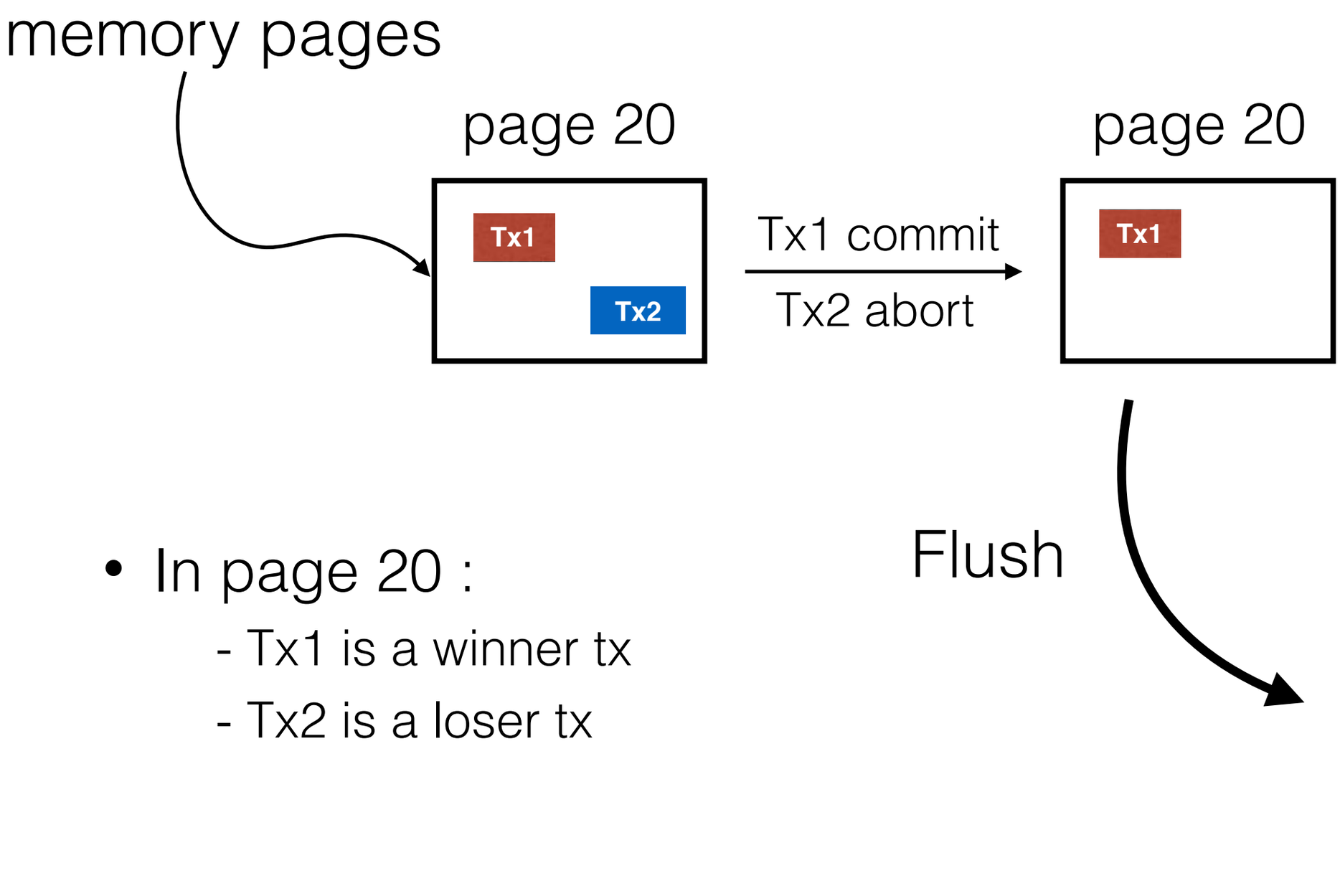
Overview of ARIES

ctsai@DataDB

Cloud Database

2017 Spring

Winner ? Loser ?



