Spanner pt. 2

Doug Woos

Logistics notes

Problem Set 4 out tonight

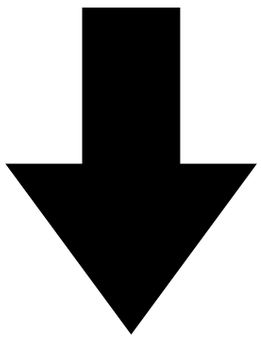
Spanner pt 2

1. Distributed transactions, in detail
    - On one Paxos group
    - Between Paxos groups
2. Fast read-only transactions with TrueTime
3. Discussion

Leader

Client

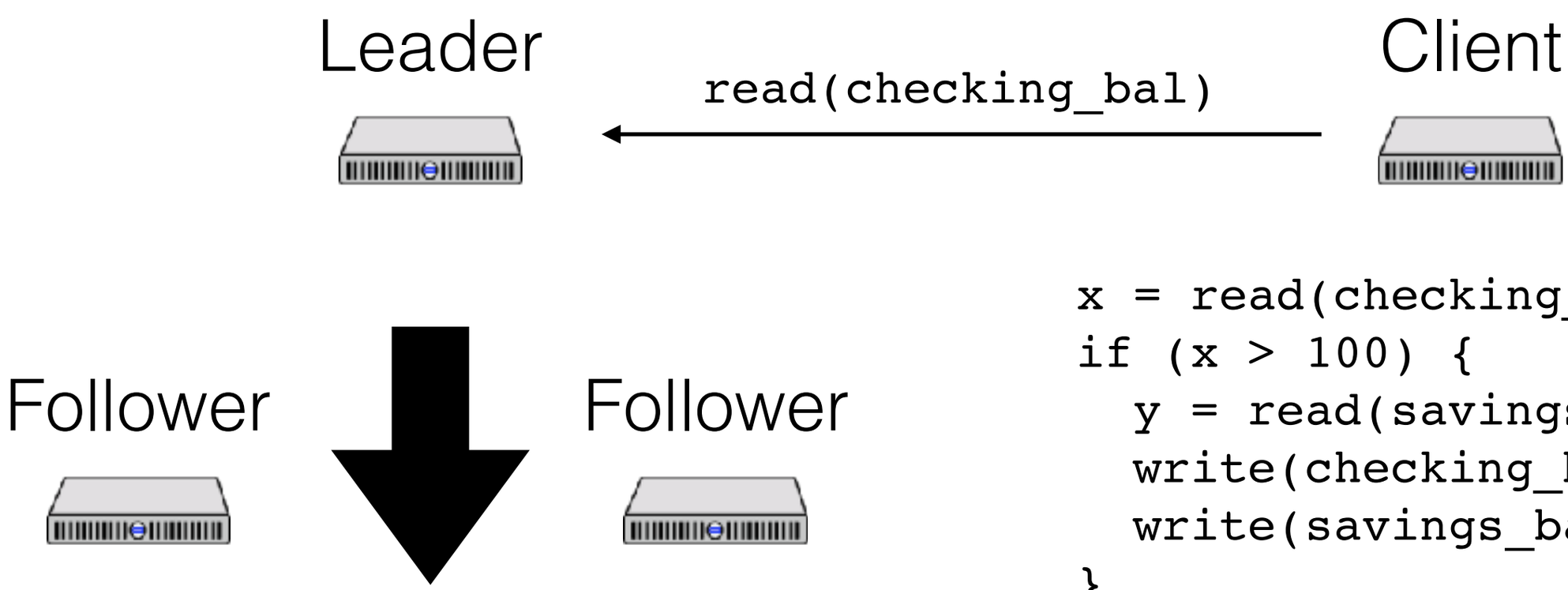
Follower



Follower

```
x = read(checking_bal)
if (x > 100) {
  y = read(savings_bal)
  write(checking_bal, x - 100)
  write(savings_bal, y + 100)
}
```

```
LOG:
  1.
  2.
  3.
  4.
  5.
  6.
  7.
```

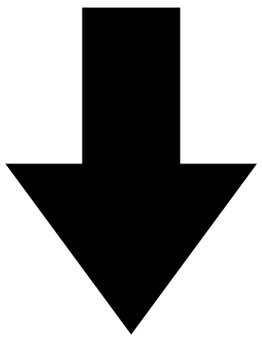



```
x = read(checking_bal)
if (x > 100) {
  y = read(savings_bal)
  write(checking_bal, x - 100)
  write(savings_bal, y + 100)
}
```

```
LOG:
  1.
  2.
  3.
  4.
  5.
  6.
  7.
```

Leader

Follower

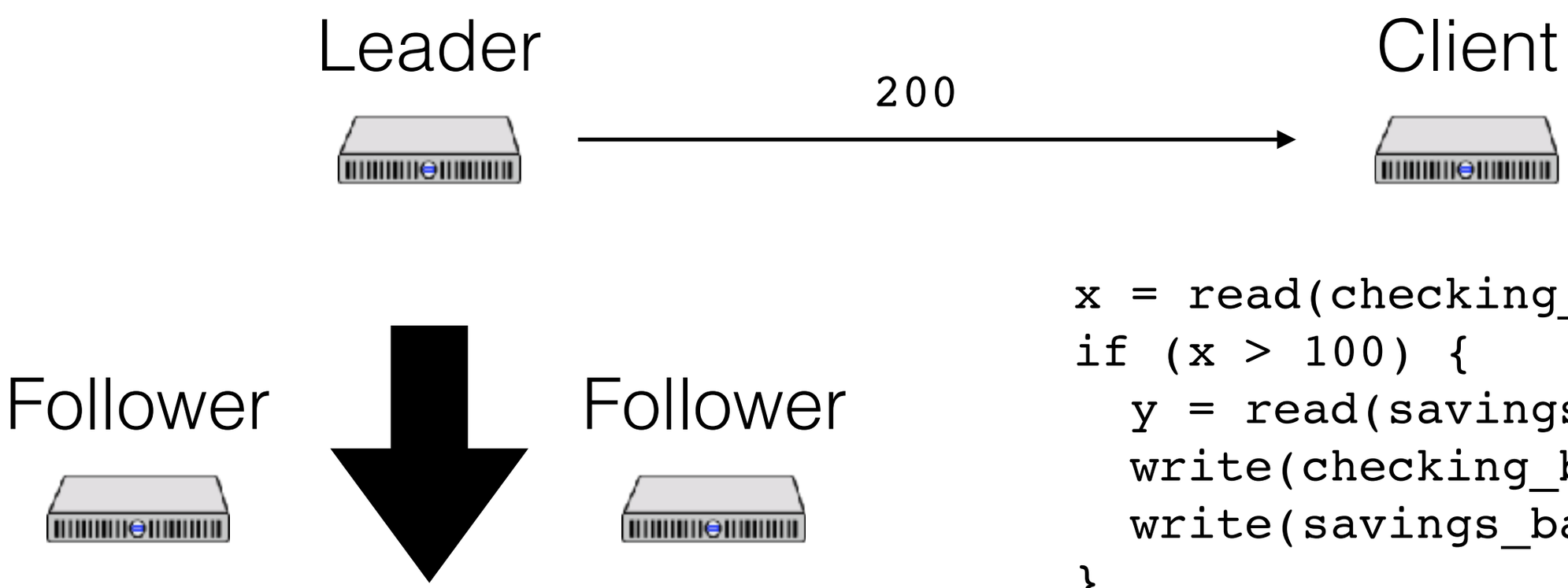


Follower

Client

```
x = read(checking_bal)
if (x > 100) {
  y = read(savings_bal)
  write(checking_bal, x - 100)
  write(savings_bal, y + 100)
}
```

```
LOG:
 1. A: Lock(R, checking_bal)
 2.
 3.
 4.
 5.
 6.
 7.
```

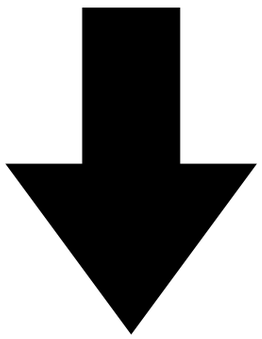



```
x = read(checking_bal)
if (x > 100) {
  y = read(savings_bal)
  write(checking_bal, x - 100)
  write(savings_bal, y + 100)
}
```

```
LOG:
 1. A: Lock(R, checking_bal)
 2.
 3.
 4.
 5.
 6.
 7.
```

Leader

Follower

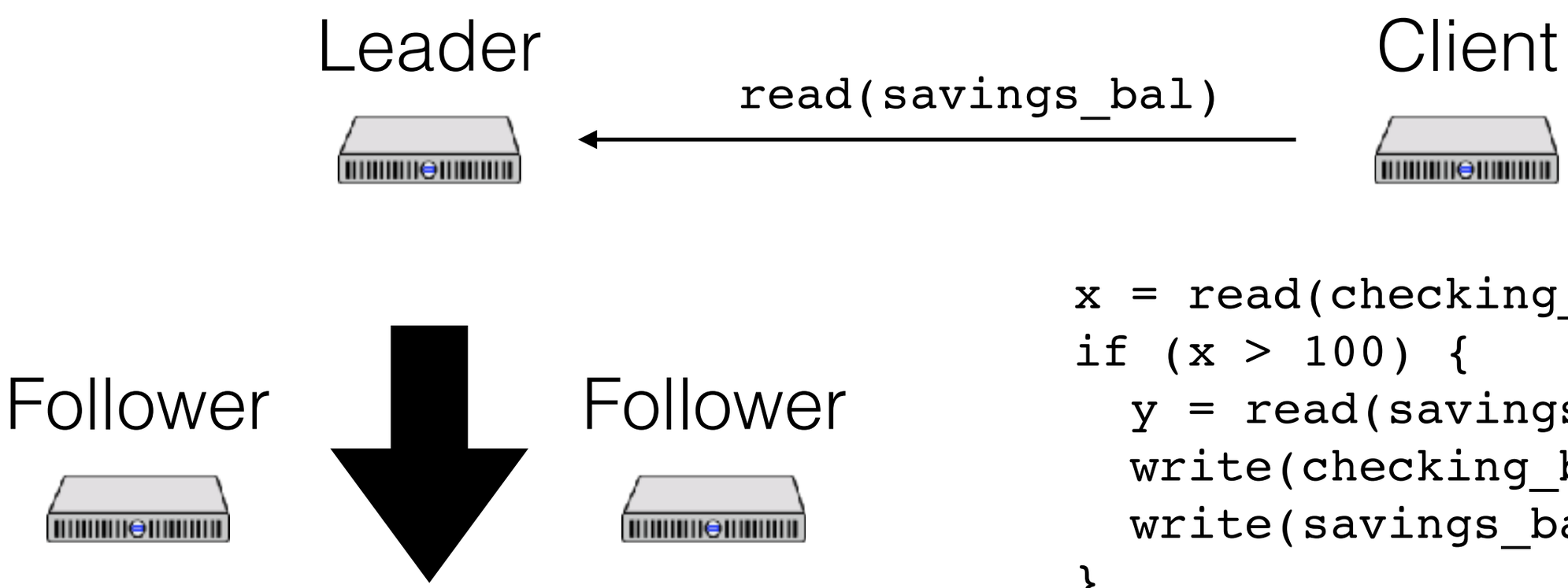


Follower

Client

```
x = read(checking_bal)
if (x > 100) {
  y = read(savings_bal)
  write(checking_bal, x - 100)
  write(savings_bal, y + 100)
}
```

```
LOG:
 1. A: Lock(R, checking_bal)
 2.
 3.
 4.
 5.
 6.
 7.
```



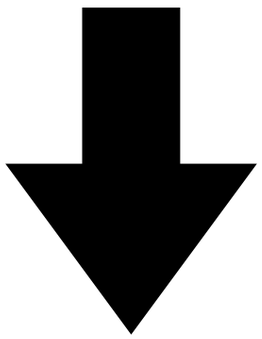

```
x = read(checking_bal)
if (x > 100) {
  y = read(savings_bal)
  write(checking_bal, x - 100)
  write(savings_bal, y + 100)
}
```

```
LOG:
 1. A: Lock(R, checking_bal)
 2.
 3.
 4.
 5.
 6.
 7.
```

