Distributed Transactions
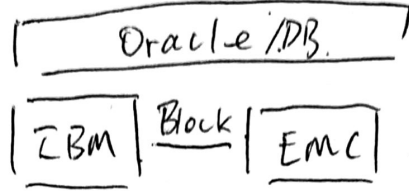
I. O. E.
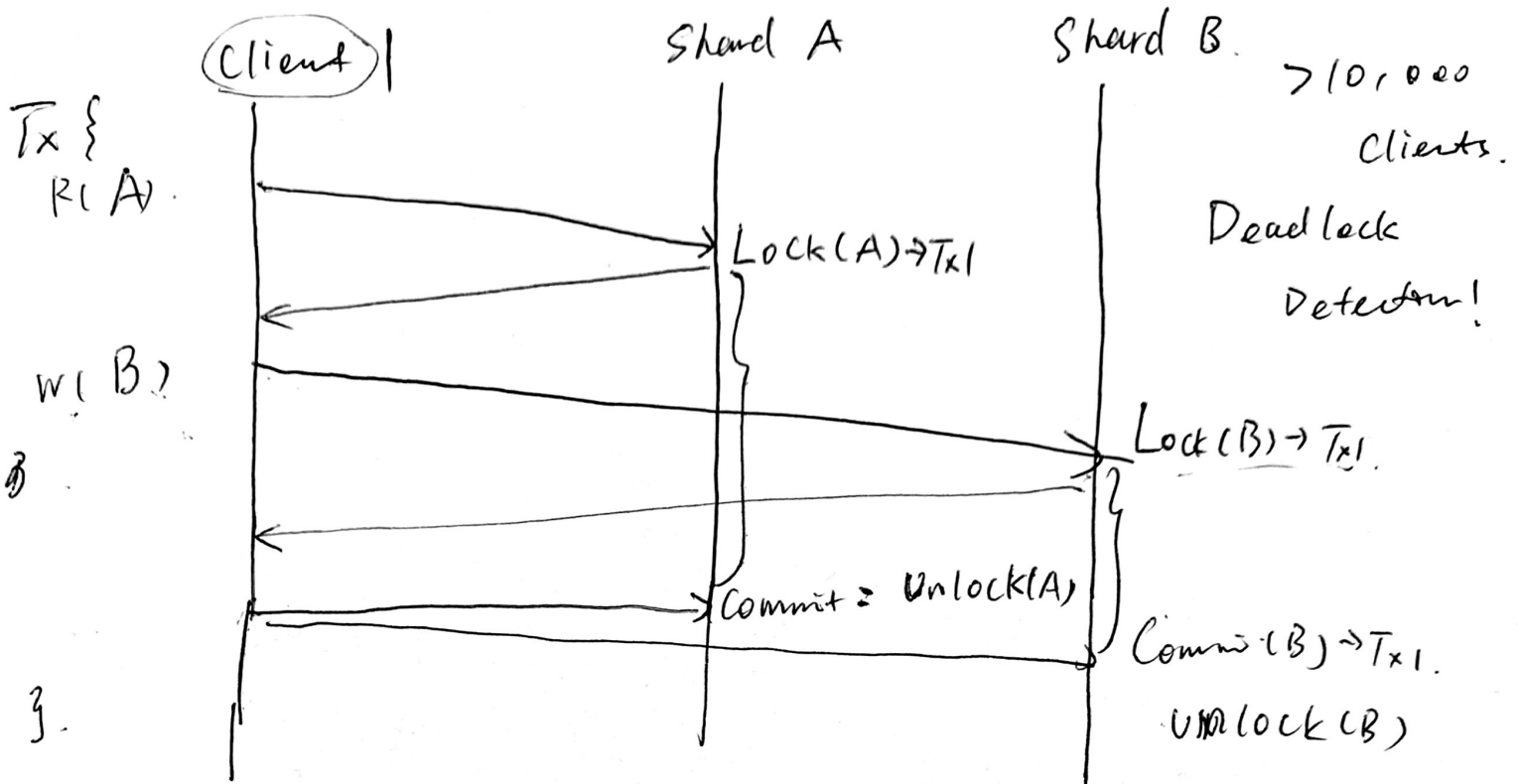
{ MongoDB.
  RethinkDB.
  Cas — — —
  Cockroach DB.
  Aurona. }

Oracle /DB.

| IBM | Block | EMC |

Expensive

Slow

App/Shard

MySa — — — — — — — — MySa.