- In page 20 :
  - Tx1 is a winner tx
  - Tx2 is a loser tx

Why ARIES ?

- Steal

  - Due to buffer management, dirty pages may be flushed to disk before txs commit

  -> The changes made by ***loser*** txs must be UNDO

Why ARIES ?

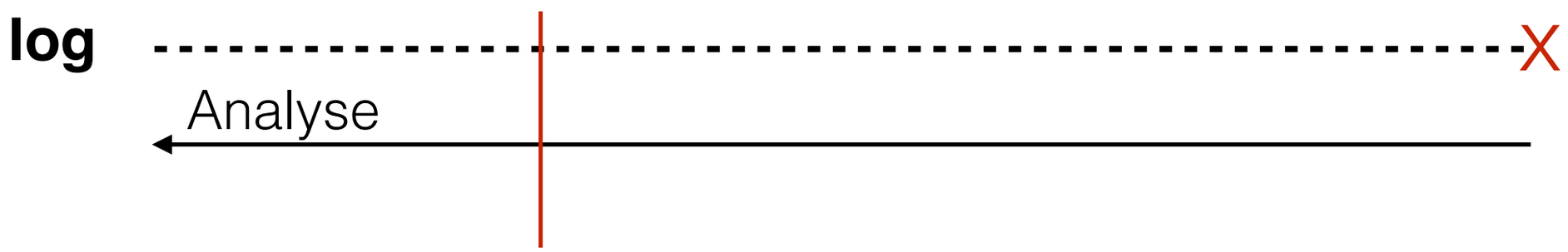
- Steal
- No Force

  - Due to performance reason, dirty page won't be flush immediately after txs commit

  -> The changes made by ***winner*** txs must be REDO

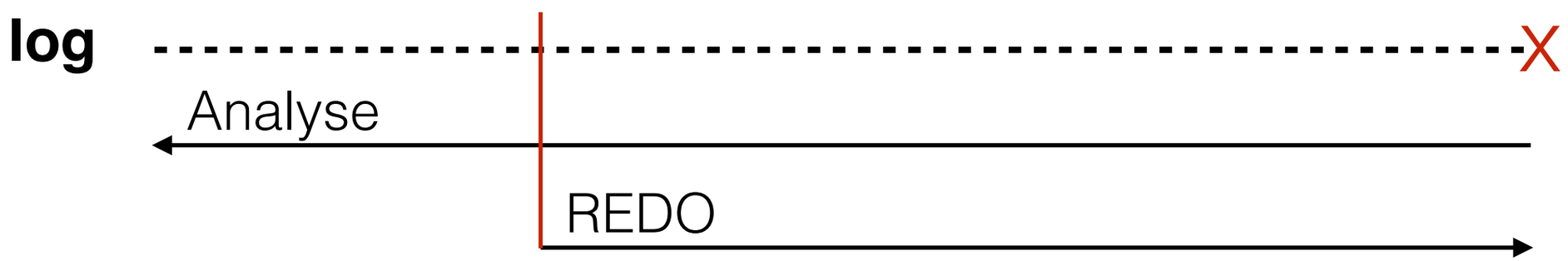
Three Phases of ARIES

- Analyse Phase
  - Find the earliest possibly start point of dirty page
  - Find loser txs



