第 21 讲:数据库崩溃恢复

15-445/645 数据库系统(2022 年秋季)

https://15445.courses.cs.cmu.edu/fall2022/

卡内基梅隆大学安迪·帕夫洛

1 事故恢复

DBMS 依赖于它的恢复算法来确保数据库一致性、事务原子性和故障时的持久性。每个恢复算法由两部分组成:

- 正常事务处理期间的操作,以确保 DBMS 可以从故障中恢复
- 将数据库恢复到确保事务原子性、一致性和持久性的状态失败后的操作。

利用**语义的**恢复和**隔离算法**(ARIES)是**IBM 研究院在 20 世纪 90 年代初为 DB2 系统开发的一种**恢复算法。

在 ARIES 恢复协议中有三个关键概念:

- **提前写日志:**在数据库更改写入磁盘之前,任何更改都将记录在稳定存储的日志中(窃取+不强制)。

重做期间重复历史:重新启动时,回溯操作并将数据库恢复到崩溃前的确切状态。

- **在撤销期间记录更改:**将撤销操作记录到日志中,以确保在重复失败的情况下不重复操作。

2 WAL 记录

预写日志记录扩展了 DBMS 的日志记录格式,以包含一个全局唯一的*日志序列号*(LSN)。图 1 显示了如何编写带有 LSN 的日志记录的高级图表。

所有日志记录都有一个 LSN。每当事务修改页面中的记录时,pageLSN 就会更新。每次 DBMS 将 WAL 缓冲区写到磁盘时,都会更新内存中的 flushedLSN。

系统中的各种组件跟踪与**它们相关的 lsn**。图 2 显示了这些 lsn 的表。

每个数据页都包含一个 pageLSN,这是该页最近更新的 LSN。DBMS 还跟踪到目前为止刷新的最大*LSN*(flushedLSN)。在 DBMS 将第 i 页写入磁盘之前,它必须至少刷新日志到 $\text{pageLSN}_i \leq \text{flushedLSN}$ 的点

3 正常执行

每个事务都会调用一系列的读写操作,然后是提交或中止。恢复算法必须具备的就是这一系列事件。

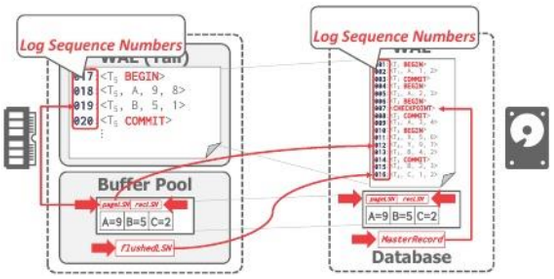

图 1:写入日志记录——每个 WAL 都有一个 lsn 计数器，每一步都增加一个。该页还保留一个 pageLSN 和一个 recLSN，后者存储使该页变脏的第一条日志记录。flushedLSN 是指向最后写入磁盘的 LSN 的指针。MasterRecord 指向最后一个成功通过的检查点。

Name	Where	Definition
flushedLSN	Memory	Last LSN in log on disk
pageLSN	$page_x$	Newest update to $page_x$
recLSN	$page_x$	Oldest update to $page_x$ since it was last flushed
lastLSN	T_i	Latest record of txn T_i
MasterRecord	Disk	LSN of latest checkpoint

图 2:LSN 类型——系统的不同部分也维护着存储相关信息的不同类型的 LSN。

事务提交

当事务提交时，DBMS 首先将 COMMIT 记录写入内存中的日志缓冲区。然后，DBMS 将所有日志记录(包括事务的 COMMIT 记录)刷新到磁盘。请注意，这些日志刷新是顺序的、同步的写到磁盘的。每个日志页面可以有多个日志记录。图 3 显示了事务提交的关系图。

一旦 COMMIT 记录被安全地存储在磁盘上，DBMS 就会向应用程序返回事务已提交的确认信息。稍后，DBMS 将向日志中写入一条特殊的 TXN-END 记录。这表明事务在系统中已经完全完成，不再有它的日志记录。这些 TXN-END 记录用于内部簿记，不需要立即刷新。