App. DB □ . □ □ □ — — — — — — □

DB . 0 □ □ □ — — — — — □ .

Transaction

⋮

END.
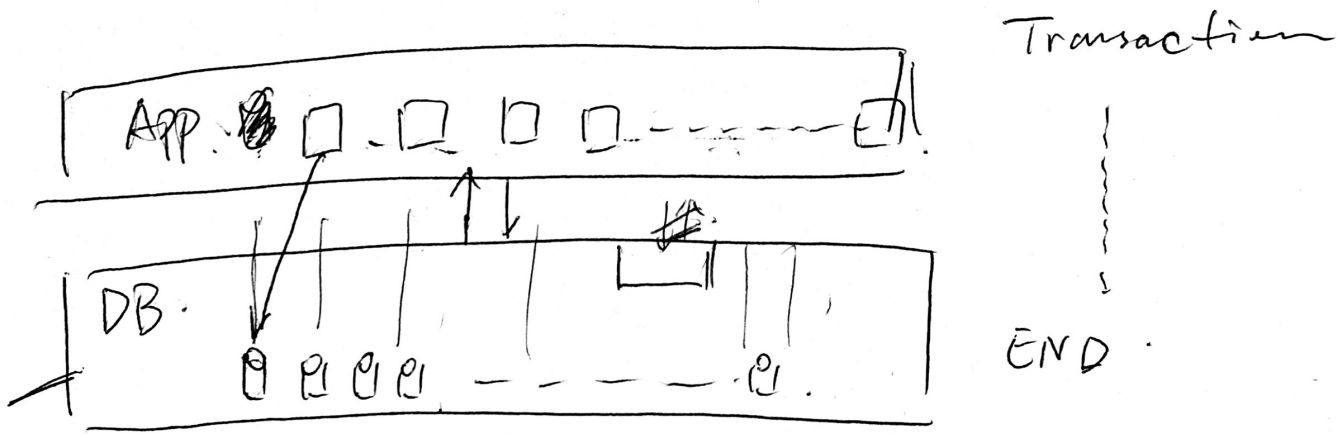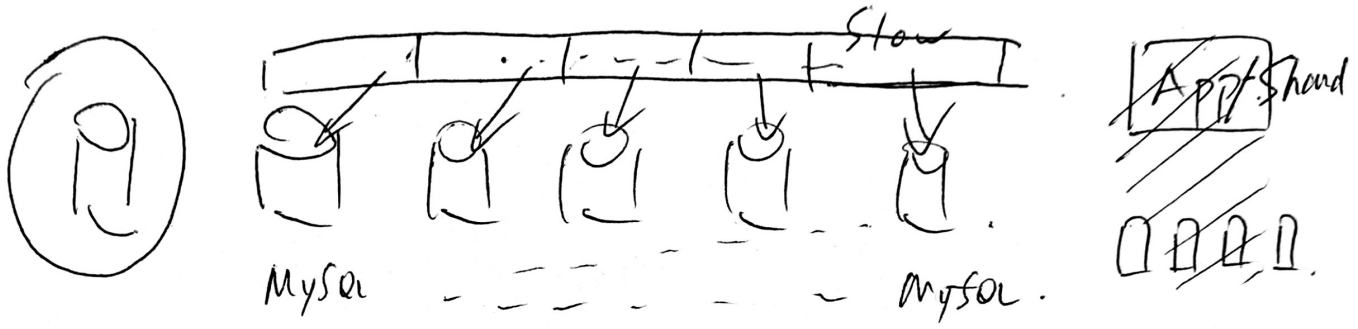
① . Scalibility. / Shard.

