CS 347 Review
Date: 4/14/2016
Topics

● Concurrency Control

● Failure Recovery
ACID

- Atomicity
- Consistency
- Isolation
- Durability
Concurrency Control

- Why do we need concurrency control?
  - Consistency
  - Isolation

- What’s the solution?
  - Serializable Schedules
It’s All About Locking

1. Well-formed Transactions
   a. Transaction must lock resource before acting on it

2. Legal Scheduler
   a. Only one transaction may hold lock on resource

3. 2 Phase Locking (2PL)
   a. All locks held until end of transaction
Failure Recovery

- Failure in a Consolidated System
  - System crash, memory loss, CPU resets
  - Disk data lost
Undo Logging

1. For every action generate undo log record (containing old value)

2. Before x is modified on disk, log records pertaining to x must be on disk

3. Before commit is flushed to log, all writes of transaction must be reflected on disk
Recovery Under Undo Logging

1. Let $S$ = set of transactions with $<Ti$ start$>$ in log, but no $<Ti$ commit$>$ (or $<Ti$ abort$>$) record in log

2. For each $<Ti, X, v>$ in log, in reverse order (latest to earliest) do:
   a. if Ti is in S then:
      i. write $(X, v)$
      ii. output $(X)$

3. For each Ti in S do:
   a. Write $<Ti, abort>$ to log
Other Variants

- Redo Logging
- Undo/Redo Logging
- Checkpoints
Questions?