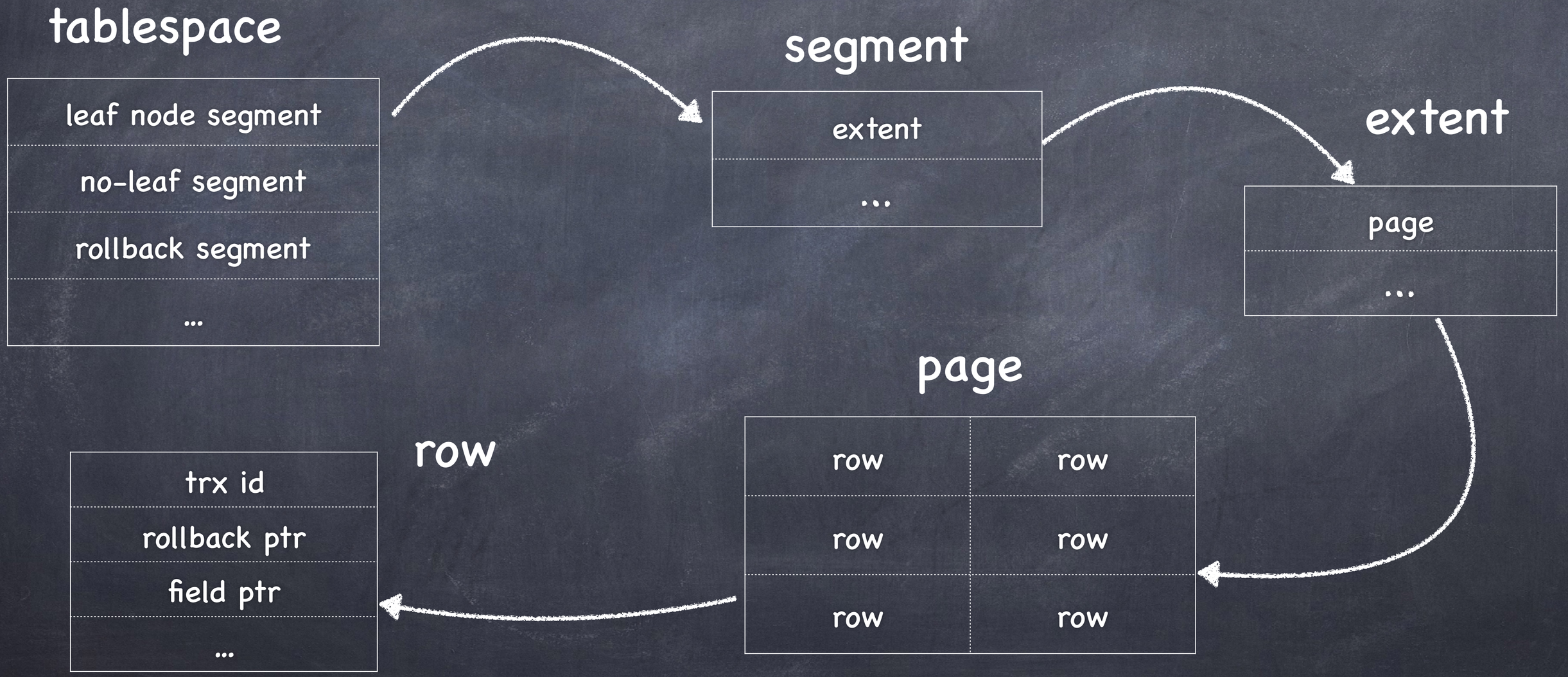
分析mysql acid设计实现

blog- xiaorui.cc
author- rfyiamcool

List

- innodb 基本结构
- innodb acid 概念
- innodb buffer pool
- mvcc
- 各种延伸
- 锁，各种锁

innodb的抽象结构



b+tree 数据结构

page

22 row
20 row
17 row
16 row
[16, 17, 20, 22]

31 row
35 row
44 row
32 row
[31, 32, 35, 44]