② . Reliability . / Replicas

{ Consistency.
  / Isolation level.
  throny Failures

Transaction w/ Sharding (Partition)

```
                    Client              Shard A            Shard B
                                                                    > 10,000
Tx {                                                                  Clients.
  R( A)  ←————————————————→ Lock(A)→Tx1                              Deadlock
                                                                     Detector!
  W( B)  ————————————————————————————————→ Lock(B)→Tx1
  B      ←————————————————————————————————
                            Commit: Unlock(A)
  }      ————————————————————————————————————→ Commit(B)→Tx1.
                                                         Unlock(B)
}
```

Global lock service. (MySQL Cluster).

Lock   A   B   C . . . .

Pro { Deadlock Detect easy .
      easy develop / debug .
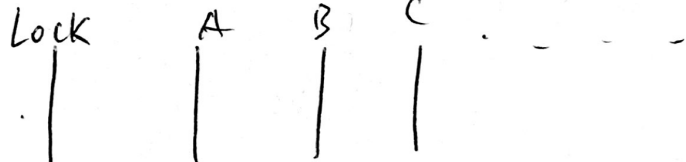
Cons { extra. commu → lower thp.
       single-point failure
       —————————————————————
       hard to scale
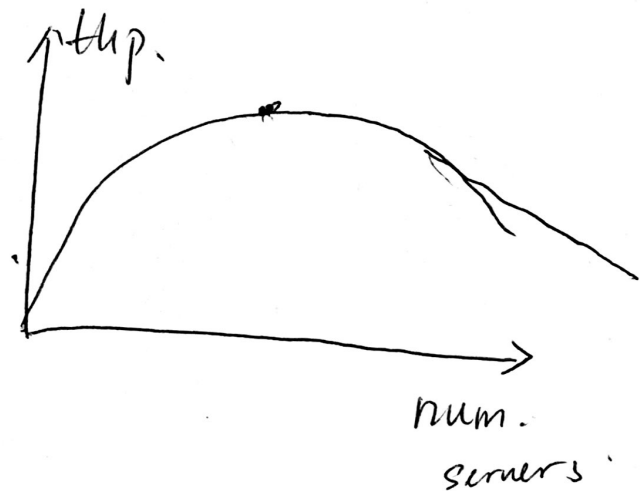
Dead Lock   Detection ⟶ Prevention

Ordering ?

Tx1   A   B   Tx 2.

A
                      Lock (A , tx1)   Lock (B, tx2)   B
                        → ok.   → OK .
    ← locked (A)   → wounded.