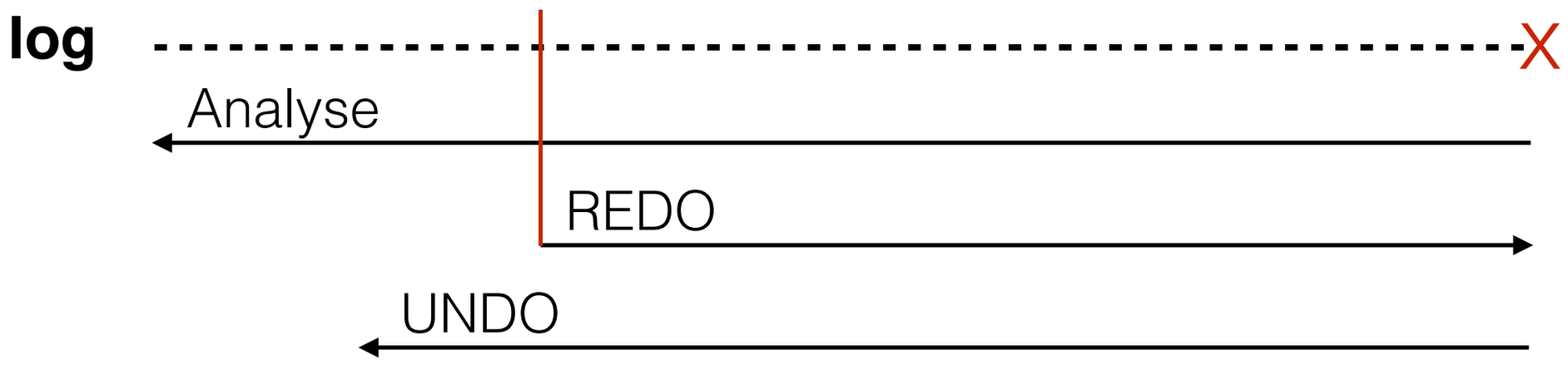
Three Phases of ARIES

- Analyse Phase
  - Find the earliest possibly start point of dirty page
  - Find loser txs
- REDO Phase
  - Repeat history (***both*** winner and loser changes)
  - Recovery exact page status when the failure occurred



Three Phases of ARIES

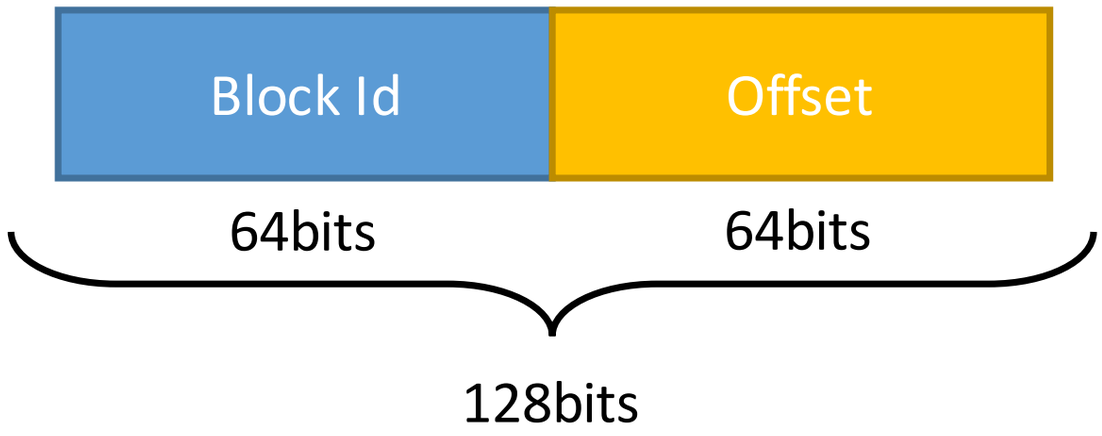
- Analyse Phase
  - Find the earliest possibly start point of dirty page
  - Find loser txs
- REDO Phase
  - Repeat history (***both*** winner and loser changes)
  - Recovery exact page status when the failure occurred
- UNDO Phase
  - Rollback ***loser*** txs changes

