2023-2024第二学期第二次期中考查（共4题，总计32分）

1. 画出 DBMS 缓冲池的结构示意图，并结合缓冲池的内部结构说明一般情况下执行引擎访问数据页面过程。说明其中什么情况下缓冲池将什么样的页面换出缓冲池？（8分）

答：访问过程：（1）执行引擎向缓冲池管理器发出某个页面p的访问请求；

（2）缓冲池管理器查找页表判断若p在其中，则增加页表中p的引用计数，依据页表中记录的p在帧数组的位置访问p，访问对应的事务不再需要页面p后，减少p的引用计数；

（3）若缓冲池管理器发现p不在页表中，则将p加入页表，然后考虑将页面p（数据块p）从数据文件中读入缓冲池的帧数组。此时若帧数组有空闲空间则直接读入p到某个空闲帧；若帧数组已满，则采用缓存替换策略选择某个引用计数为0的帧页面替换。若被替换页的脏标记显示其为脏页，则结合先写日志协议完成被替换页的写出到数据文件的操作。

当执行引擎要读取的页需要新进入帧数组而导致其中的某个脏页要被替换出缓冲池时，会触发写操作。

2**假设某企业实施指纹考勤，需要将用户指纹信息上传至指纹考情机。为此在考勤数据库中设计三张表，分别是：**

**（1）UserInfo(UserID,UserName,FingerTemplate)**

**用户信息表（用户号，用户名，用户指纹模板）；**

**（2）FingerDevice(DeviceID，DeviceLoc)**

**指纹考情机表（考勤机号，考情机所在位置）；**

**(3) FingTplUploadInfo(UserID,DeviceID,UploadState)**

**指纹上传信息表（用户号，考勤机号，上传状态）；其中UploadState只取四个状态（-1：上传失败，0：没有上传，1：正在上传，2：上传成功）。**

**用户需要查询查询已经上传指纹信息的用户信息，指纹信息及其上传设备的设备位置信息。下述SQL查询语句可以完成该查询。**

**SELECT UserName**

**FROM FingerDevice,FingTplUploadInfo,UserInfo**

**WHERE FingerDevice.DeviceID = FingTplUploadInfo.DeviceID**

**AND UserInfo.UserID = FingTplUploadInfo.UserID**

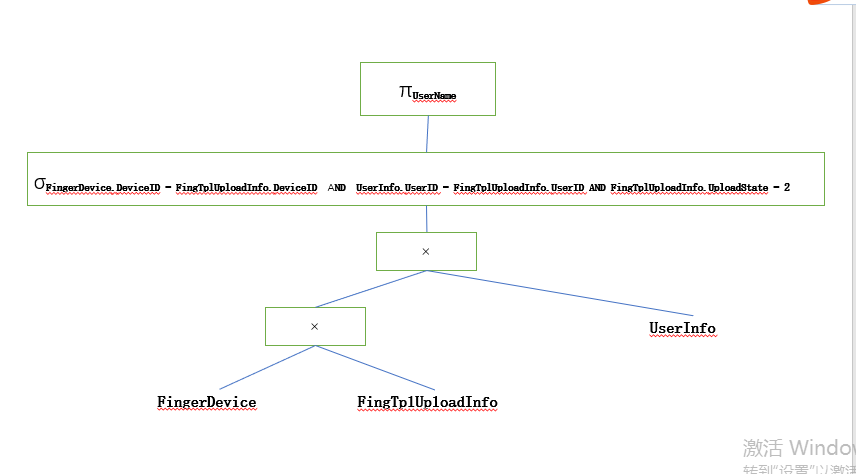
**AND FingTplUploadInfo.UploadState = 2**

**完成下列两小题：（8分）**

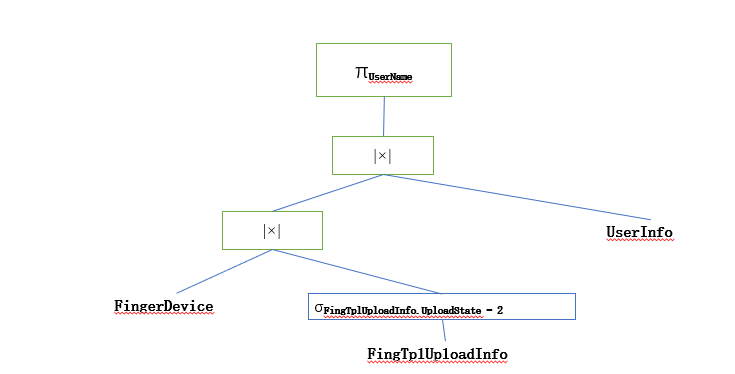
**（1）给出该查询的语法树；**

**（2）对该查询语法树进行代数优化，并给出优化后的语法树。**

答：(1)



