In a world of big data, we want transactions of sharded data with ACID guarantees:

- Atomic
- Consistent
- Isolated
- Durable

Atomic transactions ensure high availability of sharded data.

**ACID-RAIN: Ordering with Prediction, Committing with Independent Logs**

**Concurrency Control**
1. Optimistic, transactions run speculatively and then certify.
2. Conflict detection with timestamps.
3. Version reservation (lock on future version) by prediction.
4. Final certification at transaction end → lock-free: can replace slow/failed nodes immediately; reservations are only hints.

**Log Structure**
- After end_txn (read-set, write-set)
- Result from TM
- txnEntry can be garbage-collected

**Benefits of Prediction**
Different recall ratios with perfect precision (no wrong guesses).

- recall = 0: no prediction and no reservation (classical approach)
- recall = 1.0: predicting all accesses

Better recall → higher commit ratio

Different precision ratios (wrong guesses) with perfect recall.

Bad precision → more conflicts in small data sets

**Simulation Results**
- Custom-made simulator.
- Transactional YCSB workloads.
- Uniform random object access.

**Certification Scalability**
- Global log: Forms a bottleneck.
- 2PC with SMR TMs: longer certification time so higher contention.

**Execution Example with Prediction**
1. Prediction and reservation.
2. Transaction run.
3. Certification.
4. Garbage collection (asynchronous)