有序数组

file system vs disk

innodb engine - - - > 16k

---

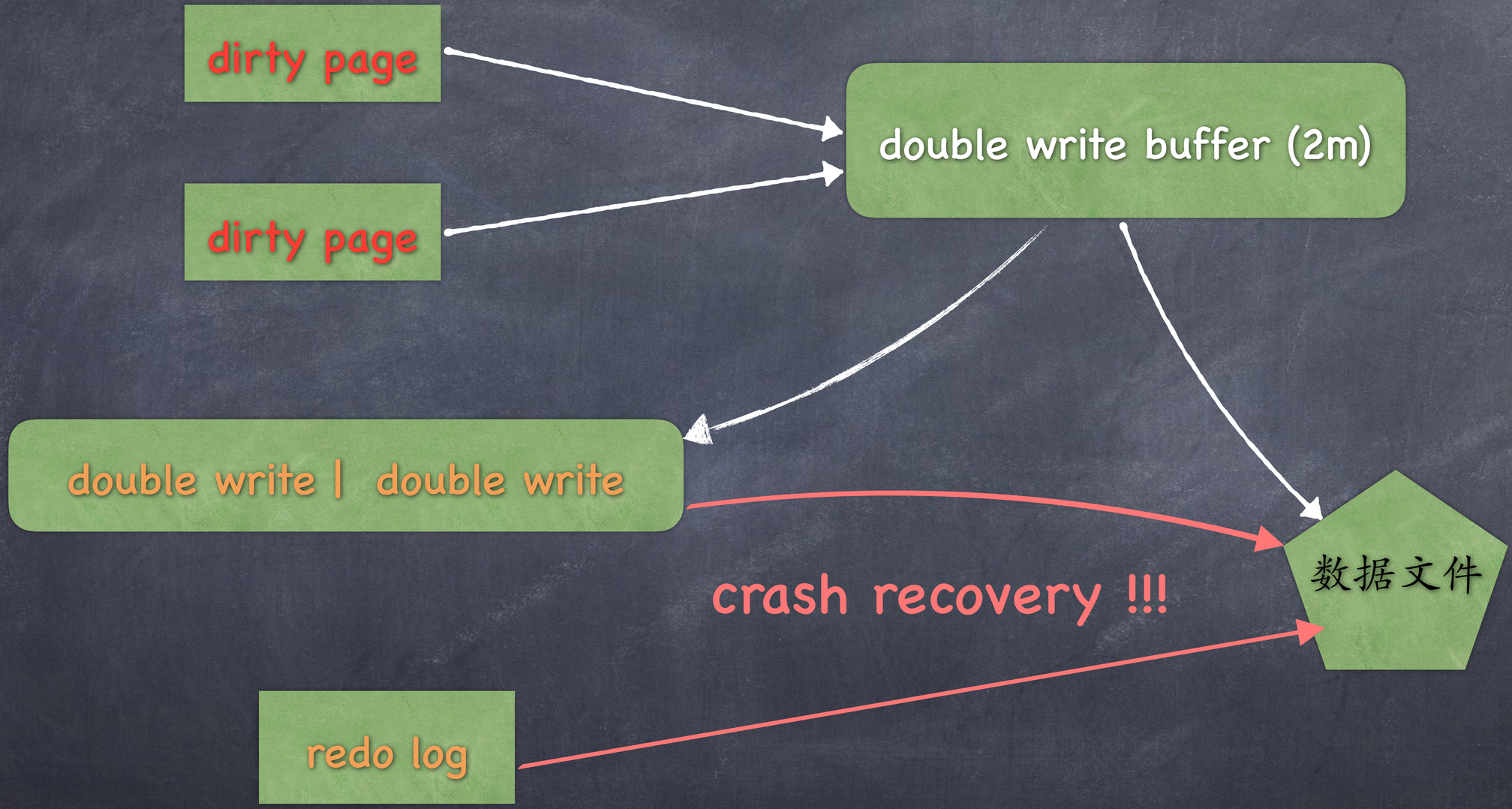
file system - - -> 4k

---

disk - - - > 512 bytes

size不同如何上下对齐?