Leader

Client

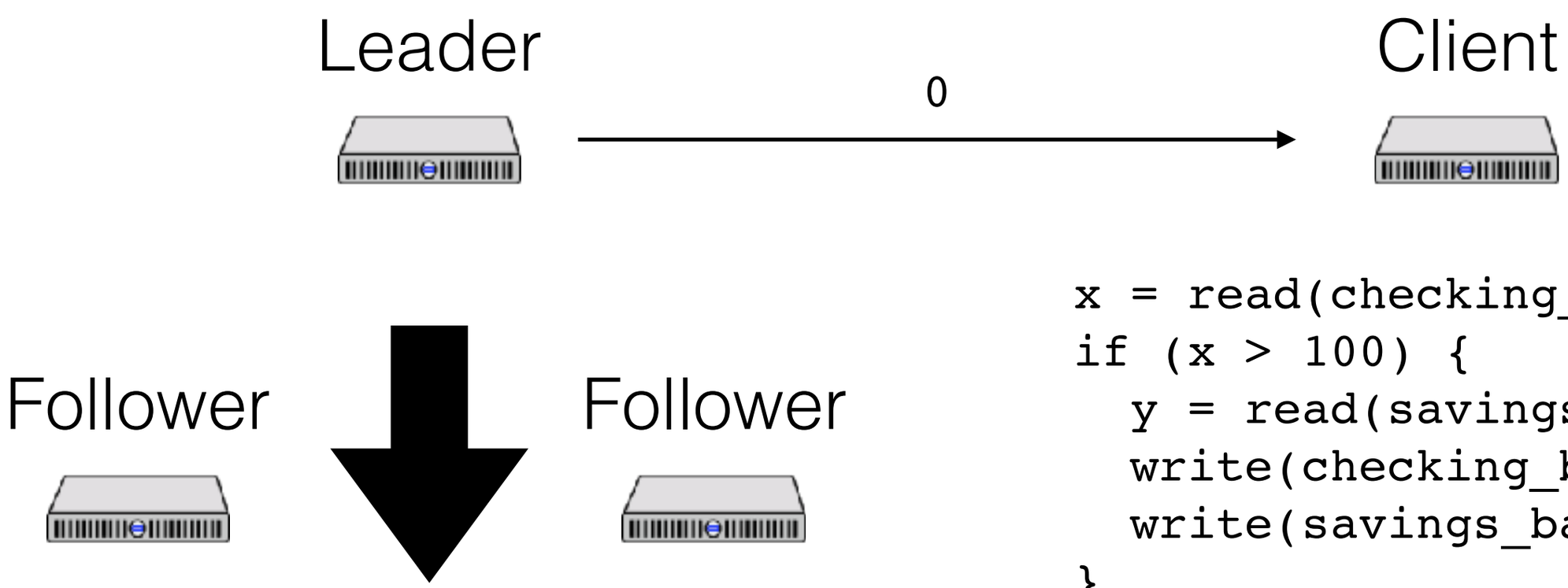
Follower



Follower

```
x = read(checking_bal)
if (x > 100) {
  y = read(savings_bal)
  write(checking_bal, x - 100)
  write(savings_bal, y + 100)
}
```

```
LOG:
 1. A: Lock(R, checking_bal)
 2. A: Lock(R, savings_bal)
 3.
 4.
 5.
 6.
 7.
```



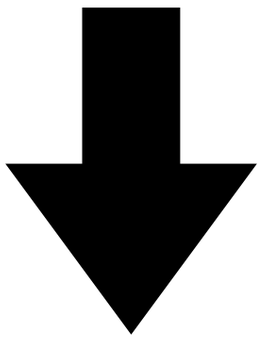

```
x = read(checking_bal)
if (x > 100) {
  y = read(savings_bal)
  write(checking_bal, x - 100)
  write(savings_bal, y + 100)
}
```

```
LOG:
 1. A: Lock(R, checking_bal)
 2. A: Lock(R, savings_bal)
 3.
 4.
 5.
 6.
 7.
```

Leader

Client

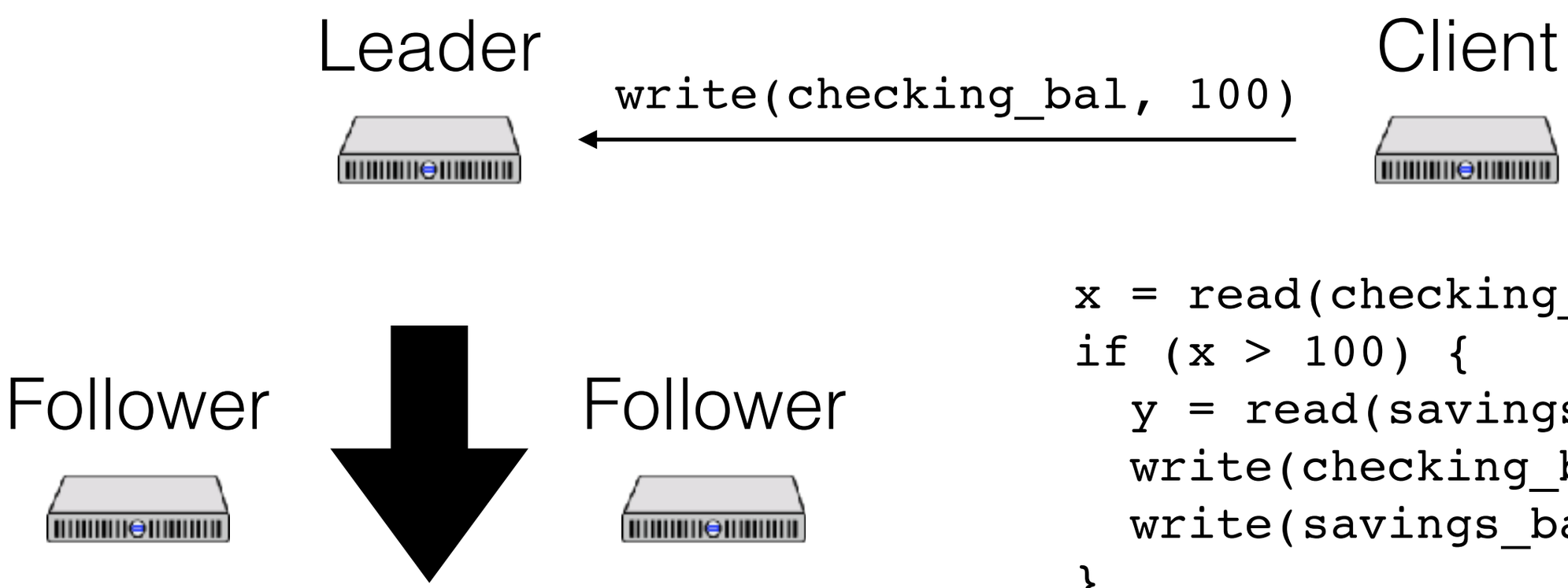
Follower



Follower

```
x = read(checking_bal)
if (x > 100) {
  y = read(savings_bal)
  write(checking_bal, x - 100)
  write(savings_bal, y + 100)
}
```

```
LOG:
 1. A: Lock(R, checking_bal)
 2. A: Lock(R, savings_bal)
 3.
 4.
 5.
 6.
 7.
```



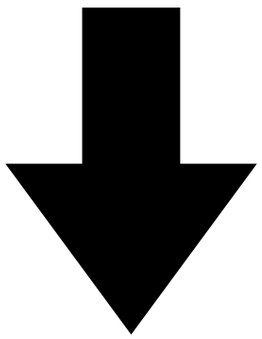

```
x = read(checking_bal)
if (x > 100) {
  y = read(savings_bal)
  write(checking_bal, x - 100)
  write(savings_bal, y + 100)
}
```

```
LOG:
 1. A: Lock(R, checking_bal)
 2. A: Lock(R, savings_bal)
 3.
 4.
 5.
 6.
 7.
```

Leader

Client

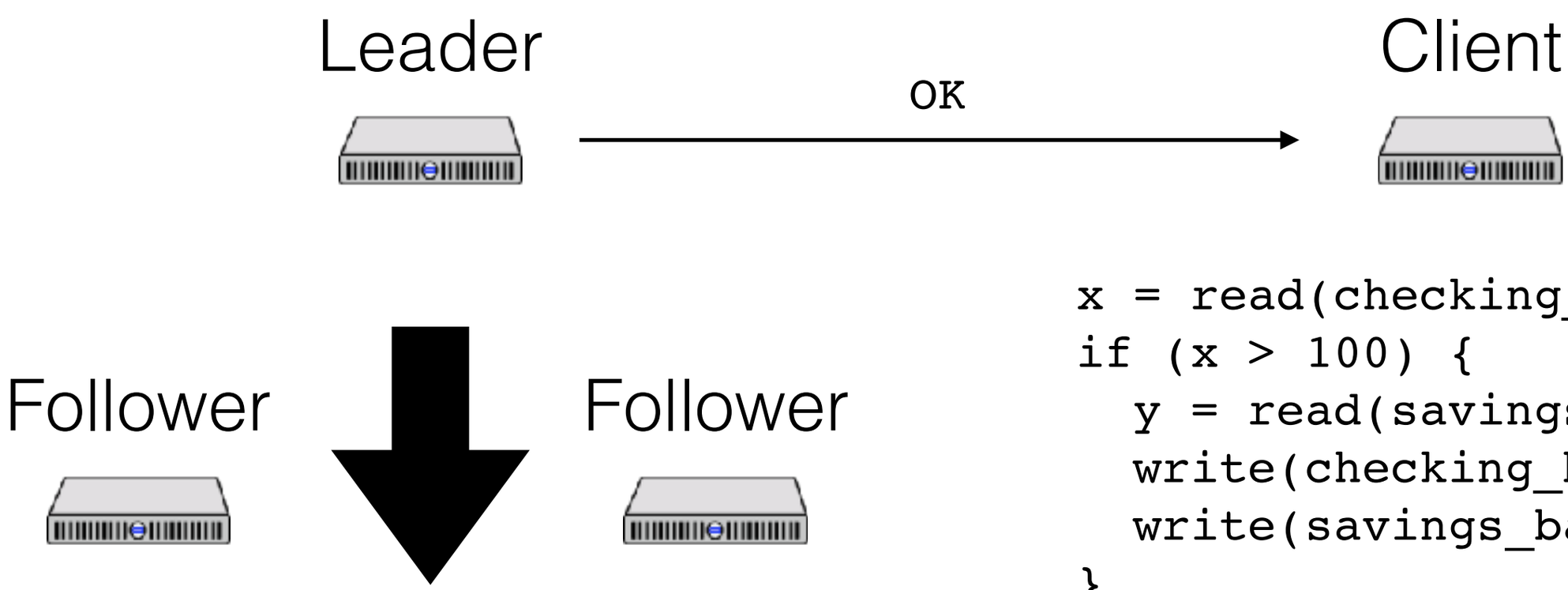
Follower



Follower

```
x = read(checking_bal)
if (x > 100) {
  y = read(savings_bal)
  write(checking_bal, x - 100)
  write(savings_bal, y + 100)
}
```

```
LOG:
 1. A: Lock(R, checking_bal)
 2. A: Lock(R, savings_bal)
 3. A: Lock(W, checking_bal)
 4. A: Write(checking_bal, 100)
 5.
 6.
 7.
```



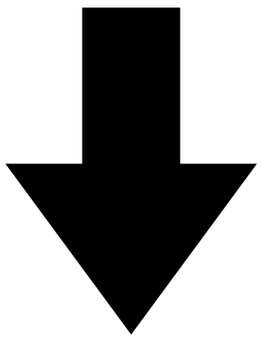

```
x = read(checking_bal)
if (x > 100) {
  y = read(savings_bal)
  write(checking_bal, x - 100)
  write(savings_bal, y + 100)
}
```

```
LOG:
 1. A: Lock(R, checking_bal)
 2. A: Lock(R, savings_bal)
 3. A: Lock(W, checking_bal)
 4. A: Write(checking_bal, 100)
 5.
 6.
 7.
```