B .
                                              A .
                      ← Lock (A, tx2)   Lock (B , tx1)
                        → OK.   → wait.
    ← locked (B)
                        ← comit   ← comit / unlock .
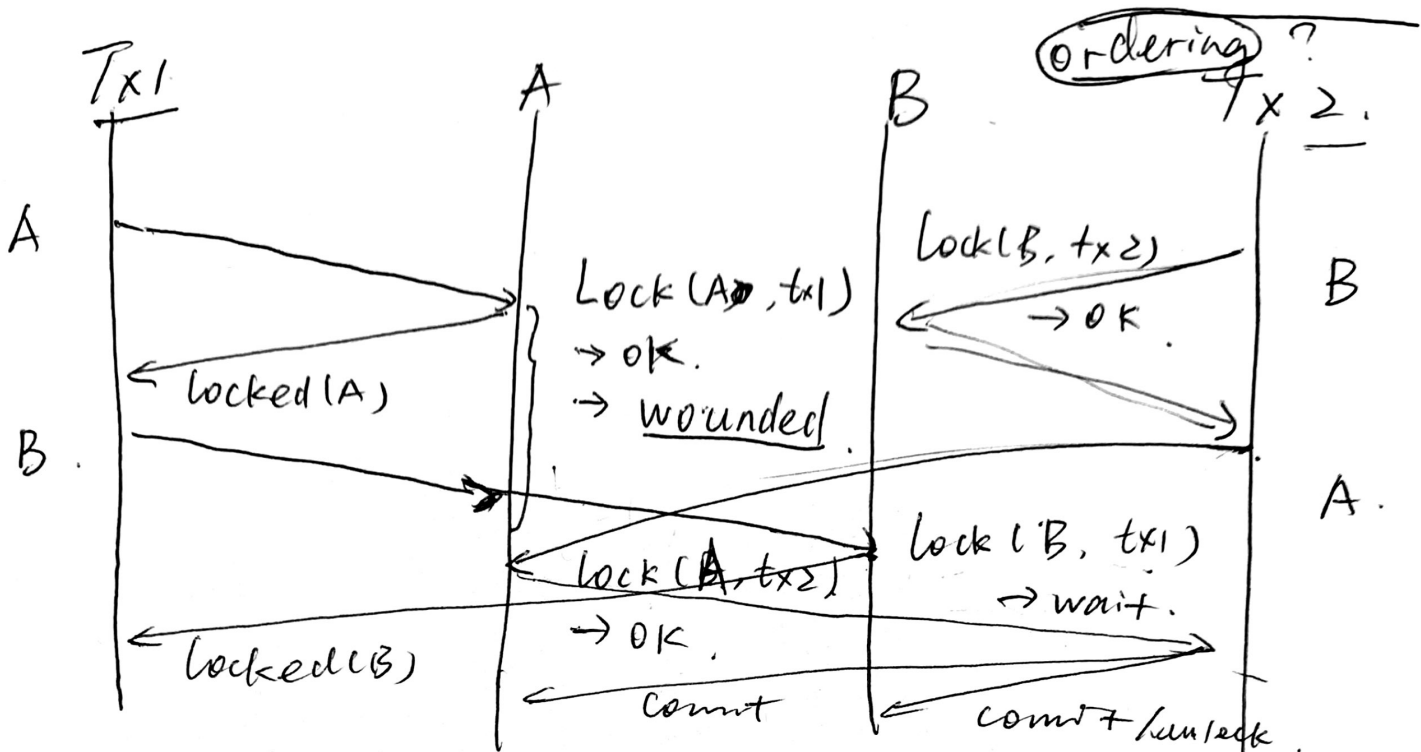
greater id   can   wound   tx   w/   smaller id.
        Smaller id   just wait.

Tx 1   A=0   B=0   Tx2 .