（2）优化后(多解)



3 **设数据库中初始值为X=1，有事务T1、T2分别如下：**

**T1: X=X\*3；**

**T2: X=X+6。请完成下列小题（共8分））**

**（1）若这两个事务允许并发执行，则有多少种可能的正确结果？。请一一列举出来；**

**（2）用表格形式写出两个事务并发执行出现丢失修改的一个调度序列；**

**（3）用表格形式写出两事务并发交替执行且满足两段锁协议，但是死锁的一个调度序列；**

**（4）用表格形式写出两事务并发交替执行且满足两段锁协议的一个成功调度序列。**

(1)答：串行执行（也就是正确的执行结果）只有T1T2和T2T1，

T1T2：T1：x=1\*3=3 T2:X=3+6=9,最后的结果X=9

T2T1：T2：x=1+6=7 T2:X=3\*7=21,最后的结果X=21

（2）(注意有多解)

|  |  |
| --- | --- |
| T1 | T2 |
| SLOCK（X） |  |
| R（X）=1 |  |
| UNLOCK（X） |  |
|  | SLOCK（X） |
|  | R（X）=1 |
|  | UNLOCK（X） |
| XLOCK（X） |  |
| X←1\*3  W(X)=3 |  |
| UNLOCK（X） |  |
|  | XLOCK（X） |
|  | X←1+6  W(X)=7 |
|  | UNLOCK（X） |

（3）死锁有多解

|  |  |
| --- | --- |
| T1 | T2 |
| SLOCK（X） |  |
| R（X）=1 |  |
|  | SLOCK（X） |
|  | R（X）=1 |
| XLOCK（X） |  |
| 等待 | XLOCK（X） |
|  | 等待 |

（4）（两段锁且成功）

|  |  |
| --- | --- |
| T1 | T2 |
| SLOCK（X） |  |
| R（X）=1 |  |
| XLOCK（X） |  |
|  | SLOCK（X） |
|  | 等待 |
| X←1\*3  W(X)=3 |  |
| UNLOCK（X） |  |
| UNLOCK（X） |  |
|  | 得到SLOCK（X） |
|  | R（X）=3 |
|  | XLOCK（X） |
|  | X←3+6  W(X)=9 |
|  | UNLOCK（X） |
|  | UNLOCK（X） |

4 **假设数据库发生系统故障时有如下所示的日志文件，且A、B、C的初始值均为1，请结合日志内容简述故障恢复过程，要求写出恢复REDO、UNDO队列以及A、B、C值的恢复过程。（8分）**