- partial page write ?
- 解决办法是 doublewrite buffer + doublewrite
- doublewrite 性能影响? 5% - 10%


dirty page
dirty page
double write buffer (2m)
double write | double write
数据文件
crash recovery !!!
redo log

what is acid

- atomicity
- consistency
- isolation
- durability

atomicity

- 一个事务为一个原子　不可分割
- 要么成功　要么失败

consistency

- 经过一系列改动及异常崩溃，最后的结果是我要的.
  - 最终一致性
  - 强一致性

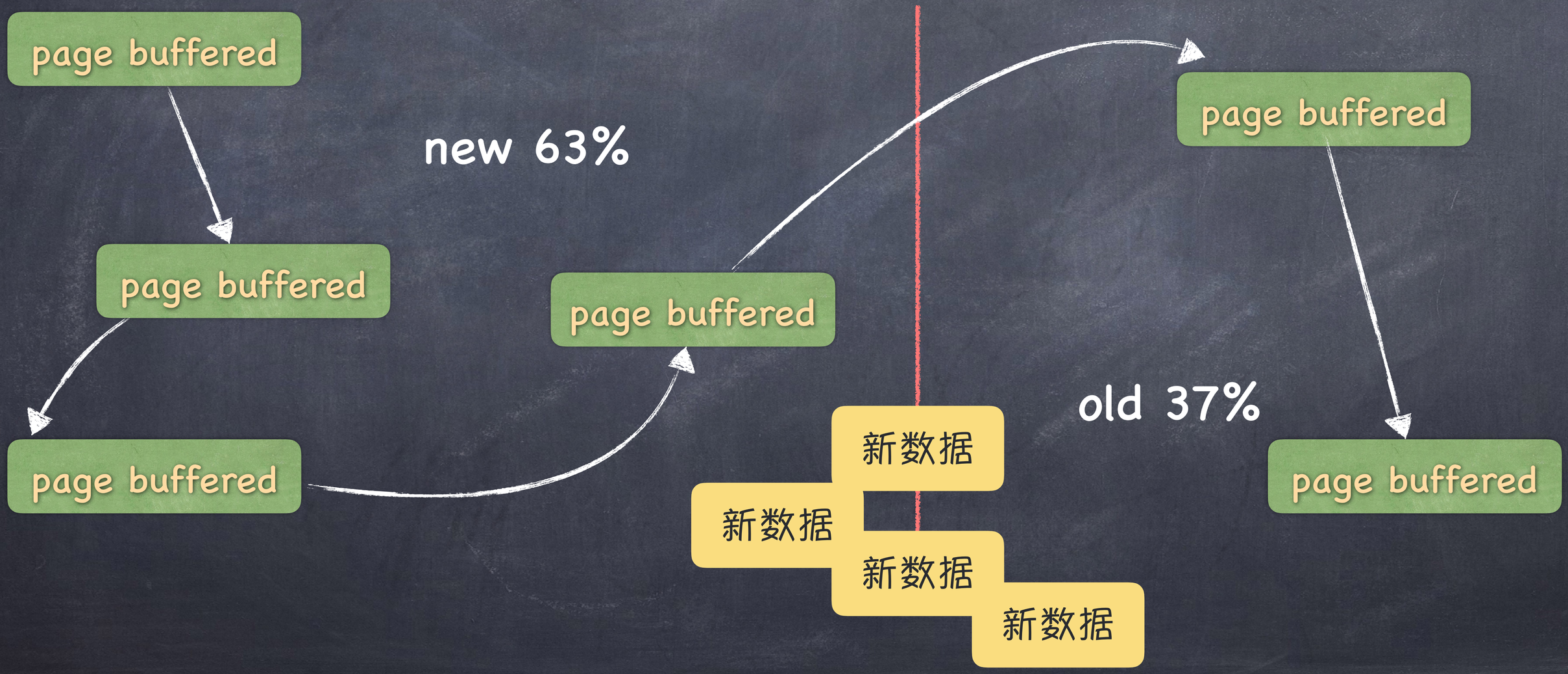
isolation

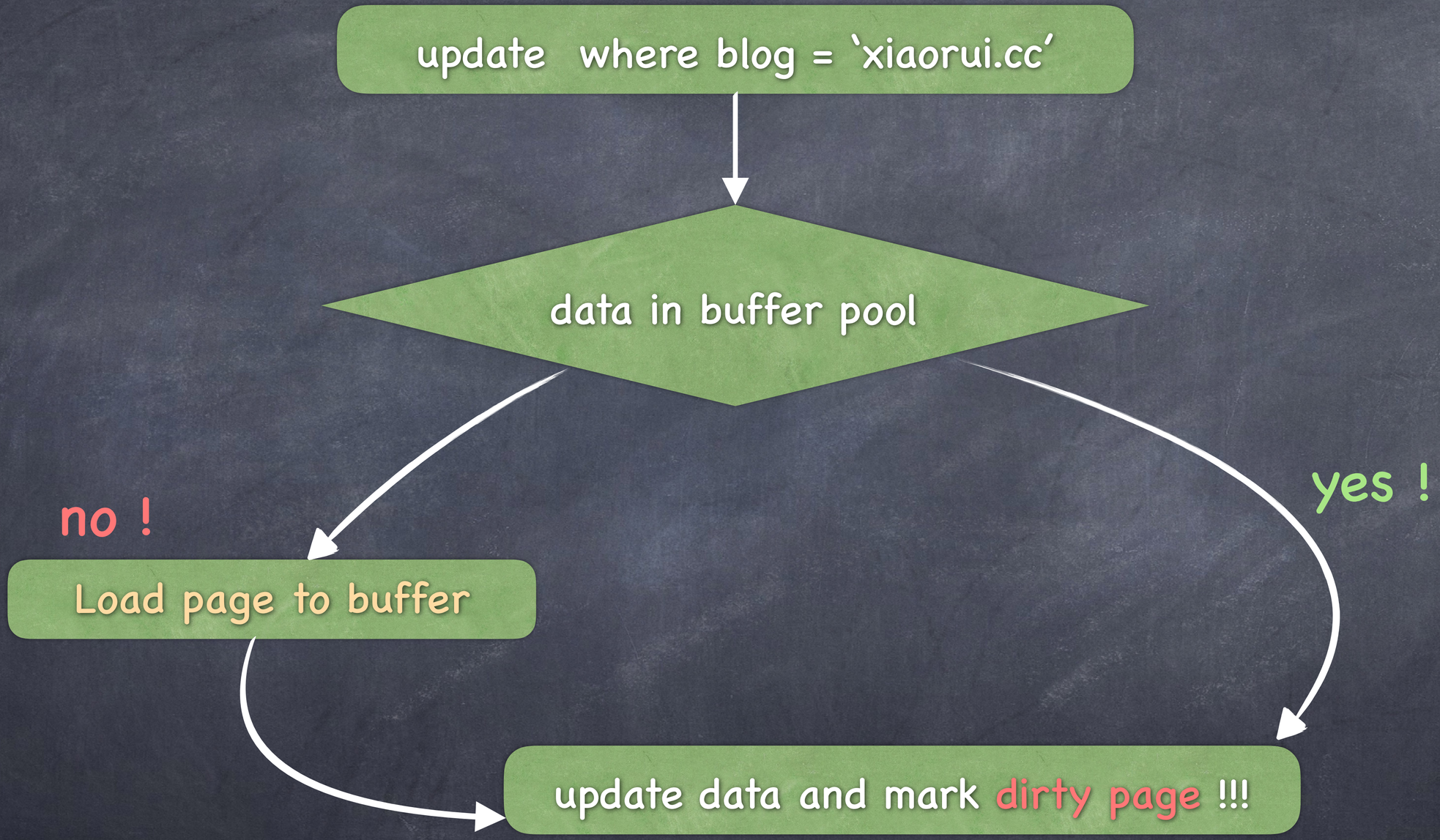
- mvcc
- 隔离级别
  - READ UNCOMMITTED
  - READ COMMITTED
  - REPEATABLE READ
  - SERIALIZABLE

durability

- 一旦提交成功，那么事务涉及的数据都会持久化
- 即使系统崩溃，修改的数据也不会丢失

innodb buffer pool




update where blog = 'xiaorui.cc'
data in buffer pool
no !
Load page to buffer
yes !
update data and mark dirty page !!!

