

L17: Isolation

Frans Kaashoek & Nickolai Zeldovich
6.033 Spring 2012

Concurrent actions

```
xfer(a, b, amt):  
  begin  
    a = a - amt  
    b = b + amt  
  commit
```

```
interest(rate):  
  begin  
    for each account x:  
      x = x * (1+rate)  
  commit
```

Locking protocol

```
read(var):  
    if var.lock not held:  
        acquire(var.lock)  
    return var.value
```

```
write(var, newval):  
    if var.lock not held:  
        acquire(var.lock)  
    var.value = newval
```

Locking protocol with release

read(var):

```
    if var.lock not held:  
        acquire(var.lock)  
    return var.value
```

write(var, newval):

```
    if var.lock not held:  
        acquire(var.lock)  
    var.value = newval
```

commit():

```
    write commit record  
    release all locks
```

Locking with reader-writer locks

```
read(var):
```

```
    if var.lock not held:
```

```
        acquire_reader(var.lock)
```

```
        # block if any writers
```

```
    return var.value
```

```
write(var, newval):
```

```
    if var.lock not held as writer:
```

```
        acquire_writer(var.lock)
```

```
        # block if any readers or writers
```

```
    var.value = newval
```

Snap-shot isolation

Setup: table with doctors, oncall=true

T1:

```
select count(*) from doctors where oncall=true;  
update doctors set oncall=false where username = 'alice';
```

T2:

```
select count(*) from doctors where oncall=true;  
update doctors set oncall=false where username = 'bob';
```

read-committed isolation

Setup: table with doctors, oncall=false

T1:

```
select count(*) from doctors where oncall=false;
```

```
select count(*) from doctors where oncall=false;
```

T2:

```
update doctors set oncall=true where username = 'bob';  
commit;
```