



ACID in SQL Server

what it is, how it works, and how to live more adventurously.

Thank you to our sponsors

In this session

- What is ACID compliance
- Isolation levels in SQL Server
- Durability in SQL Server

A

Atomic

C

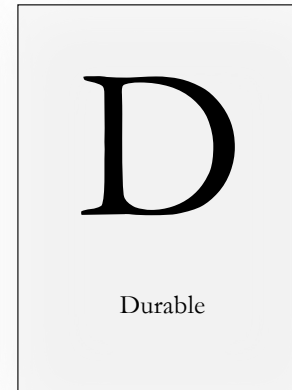
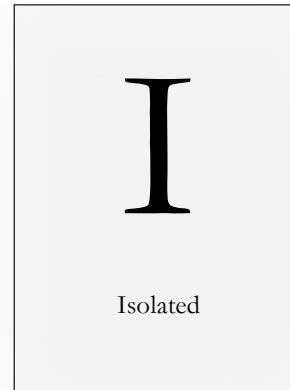
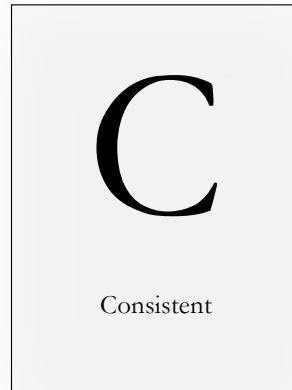
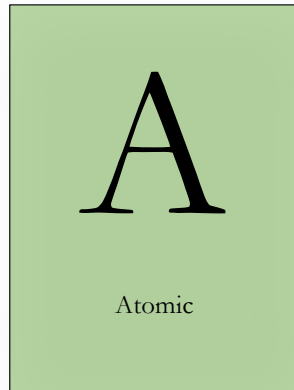
Consistent

I

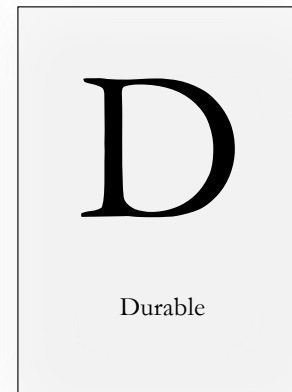
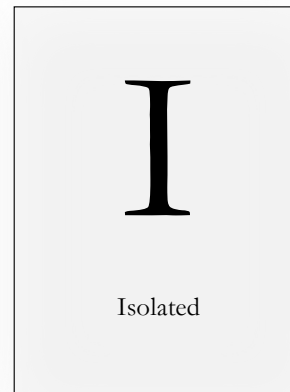
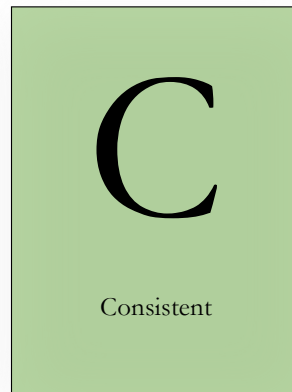
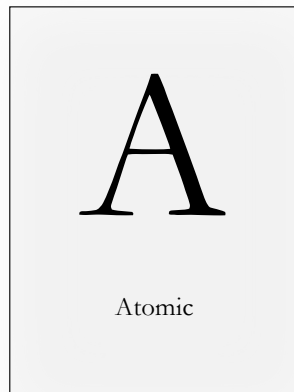
Isolated

D

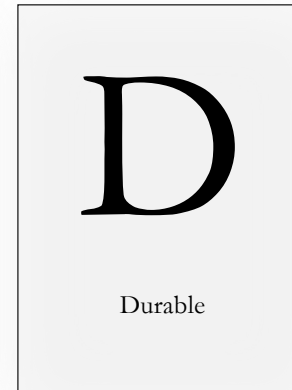
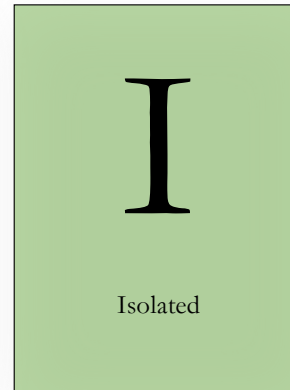
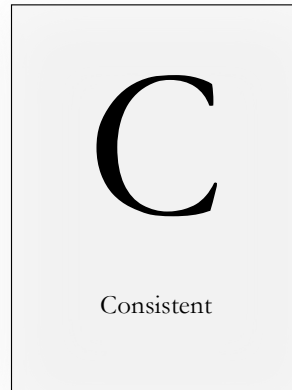
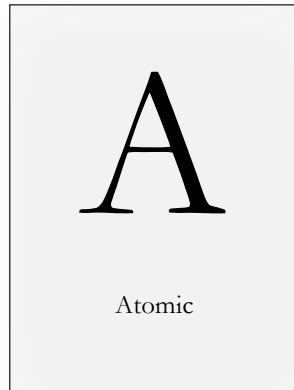
Durable



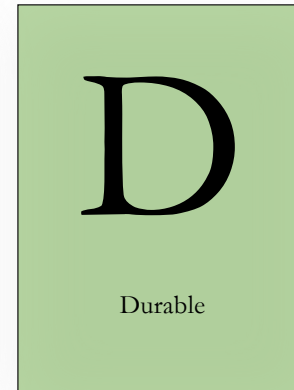
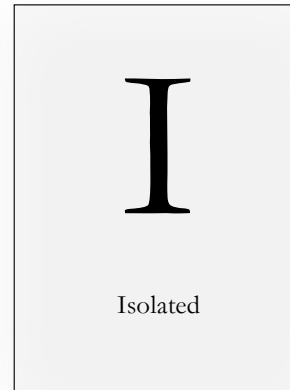
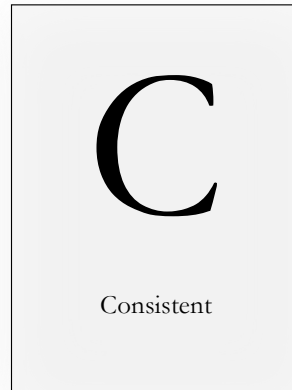
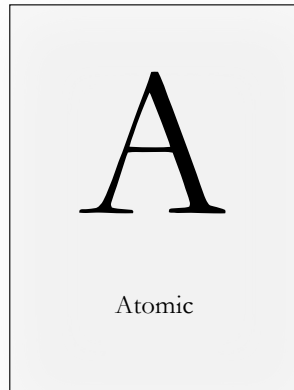
All or nothing.
The *business* logic.



A single truth.
The *database* logic.



No interference
when changing data.



Keep all promises.

Who am I?

- SQL Server developer since 1997
- Consultant
- Organizer of Data Saturday Stockholm
- Data Platform MVP
- Dog person

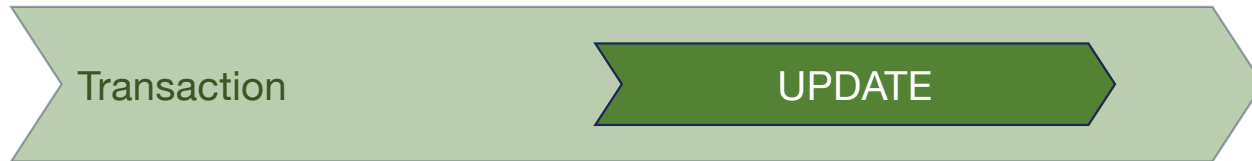


@dhmacher on (almost) all the socials.

Isolation: Quirks and features of multi-user databases

Dirty reads

- no Nobel literature prize for you.



Read Uncommitted (a.k.a. NOLOCK)



Dirty reads

- no Nobel literature prize for you.



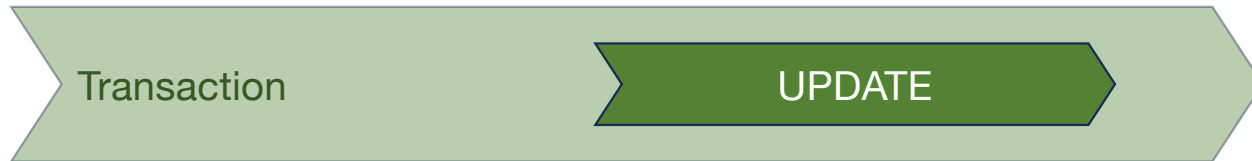
Dirty reads

- no Nobel literature prize for you.

- DML does not respect READ UNCOMMITTED, and cannot make changes to uncommitted data.

Dirty reads

- no Nobel literature prize for you.



Read Uncommitted (a.k.a. NOLOCK)



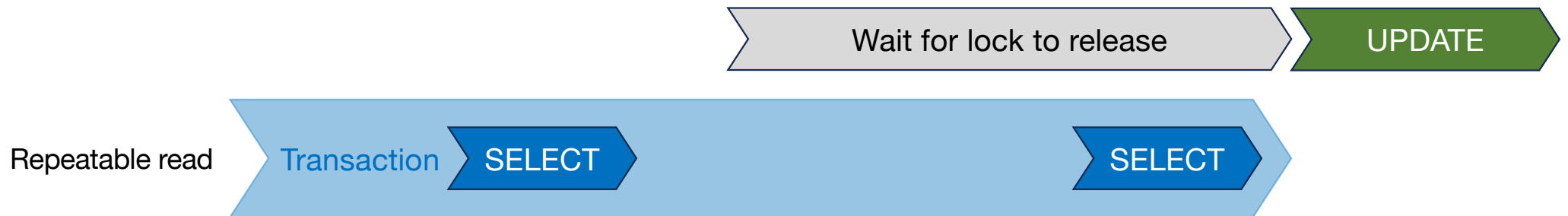


DEMO

Non-repeatable reads



Non-repeatable reads





DEMO

Phantom reads

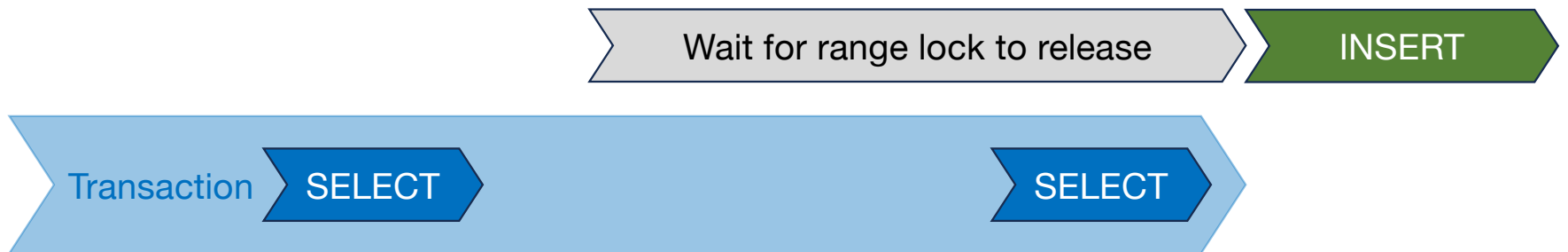
Your database is gaslighting you.

Phantom reads



Phantom reads

Serializable





DEMO

What are isolation levels?

What are isolation levels?

Read
uncommitted

Read
committed

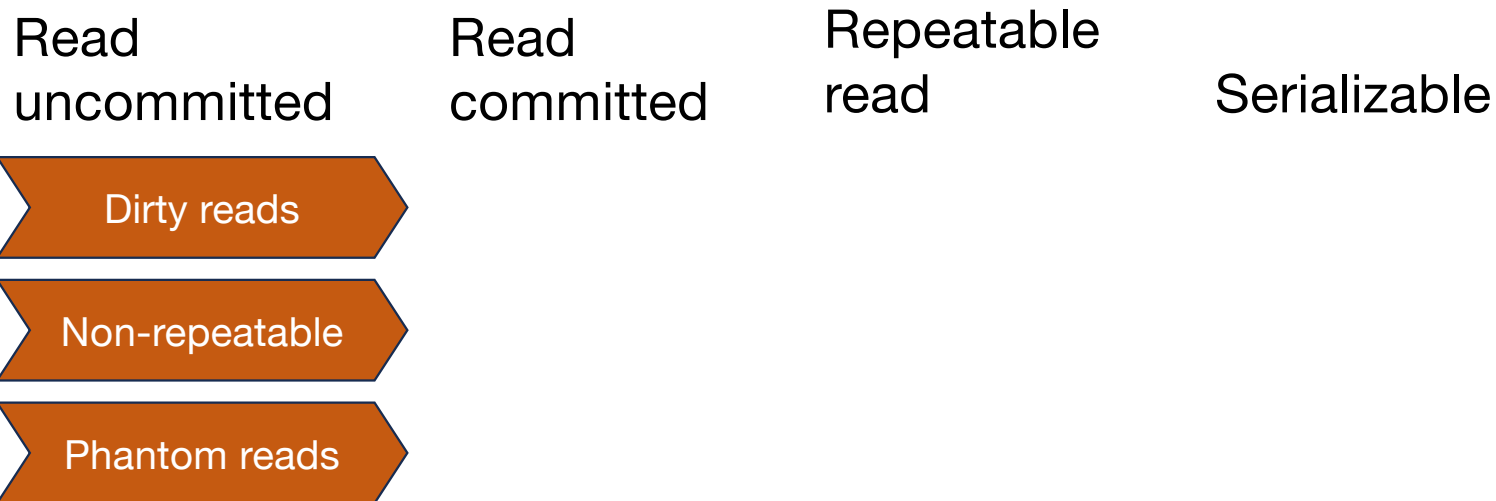
Repeatable
read

Serializable

← less likely to block

more likely to block →

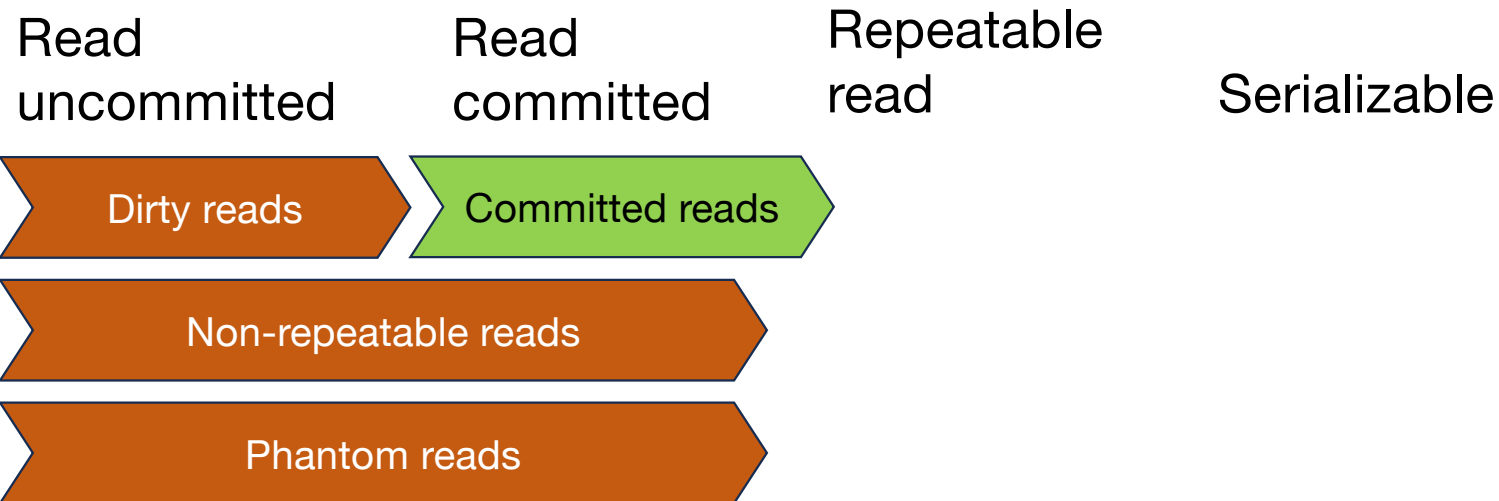
What are isolation levels?