**重新开始文件**

|  |  |
| --- | --- |
| **检查点1** | **日志序号9** |
|  |  |

**日志文件**

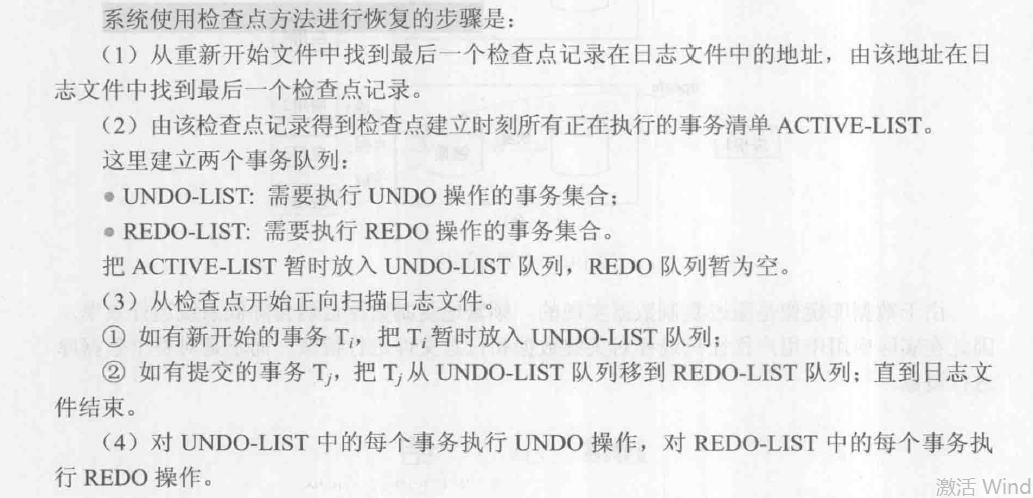
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **日志序号** | **日志记录** |  | **日志序号** | **日志记录** |
| **1** | **T1:事务开始** |  | **9** | **检查点1** |
| **2** | **T1:写A，A=3** |  | **10** | **T2:事务提交** |
| **3** | **T2:事务开始** |  | **11** | **T4:事务开始** |
| **4** | **T2:写B，B=5** |  | **12** | **T4：写B，B=11** |
| **5** | **T1:提交** |  | **13** | **T5:事务开始** |
| **6** | **T2:写B，B=7** |  | **14** | **T5:写C，C=4** |
| **7** | **T3：事务开始** |  | **15** | **T4:写B，B=13** |
| **8** | **T3：写A，A=9** |  | **16** | **T5：事务提交** |

**（1）写出检查点1记载的活动事务集合；**

（2）当重新开始文件中记录了检查点1的位置，且日志记录10还未写入日志日志文件是，请分析数据库文件中A,B，C的取值情况，并简述理由；

（3）简述该系统故障的恢复过程，写出最终开始执行REDO和UNDO操作时两个队列中的事务集合。

答：



（1）根据重新开始文件，找到最后一个检查点记录在日志文件，由该地址在日志文件中找到最后一个检查点记录。

得到日志序号9，检查点1

（2）首先建立活动事务清单，Active-LIST={T2,T3},其中T1在检查点之前提交，T4，T5在检查点之后开始，因此他们都不在活动事务列表中

紧接着创建UNDO-LIST={}.REDO-LIST={}

然后将Active-LIST放到UNDO-LIST进行初始化，得到

UNDO-LIST={T2,T3}，而REDO-LIST={}

(3)从检查点（也就是时序9）开始，扫描日志

①发现新开始的事务T4，T5，于是UNDO-LIST={T2,T3,T4,T5}

②发现T2，T5提交，于是将T2，T5从UNDO-LIST移入REDO-LIST。

最后得到UNDO-LIST={T3,T4}，REDO-LIST= { T2，T5}

（4）现在可以开始恢复，先UNDO，再REDO

UNDO-LIST={T3,T4}

反向扫描日志，15,12，8,分别属于T4，T4，T3，依次对这些操作进行UNDO，

对15UNDO,C由4变为1，对12UNDO，B由11变成7，对8进行UNDO，A由9变成3，

REDO-LIST= { T2，T5}

正向扫描日志，4,6，14,分别属于T4，T4，T3，依次对这些操作进行REDO，对4REDO，B变成5，6REDO，B变成7,14REDO，C变成4

最后的结果：A=3 B=7 C=4

对于第（2）题，在时刻9，检查点将缓冲区中所有的数据写入磁盘。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **日志序号** | **日志记录** |  | **日志序号** | **日志记录** |
| **1** | **T1:事务开始** |  | **9** | **检查点1** |
| **2** | **T1:写A，A=3（1→3）** |  | **10** | **T2:事务提交** |
| **3** | **T2:事务开始** |  | **11** | **T4:事务开始** |
| **4** | **T2:写B，B=5（1→5）** |  | **12** | **T4：写B，B=11（7→11）** |
| **5** | **T1:提交** |  | **13** | **T5:事务开始** |
| **6** | **T2:写B，B=7（5→7）** |  | **14** | **T5:写C，C=4（1→4）** |
| **7** | **T3：事务开始** |  | **15** | **T4:写B，B=13** |
| **8** | **T3：写A，A=9（3→9）** |  | **16** | **T5：事务提交** |

所以此时A=9,B=7,C=1