

---

---

# Crash Recovery

18 April 2010

Lecture 5

---

# Topics for Today

---

- Overview of Crash Recovery
  - What is the log?
  - Steal/Force
  - What is ARIES?
- Source: R&G 16.7

# Crash Recovery

---

- We will talk about the recovery manager in the DBMS
- Its job is to take over in case of
  - System Crashes
  - Media Failure
  - Transaction failure or aborts
- We'll outline ARIES
  - Fairly simple, getting popular (according to the authors)
- Who uses ARIES?
  - IBM DB2
  - Informix, MS SQL Server, Oracle 8, Sybase ASE all use Write-Ahead Logging
  - They are similar to ARIES

# Recovery Manager

---

- The DBMS Recovery Manager is responsible for crashes and aborted transactions
  - It helps ensure the Atomicity and Durability
  - It must be resilient against crashes and media failures
- The Recovery Manager runs when:
  - A transaction aborts and must be undone
  - The database experiences a failure (power, media, etc)
- Its job is to:
  - Find recent transactions which committed → make sure they are saved
  - Everyone else → make sure they are not saved

# The WAL

---

- Every action is written to the log before it's performed – **Write Ahead Log (WAL)**
- The log stores the following information:
  - Transaction ID
  - Table ID, Record ID
  - Previous Record Data
  - New Record Data
- Also notes every commit or abort
- Assumed to be written to **stable storage**
  - Very low likelihood of failure
  - May be RAID (that's the DBA's job)

# Crash Recovery Model

---

---

- We already have a log – what's so hard?
- 
- Let's also assume that the database server has a hard disk and memory and uses a standard Virtual Memory Paging system
    - Files are divided into pages on disk
    - Pages are moved in and out of memory
  - Assume that writing a page of memory to disk is an atomic action
  - Two questions
    - **Stealing Frames**: Can a dirty page in memory be forced to disk before it's committed?
    - **Force Pages**: Does a transaction force all of its dirty pages to disk after it commits?

If the answers are Yes and No, the files on the disk may be wrong –  
having some bad updates and missing some good ones

# Why Steal and Not Force?

---

- Why Steal?
  - Memory is a limited resource
  - Expecting all dirty pages to sit in memory may require huge amounts of it
  - Allowing the Virtual Memory paging to work normally is a big boost
  - The result: **The disk files may contain writes from uncommitted transactions.**

---
- Why not Force?
  - Forcing dirty pages to disk would raise the burden of I/O
  - Making all COMMITs involve writing all of the transaction's dirty pages to disk would take a long time
  - The result: **The disk files may be missing some writes from committed transactions.**

# Overview of ARIES

---

- ARIES is a recovery algorithm for a steal/no force DBMS
- Major steps
  - Analyze the log to discover all transactions.
  - Redo all actions in the log up to the moment the database crashed
  - Undo the actions of transactions that didn't commit
- End result: The database contains only the writes of previously committed actions
- For abort, similar steps to ARIES, but simpler
- Recovery is complicated:
  - What if it fails during recovery?
  - More on this in Chapter 18

# Watching your back

---

- To reduce the risk of expensive recoveries, the recovery manager **forces pages to disk every so often**
  - The less outstanding dirty page there are, the easier it is for the algorithm
- **Checkpoints**: writing information about outstanding transactions and dirty pages to disk every so often
- More on this in 18.5

# Conclusion

---

---

- Overview of Crash Recovery
  - What is the log?
  - Steal/Force
  - What is ARIES?