← less likely to block

more likely to block →

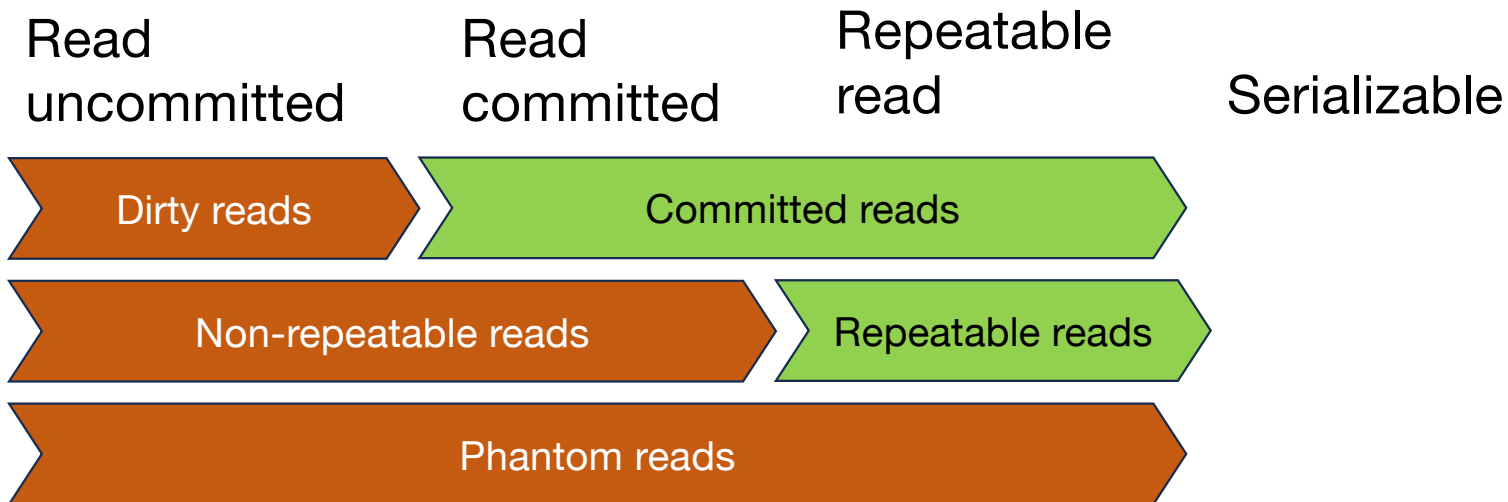
What are isolation levels?



← less likely to block

more likely to block →

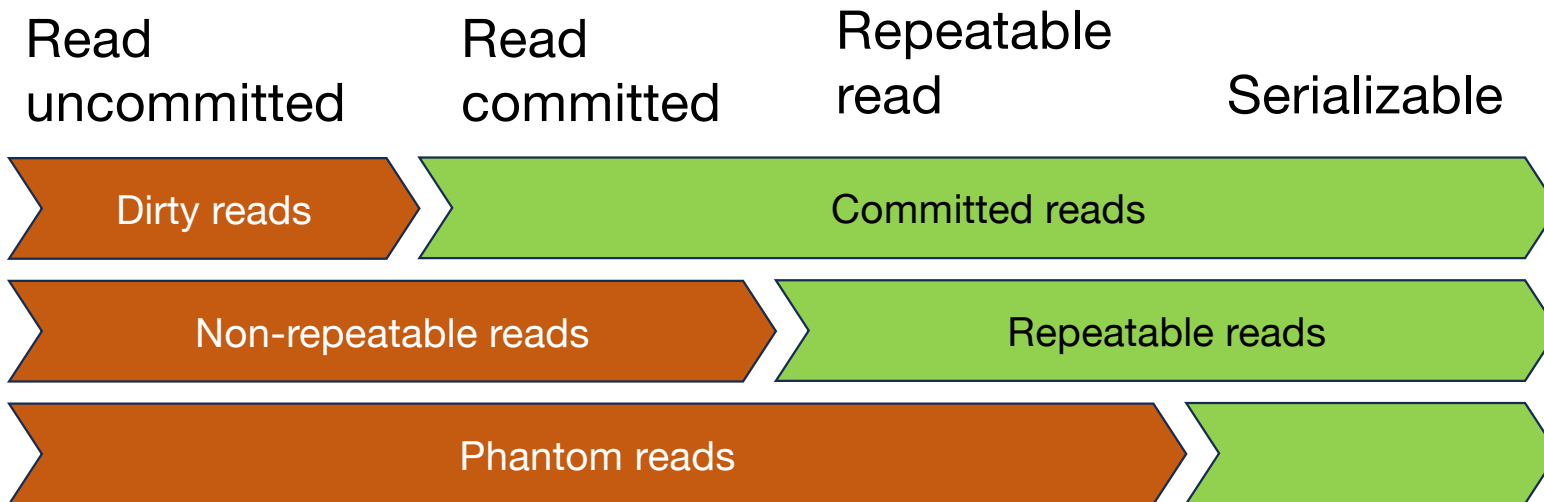
What are isolation levels?



← less likely to block

more likely to block →

What are isolation levels?



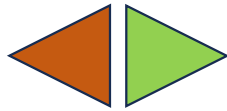
← less likely to block

more likely to block →

So... Serialize all the things, then?

Less locking

- Less isolation
- Less predictable

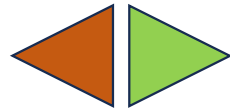


More locking

- Better isolation
- More predictable

So... Serialize all the things, then?

Less locking



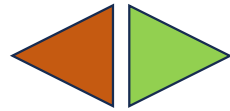
More locking

- Less isolation
- Less predictable
- Better concurrency

- Better isolation
- More predictable
- Lower concurrency

So... Serialize all the things, then?

Less locking

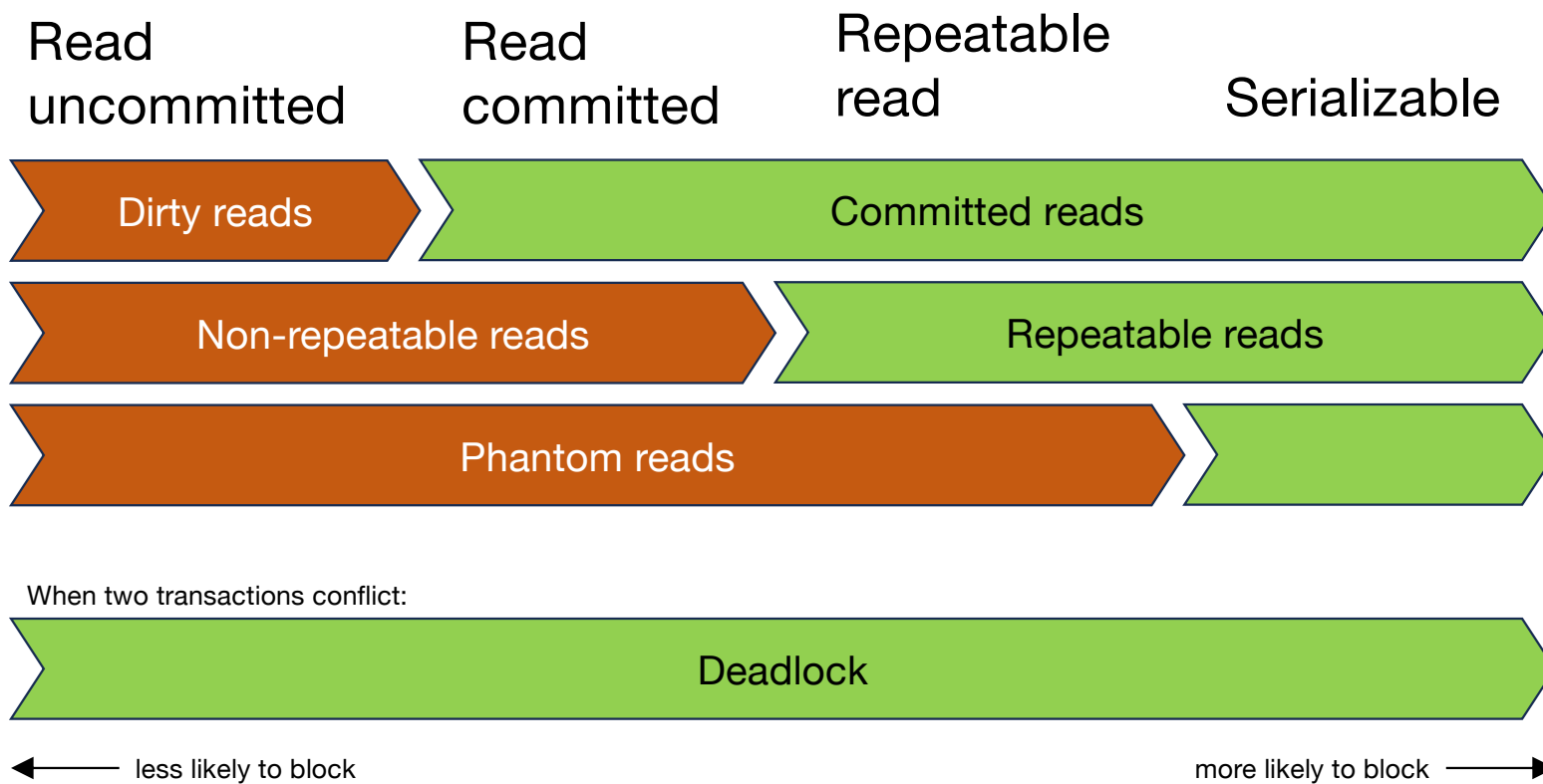


- Less isolation
- Less predictable
- Better concurrency
- Fewer conflicts

More locking

- Better isolation
- More predictable
- Lower concurrency
- More conflicts

What are isolation levels?