innodb buffer pool

- lru
- dirty page
- free list
- flush list

一些概念总览

- redo
- undo
- checkpoint
- purge

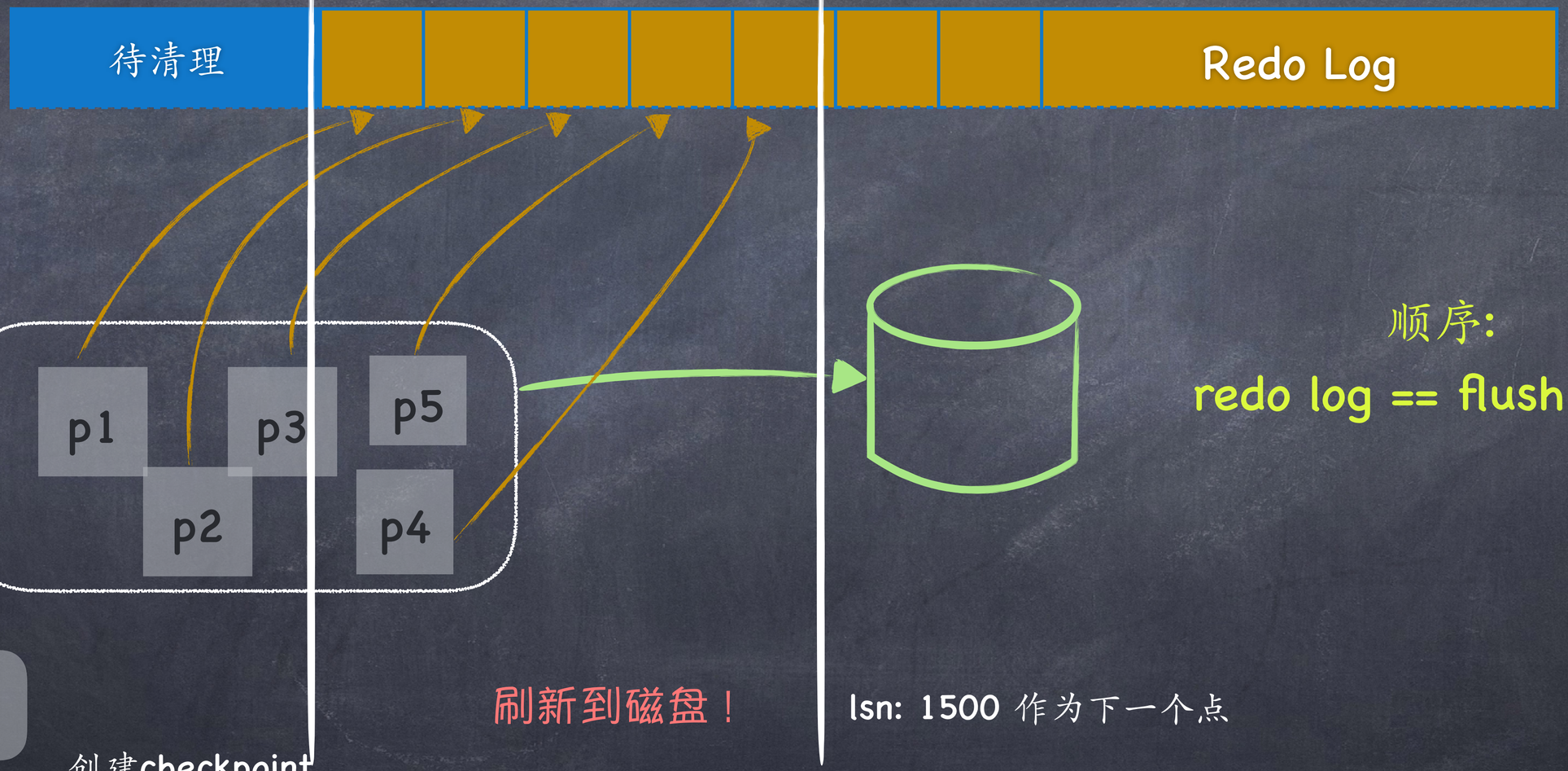
if not checkpoint

- 写操作都是在buffer pool进行，每次提交都会顺序io增加redo日志.
  - buffer pool里面的dirty page越来越多 .
  - redo 越来越大，导致recovery时间过长 .

if checkpoint

- 当 buffer pool 内存不够用时, 将dirty page刷新到文件.
- 当redo文件过大又不可循环写入时，同上
- 缩短crash recovery时间

checkpoint + lsn



一个事务过程

1. begin
2. undo log
3. udate where id = xxx;
4. redo buffer
5. redo fsync; binlog; commit

if crash ?

if crash ?

if crash ?

one of three happen crash ?