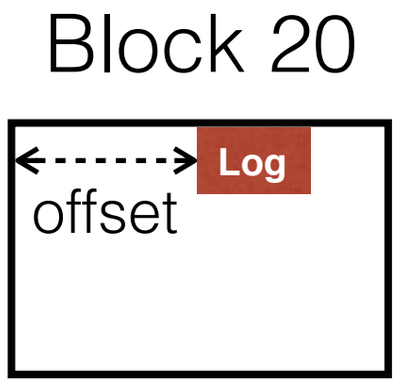


Logs in ARIES

Log Sequence Number

- Format





- Class
  - LogSeqNum – Store Block , offset

Physical Log Record

- Record format :
  - Set Value Record:

    <Op Code, txNum, fileName, blockNum, offset, sqlType, oldVal, newVal >
  - Index Page Insert/Delete Record :

    <Op Code, txNum, fileName, blockNum, insertSlot, insertKey, insertRidBlkNum, insertRidId>
- REDO :
- UNDO :

Physical Log Record

- Record format :
  - Set Value Record:

    <Op Code, txNum, fileName, blockNum, offset, sqlType, oldVal, newVal >
  - Index Page Insert/Delete Record :

    <Op Code, txNum, fileName, blockNum, insertSlot, insertKey, insertRidBlkNum, insertRidId>
- REDO :
  - Apply newVal to the page
- UNDO :

Physical Log Record

- Record format :
  - Set Value Record:

    <Op Code, txNum, fileName, blockNum, offset, sqlType, oldVal, newVal >
  - Index Page Insert/Delete Record :

    <Op Code, txNum, fileName, blockNum, insertSlot, insertKey, insertRidBlkNum, insertRidId>
- REDO :
  - Apply newVal to the page
- UNDO :
  - Apply oldVal to the page

Physical Log Record

- Record format :
  - Set Value Record:

    <Op Code, txNum, fileName, blockNum, offset, sqlType, oldVal, newVal >
  - Index Page Insert/Delete Record :

    <Op Code, txNum, fileName, blockNum, insertSlot, insertKey, insertRidBlkNum, insertRidId>
- REDO :
  - Apply newVal to the page
- UNDO :
  - Apply oldVal to the page
  - Append its ***Compensation Log***

Compensation Log Record

- Feature :
  - REDO-ONLY Physical Log Record
  - UNDO procedure's REDO Log
- Record format :
  - Set Value Clr and Index Page Insert/Delete Clr:

    <Op Code, UndoTxNum… , UndoNextLSN >
- REDO :
  - Apply oldVal to the page
- UNDO :
  - Do nothing

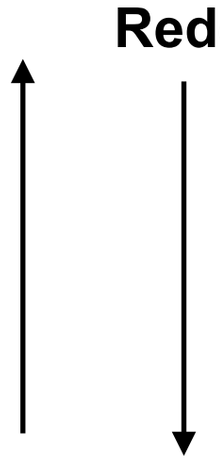
Why Clr is Redo Only ?

[0] <Start 1>

[1] <SetVal , 1 , Page 20 , 0 , 1>

[2] <SetVal , 1 , Page 20 , 1 , 2>

[3] <SetVal , 1 , Page 20 , 2 , 3>



Crash Here !

Analyse tx1 is a loser tx

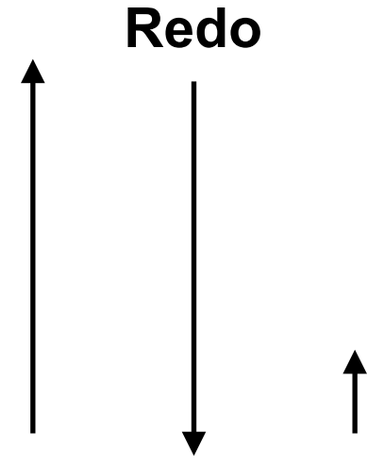
Why Clr is Redo Only ?

[0] <Start 1>

[1] <SetVal , 1 , Page 20 , 0 , 1>

[2] <SetVal , 1 , Page 20 , 1 , 2>

[3] <SetVal , 1 , Page 20 , 2 , 3>



Crash Here ! **Analyse** **Undo**

[4]<SetValClr 1, Page 20 , 3 , 2 > // Append Undo [3] Redo log

Why Clr is Redo Only ?