Deadlock

Transaction 1

UPDATE a

Transaction 2

Deadlock

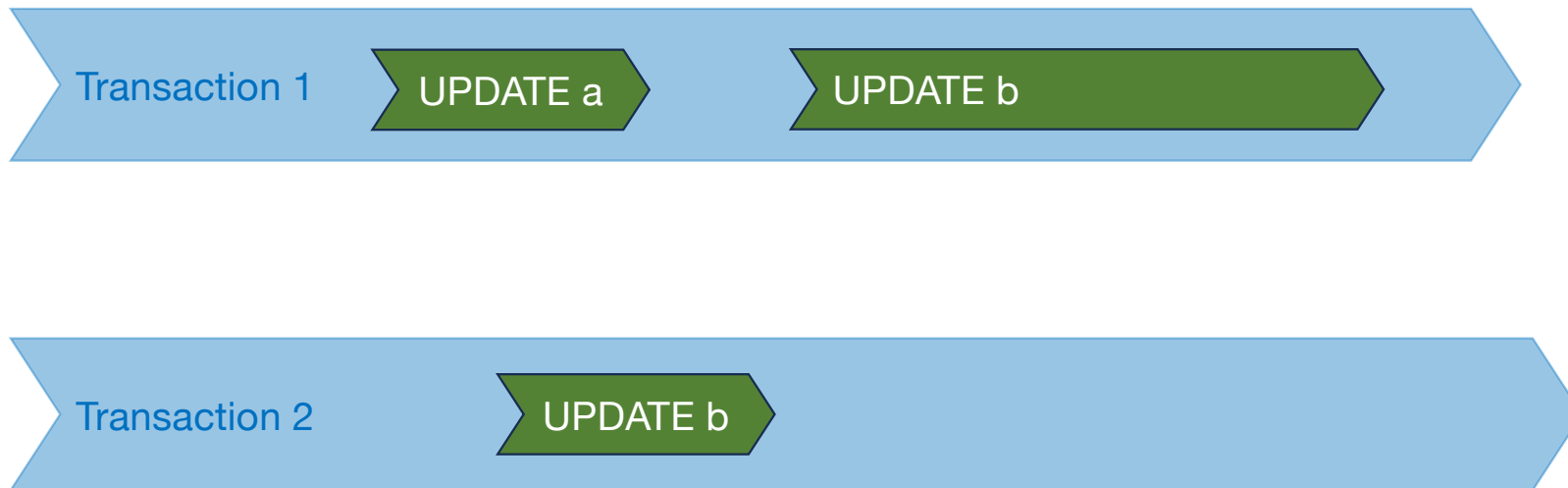
Transaction 1

UPDATE a

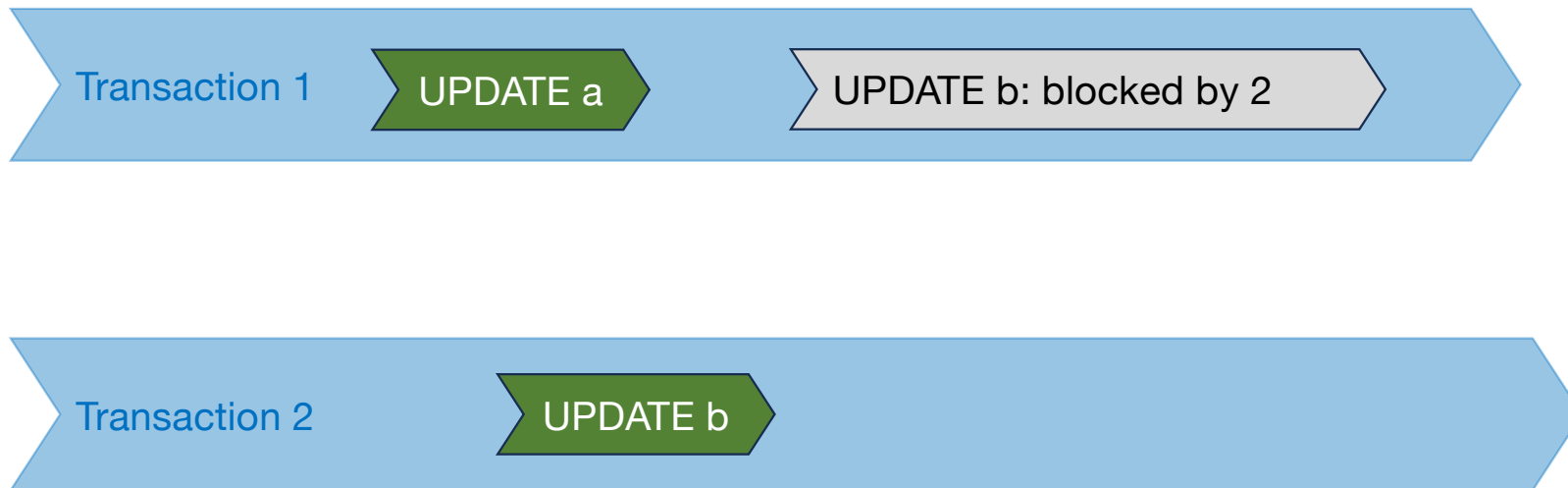
Transaction 2

UPDATE b

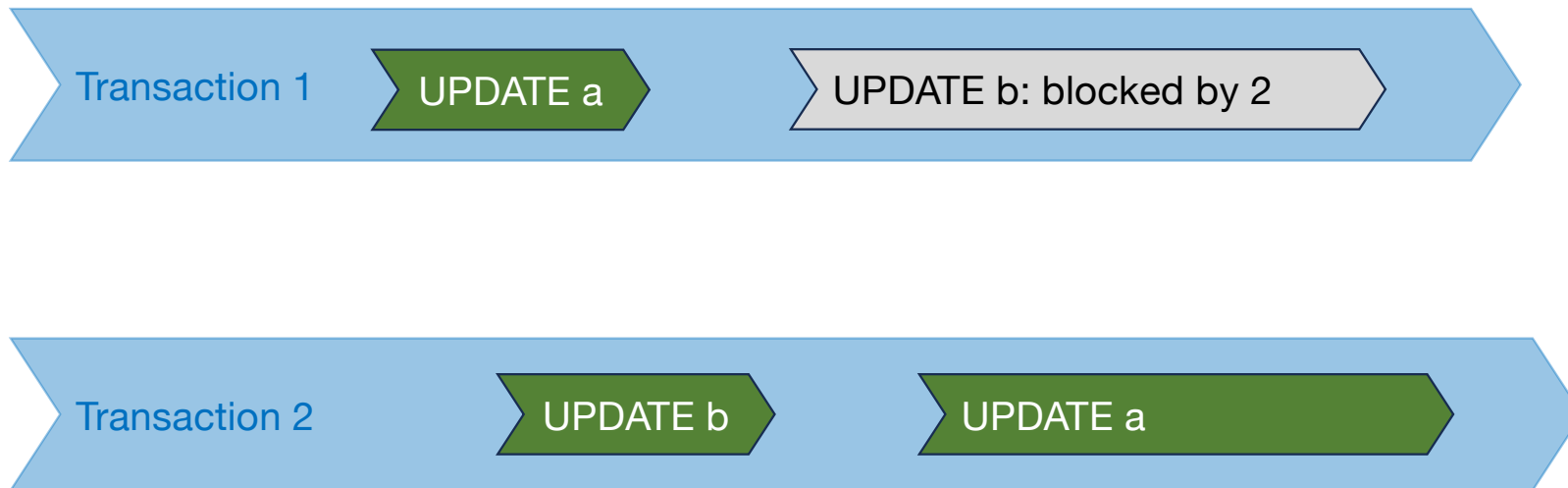
Deadlock



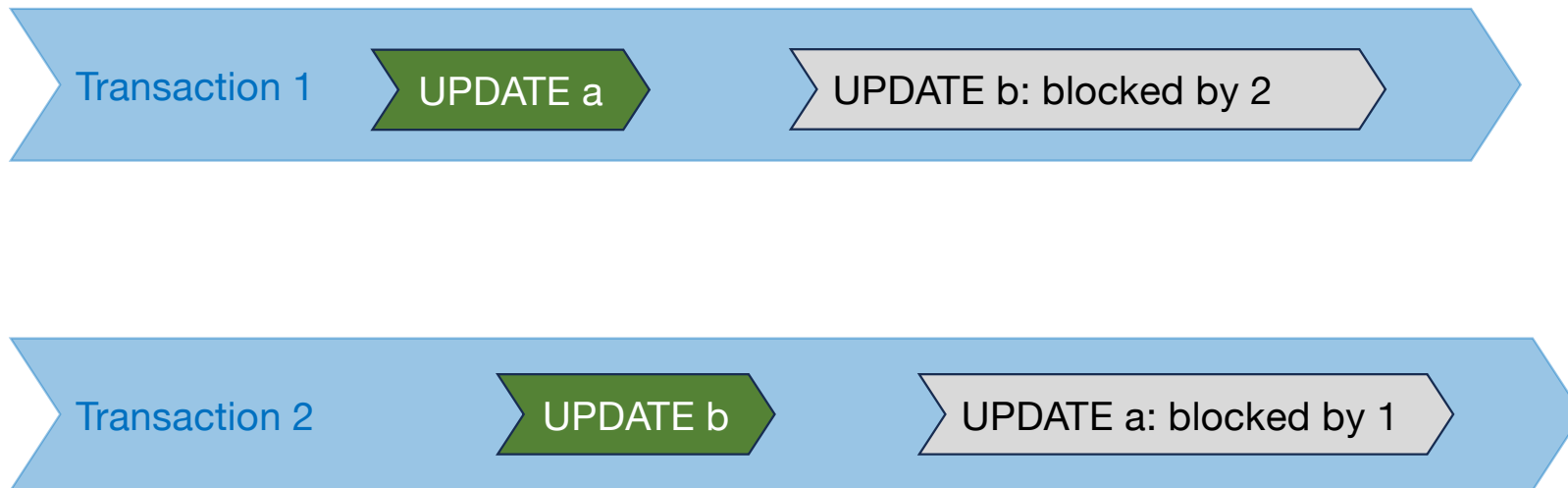
Deadlock



Deadlock



Deadlock



Deadlock

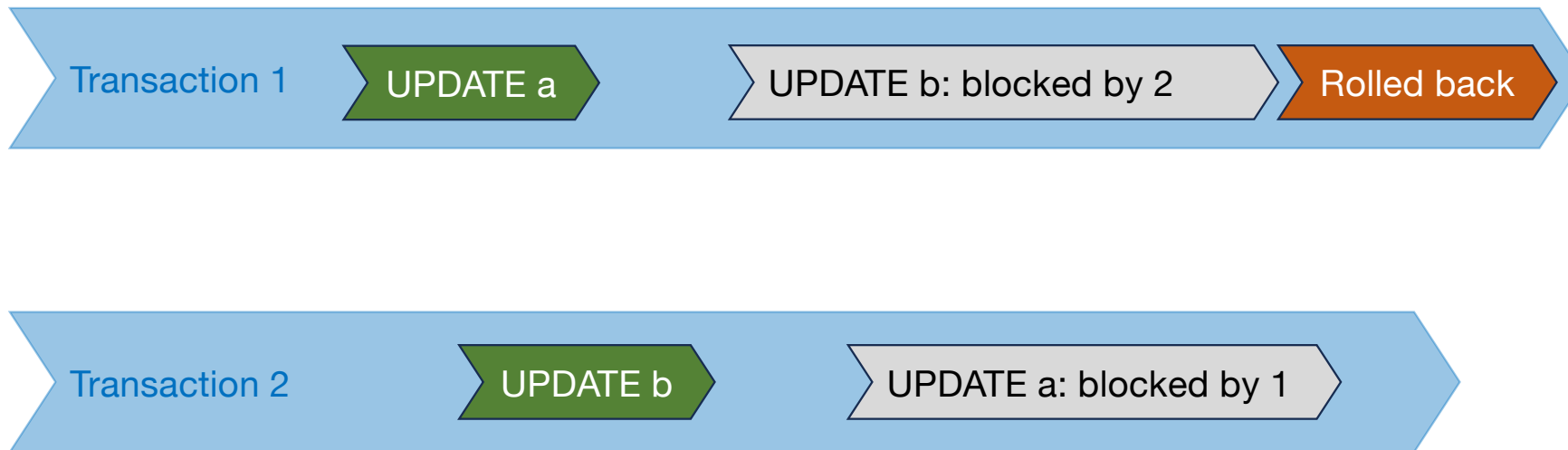


Deadlock: Transaction 1 and 2 are waiting on each other.



Deadlock

SQL Server will fail the transaction with the least amount of work to roll back:



Deadlock

Transaction 1

Rolled back

Transaction 1 is no longer blocking a.