R(A)=0   lock(B,tx2)→ok   R(B) =0

R(B)   W(A)
=B+1   wounded   W(A)=B+1

W(B)=A+1
        wound

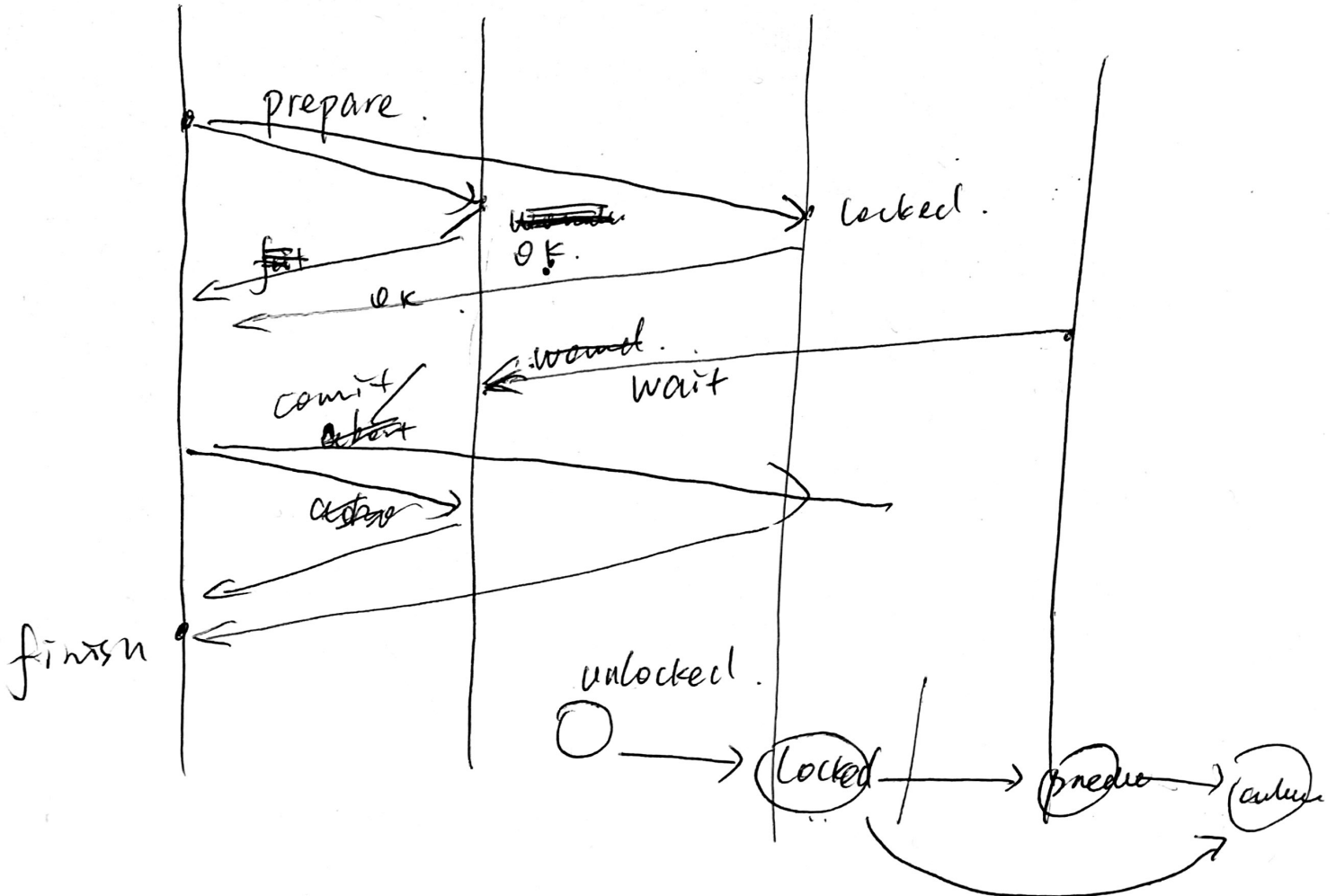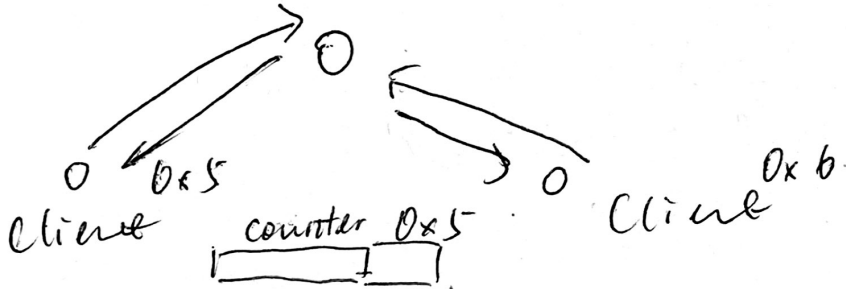   A=1