Leader

Client

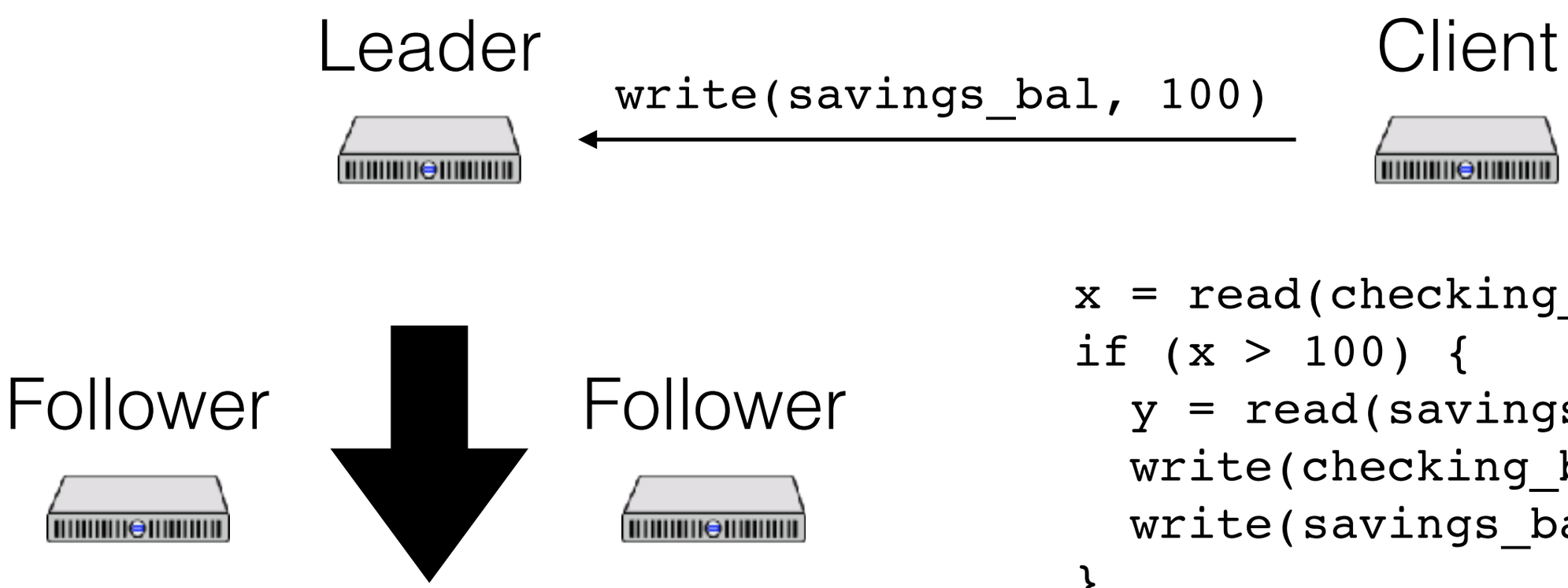
Follower



Follower

```
x = read(checking_bal)
if (x > 100) {
  y = read(savings_bal)
  write(checking_bal, x - 100)
  write(savings_bal, y + 100)
}
```

```
LOG:
 1. A: Lock(R, checking_bal)
 2. A: Lock(R, savings_bal)
 3. A: Lock(W, checking_bal)
 4. A: Write(checking_bal, 100)
 5.
 6.
 7.
```

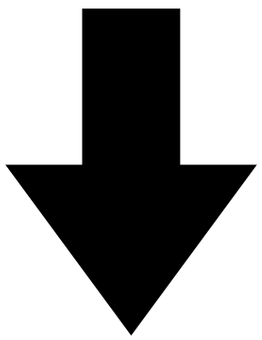



```
x = read(checking_bal)
if (x > 100) {
  y = read(savings_bal)
  write(checking_bal, x - 100)
  write(savings_bal, y + 100)
}
```

```
LOG:
 1. A: Lock(R, checking_bal)
 2. A: Lock(R, savings_bal)
 3. A: Lock(W, checking_bal)
 4. A: Write(checking_bal, 100)
 5.
 6.
 7.
```

Leader

Follower

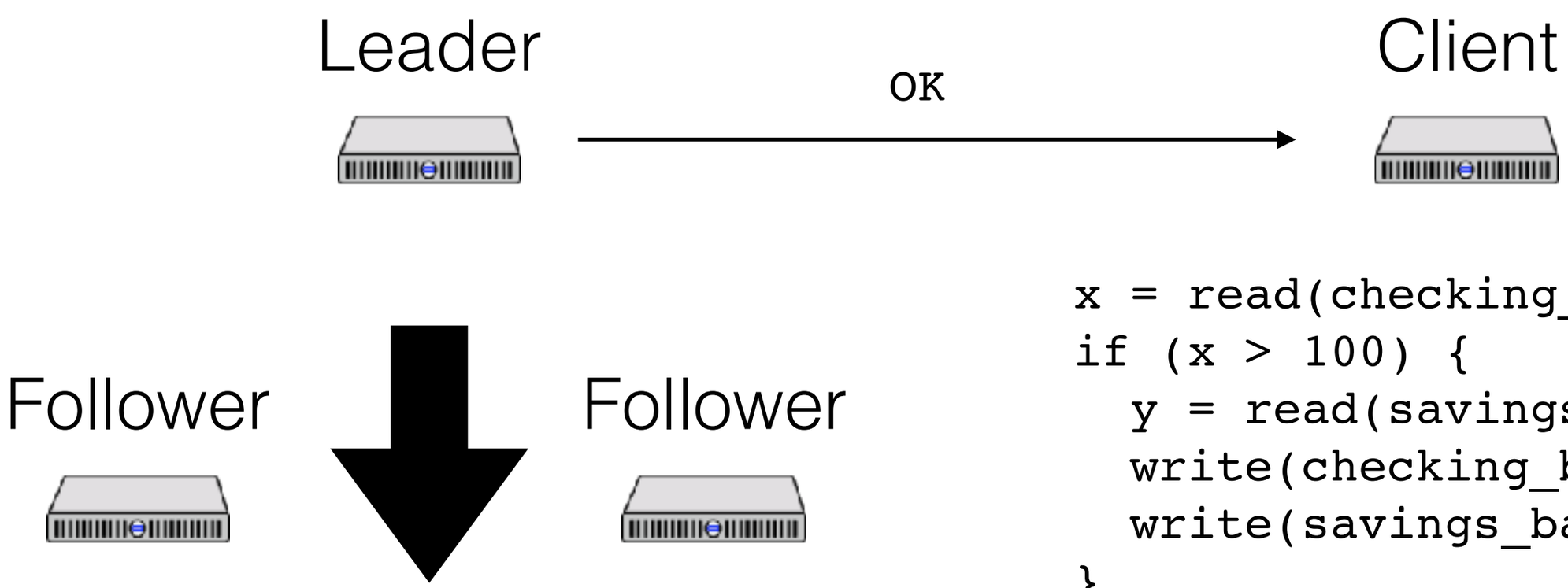


Follower

Client

```
x = read(checking_bal)
if (x > 100) {
  y = read(savings_bal)
  write(checking_bal, x - 100)
  write(savings_bal, y + 100)
}
```

```
LOG:
 1. A: Lock(R, checking_bal)
 2. A: Lock(R, savings_bal)
 3. A: Lock(W, checking_bal)
 4. A: Write(checking_bal, 100)
 5. A: Lock(W, savings_bal)
 6. A: Write(savings_bal, 100)
 7.
```

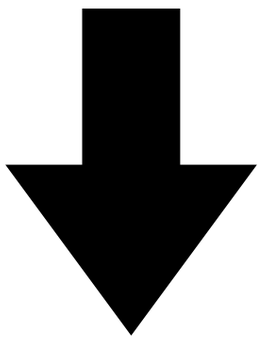



```
x = read(checking_bal)
if (x > 100) {
  y = read(savings_bal)
  write(checking_bal, x - 100)
  write(savings_bal, y + 100)
}
```

```
LOG:
 1. A: Lock(R, checking_bal)
 2. A: Lock(R, savings_bal)
 3. A: Lock(W, checking_bal)
 4. A: Write(checking_bal, 100)
 5. A: Lock(W, savings_bal)
 6. A: Write(savings_bal, 100)
 7.
```

Leader

Follower

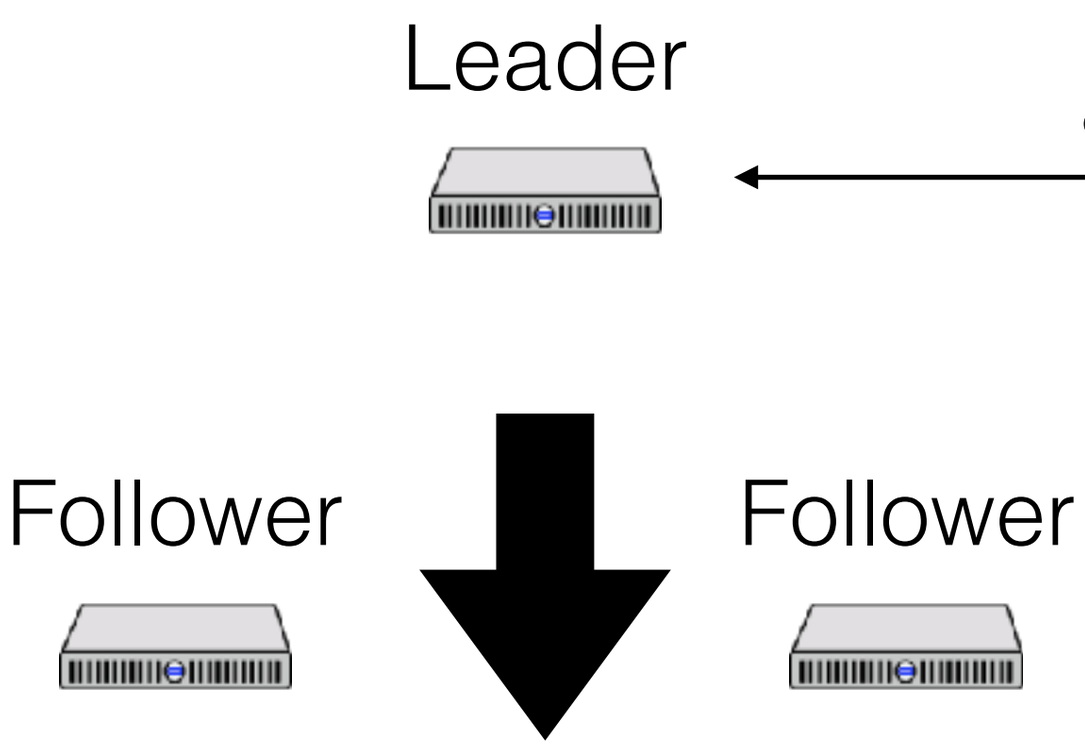


Follower

Client

```
x = read(checking_bal)
if (x > 100) {
  y = read(savings_bal)
  write(checking_bal, x - 100)
  write(savings_bal, y + 100)
}
```

```
LOG:
 1. A: Lock(R, checking_bal)
 2. A: Lock(R, savings_bal)
 3. A: Lock(W, checking_bal)
 4. A: Write(checking_bal, 100)
 5. A: Lock(W, savings_bal)
 6. A: Write(savings_bal, 100)
 7.
```



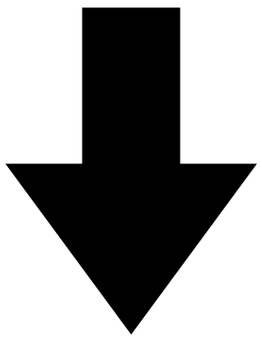

```
x = read(checking_bal)
if (x > 100) {
  y = read(savings_bal)
  write(checking_bal, x - 100)
  write(savings_bal, y + 100)
}
```

```
LOG:
 1. A: Lock(R, checking_bal)
 2. A: Lock(R, savings_bal)
 3. A: Lock(W, checking_bal)
 4. A: Write(checking_bal, 100)
 5. A: Lock(W, savings_bal)
 6. A: Write(savings_bal, 100)
 7.
```