交易中止

中止一个事务是 ARIES 撤销操作仅适用于一个事务的一个特殊情况。

在日志记录中添加一个名为 prevLSN 的附加字段。这对应于事务的前一个 LSN。DBMS 使用这些 prevLSN 值为每个事务维护一个链表，这样可以更容易地遍历日志以查找其记录。

还引入了一种称为*补偿日志记录*(CLR)的新记录类型。CLR 描述

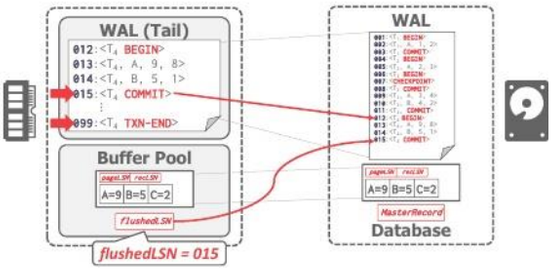

图 3:事务提交-在事务提交(015)之后，日志被刷出，flushedLSN 被修改为指向生成的最后一个日志记录。在稍后的某个点，写入事务结束消息，以在日志中表示此事务将不再出现。

为撤销前一个更新记录的操作而采取的操作。它有一个更新日志记录的所有字段加上 *undoNext* 指针(即，下一个要被撤销的 LSN)。DBMS 像添加任何其他记录一样将 clr 添加到日志中，但它们永远不需要被撤销。

要中止事务，DBMS 首先将 abort 记录附加到内存中的日志缓冲区中。然后，它按反向顺序撤销事务的更新，以从数据库中删除它们的影响。对于每个未完成的更新，DBMS 在日志中创建 **CLR** 条目并恢复旧值。在所有被终止的事务的更新都被逆转之后，DBMS 就会写一条 TXN-END 日志记录。图 4 显示了这一过程的示意图。

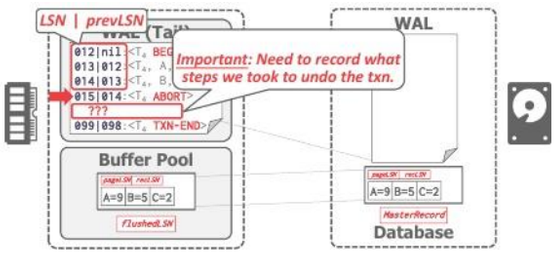

图 4:事务中止——DBMS 为事务创建的每个日志记录维护一个 LSN 和 prevLSN。当事务终止时，所有先前的更改都被逆转。在反向更改的日志条目保存到磁盘之后，DBMS 将 TXN-END 记录追加到中止事务的日志中。

4 检查点

DBMS 定期设置*检查点*，将缓冲池中的脏页写到磁盘上。这是用来减少恢复时重放日志的数量。

下面讨论的前两种阻塞检查点方法在检查点 pro 期间暂停事务

转运。这个暂停是必要的，以确保 DBMS 在检查点期间不会错过对页面的更新。然后，提供了一种更好的方法，该方法允许事务在检查点期间继续执行，但要求 DBMS 记录额外的信息，以确定它可能错过了哪些更新。

阻塞检查点

当 DBMS 采用检查点以确保将数据库的一致快照写入磁盘时，它会停止事务和查询的执行。这与上一讲中讨论的方法相同:

停止任何新事务的开始。

等待所有活动事务执行完毕。

将脏页刷写到磁盘。

更好地阻塞检查点

与之前的检查点方案类似，DBMS 不需要等待活动事务完成执行。DBMS 现在记录检查点开始时的内部系统状态。

停止任何新事务的开始。

•在 DBMS 执行检查点时暂停事务。

活动事务表(Active Transaction Table, ATT): ATT 表示 DBMS 中活动运行的事务的状态。事务的条目在 DBMS 完成事务的提交/中止过程后被删除。对于每个交易分录，ATT 包含以下信息:

- transactionId:唯一的交易标识符
- status:事务的当前“模式”(Running, commit, Undo Candidate)。
- lastLSN:最近由事务写入的 LSN

注意，ATT 包含没有 TXN-END 日志记录的每个事务。这包括正在提交或中止的两个事务。

脏页表(Dirty Page Table, DPT): DPT 包含缓冲池中被未提交事务修改的页的信息。每个脏页都有一个包含 recsn 的条目(即，首先导致该页变脏的日志记录的 LSN)。

