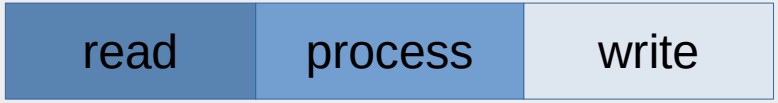
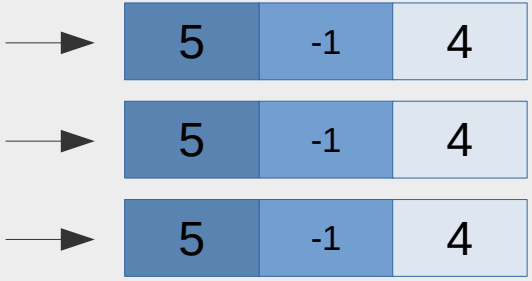
# Concurrency Issue In Product Service

PayPing.




Sequential


Concurrent



# Concurrency Control

- Optimistic
  - No checking and last one in wins
  - Comparing the row version (soft lock)
- Pessimistic
  - lock

# Locks

- Pros
  - Very secure (most useful when correctness is important)
- Cons
  - Not scalable (Significant database management resources)
  - Prone to deadlocks
  - Locks can last a long time

# Why distributed

- Access to lock from different instances and programs (Not just a row)
- Redis (Fast and efficient at managing temporary values)
- Less database resource

# Safety of distributed locking

- Single node
- Cluster of nodes (Redlock pattern by Redis)

links:

- https://martin.kleppmann.com/2016/02/08/how-to-do-distributed-locking.html
- http://antirez.com/news/101

# Library

- DistributedLock
  - https://github.com/madelson/DistributedLock
  - https://www.nuget.org/packages/DistributedLock/

# Types of lock

- Locks
- Reader-writer locks
- Semaphores

# Resources

- https://www.codemag.com/article/0607081/Database-Concurrency-Conflicts-in-the-Real-World
- https://redis.io/topics/distlock
- https://martin.kleppmann.com/2016/02/08/how-to-do-distributed-locking.html
- http://antirez.com/news/101
- https://github.com/samcook/RedLock.net
- https://github.com/madelson/DistributedLock
- https://github.com/aRmanNM/concurrency-test