Leader

Client

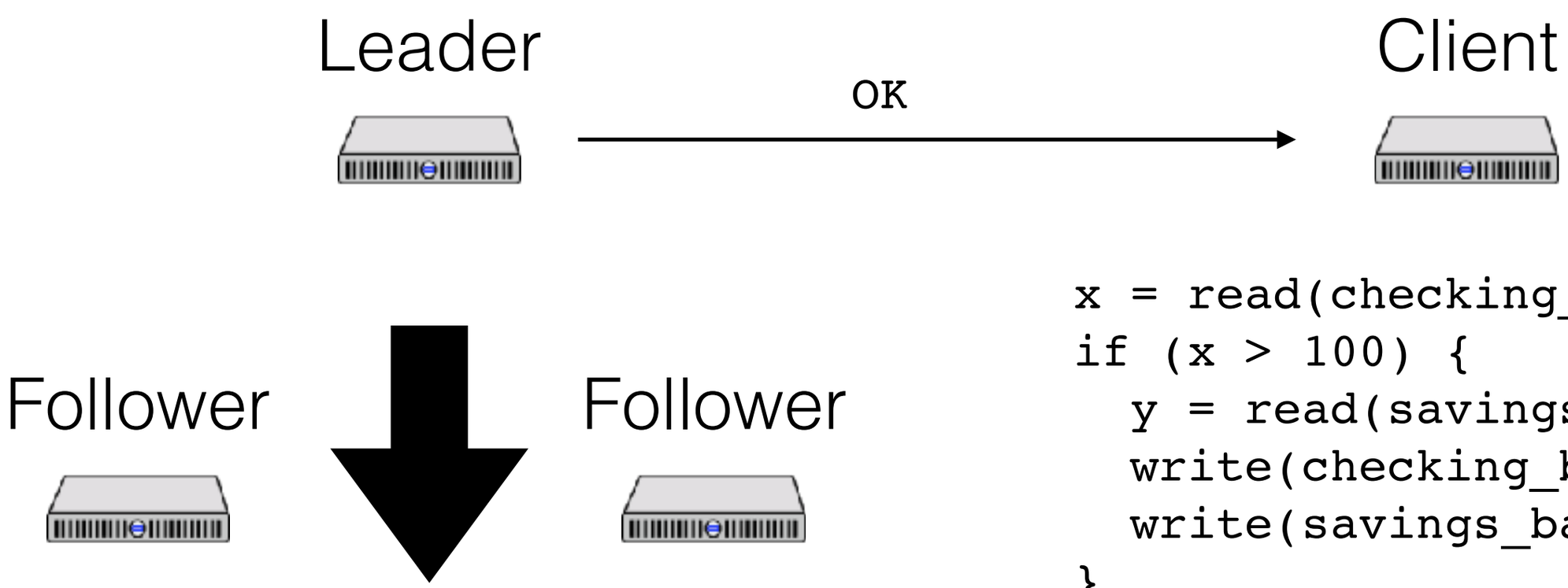
Follower



Follower

```
x = read(checking_bal)
if (x > 100) {
  y = read(savings_bal)
  write(checking_bal, x - 100)
  write(savings_bal, y + 100)
}
```

```
LOG:
 1. A: Lock(R, checking_bal)
 2. A: Lock(R, savings_bal)
 3. A: Lock(W, checking_bal)
 4. A: Write(checking_bal, 100)
 5. A: Lock(W, savings_bal)
 6. A: Write(savings_bal, 100)
 7. A: Commit
```



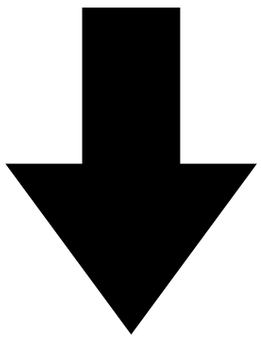

```
x = read(checking_bal)
if (x > 100) {
  y = read(savings_bal)
  write(checking_bal, x - 100)
  write(savings_bal, y + 100)
}
```

```
LOG:
 1. A: Lock(R, checking_bal)
 2. A: Lock(R, savings_bal)
 3. A: Lock(W, checking_bal)
 4. A: Write(checking_bal, 100)
 5. A: Lock(W, savings_bal)
 6. A: Write(savings_bal, 100)
 7. A: Commit
```

Leader

Client

Follower



Follower

```
x = read(checking_bal)
if (x > 100) {
  y = read(savings_bal)
  write(checking_bal, x - 100)
  write(savings_bal, y + 100)
}
```

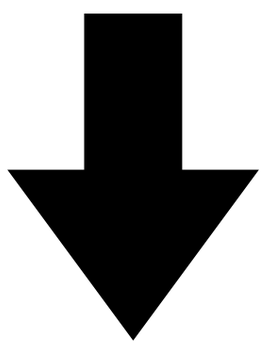
```
LOG:
 1. A: Lock(R, checking_bal)
 2. A: Lock(R, savings_bal)
 3. A: Lock(W, checking_bal)
 4. A: Write(checking_bal, 100)
 5. A: Lock(W, savings_bal)
 6. A: Write(savings_bal, 100)
 7. A: Commit
 8. A: Unlock(checking_bal)
 9. A: Unlock(savings_bal)
```

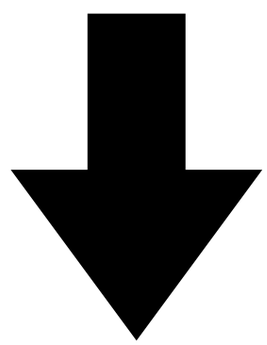
Leader 1

Client

Leader 2

```
x = read(checking_bal)
if (x > 100) {
  y = read(savings_bal)
  write(checking_bal, x - 100)
  write(savings_bal, y + 100)
}
```





LOG:

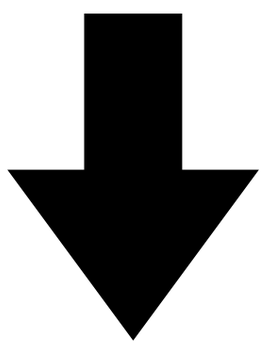
LOG:

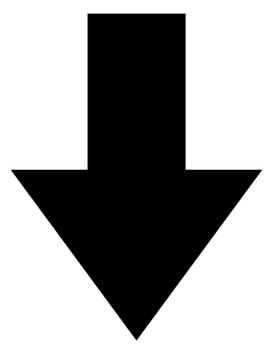
Leader 1

Client

Leader 2

```
x = read(checking_bal)
if (x > 100) {
  y = read(savings_bal)
  write(checking_bal, x - 100)
  write(savings_bal, y + 100)
}
```



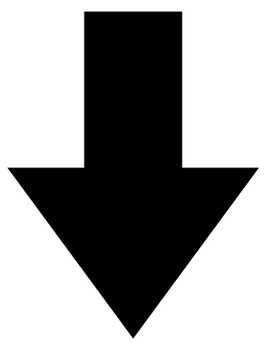


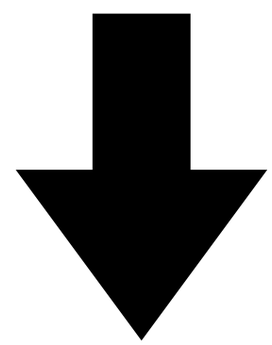
LOG:

LOG:

Leader 1 Client Leader 2

```
x = read(checking_bal)
if (x > 100) {
  y = read(savings_bal)
  write(checking_bal, x - 100)
  write(savings_bal, y + 100)
}
```





```
LOG:
 1. A: Lock(R, checking_bal)
```

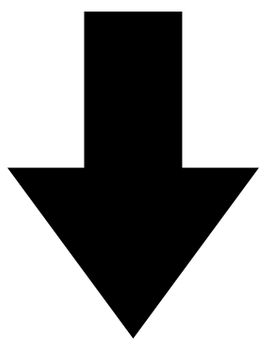
```
LOG:
```

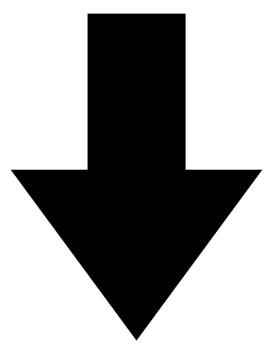
Leader 1

Client

Leader 2

```
x = read(checking_bal)
if (x > 100) {
  y = read(savings_bal)
  write(checking_bal, x - 100)
  write(savings_bal, y + 100)
}
```





```
LOG:
 1. A: Lock(R, checking_bal)
```

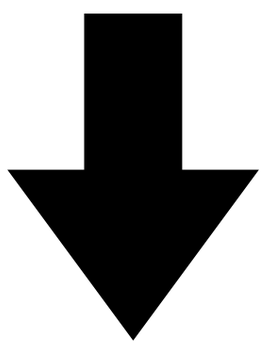
```
LOG:
```

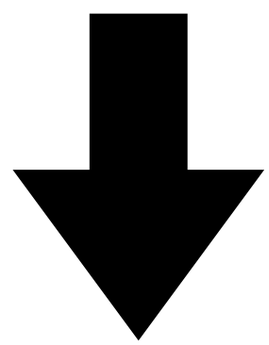
Leader 1

Client

Leader 2

```
x = read(checking_bal)
if (x > 100) {
  y = read(savings_bal)
  write(checking_bal, x - 100)
  write(savings_bal, y + 100)
}
```

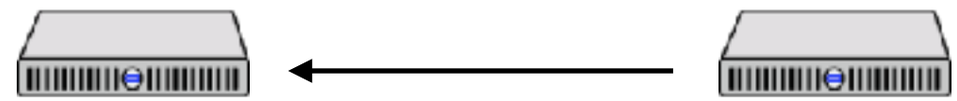




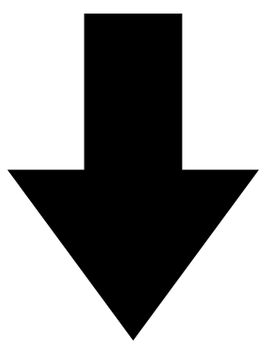
```
LOG:
 1. A: Lock(R, checking_bal)
```

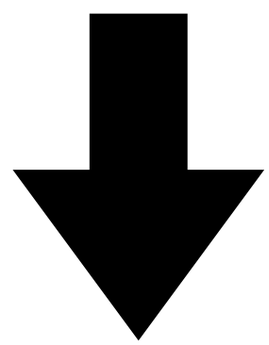
```
LOG:
 1. A: Lock(R, savings_bal)
```

Leader 1 Client Leader 2



```
x = read(checking_bal)
if (x > 100) {
  y = read(savings_bal)
  write(checking_bal, x - 100)
  write(savings_bal, y + 100)
}
```

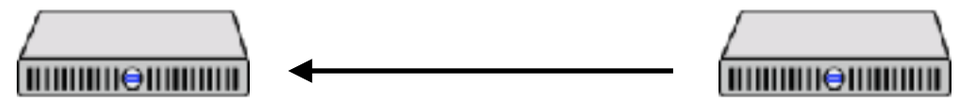




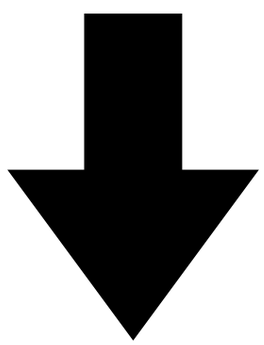
```
LOG:
 1. A: Lock(R, checking_bal)
```

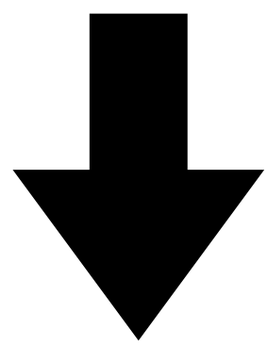
```
LOG:
 1. A: Lock(R, savings_bal)
```

Leader 1 Client Leader 2



```
x = read(checking_bal)
if (x > 100) {
  y = read(savings_bal)
  write(checking_bal, x - 100)
  write(savings_bal, y + 100)
}
```



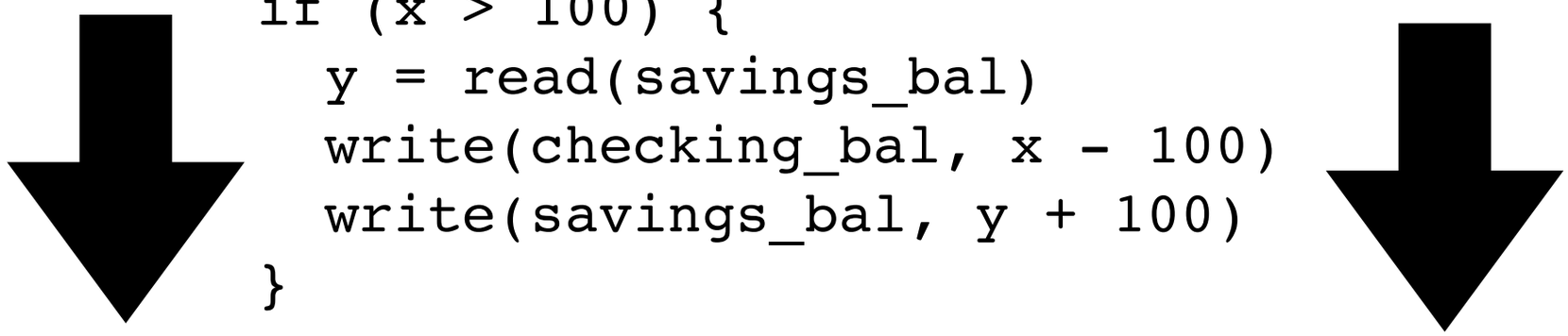


```
LOG:
 1. A: Lock(R, checking_bal)
 2. A: Lock(W, checking_bal)
 3. A: Write(checking_bal, 100)
```

```
LOG:
 1. A: Lock(R, savings_bal)
```