DPT 包含缓冲池中所有脏的页面。这些更改是由正在运行的事务、提交的事务还是中止的事务引起的并不重要。

总的来说，ATT 和 DPT 通过 ARIES 恢复协议帮助 DBMS 恢复崩溃前的数据库状态。

模糊检查点

*模糊检查点*是 DBMS 允许其他事务继续运行的地方。这就是 ARIES 在其协议中使用的。

DBMS 使用额外的日志记录来跟踪检查点边界:

- < CHECKPOINT-BEGIN >:检查点的起始点。此时，DBMS 获取当前 ATT 和 DPT 的快照，它们在<CHECKPOINT-END>记录中被引用。
- < CHECKPOINT-END >:当检查点完成时。它包含 ATT + DPT，在写入<CHECKPOINT-BEGIN>日志记录时捕获。

5ARIES 恢复

ARIES 协议由三个阶段组成。在崩溃后启动时，DBMS 将执行以下阶段，如图 5 所示:

1. **分析**:读取 WAL 来识别缓冲池中的脏页和崩溃时的活动事务。在分析阶段结束时，*ATT* 告诉 DBMS 哪些事务在崩溃时处于活动状态。*DPT* 告诉 DBMS 哪些脏页可能没有保存到磁盘上。
2. **重做**:从日志中的适当点开始重复所有操作。
3. **撤销**:对崩溃前未提交的事务进行反向操作。

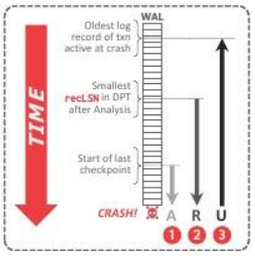

图 5:ARIES 恢复:DBMS 通过检查从通过主记录找到的最后一个 BEGIN-CHECKPOINT 开始的日志来启动恢复过程。然后开始分析阶段，通过时间向前扫描来构建 ATT 和 DPT。在重做阶段，算法跳转到最小的 recLSN，这是可能修改了未写入磁盘的页面的最老的日志记录。然后，DBMS 应用最小 recLSN 的所有更改。撤销阶段从崩溃时活动的事务的最旧日志记录开始，并撤销到该点为止的所有更改。

分析阶段

从通过数据库的主记录 *LSN* 找到的最后一个检查点开始。

1.从检查点扫描日志转发。

2.如果 DBMS 发现 TXN-END 记录，则从 ATT 中删除其事务。

3. 所有其他记录，将事务以 **UNDO** 状态添加到 ATT，并在提交时将事务状态更改为 **commit**。
4. 对于 UPDATE 日志记录，如果页 P 不在 DPT 中，则将 P 添加到 DPT 中，并将 P 的 recLSN 设置为日志记录的 *LSN*。

重做阶段

这个阶段的目标是让 DBMS 重复历史，以重建其直到崩溃时刻的状态。它将重新应用所有更新(甚至是中止的事务)并重做 **clr**。

DBMS 从 DPT 中包含最小 recLSN 的日志记录向前扫描。对于每个具有给定 *LSN* 的更新日志记录或 CLR, DBMS 将重新应用更新，除非:

- 受影响的页面不在 DPT 中，或者
- 受影响的页在 DPT 中，但该记录的 *LSN* 小于 DPT 中该页的 recLSN，或者
- 受影响的 pageLSN(磁盘上)≥*LSN*。

要重做一个操作，DBMS 重新应用日志记录中的更改，然后将受影响页面的 *pageLSN* 设置为该日志记录的 *LSN*。

在重做阶段结束时，为状态为 COMMIT 的所有事务写入 TXN-END 日志记录，并将它们从 ATT 中删除。

撤销阶段

在最后一个阶段，DBMS 反转在崩溃时处于活动状态的所有事务。这些都是分析阶段之后 ATT 中具有 UNDO 状态的事务。

DBMS 以反向 *LSN* 顺序处理事务，使用最后的 LSN 来加快遍历。当它反转事务的更新时，DBMS 会为每次修改写入一个 CLR 条目到日志中。

一旦成功终止了最后一个事务，DBMS 就会清空日志，然后准备开始处理新的事务。