[0] <Start 1>

[1] <SetVal , 1 , Page 20 , 0 , 1>

[2] <SetVal , 1 , Page 20 , 1 , 2>

[3] <SetVal , 1 , Page 20 , 2 , 3>

Crash Here !

[4]<SetValClr 1, Page 20 , 3 , 2 >

Crash Again !

Why Clr is Redo Only ?

[0] <Start 1>

[1] <SetVal , 1 , Page 20 , 0 , 1>

[2] <SetVal , 1 , Page 20 , 1 , 2>

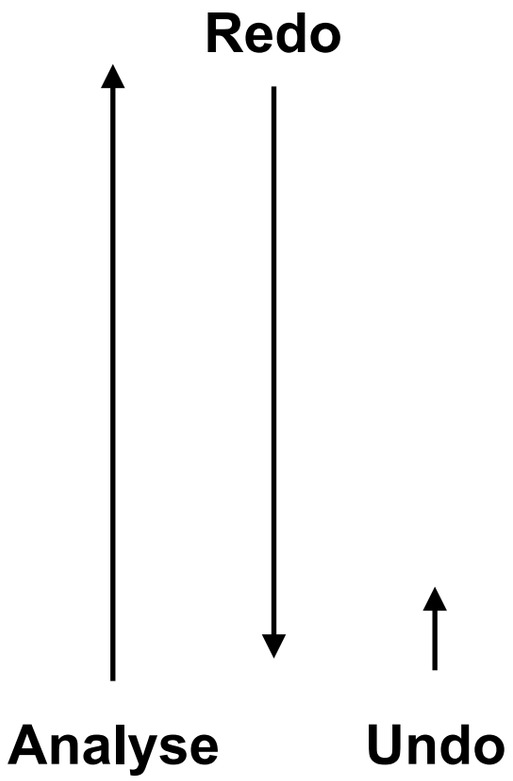
[3] <SetVal , 1 , Page 20 , 2 , 3>

Crash Here !

[4]<SetValClr 1, Page 20 , 3 , 2 >

Crash Again !

[5]<SetValClr 1, Page 20 , 2 , 3 >



// Append Undo { Undo [3] Redo log } Redo log

Why Clr is Redo Only ?

[0] <Start 1>

[1] <SetVal , 1 , Page 20 , 0 , 1>

[2] <SetVal , 1 , Page 20 , 1 , 2>

[3] <SetVal , 1 , Page 20 , 2 , 3>

Crash Here !

[4]<SetValClr 1, Page 20 , 3 , 2 >

Crash Again !

[5]<SetValClr 1, Page 20 , 2 , 3 >

Crash Again !

Why Clr is Redo Only ?

[0] <Start 1>

[1] <SetVal , 1 , Page 20 , 0 , 1>

[2] <SetVal , 1 , Page 20 , 1 , 2>

[3] <SetVal , 1 , Page 20 , 2 , 3>

Crash Here !