Leader 1 Client Leader 2

```
x = read(checking_bal)
if (x > 100) {
  y = read(savings_bal)
  write(checking_bal, x - 100)
  write(savings_bal, y + 100)
}
```

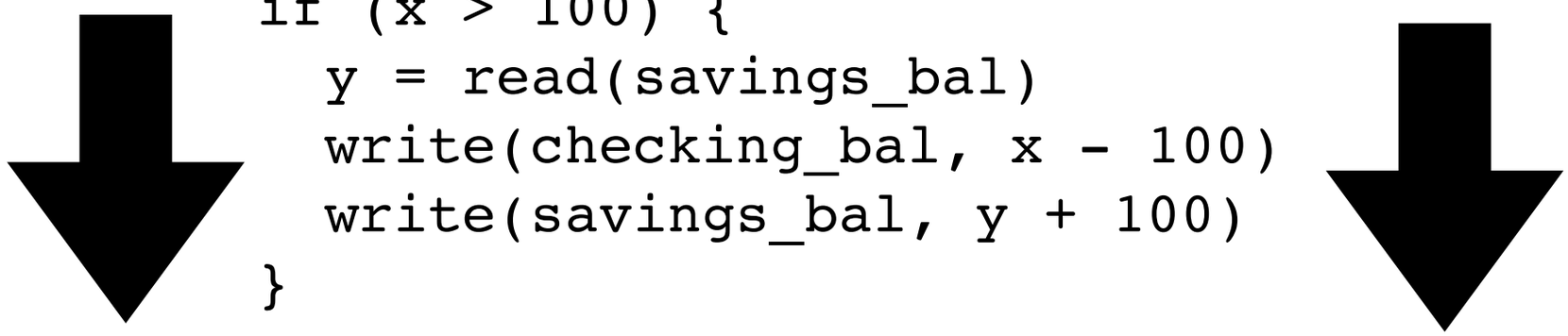


```
LOG:
 1. A: Lock(R, checking_bal)
 2. A: Lock(W, checking_bal)
 3. A: Write(checking_bal, 100)
```

```
LOG:
 1. A: Lock(R, savings_bal)
```

Leader 1 Client Leader 2

```
x = read(checking_bal)
if (x > 100) {
  y = read(savings_bal)
  write(checking_bal, x - 100)
  write(savings_bal, y + 100)
}
```



```
LOG:
 1. A: Lock(R, checking_bal)
 2. A: Lock(W, checking_bal)
 3. A: Write(checking_bal, 100)
```

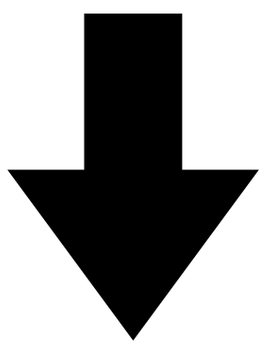
```
LOG:
 1. A: Lock(R, savings_bal)
 2. A: Lock(W, savings_bal)
 3. A: Write(savings_bal, 100)
```

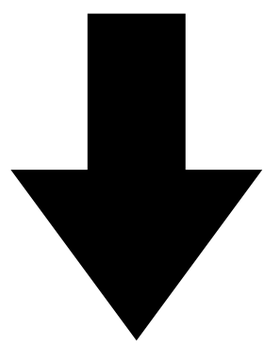
Leader 1

Client

Leader 2

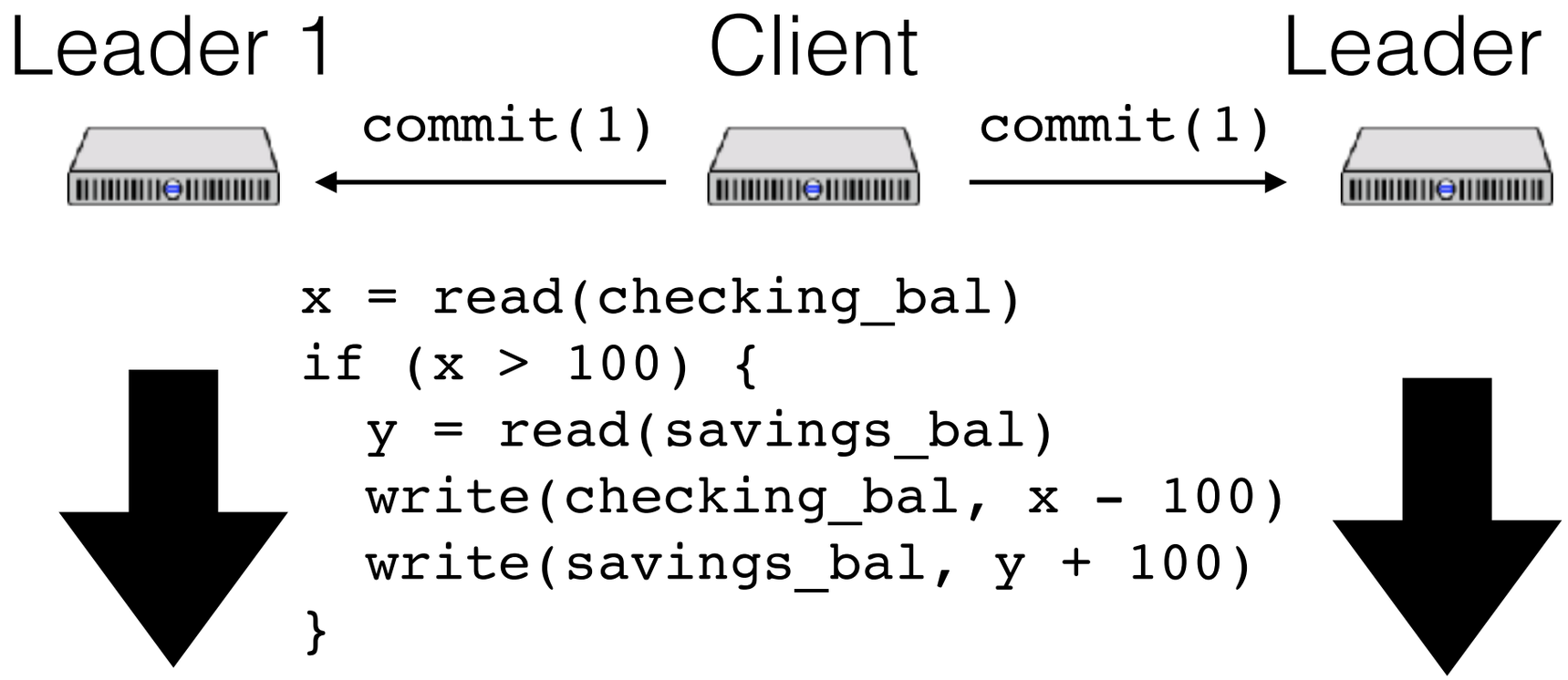
```
x = read(checking_bal)
if (x > 100) {
  y = read(savings_bal)
  write(checking_bal, x - 100)
  write(savings_bal, y + 100)
}
```





```
LOG:
 1. A: Lock(R, checking_bal)
 2. A: Lock(W, checking_bal)
 3. A: Write(checking_bal, 100)
```

```
LOG:
 1. A: Lock(R, savings_bal)
 2. A: Lock(W, savings_bal)
 3. A: Write(savings_bal, 100)
```




```
LOG:
 1. A: Lock(R, checking_bal)
 2. A: Lock(W, checking_bal)
 3. A: Write(checking_bal, 100)
```

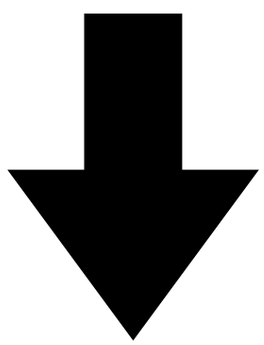
```
LOG:
 1. A: Lock(R, savings_bal)
 2. A: Lock(W, savings_bal)
 3. A: Write(savings_bal, 100)
```

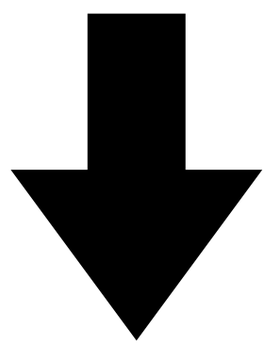
Leader 1

Client

Leader 2

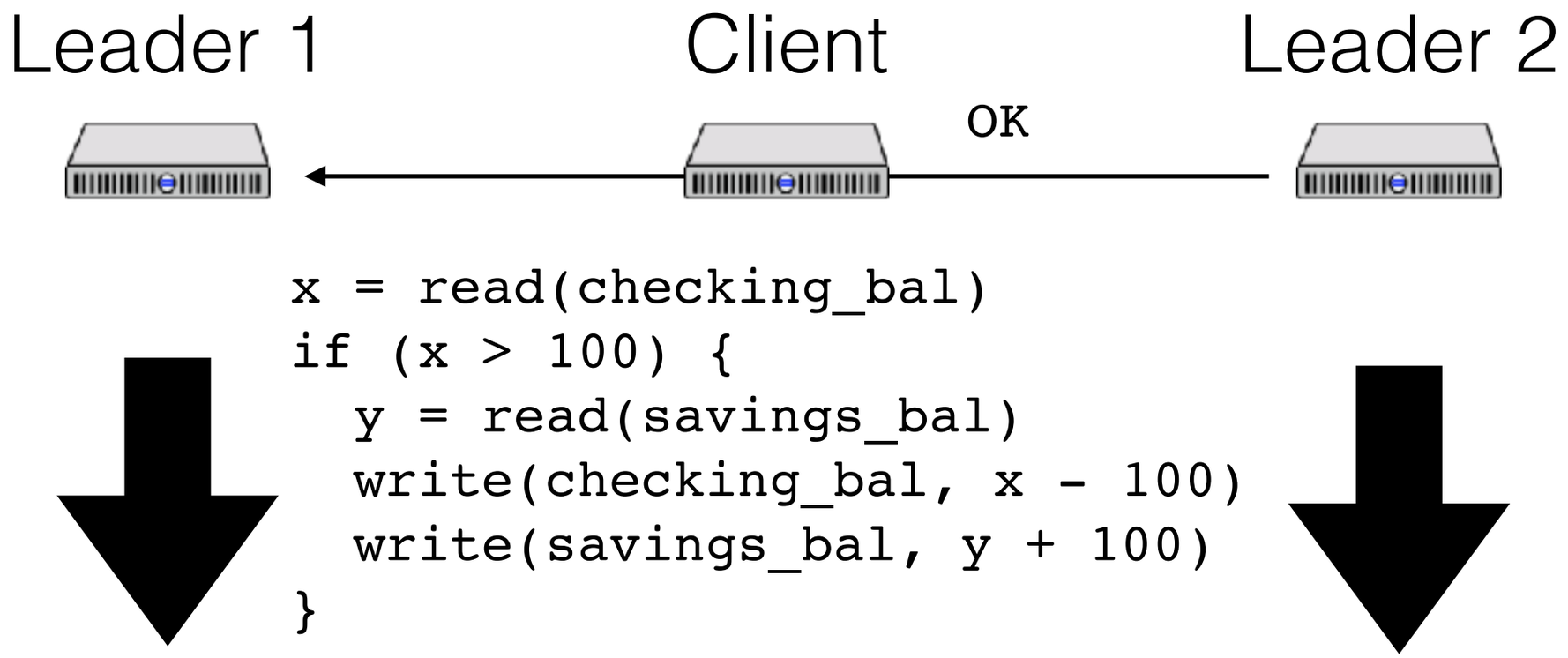
```
x = read(checking_bal)
if (x > 100) {
  y = read(savings_bal)
  write(checking_bal, x - 100)
  write(savings_bal, y + 100)
}
```





```
LOG:
 1. A: Lock(R, checking_bal)
 2. A: Lock(W, checking_bal)
 3. A: Write(checking_bal, 100)
```

```
LOG:
 1. A: Lock(R, savings_bal)
 2. A: Lock(W, savings_bal)
 3. A: Write(savings_bal, 100)
 4. A: Prepare
```




```
LOG:
 1. A: Lock(R, checking_bal)
 2. A: Lock(W, checking_bal)
 3. A: Write(checking_bal, 100)
```

