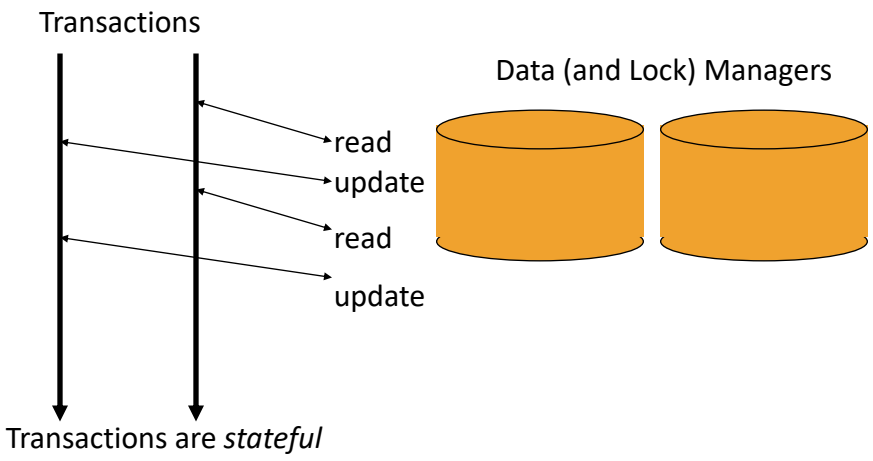
Transactions

Dominic Duggan
Stevens Institute of Technology

1

Transaction and Data Managers



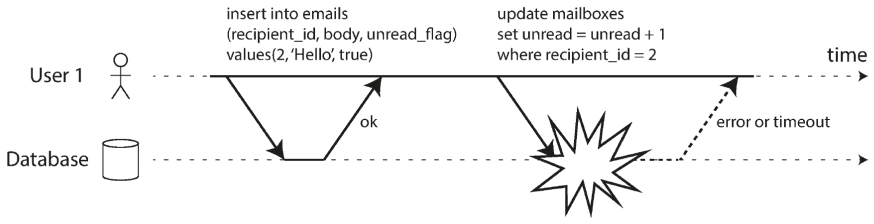
2

Atomicity

- Either a transaction happens completely, or it does not happen at all
- If a transaction happens, it appears to happen as a single indivisible action

3

Violating Atomicity



4

Consistency

- If the system has certain invariants, then if they held before a transaction, they hold after the completion of that transaction
  - Law of conservation of money
- That doesn't mean that the invariants need to be true throughout the transaction
- **Application Property**

5

5

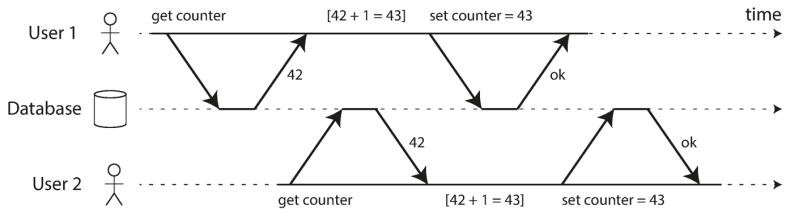
Isolation

- Final result looks as though all transactions ran sequentially in some (system-dependent) order
  - They appear to have run serially, rather than in parallel
- Implemented by no database...

6

6

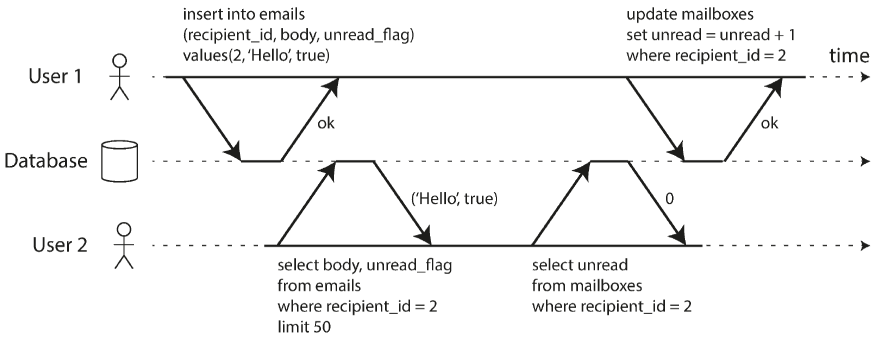
Violating Isolation



7

7

Violating Isolation



8

Durability

- Once a transaction is successfully committed, regardless of what happens afterward, its results become permanent
- Specifically, no failure after the commit can undo the results or cause them to be lost
- At least written to (stable) log…