Transaction 2

UPDATE b

UPDATE a

Deadlock

Transaction 1

Rolled back

Transaction 2

UPDATE b

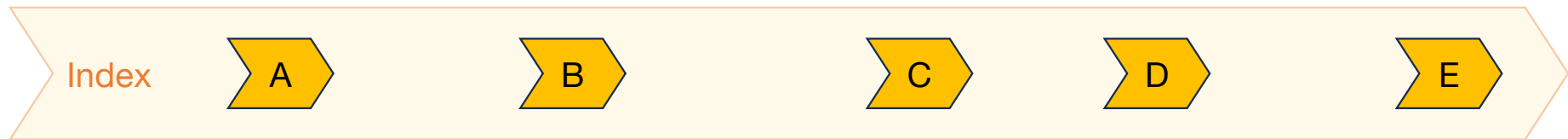
UPDATE a

COMMIT



DEMO

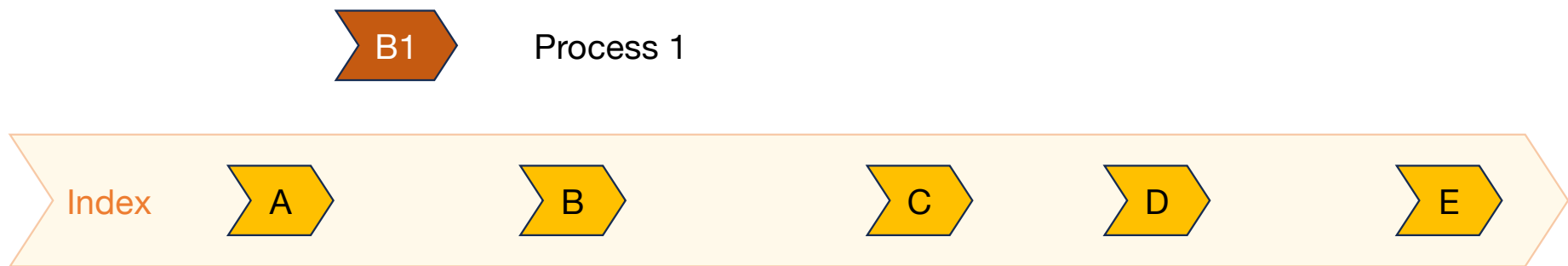
The serializable merge deadlock



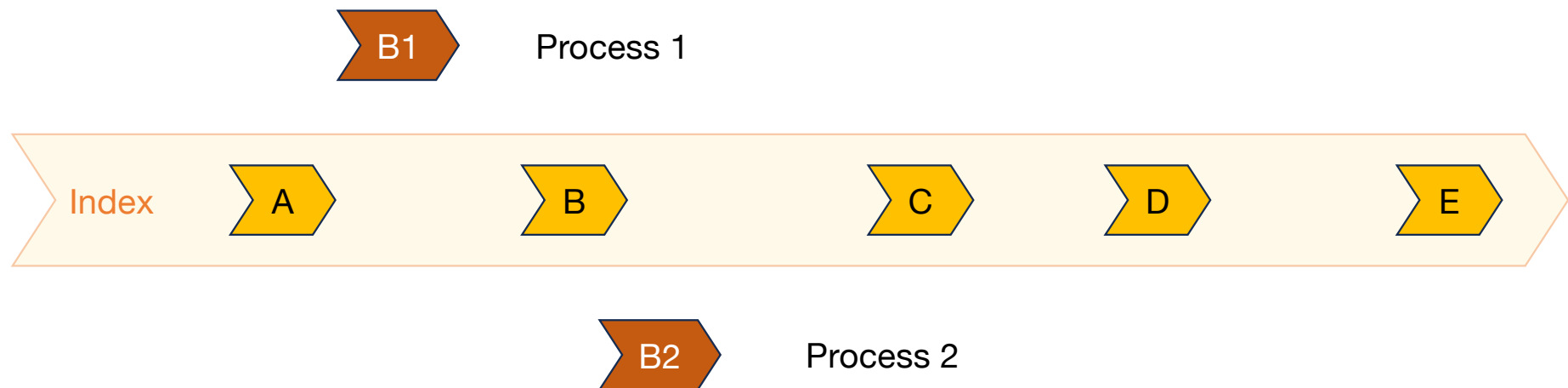


DEMO

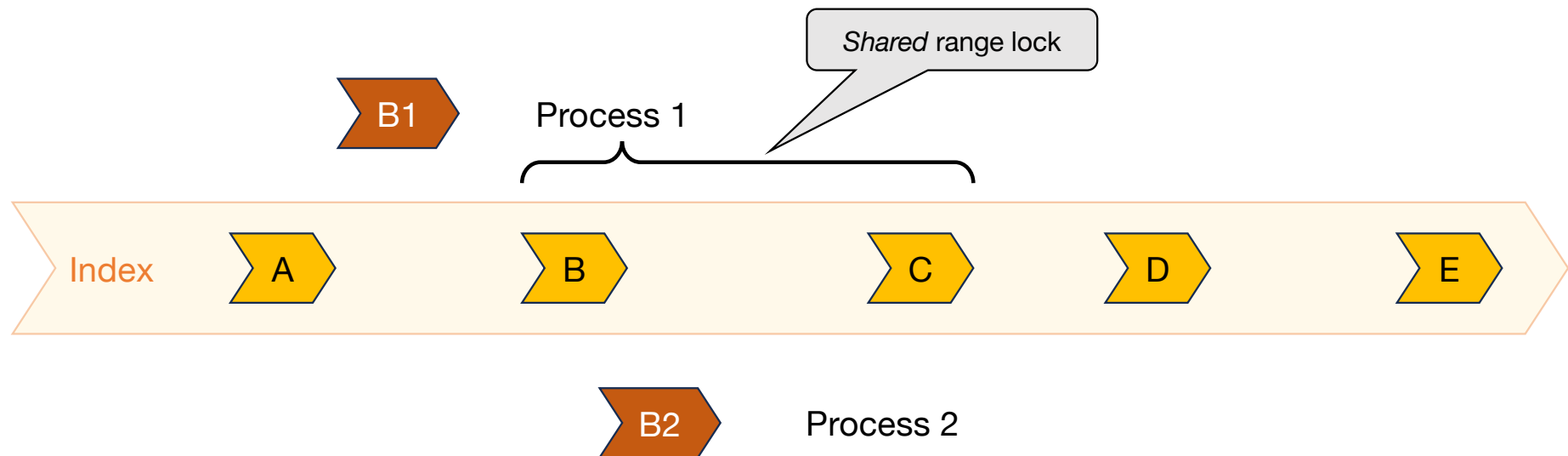
The serializable merge deadlock



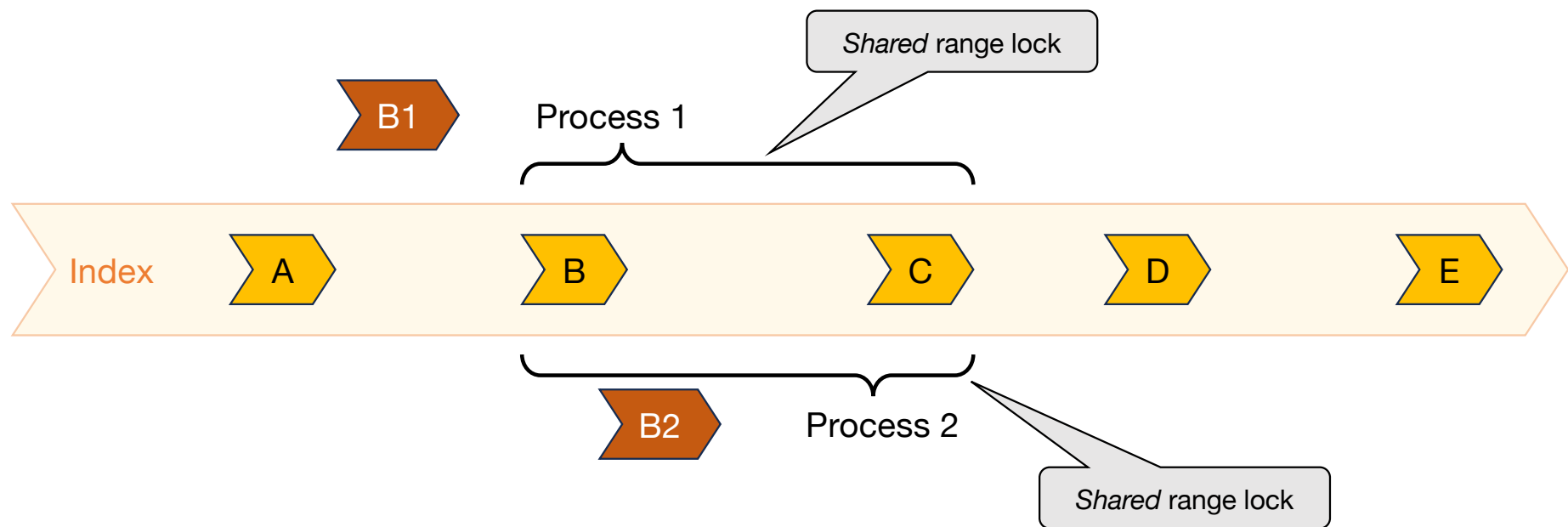
The serializable merge deadlock



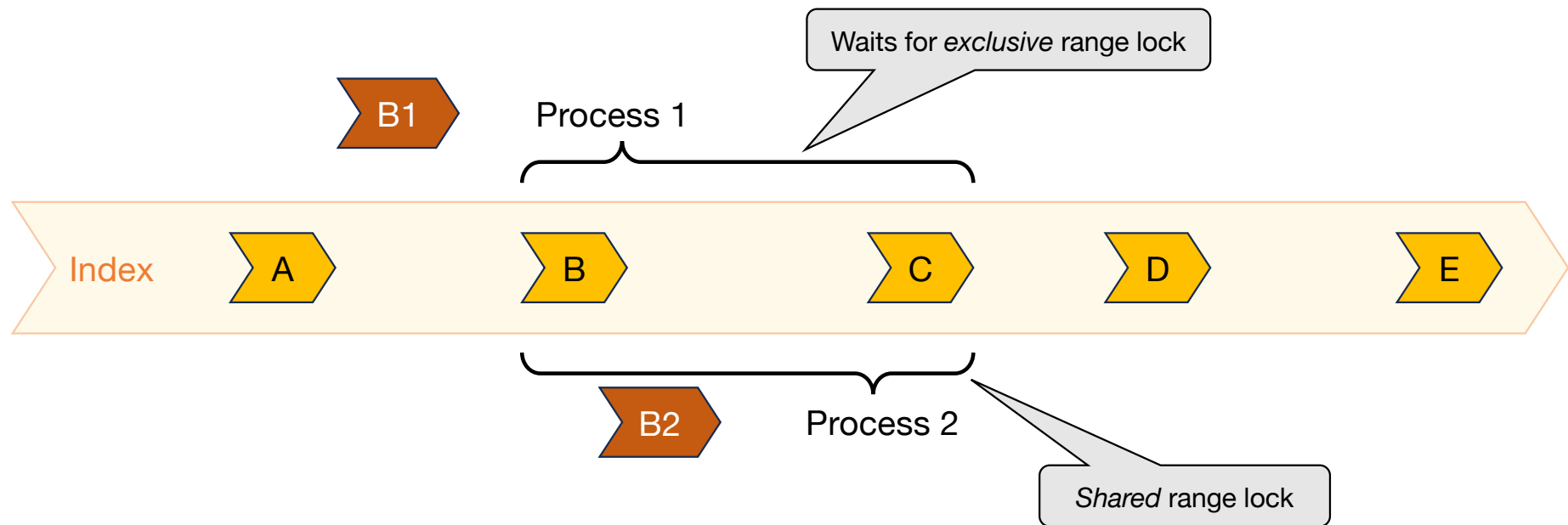
The serializable merge deadlock



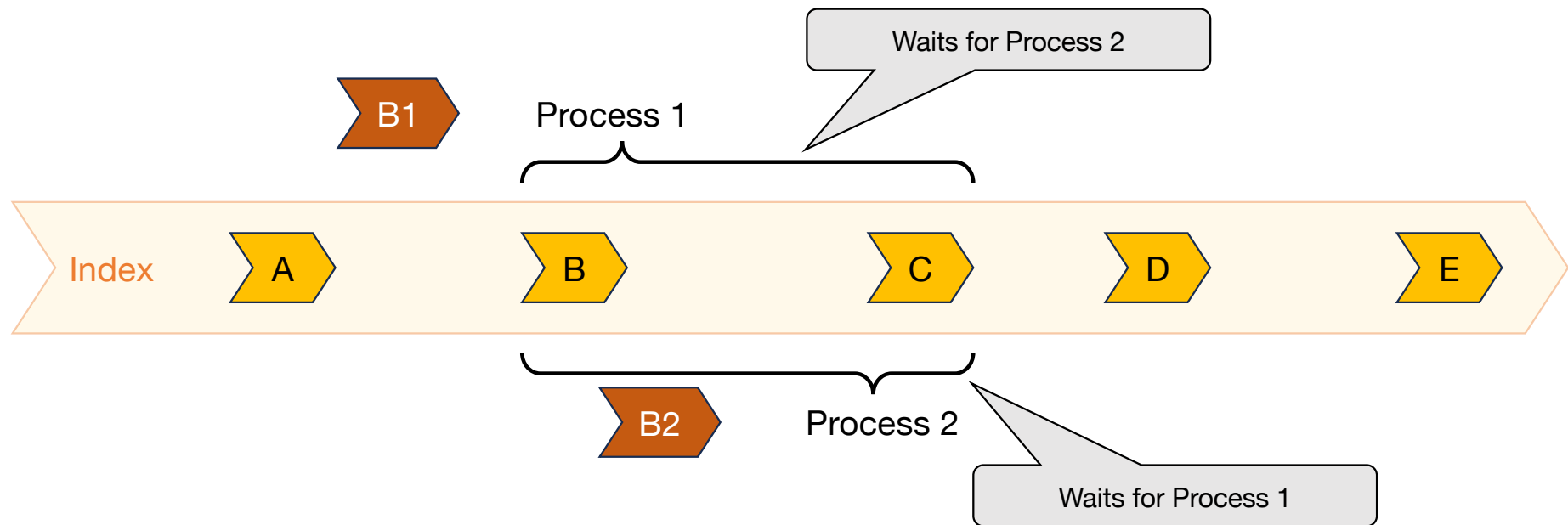