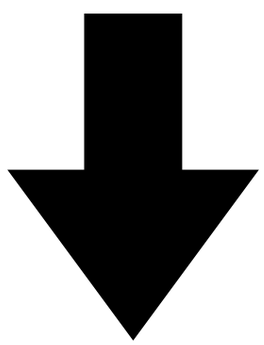
```
LOG:
 1. A: Lock(R, savings_bal)
 2. A: Lock(W, savings_bal)
 3. A: Write(savings_bal, 100)
 4. A: Prepare
```

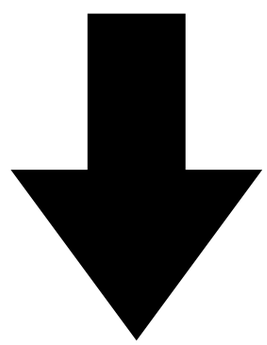
Leader 1

Client

Leader 2

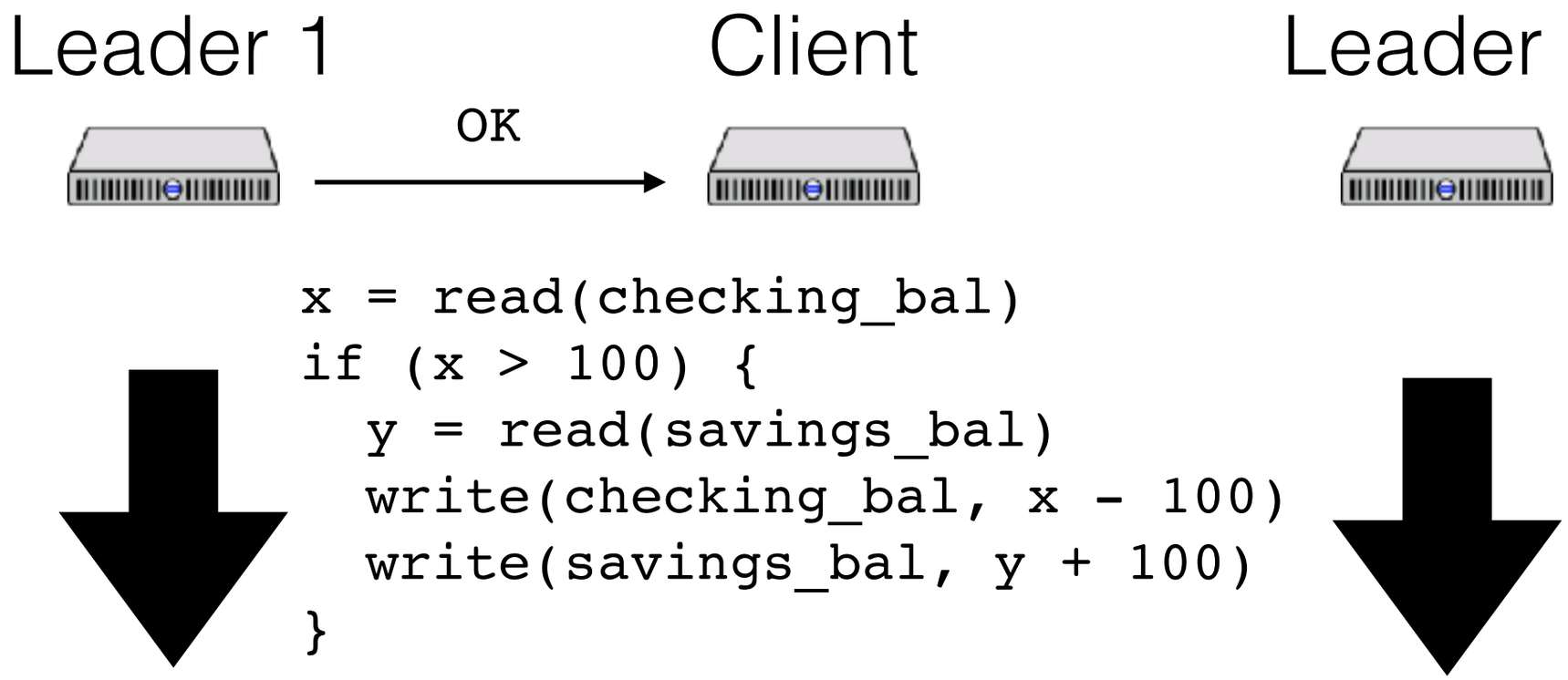
```
x = read(checking_bal)
if (x > 100) {
  y = read(savings_bal)
  write(checking_bal, x - 100)
  write(savings_bal, y + 100)
}
```





```
LOG:
 1. A: Lock(R, checking_bal)
 2. A: Lock(W, checking_bal)
 3. A: Write(checking_bal, 100)
 4. A: Commit
```

```
LOG:
 1. A: Lock(R, savings_bal)
 2. A: Lock(W, savings_bal)
 3. A: Write(savings_bal, 100)
 4. A: Prepare
```




```
LOG:
 1. A: Lock(R, checking_bal)
 2. A: Lock(W, checking_bal)
 3. A: Write(checking_bal, 100)
 4. A: Commit
```

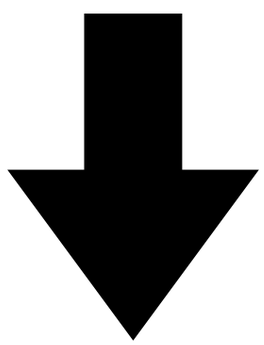
```
LOG:
 1. A: Lock(R, savings_bal)
 2. A: Lock(W, savings_bal)
 3. A: Write(savings_bal, 100)
 4. A: Prepare
```

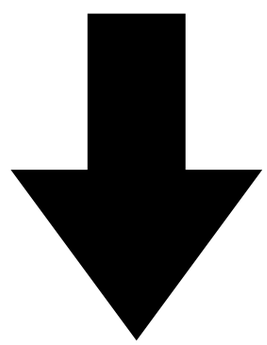
Leader 1

Client

Leader 2

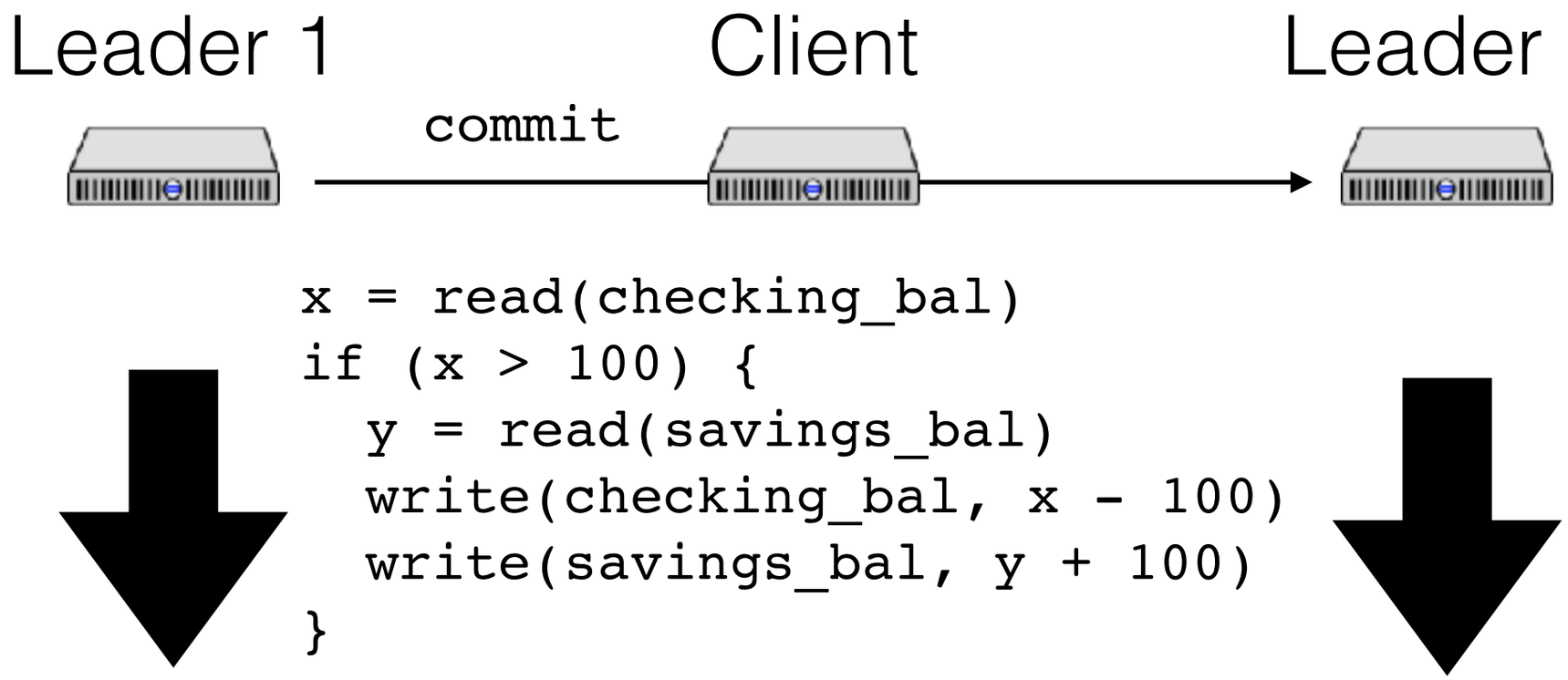
```
x = read(checking_bal)
if (x > 100) {
  y = read(savings_bal)
  write(checking_bal, x - 100)
  write(savings_bal, y + 100)
}
```





```
LOG:
 1. A: Lock(R, checking_bal)
 2. A: Lock(W, checking_bal)
 3. A: Write(checking_bal, 100)
 4. A: Commit
 5. A: Unlock(checking_bal)
```

```
LOG:
 1. A: Lock(R, savings_bal)
 2. A: Lock(W, savings_bal)
 3. A: Write(savings_bal, 100)
 4. A: Prepare
```




```
LOG:
 1. A: Lock(R, checking_bal)
 2. A: Lock(W, checking_bal)
 3. A: Write(checking_bal, 100)
 4. A: Commit
 5. A: Unlock(checking_bal)
```

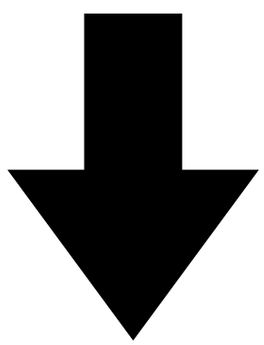
```
LOG:
 1. A: Lock(R, savings_bal)
 2. A: Lock(W, savings_bal)
 3. A: Write(savings_bal, 100)
 4. A: Prepare
```

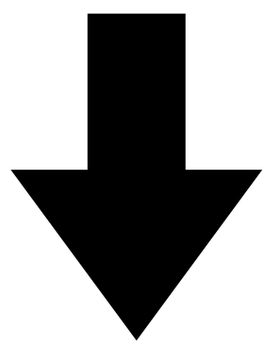
Leader 1

Client

Leader 2

```
x = read(checking_bal)
if (x > 100) {
  y = read(savings_bal)
  write(checking_bal, x - 100)
  write(savings_bal, y + 100)
}
```





```
LOG:
 1. A: Lock(R, checking_bal)
 2. A: Lock(W, checking_bal)
 3. A: Write(checking_bal, 100)
 4. A: Commit
 4. A: Unlock(checking_bal)
```

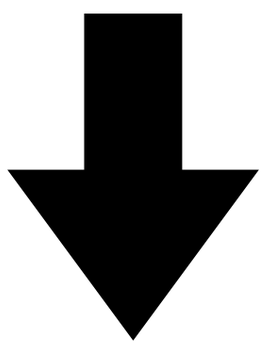
```
LOG:
 1. A: Lock(R, savings_bal)
 2. A: Lock(W, savings_bal)
 3. A: Write(savings_bal, 100)
 4. A: Prepare
 5. A: Commit
```

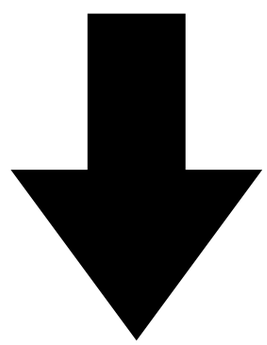
Leader 1

Client

Leader 2

```
x = read(checking_bal)
if (x > 100) {
  y = read(savings_bal)
  write(checking_bal, x - 100)
  write(savings_bal, y + 100)
}
```





```
LOG:
 1. A: Lock(R, checking_bal)
 2. A: Lock(W, checking_bal)
 3. A: Write(checking_bal, 100)
 4. A: Commit
 4. A: Unlock(checking_bal)
```

```
LOG:
 1. A: Lock(R, savings_bal)
 2. A: Lock(W, savings_bal)
 3. A: Write(savings_bal, 100)
 4. A: Prepare
 5. A: Commit
 6. A: Unlock(savings_bal)
```

How can we get fast reads?

