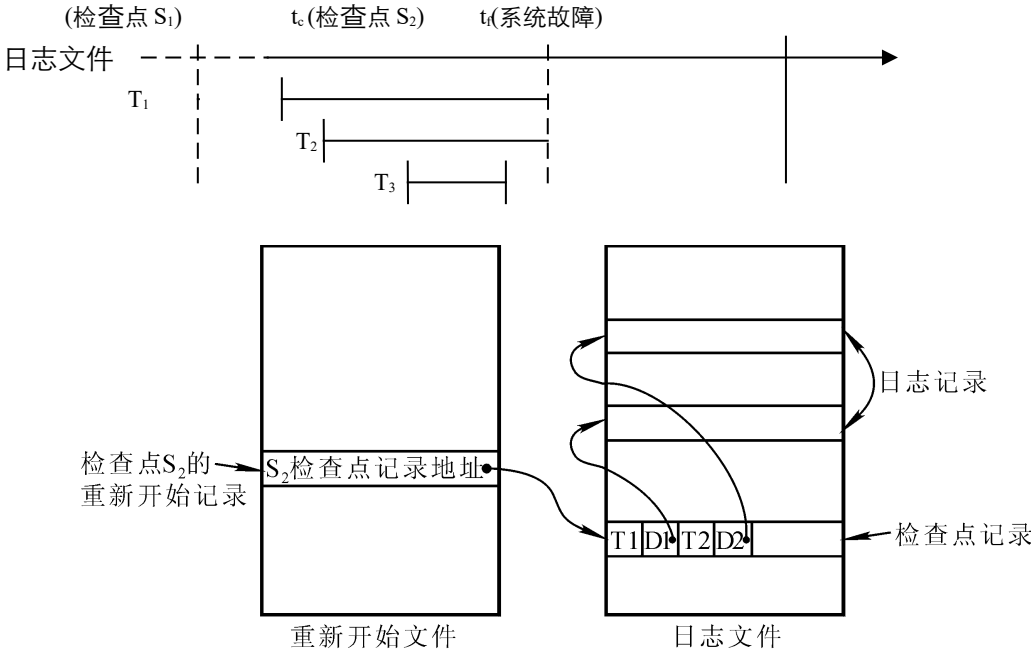
b. 再一次正向扫描日志文件，对重做队列中的所有事务进行重做处理。即将日志记录中“更新后的值”写入数据库。

10. 答：

检查点记录是一类新的日志纪录。它的内容包括：

（1）建立检查点时刻所有正在执行的事务清单，如图中的 T1、T2。

（2）这些事务的最近一个日志记录的地址，如图中的 D1、D2。



11. 答：

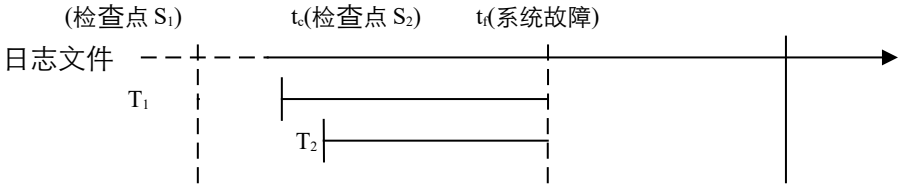
利用日志技术进行数据库恢复时，恢复子系统必须搜索日志，确定哪些事务需要 REDO，哪些事务需要 UNDO。一般来说，需要检查所有日志记录。这样做有两个问题：

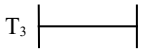
一是搜索整个日志将耗费大量的时间。

二是很多需要 REDO 处理的事务实际上已经将它们的更新操作结果写到数据库中了，恢复子系统又重新执行了这些操作，浪费了大量时间。

检查点技术就是为了解决这些问题。

例如：





在采用检查点技术之前，恢复时需要从头扫描日志文件，而利用检查点技术只需要从 T_c 开始扫描日志，这就缩短了扫描日志的时间。

事务 T_3 的更新操作实际上已经写到数据库中了，进行恢复时没有必要再 REDO 处理，采用检查点技术做到了这一点。

12. 答:

（1）在重新开始文件（见第 11 题的图）中找到最后一个检查点记录在日志文件中的地址，由该地址在日志文件中找到最后一个检查点记录。

（2）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。

这里建立两个事务队列:

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。

（3）从检查点开始正向扫描日志文件

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，直到日志文件结束;

（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。

13. 答:

数据库镜像即根据 DBA 的要求，自动把整个数据库或者其中的部分关键数据复制到另一个磁盘上。每当主数据库更新时，DBMS 自动把更新后的数据复制过去，即 DBMS 自动保证镜像数据与主数据的一致性。

数据库镜像的用途有:

一是用于数据库恢复。当出现介质故障时，可由镜像磁盘继续提供使用，同时 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。

二是提高数据库的可用性。在没有出现故障时，当一个用户对某个数据加排它锁进行修改时，其他用户可以读镜像数据库上的数据，而不必等待该用户释放锁。

14. 答:

下面简单介绍一下 Oracle 的恢复技术:

Oracle 中恢复机制也采用了转储和登记日志文件两个技术。

Oracle 向 DBA 提供了多种转储后备副本的方法，如文件拷贝、利用 Oracle 的 Export 实用程序、用 SQL 命令 Spool 以及自己编程实现等。相应地，Oracle 也提供了多种重装后备副本的方法，如文件拷贝、利用 Oracle 的 Import 实用程序、利用 SQL*LOADER 以及自己编程实现等。

在 Oracle 早期版本（V.5）中，日志文件以数据块为单位，也就是说，Oracle 的恢复操作是基于数据块的，不是基于操作的。Oracle 中记录数据库更新前的旧值的日志文件称为数据库前像文件（Before Image，简称 BI 文件），记录数据库更新后的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 文件是必须配置的，AI 文件是可以任选的。

Oracle7 为了能够在出现故障时更有效地恢复数据，也为了解决读“脏”数据问题，提供了 REDO 日志文件和回滚段(Rollback Segment)。REDO 日志文件中记录了被更新数据的前像和后像。回滚段记录更新数据的前像，设在数据库缓冲区中。在利用日志文件进行故障恢复时，为减少扫描日志文件的遍数，Oracle7 首先扫描 REDO 日志文件，重做所有操作，包括未正常提交的事务的操作，然后再根据回滚段中的数据，撤销未正常提交的事务的操作。

详细技术希望读者自己设法了解 Oracle 最新版本的介绍，例如通过 INTERNET 访问 Oracle 公司的网站。也可以了解其他 DBMS 厂商的产品情况。

*15. 解析

这是一个大作业。可以综合复习和运用学到的知识。读者可以参考《概论》书上 10.4.2、10.5、10.6，设计一个恢复子系统。

例如，日志文件的结构你可以记录为单位，也可以以数据块为单位。不同的日志文件结构，登记的日志内容，日志文件恢复事务的方法也就不同了。

对于研究生，还应该上机模拟实现你设计的恢复子系统。