The serializable merge deadlock



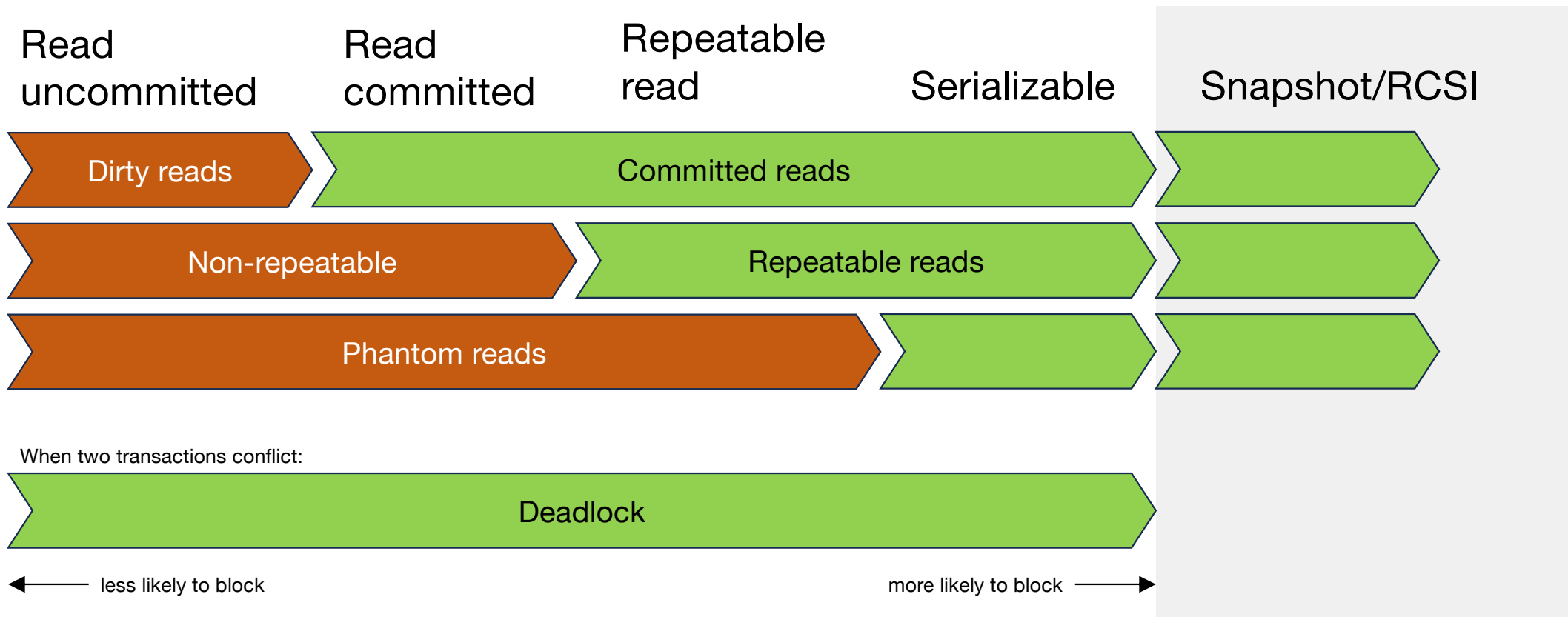
The serializable merge deadlock



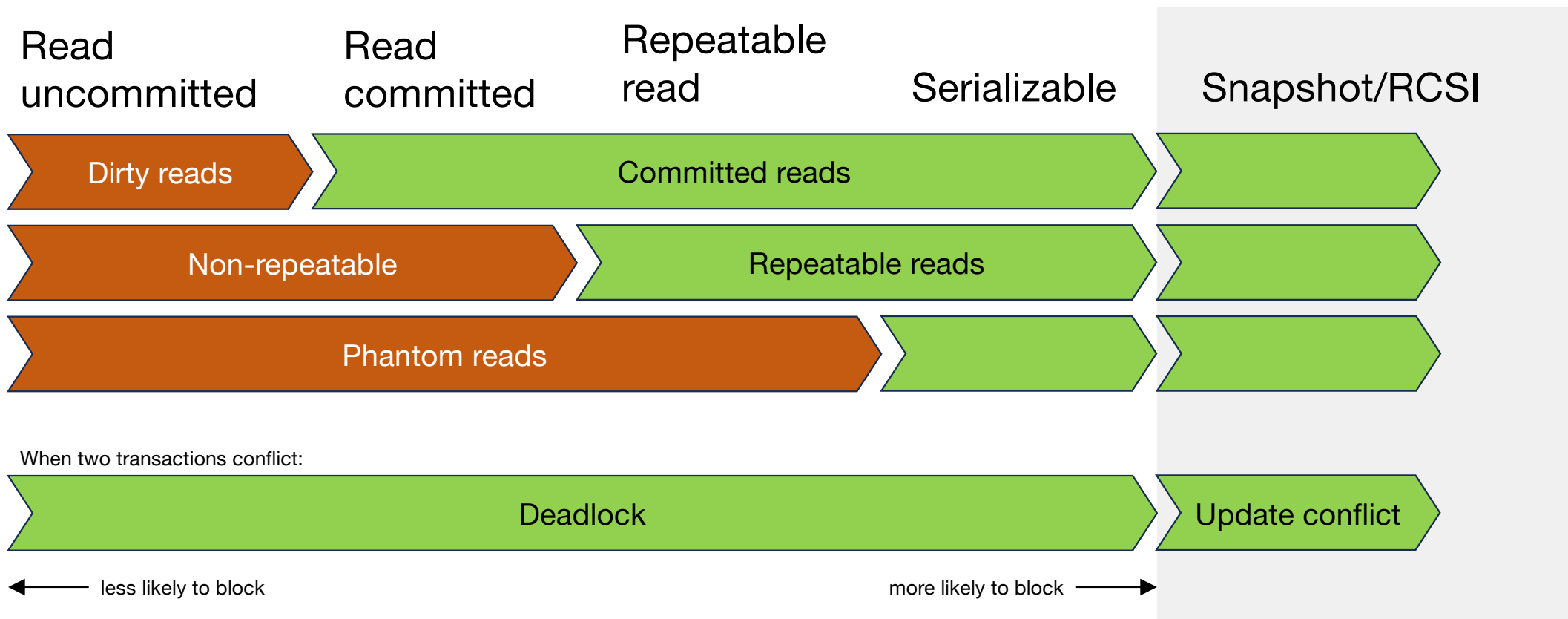
The serializable merge deadlock



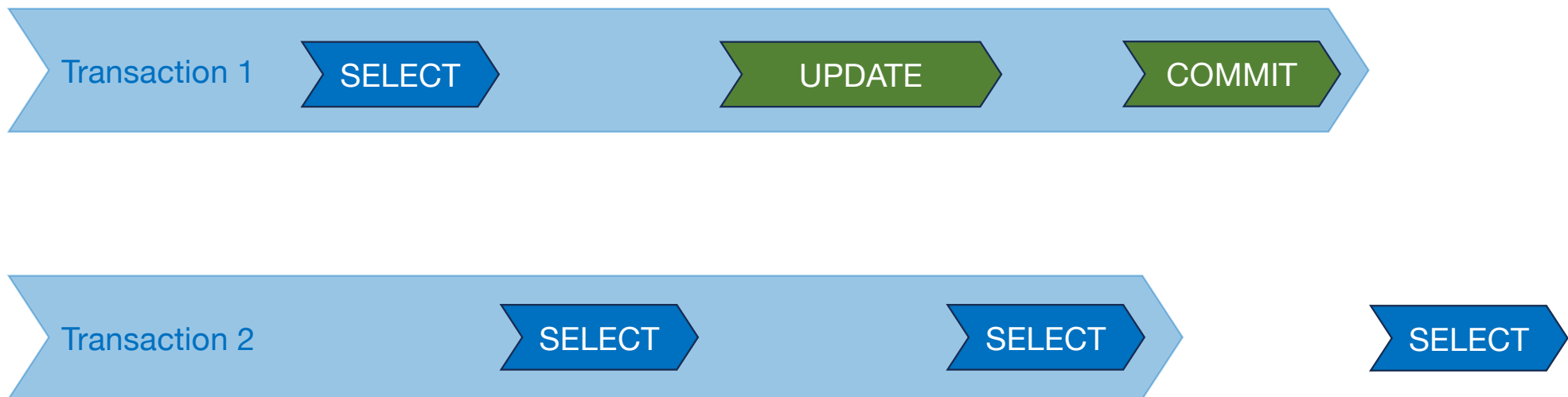
Snapshot



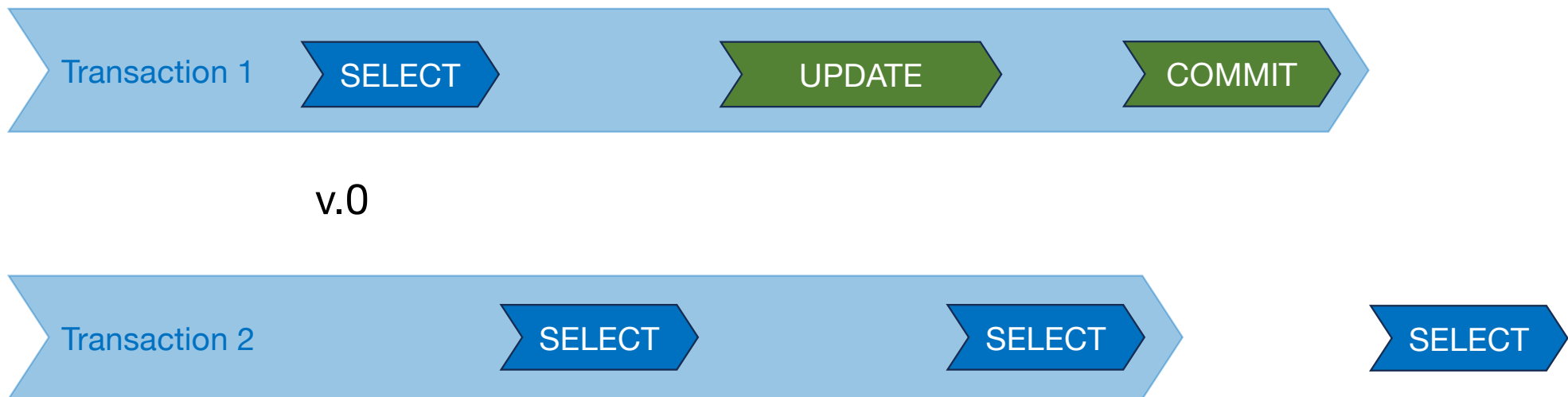
Snapshot



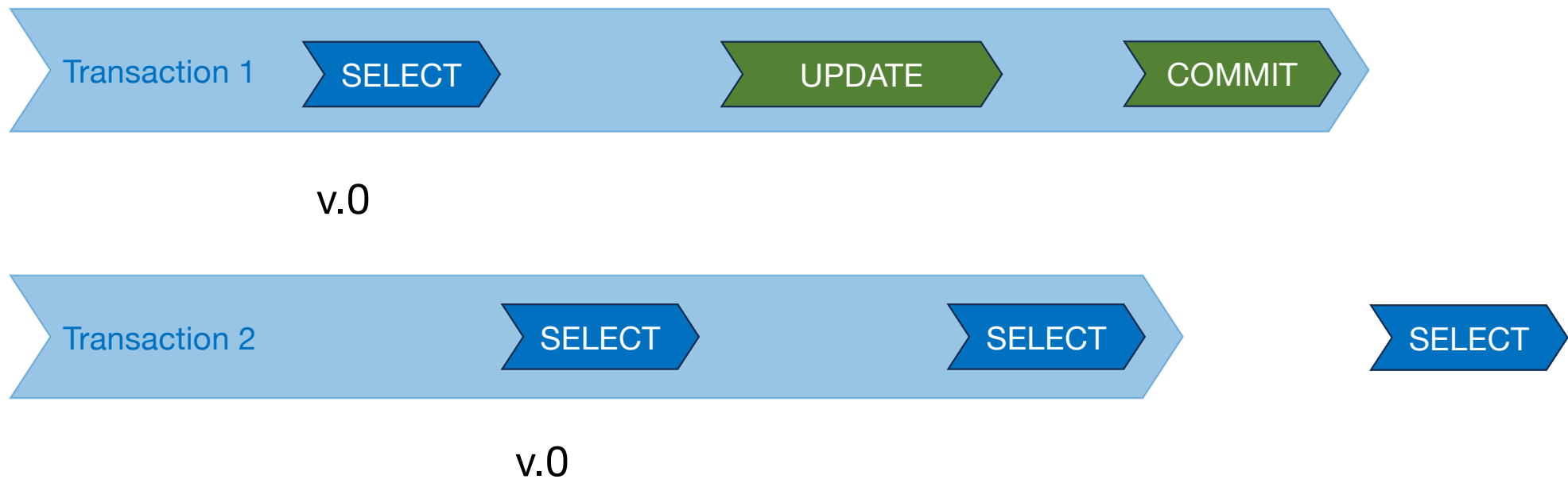
Snapshot isolation



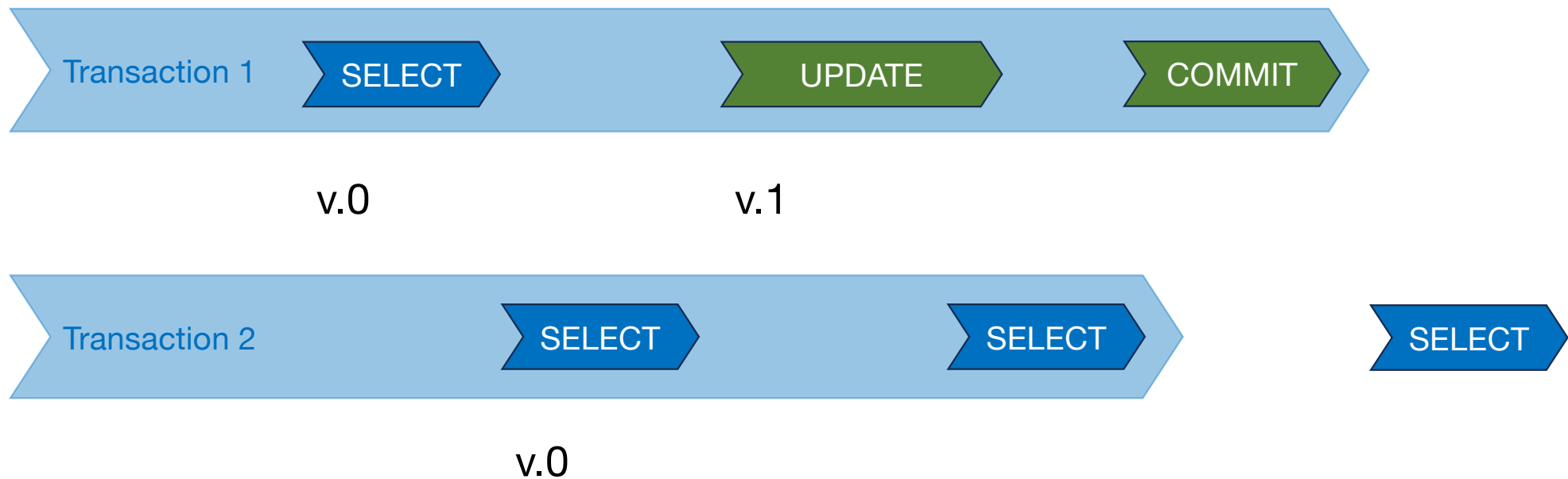
Snapshot isolation



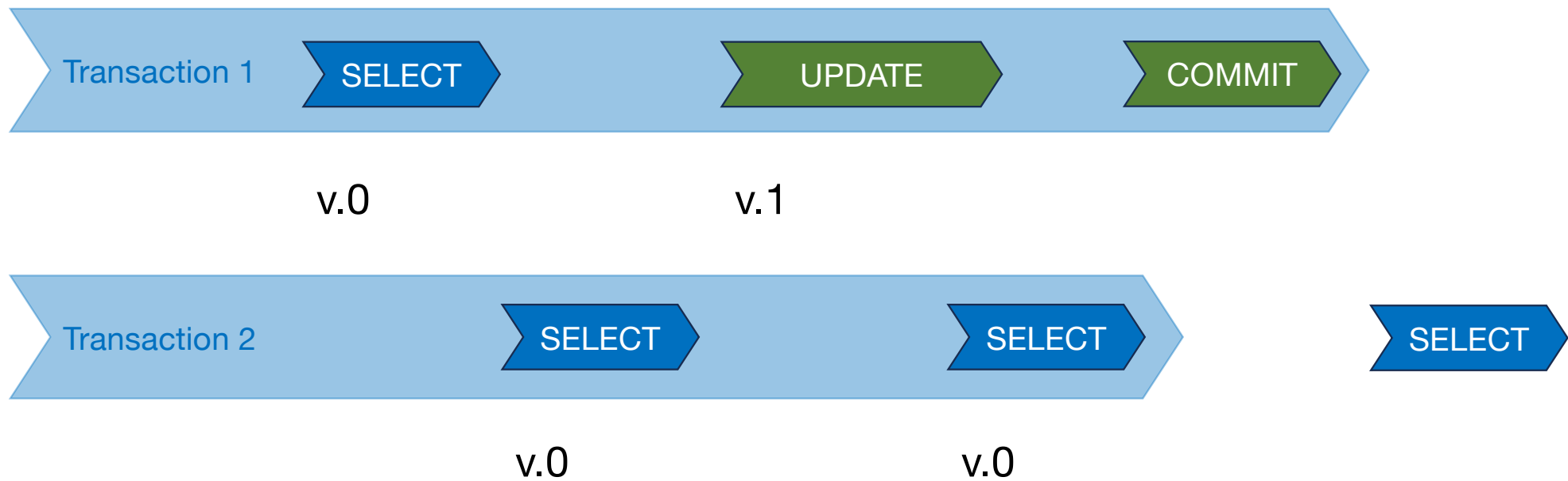
Snapshot isolation



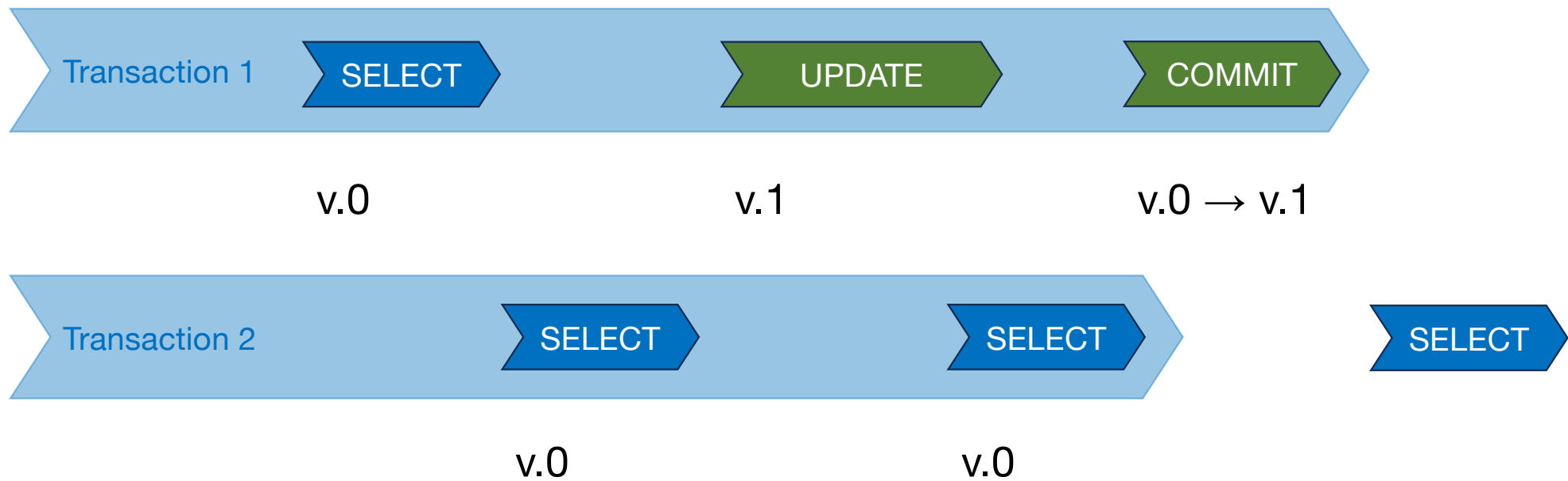
Snapshot isolation



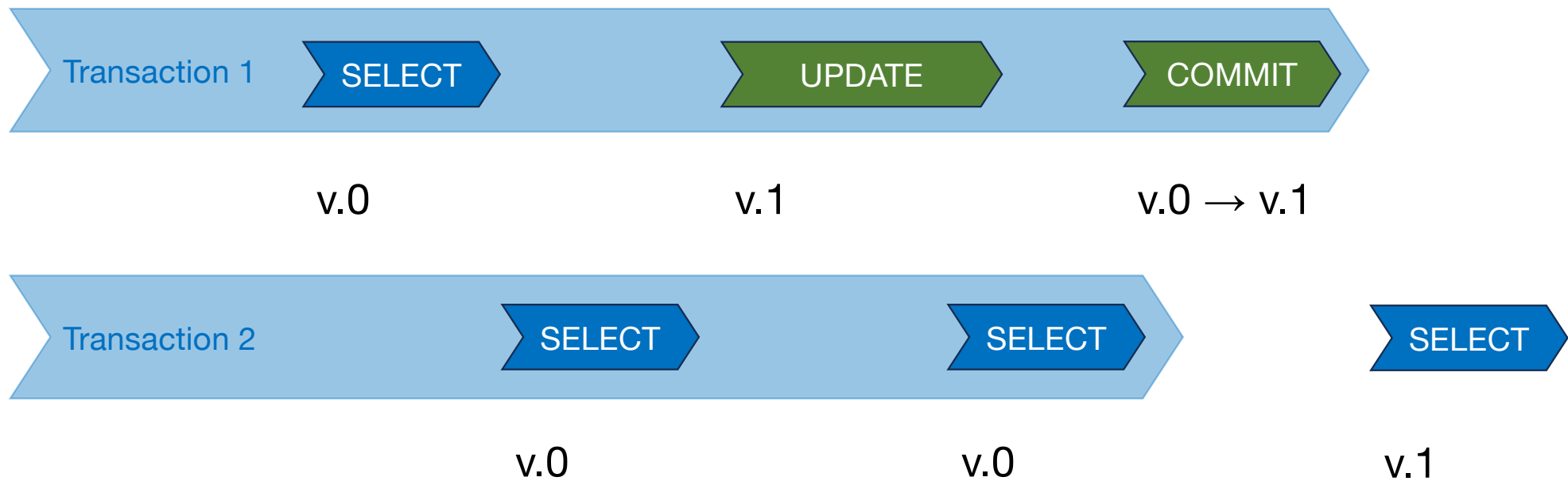