9

9

IMPLEMENTATION

10

10

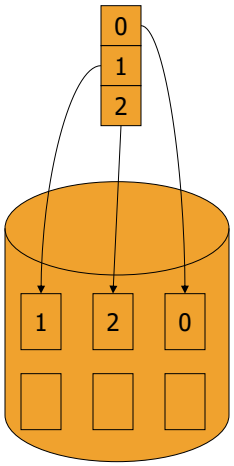
Two Implementation Methods

- Private Workspace
  - Shadow paging
- Writeahead Log
  - Log of updates, including original values

11

11

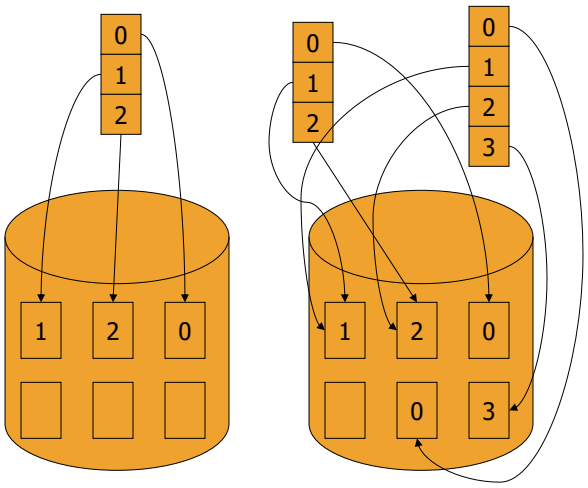
Private Workspace



12

12

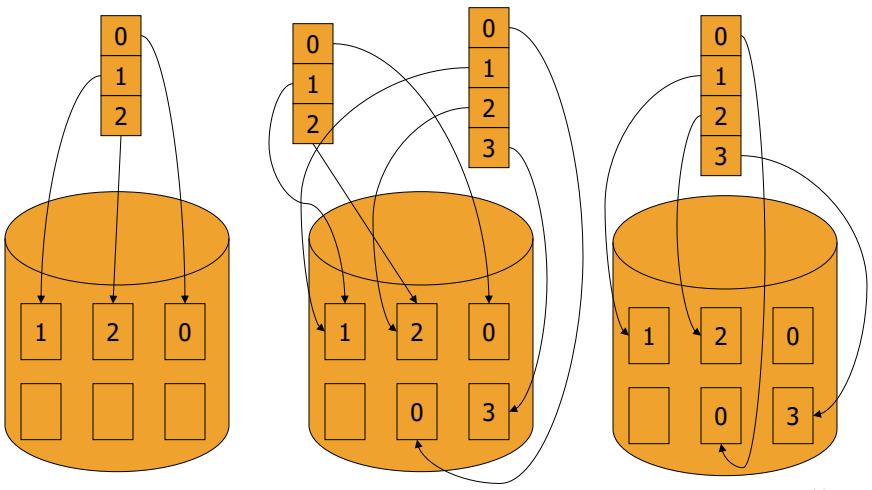
Private Workspace



13

13

Private Workspace



14

14

Writeahead Log

- DB modified in place
- Log record:
  - transaction identifier
  - file and block that are being changed
  - old and new values of the block
- DB only changed after log record has been successfully written
  - File update may happen some time after log written
  - File/database access slower than log append

15

15

Writeahead Log Example

```
x = 0;                          (1) [x = 0/1]
y = 0;
BEGIN_TRANSACTION;              (2) [x = 0/1]
 (1) x = x + 1;                     [y = 0/2]
 (2) y = y + 2;
 (3) x = y * y;                 (3) [x = 0/1]
END_TRANSACTION;                    [y = 0/2]
                                    [x = 1/4]
```