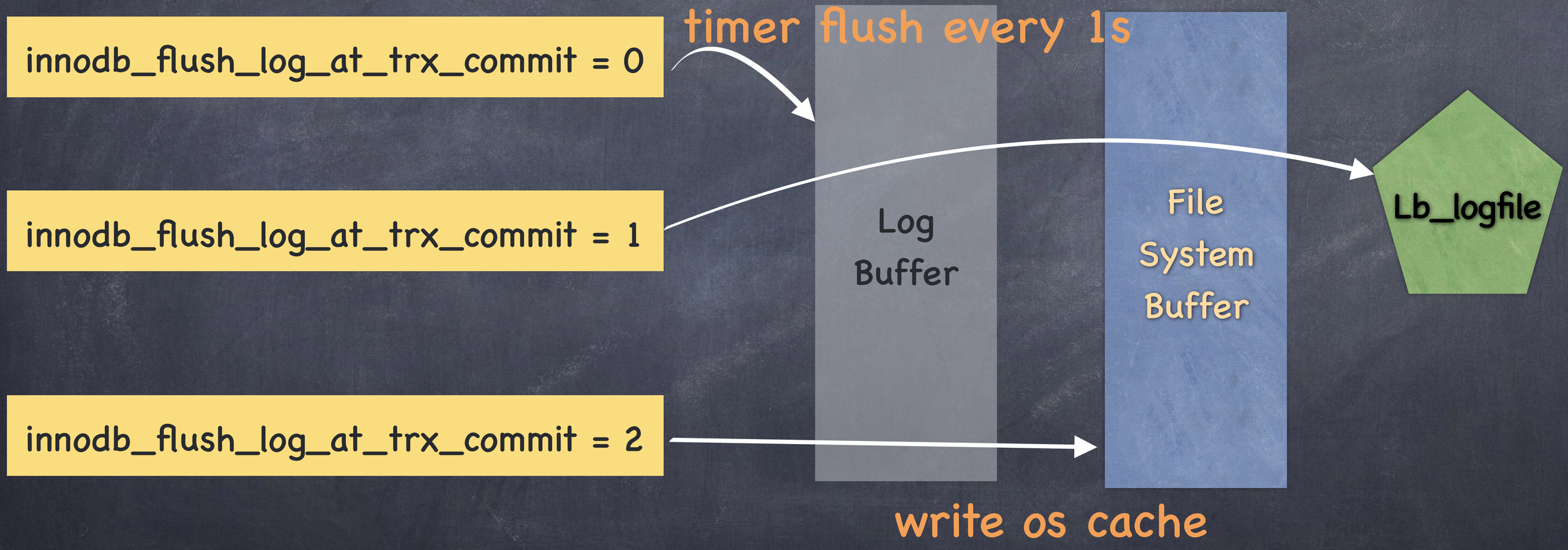
redo

- when redo save ?
  - when commit !
- 顺序io
- page number + block
- when redo reduce ?
  - Ringbuffer full
  - flush timer run