R/W transactions are complicated!

- And slow

Can we do fast, lock-free reads?

- Real time to the rescue

TrueTime

API that exposes real time, with uncertainty

`{earliest: e, latest: l} = TT.now()`

“Real time” is between `earliest` and `latest`

Time is an illusion; lunchtime, doubly so

If I call TT.now() on two nodes simultaneously, intervals *guaranteed* to overlap!

And: if intervals don’t overlap, the later one really happened later!

TrueTime usage

Assign a timestamp to each transaction

- At each Paxos group, timestamp increases monotonically

- Globally, if T1 returns before T2 starts,
    timestamp(T1) < timestamp(T2)

TrueTime usage

Timestamp for an RW transaction chosen by coordinator leader

Timestamps for R/W transactions is max of:

- Local time
- Prepare timestamps at every participant
- Timestamp of any previous local transaction

Commit wait

Need to ensure that all future transactions will get a higher timestamp

Therefore, need to wait until

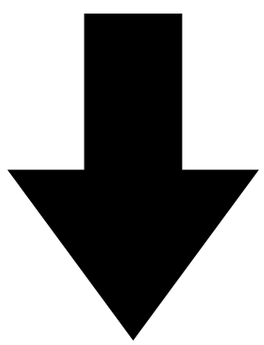
TT.now() > transaction timestamp

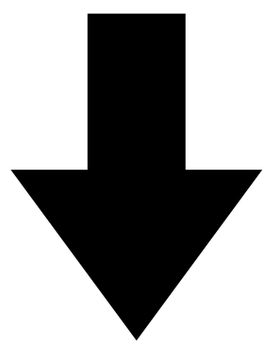
Leader 1

Client

Leader 2

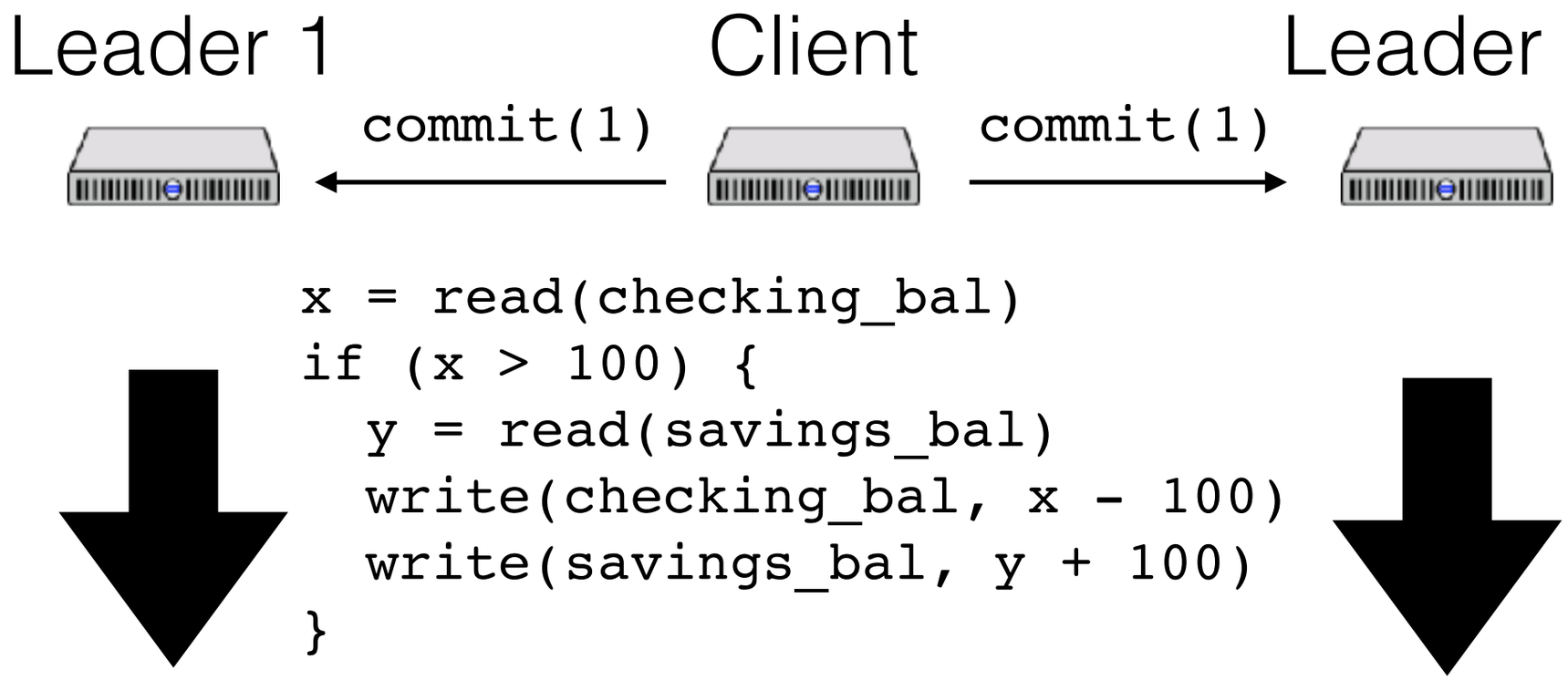
```
x = read(checking_bal)
if (x > 100) {
  y = read(savings_bal)
  write(checking_bal, x - 100)
  write(savings_bal, y + 100)
}
```





```
LOG:
 1. A: Lock(R, checking_bal)
 2. A: Lock(W, checking_bal)
 3. A: Write(checking_bal, 100)
```

```
LOG:
 1. A: Lock(R, savings_bal)
 2. A: Lock(W, savings_bal)
 3. A: Write(savings_bal, 100)
```




```
LOG:
 1. A: Lock(R, checking_bal)
 2. A: Lock(W, checking_bal)
 3. A: Write(checking_bal, 100)
```

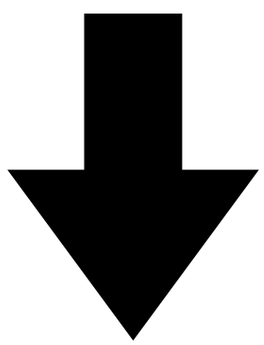
```
LOG:
 1. A: Lock(R, savings_bal)
 2. A: Lock(W, savings_bal)
 3. A: Write(savings_bal, 100)
```

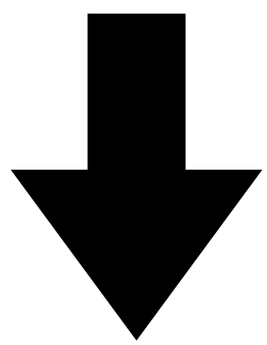
Leader 1

Client

Leader 2

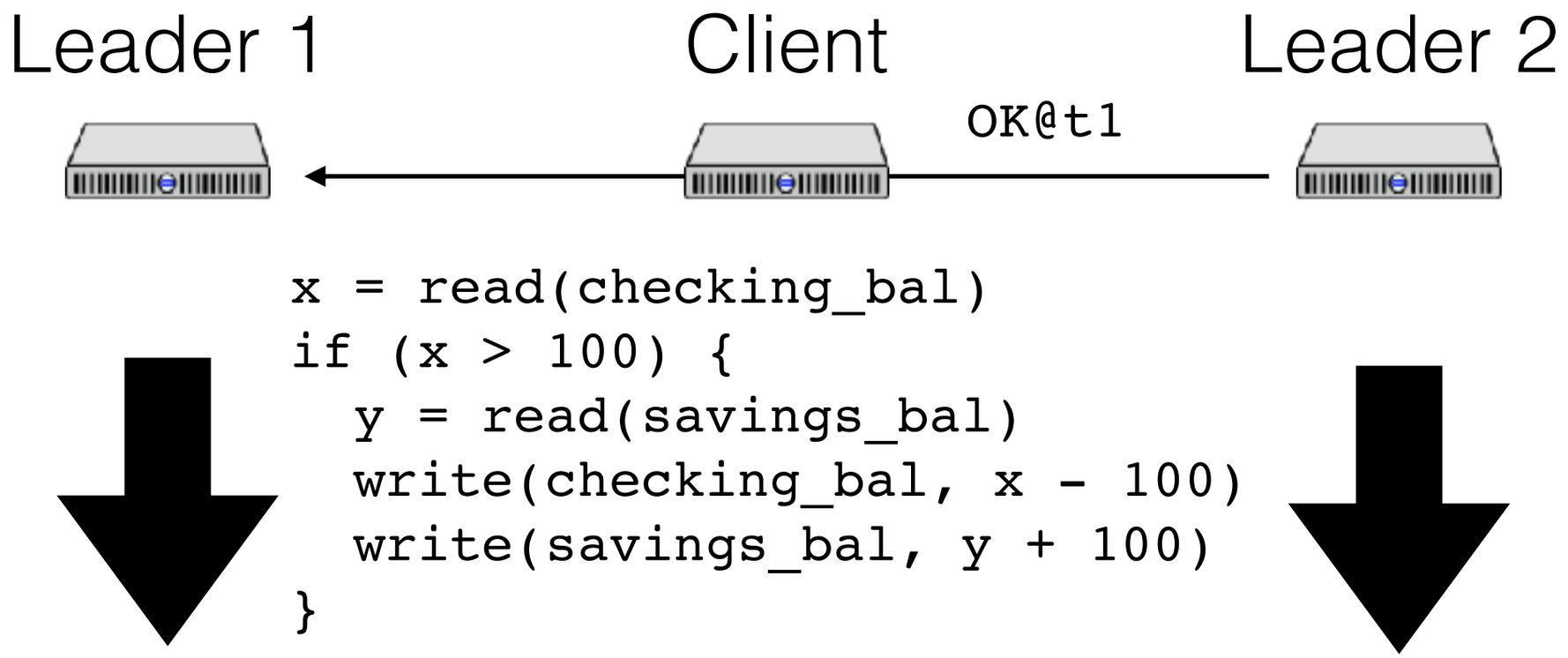
```
x = read(checking_bal)
if (x > 100) {
  y = read(savings_bal)
  write(checking_bal, x - 100)
  write(savings_bal, y + 100)
}
```





```
LOG:
 1. A: Lock(R, checking_bal)
 2. A: Lock(W, checking_bal)
 3. A: Write(checking_bal, 100)
```

```
LOG:
 1. A: Lock(R, savings_bal)
 2. A: Lock(W, savings_bal)
 3. A: Write(savings_bal, 100)
 4. A: Prepare@t1
```




```
LOG:
 1. A: Lock(R, checking_bal)
 2. A: Lock(W, checking_bal)
 3. A: Write(checking_bal, 100)
```

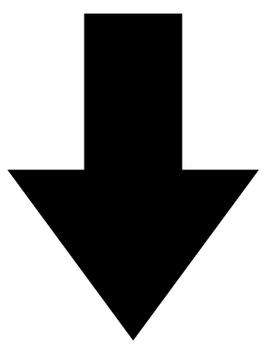
```
LOG:
 1. A: Lock(R, savings_bal)
 2. A: Lock(W, savings_bal)
 3. A: Write(savings_bal, 100)
 4. A: Prepare@t1
```

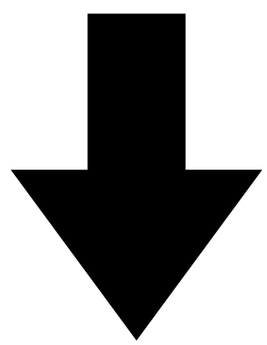
Leader 1

Client

Leader 2

```
x = read(checking_bal)
if (x > 100) {
  y = read(savings_bal)
  write(checking_bal, x - 100)
  write(savings_bal, y + 100)
}
```





```
LOG:
 1. A: Lock(R, checking_bal)
 2. A: Lock(W, checking_bal)
 3. A: Write(checking_bal, 100)
 4. A: Commit@max(t1, tleader)
```

