

binlog

目录

- 1 binlog 和主库性能
- 2 binlog 和主备同步性能
- 3 binlog 和数据恢复速度
- 4 binlog 和主备切换速度

binlog 和 数据恢复速度

开了 binlog 以后，可以做哪些准备，应对删库不用跑路？

秒级恢复方案： MVCC 闪回， binlog 闪回、回收站、延迟备份

小时级恢复方案： 全量 + binlog 回放（过滤+并行）

binlog 和数据恢复速度：MVCC 闪回

原理：利用 InnoDB RR 隔离级别特性

session1	session2	session3
START TRANSACTION WITH CONSISTENT SNAPSHOT; (id,c)=(1,1)		
	update t1 set c=c+1 where id=1 (2,1)	
		update t1 set c=c*10 where id=1 (20,1)
select id,c from t1 ; (id,c)=(1,1)		

binlog 和数据恢复速度：binlog 闪回

```
for transaction N .. 1:  
  reverse events in trans  
  for events in trans  
    insert -> delete  
    update A to B -> update B to A  
    delete -> insert  
  end for  
end for
```

对 binlog_row_image 的要求？

binlog 和数据恢复速度：回收站

drop / truncate table --> rename

binlog 和数据恢复速度：延迟备份



binlog 和数据恢复速度：全量 + binlog 回放

有哪些加速 binlog 回放的方法

1. sync_binlog 和 innodb_flush_log_at_trx_commit 关掉双1 (甚至关掉 binlog)
2. 只回放目标表
3. 并行回放 (模拟 relaylog)

目录

- 1 binlog 和主库性能
- 2 binlog 和主备同步性能
- 3 binlog 和数据恢复速度
- 4 binlog 和主备切换速度

影响主备切换速度的主要环节

- 探测
- 切换
- 恢复服务

分别怎么加速？



主备切换之：探测

各种方法和 bad case

ping

connect

connect + select 1

connect + select from . .. limit 1;

connect + update ... set last_monitor_time = now();

频率、异常处理和重试

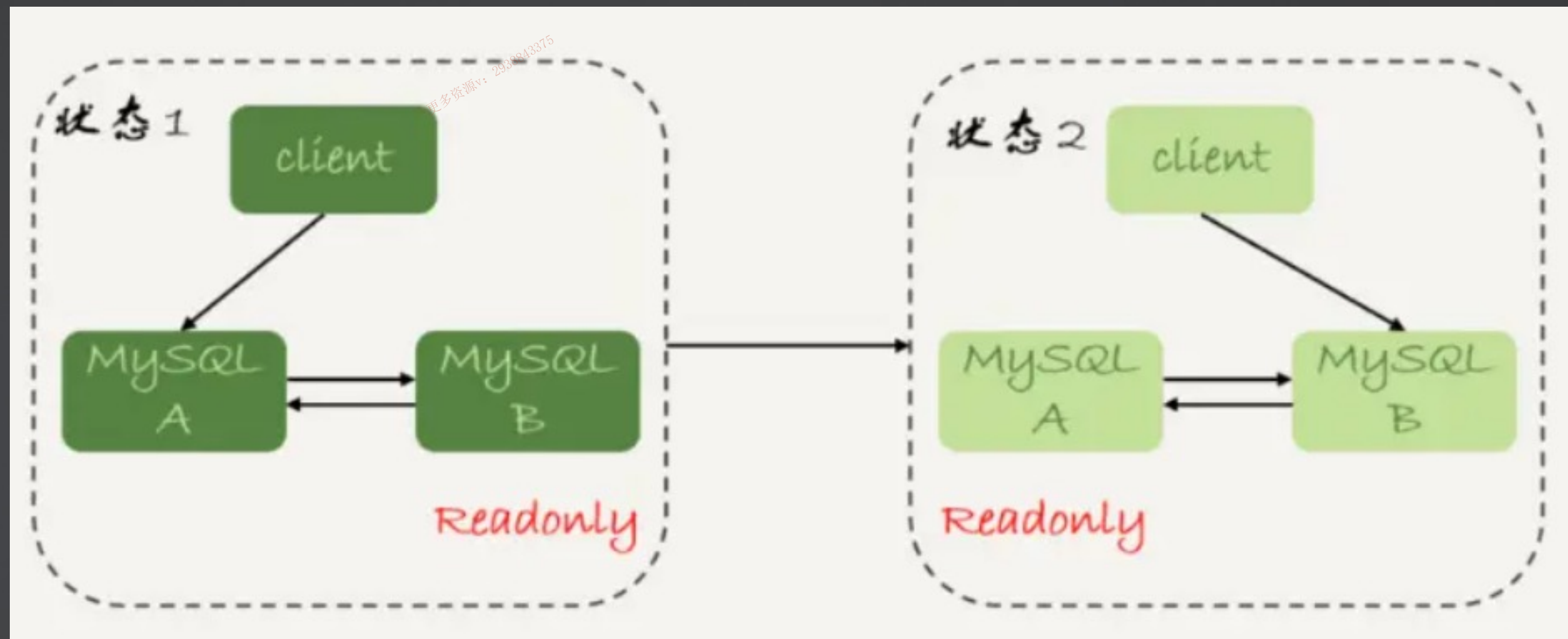
多久执行一次?