Snapshot isolation



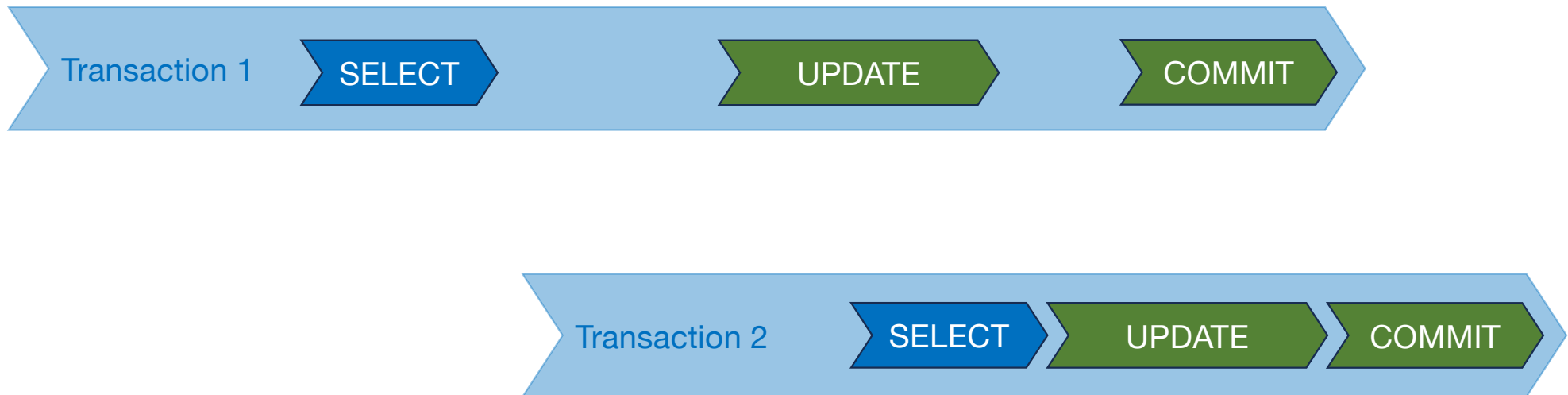
Snapshot isolation



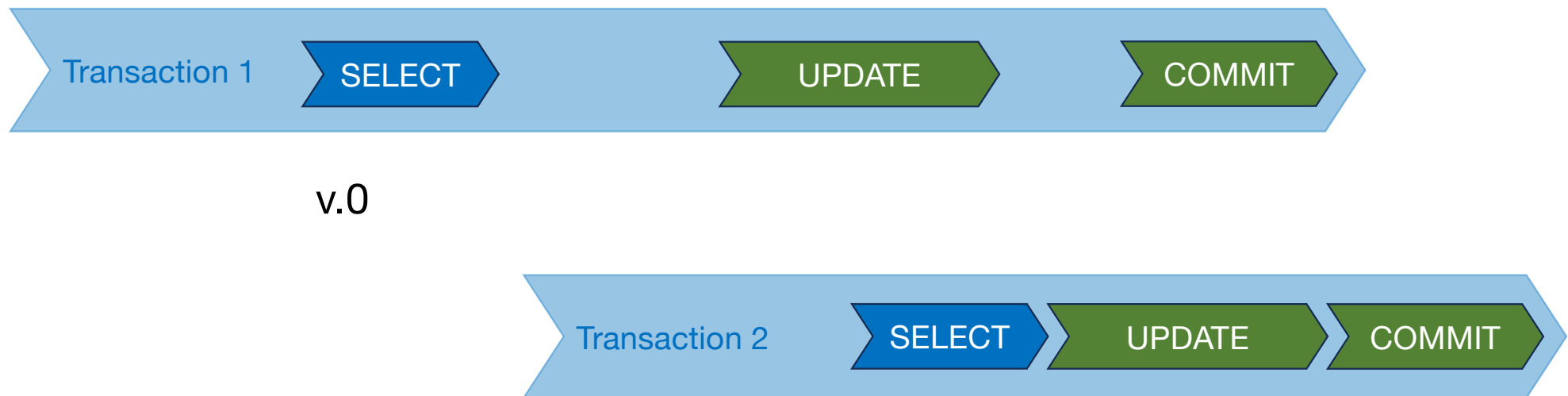
Snapshot isolation



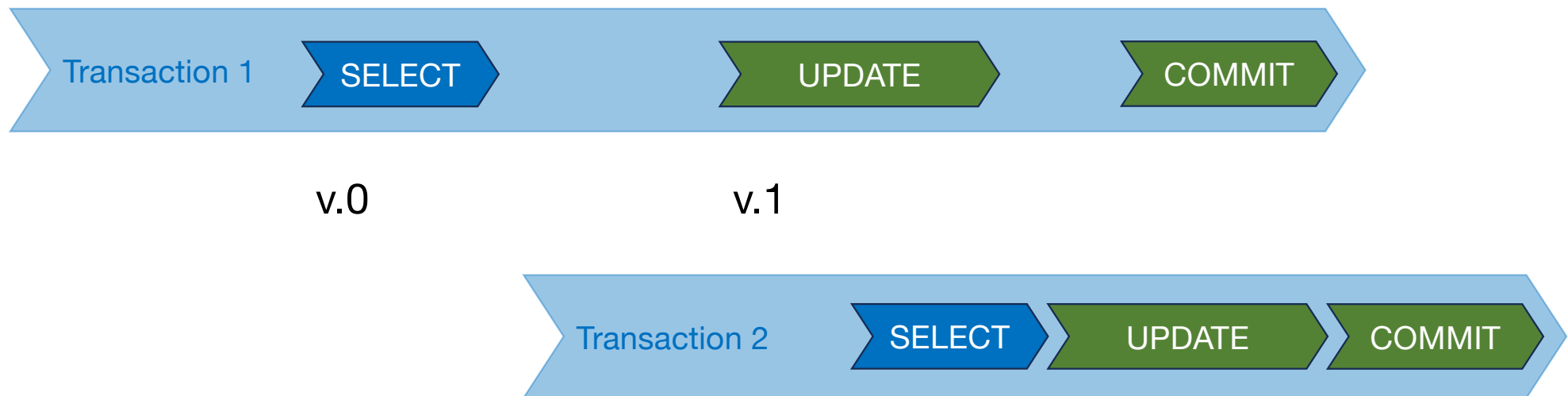
Snapshot isolation: Update conflict



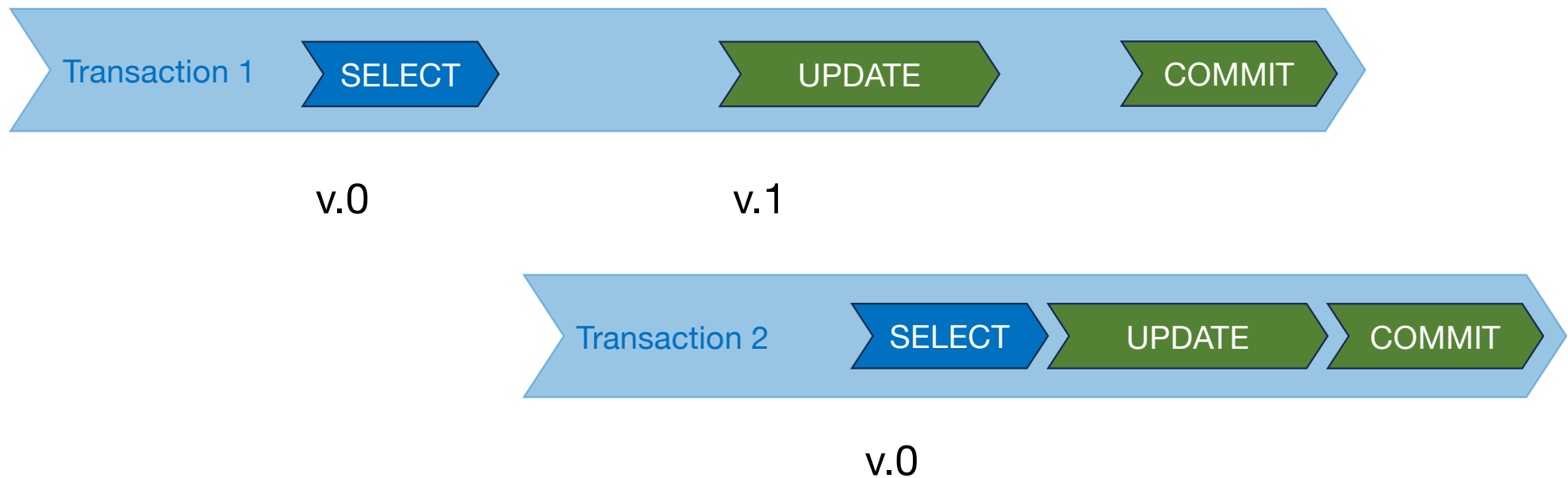
Snapshot isolation: Update conflict



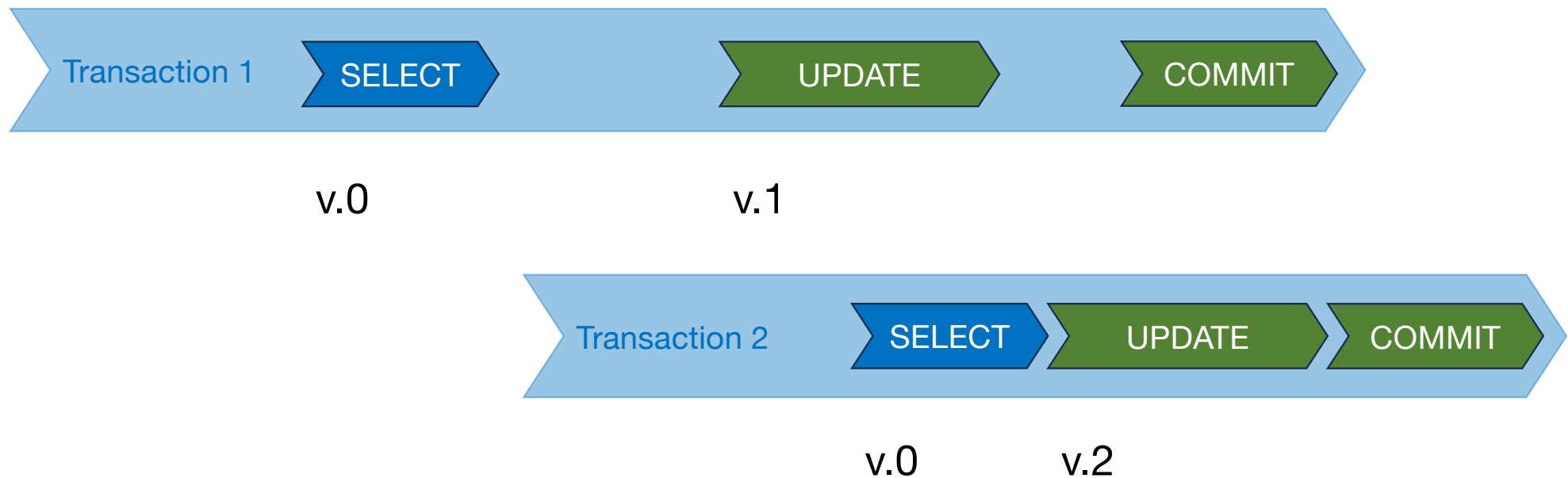
Snapshot isolation: Update conflict



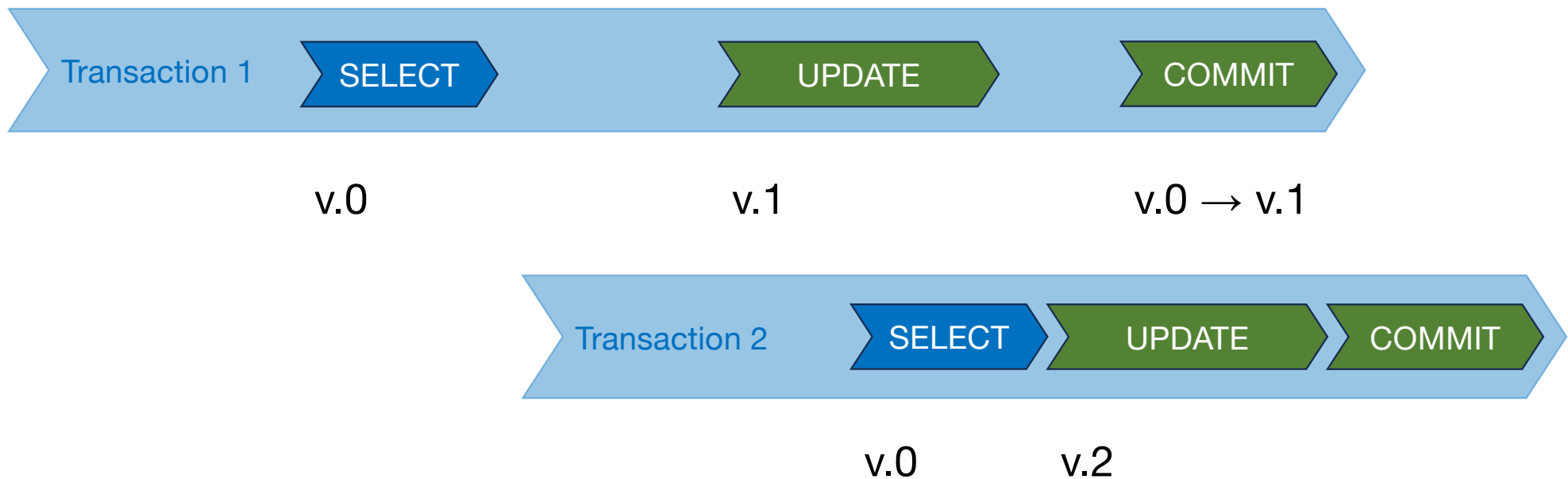
Snapshot isolation: Update conflict



Snapshot isolation: Update conflict



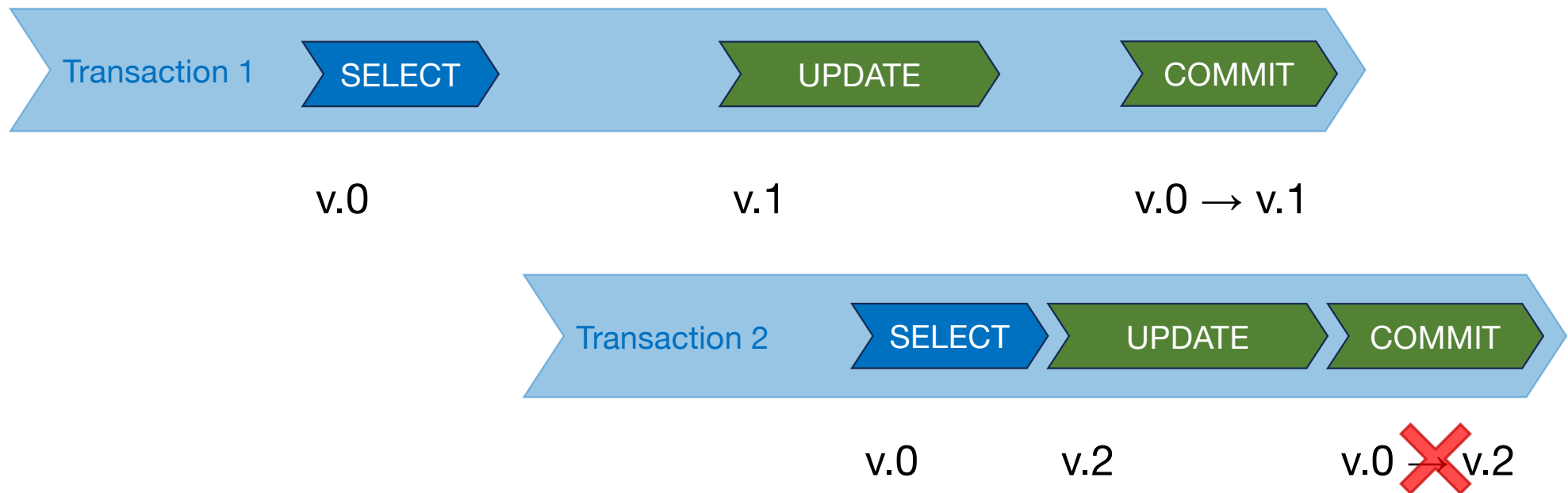