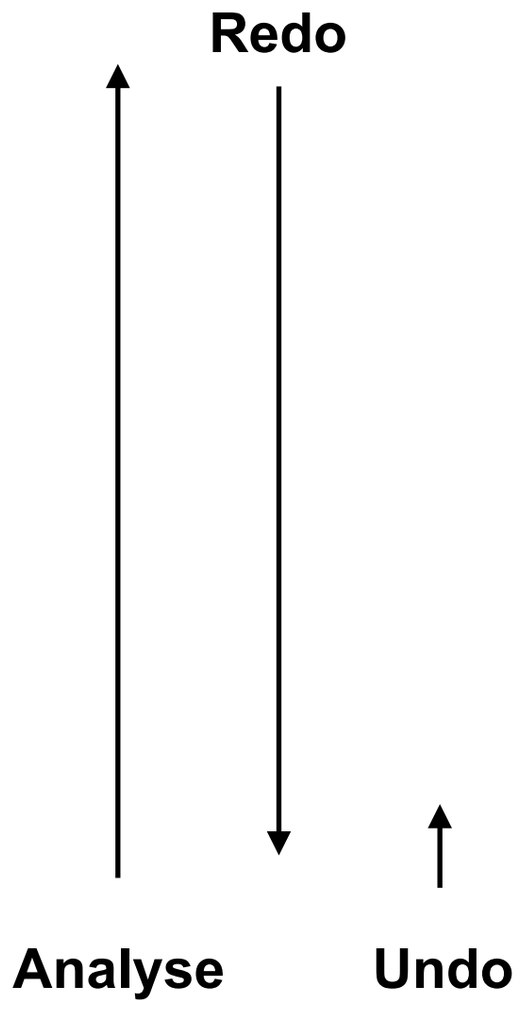
[4]<SetValClr 1, Page 20 , 3 , 2 >

Crash Again !

[5]<SetValClr 1, Page 20 , 2 , 3 >

Crash Again !

[6]<SetValClr 1, Page 20 , 3 , 2 >



// Append Undo { Undo { Undo [3] Redo log } Redo log} Redo log

… WTF

Which level of REDO am I Undoing ?

Why Clr need UndoNext ?

- Clr without UndoNext :

[0] <Start 1>

[1] <SetVal , 1 , Page 20 , 0 , 1>

[2] <SetVal , 1 , Page 20 , 1 , 2>

[3] <SetVal , 1 , Page 20 , 2 , 3>

Crash Here !

[4]<SetValClr 1, Page 20 , 3 , 2 > // Append Undo [3] Redo log

Crash Again !

[5]<SetValClr 1, Page 20 , 3 , 2 > // Append Undo [3] Redo log

[6]<SetValClr 1, Page 20 , 2 , 1 > // Append Undo [2] Redo log

[7]<SetValClr 1, Page 20 , 1 , 0 >

Why Clr need UndoNext ?

- Clr with UndoNext :

  [0] <Start 1>

  [1] <SetVal , 1 , Page 20 , 0 , 1>

  [2] <SetVal , 1 , Page 20 , 1 , 2>

  [3] <SetVal , 1 , Page 20 , 2 , 3>

  Crash Here !

  [4]<SetValClr 1, Page 20 , 3 , 2 , [3] > // Append Undo [3] Redo log

  Crash Again !

  [5]<SetValClr 1, Page 20 , 2 , 1 , [2] > // Append Undo [2] Redo log

  [6]<SetValClr 1, Page 20 , 1 , 0 , [1] >

Why Clr need UndoNext ?

- UndoNext helps skip logs which have been Undone by Redo

[0] <Start 1>

[1] <SetVal , 1 , Page 20 , 0 , 1>

[2] <SetVal , 1 , Page 20 , 1 , 2>

[3] <SetVal , 1 , Page 20 , 2 , 3>

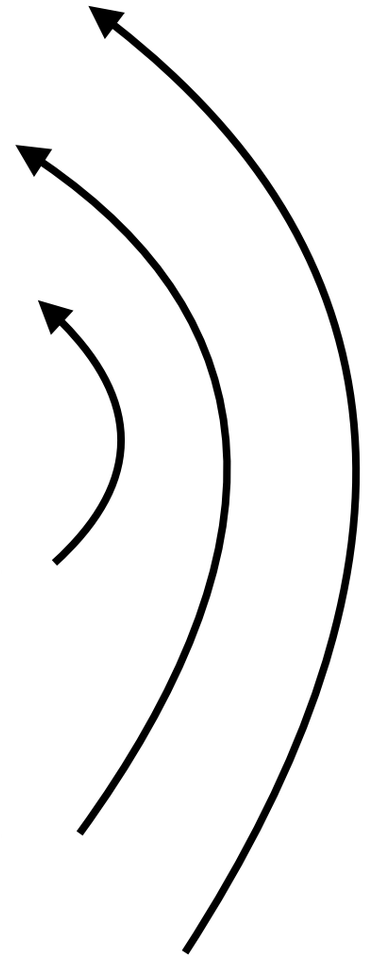
Crash Here !