16

16

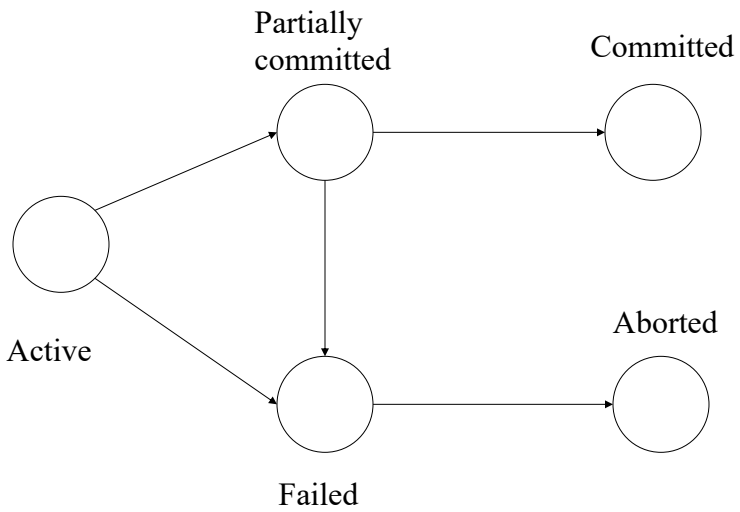
Writeahead Log

- Success: commit record is written to the log
- Abort: roll back any changes

17

17

Transaction State



18

18

READ COMMITTED

19

19

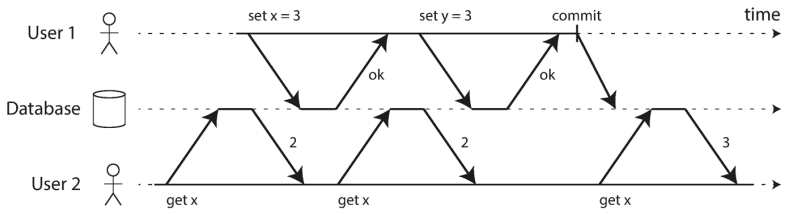
Read Committed

- No dirty reads
  - When reading from the database, you will only see data that has been committed
- No dirty writes
  - When writing to the database, you will only overwrite data that has been committed

20

20

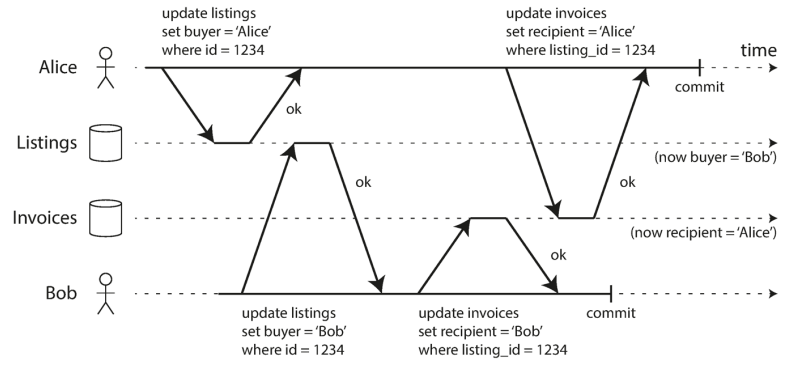
No Dirty Reads



21

21

Dirty Writes



22

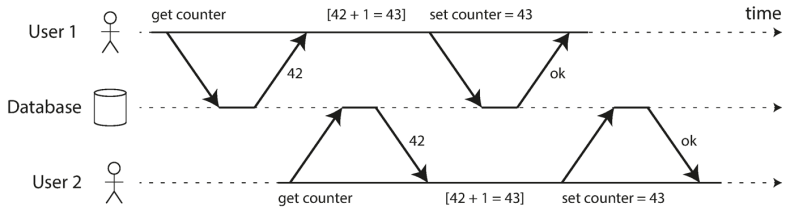
22

Implementing Read Committed

- Prevent dirty writes
  - Lock database record until commit
- Prevent dirty reads
  - Other transactions are given old version of data until commit

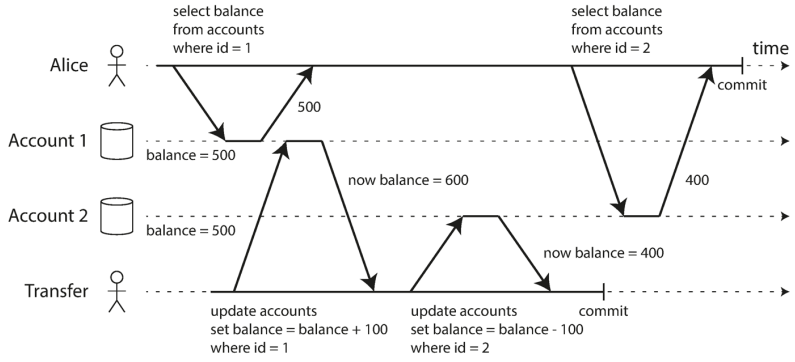
Read Committed

- Does Read Committed prevent this?



Read Skew

- Problem for e.g. backups, analytic queries



25

25