Snapshot isolation: Update conflict



Snapshot isolation: Update conflict



Snapshot isolation: Update conflict





DEMO

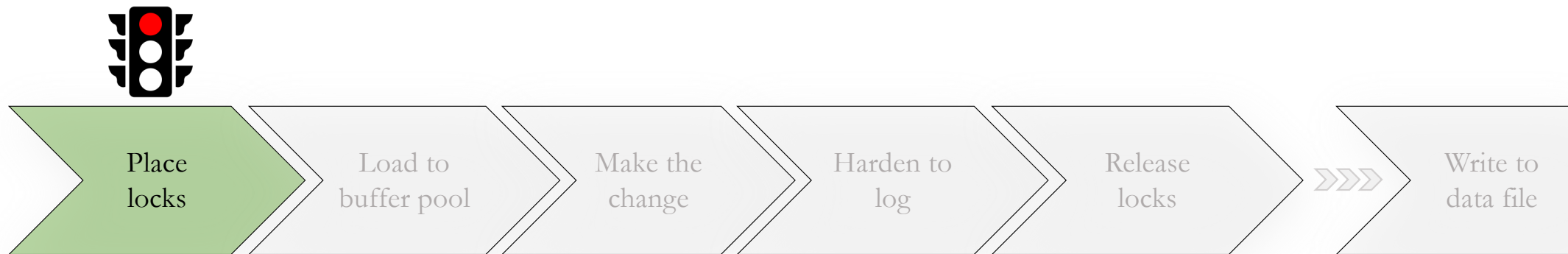
Read Committed Snapshot Isolation

vs. Snapshot Isolation

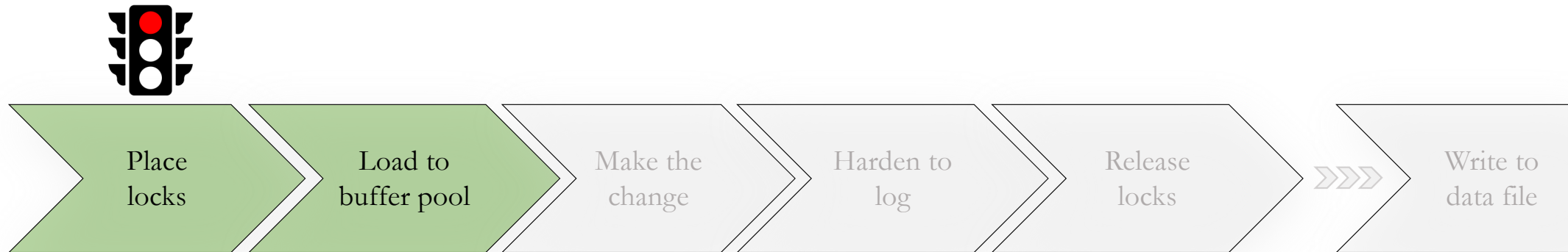
- Snapshot Isolation is set per *transaction*.
- Snapshot Isolation protects the *transaction*.
- Snapshot Isolation requires code change.
- Read Committed Snapshot Isolation is a *database* setting.
- Read Committed Snapshot Isolation protects the *statement*.
- Both require testing, because they behave differently.

