Recitation 18 — Databases

Parts of this paper recap definitions from lecture — things like atomicity, isolation, serializability, etc. These notes focus on the areas not covered in lecture

Basics of Database Management Systems (DBMSes)

- ACID = Atomicity, Concurrency, Isolation, Durability.
- Write-ahead logging: typically used to back a transactional database. Like what you've seen in lecture.
- Recovery: The recovery process for WAL is different depending on a few choices the DBMS makes:
    - STEAL vs. NO-STEAL: whether an uncommitted write can overwrite the most recent committed value of a data item on non-volatile storage (e.g., cell storage).
    - FORCE vs. NO-FORCE: whether all updates made by a transaction are reflected on non-volatile storage (e.g., cell storage) before the transaction is allowed to commit.
    - The policies we choose affect how much work UNDO/REDO have to do
- Concurrency control
    - 2PL. DBMS handles deadlock detection.
- Isolation levels, from least strict to most strict
    - READ UNCOMMITTED: transactions can read uncommitted data
    - READ COMMITTED: transactions can only read committed data, but can experience non-repeateable reads
    - REPEATABLE READ: ensures that reads of individual items are repeatable
    - SERIALIZABLE: conflict serializability. Gets rid of the "phantom problem", where a new value appears between two reads.

Tradeoffs

- Lots of trade-offs when it comes to implementing a database management system
    - Combinations of STEAL/NO-STEAL FORCE/NO-FORCE policies
    - Frequency of checkpointing the logs
    - Logical vs. physical logging
    - Degrees of isolation
    - Locking granularity
    - Optimistic vs. pessimistic concurrency control