Hybrid Solution

(optimistic concurrency + pessimistic or Multiversion Concurrency Control)

```
last_changed = obj.modified

... read phase

SomeModel.objects.select_for_update().filter(id=obj.id, modified=last_changed)

... write phase
```

Best of both worlds

> locking is limited to write-phase only
> no need for complex multi-model compare-and-swaps

MVCC is used internally by PostgreSQL

Alternative: SQL gap-locking w/ filter query on indexed col.

Dealing with money

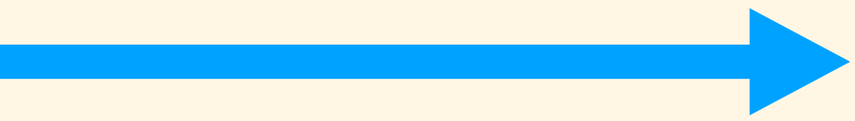
float, Decimal, and math

Avoiding concurrency

linearizing writes in a queue

Dealing with concurrency

transactions, locking, compare-and-swaps



Schema design

log-structured data, minimizing locks

The bigger picture

code layout, storage layer, NewSQL databases