More detail: <https://brentozar.com/go/rcsi>

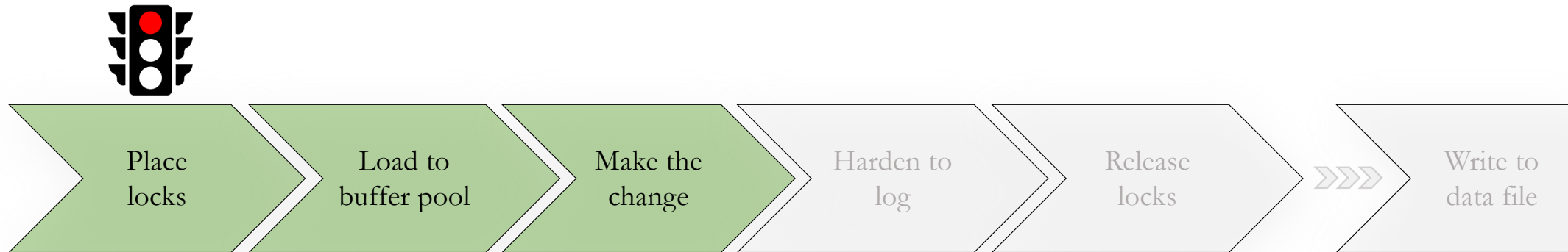
How the transaction log works



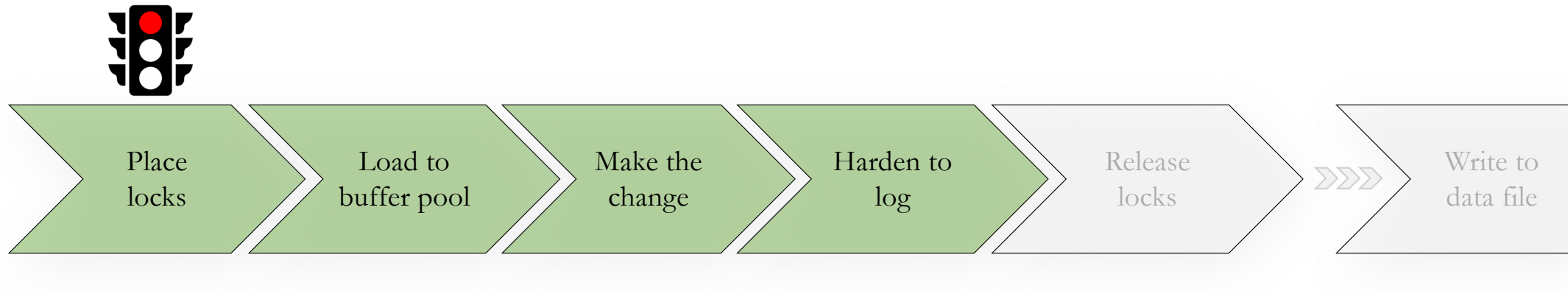
How the transaction log works



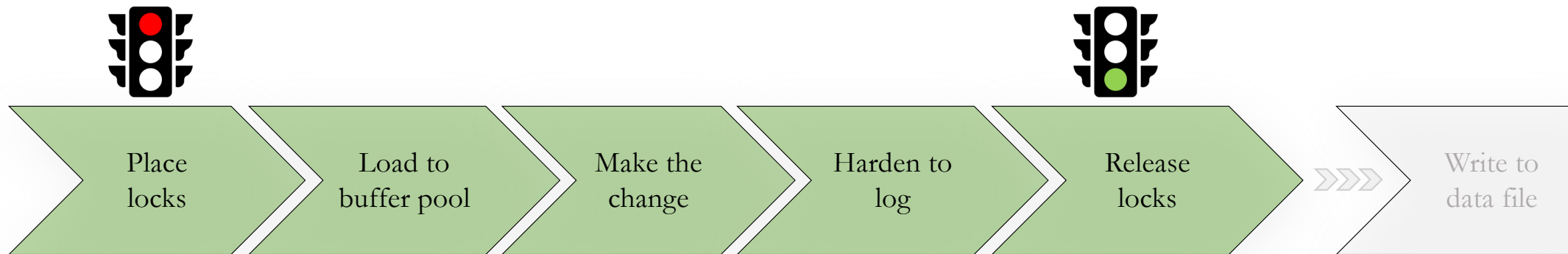
How the transaction log works



How the transaction log works



How the transaction log works



How the transaction log works



Durability



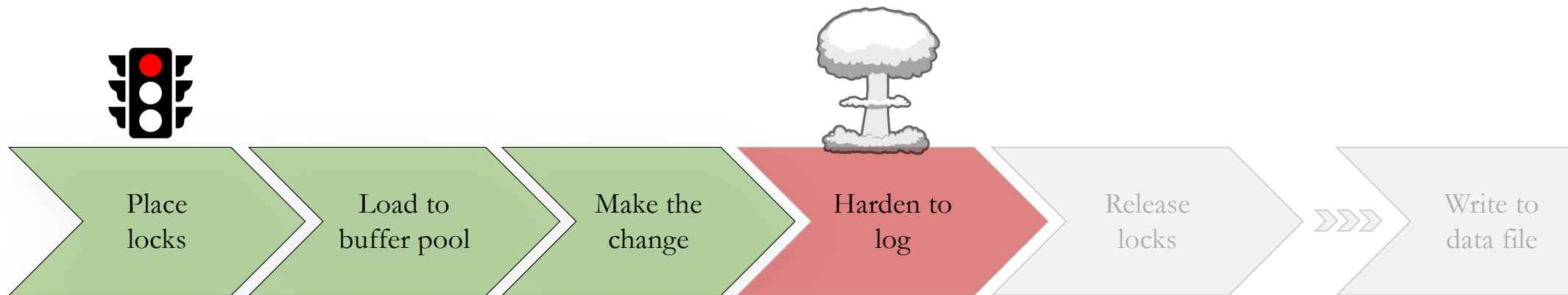
Durability



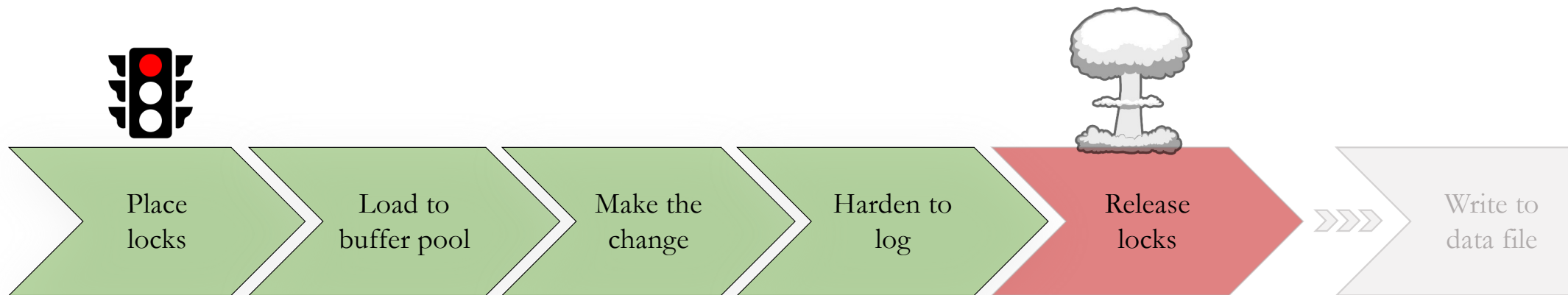
Durability



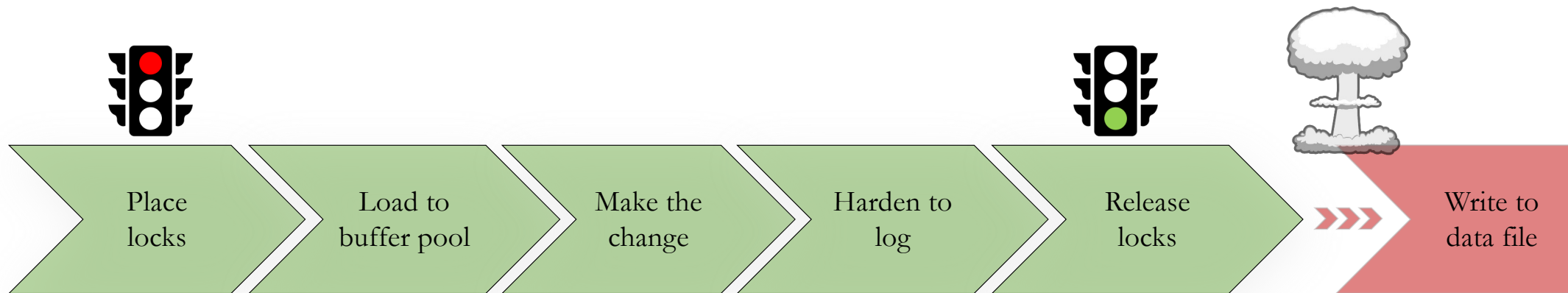
Durability



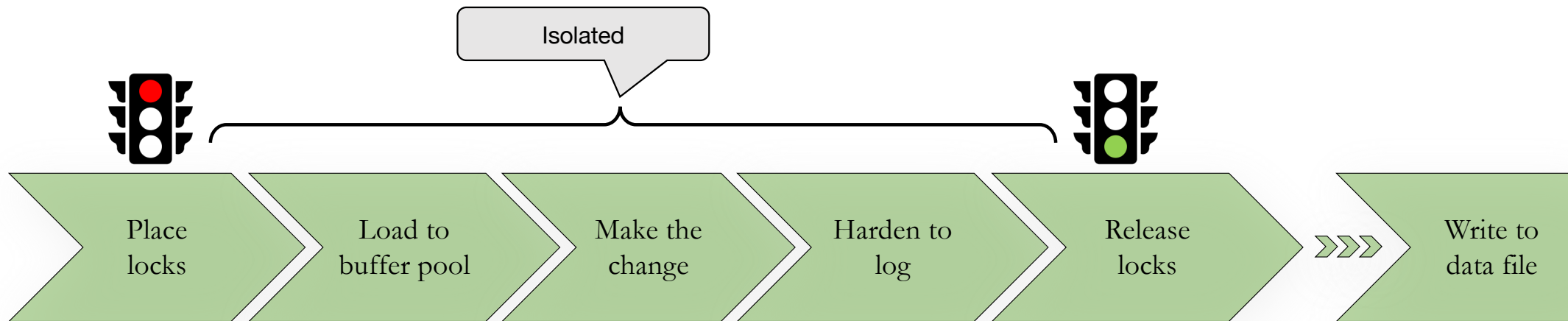
Durability



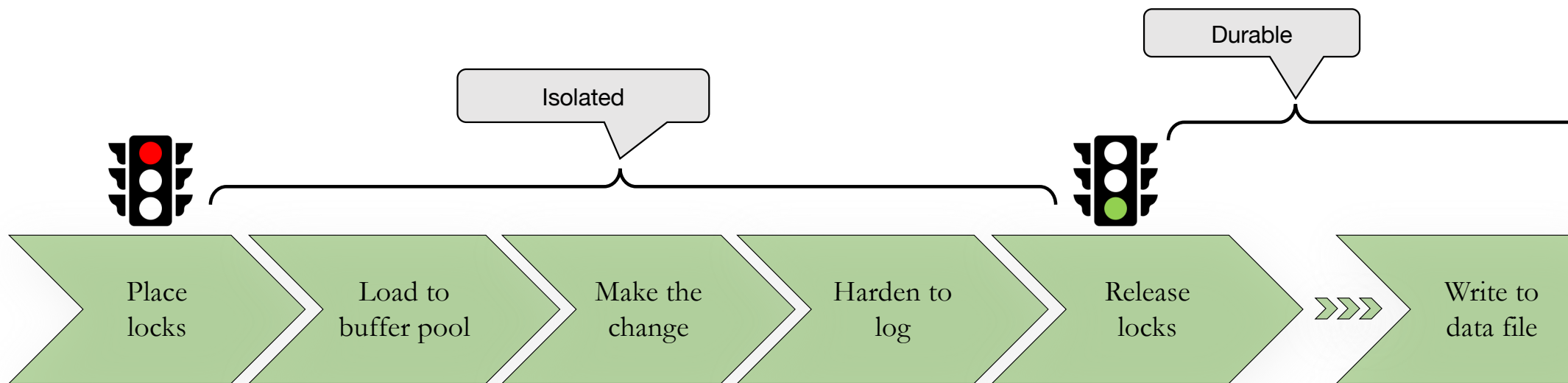
Durability



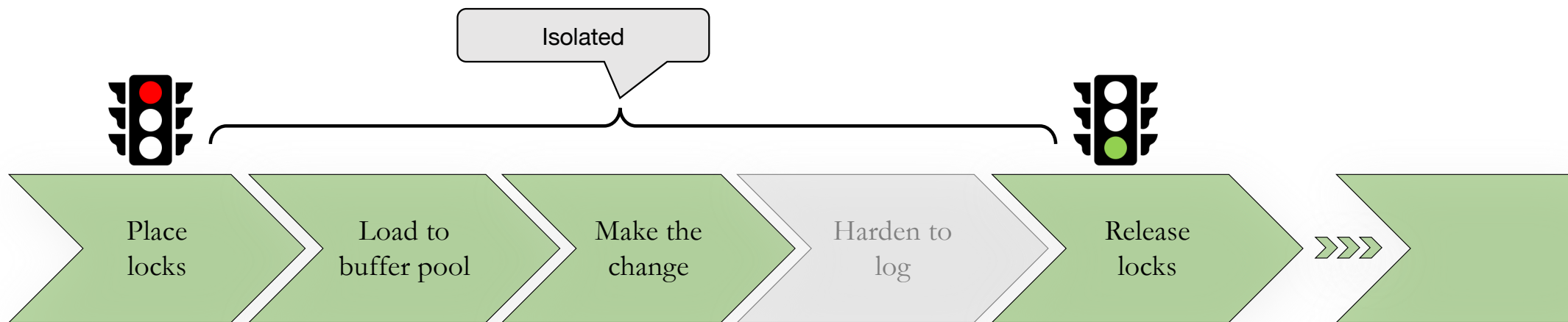
Durability



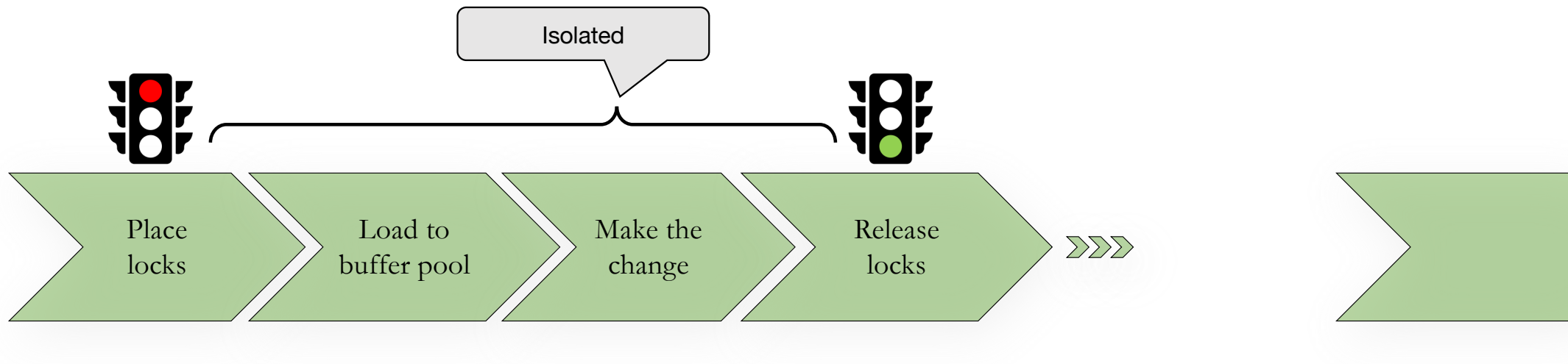
Durability



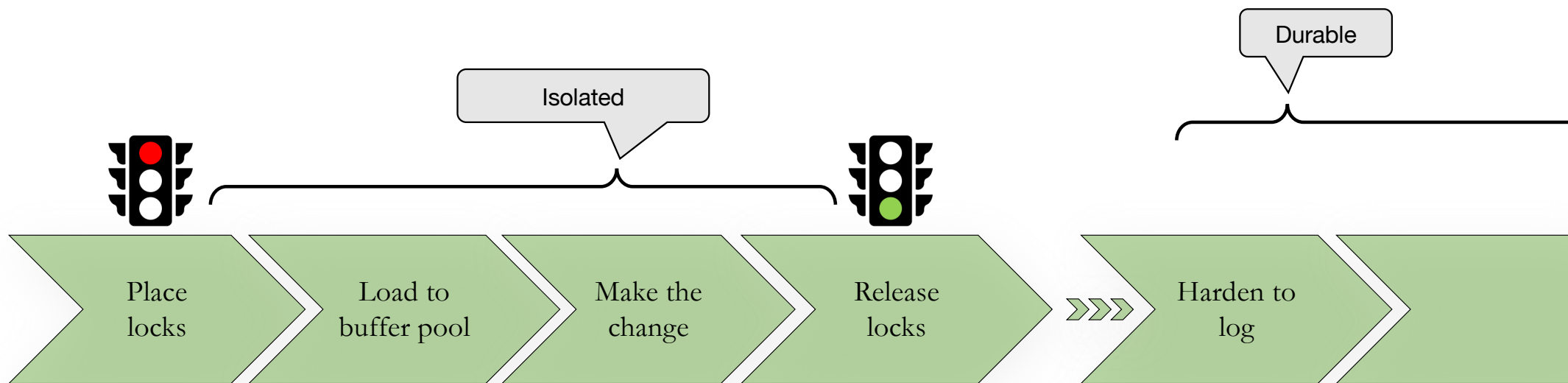
Delayed durability



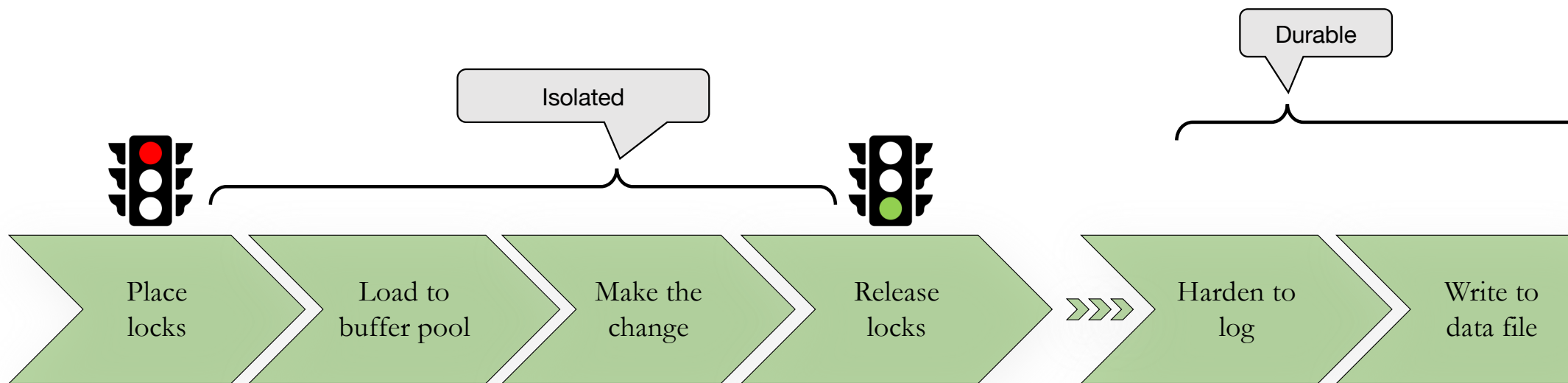
Delayed durability



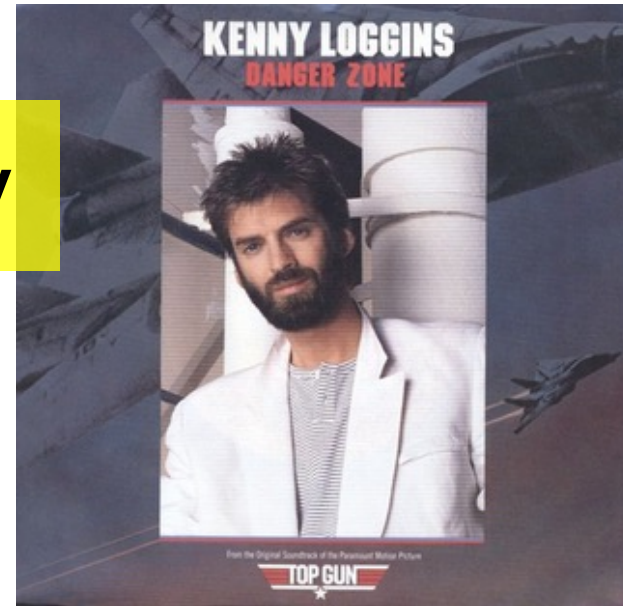
Delayed durability



Delayed durability

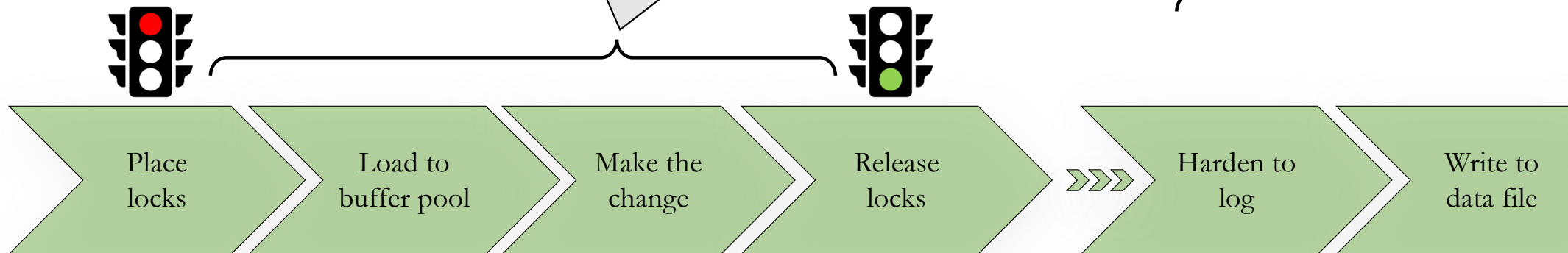


Delayed durability



Isolated

Durable



Delayed durability

- Writes are batched into the transaction log
- Significantly reduced latency for tiny transactions
- Great if you don't mind losing the data – like a staging environment
- tempdb uses a form of Delayed Durability under the hood

Key take-aways

- I was going to do this slide later, but here we are.

Give me feedback. Please?