[4]<SetValClr 1, Page 20 , 3 , 2 , [3] >

Crash Again !

[5]<SetValClr 1, Page 20 , 2 , 1 , [2] >

[7]<SetValClr 1, Page 20 , 1 , 0 , [1] >



Logical Log Record

- Record format :

  - Logical - Start Record

    <OP Code, txNum>

  - Record File Insert/Delete End Record :

    <Op Code, txNum,fileName, blockNum, slotId , logicalStartLSN>

  - Index Insert/Delete End Record :

    <Op Code, txNum, tblName, fldName, searchKey, recordBlockNum, recordSlotId , logicalStartLSN>

- REDO :

  - Do nothing

- UNDO :

  - Undo ***competed*** logical log ***logically***

  - Undo ***partial*** logical log ***physically***

  - Append ***Logical Abort*** log record

Rollback a competed Logical Log Record

[0] <Start 1>

[1] <LogicalStart, 1 >

[2] <Index Page Insert , 1 , … >

[3] <SetVal , 1 , Page 2 , 1 , 2>

[4] <SetVal , 1 , Page 20 , 2 , 3>

[5] <Record File Insert End , 1, … , [2] >

Crash Here !

Rollback a competed Logical Log Record

[0] <Start 1>

[1] <LogicalStart, 1 >

[2] <Index Page Insert , 1 , … >

[3] <SetVal , 1 , Page 2 , 1 , 2>

[4] <SetVal , 1 , Page 20 , 2 , 3>

[5] <Record File Insert End , 1, … , [2] >

Logical operations

Crash Here !

Rollback a competed Logical Log Record

[0] <Start 1>

[1] <LogicalStart, 1 >

[2] <Index Page Insert , 1 , … >

[3] <SetVal , 1 , Page 2 , 1 , 2>

[4] <SetVal , 1 , Page 20 , 2 , 3>

Physical operations

Logical operations

[5] <Record File Insert End , 1, … , [2] >

Crash Here !

Rollback a competed Logical Log Record

[0] <Start 1>

[1] <LogicalStart, 1 >

[2] <Index Page **Insert** , 1 , … >

[3] <SetVal , 1 , Page 2 , 1 , 2>

[4] <SetVal , 1 , Page 20 , 2 , 3>

[5] <Record File Insert End , 1, … , [2] >

Crash Here !

[6] <Start 2 >

[7] <LogicalStart, 2 >

[8] <Index Page **Delete** , 2 , … >

[9] <SetVal , 2 , Page 2 , 2 , 1>

[10] <SetVal , 2 , Page 20 , 2 , 3>

[11] <Record File Insert End , 2, … , [7]>

[12] <Logical Abort 1 , [1]>

Rollback a competed Logical Log Record

[0] <Start 1>

[1] <LogicalStart, 1 >

[2] <Index Page **Insert** , 1 , … >

[3] <SetVal , 1 , Page 2 , 1 , 2>

[4] <SetVal , 1 , Page 20 , 2 , 3>

[5] <Record File Insert End , 1, … , [2] >

Crash Here !

[6] <Start 2 >

[7] <LogicalStart, 2 >

[8] <Index Page **Delete** , 2 , … >

[9] <SetVal , 2 , Page 2 , 2 , 1>

[10] <SetVal , 2 , Page 20 , 2 , 3>

Physical operations

Logical operations

[11] <Record File Insert End , 2, … , [7]>

[12] <Logical Abort 1 , [1]>