    X

Error.   B=1
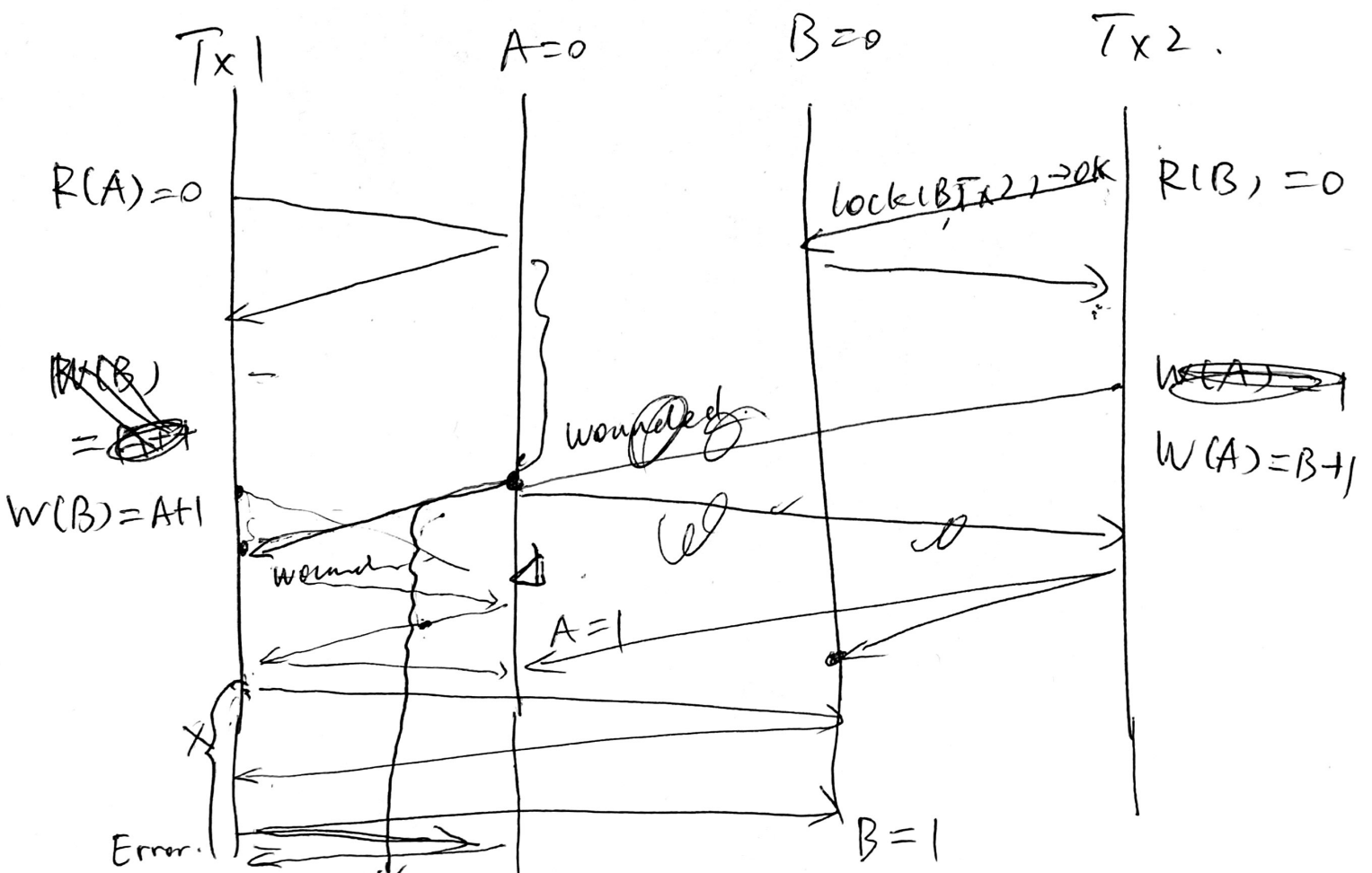
id = read ("next_id")
write (id., "buy a pen")