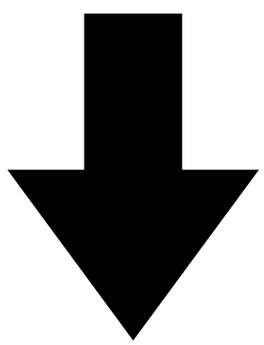
```
LOG:
 1. A: Lock(R, savings_bal)
 2. A: Lock(W, savings_bal)
 3. A: Write(savings_bal, 100)
 4. A: Prepare@t1
```

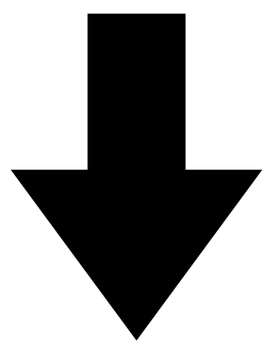
Leader 1

Client

Leader 2

```
x = read(checking_bal)
if (x > 100) {
  y = read(savings_bal)
  write(checking_bal, x - 100)
  write(savings_bal, y + 100)
}
```





```
LOG:
 1. A: Lock(R, checking_bal)
 2. A: Lock(W, checking_bal)
 3. A: Write(checking_bal, 100)
 4. A: Commit@max(t1, tleader)
```

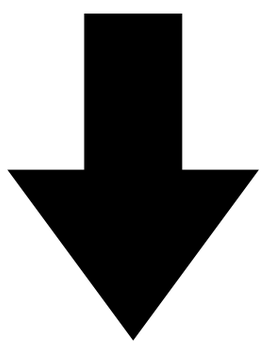
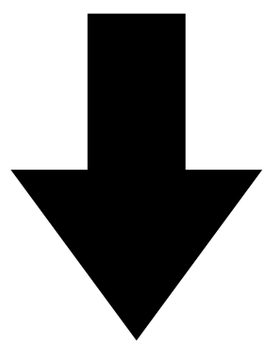
```
LOG:
 1. A: Lock(R, savings_bal)
 2. A: Lock(W, savings_bal)
 3. A: Write(savings_bal, 100)
 4. A: Prepare@t1
```

Waiting…

Leader 1 Client Leader 2

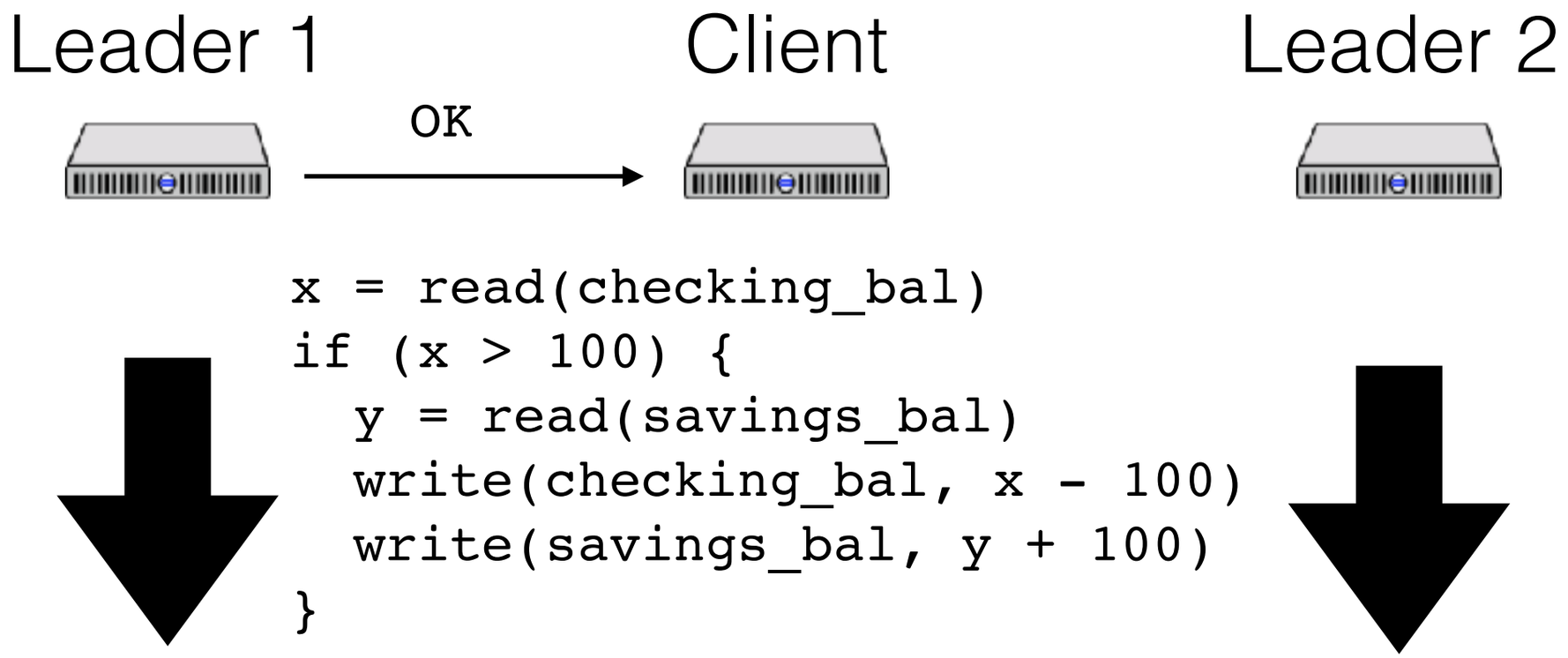


```
x = read(checking_bal)
if (x > 100) {
  y = read(savings_bal)
  write(checking_bal, x - 100)
  write(savings_bal, y + 100)
}
```

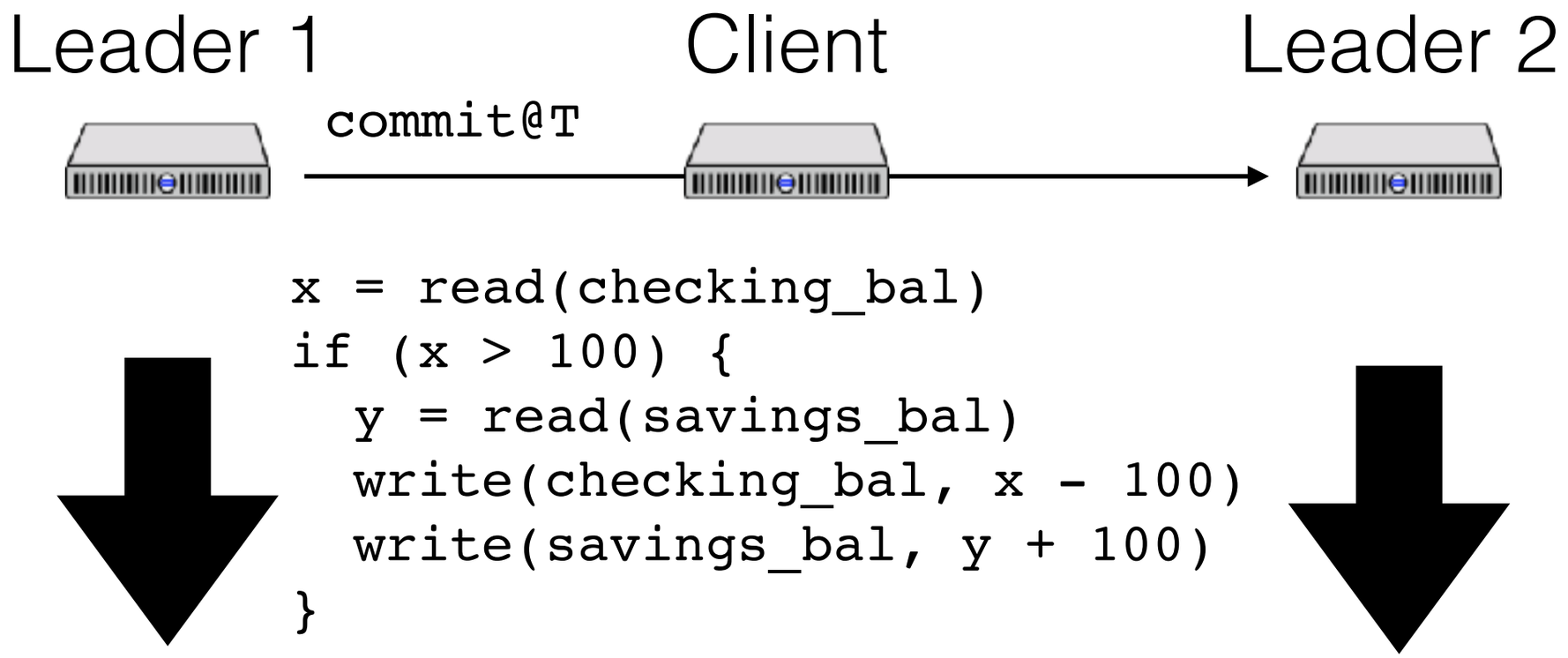
```
LOG:
 1. A: Lock(R, checking_bal)
 2. A: Lock(W, checking_bal)
 3. A: Write(checking_bal, 100)
 4. A: Commit@max(t1, tleader)
 5. A: Unlock(checking_bal)
```

```
LOG:
 1. A: Lock(R, savings_bal)
 2. A: Lock(W, savings_bal)
 3. A: Write(savings_bal, 100)
 4. A: Prepare@t1
```




```
LOG:
 1. A: Lock(R, checking_bal)
 2. A: Lock(W, checking_bal)
 3. A: Write(checking_bal, 100)
 4. A: Commit@max(t1, tleader)
 5. A: Unlock(checking_bal)
```

```
LOG:
 1. A: Lock(R, savings_bal)
 2. A: Lock(W, savings_bal)
 3. A: Write(savings_bal, 100)
 4. A: Prepare@t1
```




```
LOG:
 1. A: Lock(R, checking_bal)
 2. A: Lock(W, checking_bal)
 3. A: Write(checking_bal, 100)
 4. A: Commit@max(t1, tleader)
 5. A: Unlock(checking_bal)
```

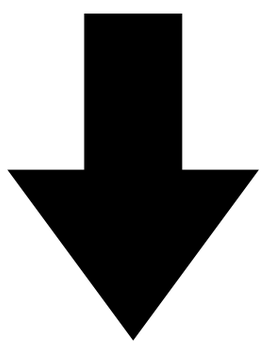
```
LOG:
 1. A: Lock(R, savings_bal)
 2. A: Lock(W, savings_bal)
 3. A: Write(savings_bal, 100)
 4. A: Prepare@t1
```

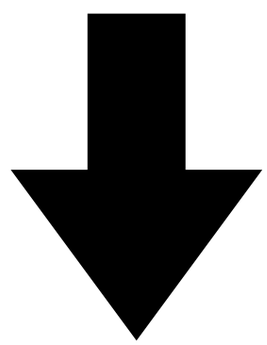
Leader 1

Client

Leader 2

```
x = read(checking_bal)
if (x > 100) {
  y = read(savings_bal)
  write(checking_bal, x - 100)
  write(savings_bal, y + 100)
}
```





```
LOG:
 1. A: Lock(R, checking_bal)
 2. A: Lock(W, checking_bal)
 3. A: Write(checking_bal, 100)
 4. A: Commit@max(t1, tleader)
 5. A: Unlock(checking_bal)
```

```
LOG:
 1. A: Lock(R, savings_bal)
 2. A: Lock(W, savings_bal)
 3. A: Write(savings_bal, 100)
 4. A: Prepare@t1
 5. A: Commit@max(t1, tleader)
 6. A: Unlock(checking_bal)
```

TrueTime usage

Timestamps for RO transactions

- Client can choose a timestamp
- For external consistency, always safe to choose `TT.now().latest`

A response from a replica is a promise: will never have a new transaction that commits before timestamp

Can read from any replica that has seen a Paxos write after timestamp

TrueTime implementation

GPS, atomic clocks

Armageddon masters and timeslave daemons

All local clocks synced with masters, and expose uncertainty to local apps

Assumptions made about local clock drift

Conclusions

What's cool about Spanner?

- Distributed transactions with decent performance
- What makes that possible?
- Read-only transactions with great performance
- What makes that possible?

Clocks are a form of communication!

Discussion

CPU errors and bad clocks

- When does Google discover errors?

Disaster preparedness

Is Spanner's complexity scary?

What happens if there is high clock skew?

- If we know about it

- If we don't?