undo

- 随机io
- 是否存入文件，取决于buffer pool

flush log rule



数据库并发处理的演变

- 基于表的读写锁
- 基于索引的读写锁 (分离锁)
- mvcc

mvcc

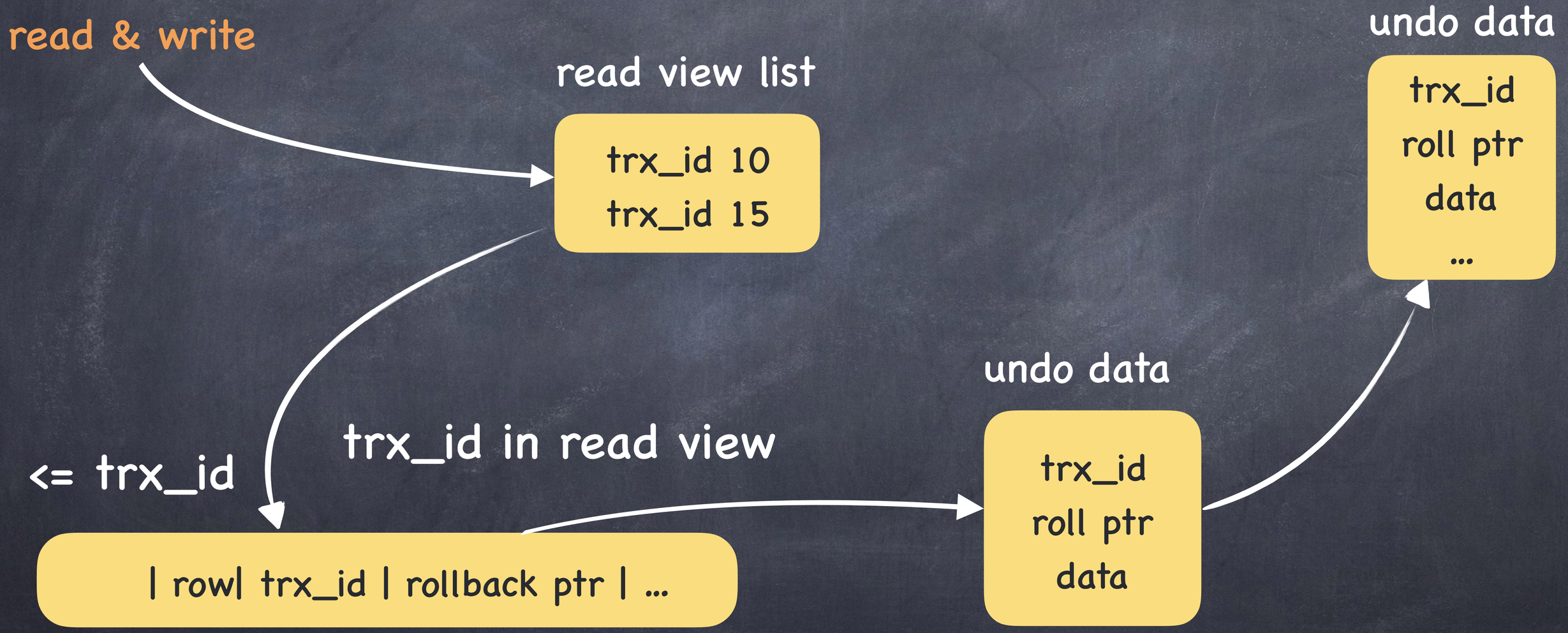
- 读读非阻塞
- 读写非阻塞
- 写写阻塞

一句话，并发的效果，串行的结果 !!!

又是概念

- read view
- 隐藏字段
- 是否可见的条件
- 当前读
- 快照读

mvcc流程图


read & write
read view list
trx_id 10
trx_id 15
undo data
trx_id
roll ptr
data
...
trx_id in read view
<= trx_id
undo data
trx_id
roll ptr
data
| row| trx_id | rollback ptr | ...

幻读

- record lock
- gap lock
- next gap lock

业务数据不一致

	事务 A	事务B
ts	begin	
ts	select stock where ticket=10 stock=1	
ts		begin
ts	update stock=stock-1 stock=0	
ts	commit	
ts		select stock where ticket = 10
ts		update stock=stock-1 stock=-1
ts		commit

最后 -1

互斥锁 vs 乐观锁

- 互斥锁
  - select * from train where ticket = 10 for update;
- 乐观锁
  - while 1:
    - update train set stock=stock-1 where ticket = 10 and stock = x;

crash recovery

- 从doublewrite 修正缺斤少两的page
- 重做提交的redo日志
- 回滚执行一半的 未提交的 事务

" Q & A"

-fengyun rui