read / write ?
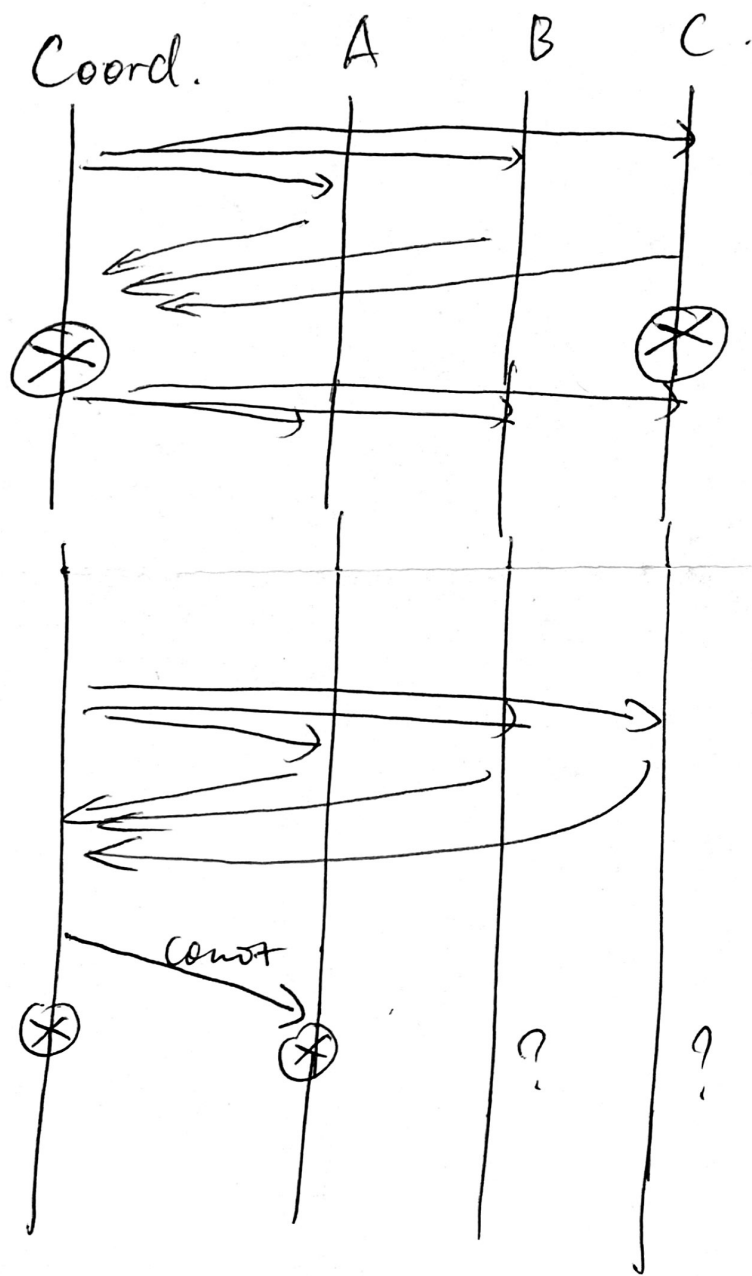set prior to execute

$N > M$.   id $\approx$ time.

$\begin{cases} N \text{ wound } M, \quad M \text{ waits.} : \text{old txs cannot finish.} \\ M \text{ wound } N, \quad N \text{ waits.} : \text{old tx slow} \end{cases}$

client   0x5   counter   0x5   Client   0x6

prepare.

locked.

OK.

OK.

wound.
wait

commit
abort

finish

unlocked.

Locked → prede → cutre

Prepare. → Commit / Abort.

→ 2 - Phase — Commit / 2PC. (No failures)

(Consensus)

Coord.   A   B   C.

Conot

2PC → 3PC. (1980s)

(Paxos.)