重试几次?

主备切换之：切换

基本流程：

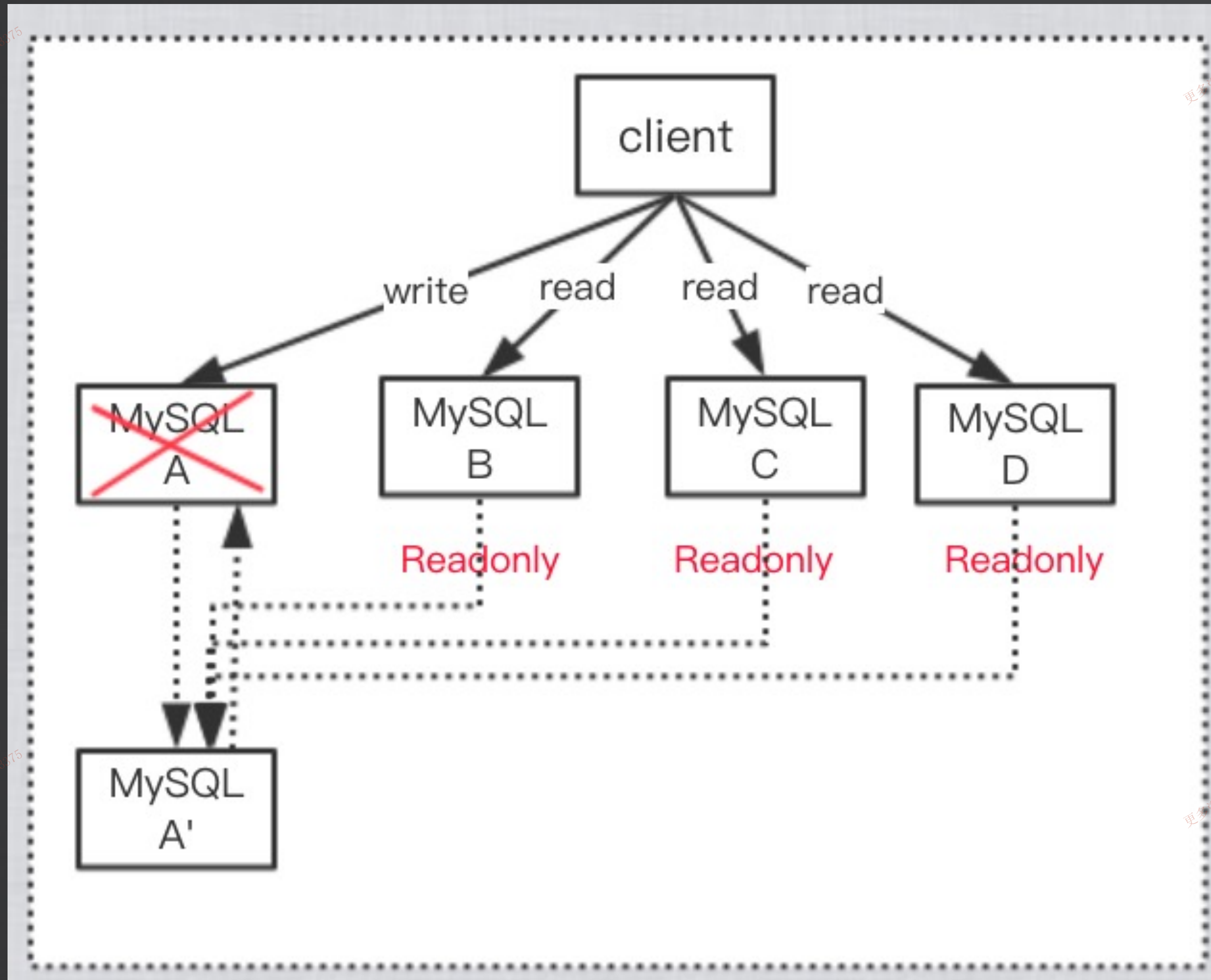
- 等延迟 < 1
- 双 RO
- 等 slave 完全追上
- 切业务流量
- [数据一致性校验]
- 新主关掉 RO
- 修改主备关系



如何判断 slave 完全追上：pos vs GTID?

$\text{immediate_commit_timestamp} -$
 $\text{original_commit_timestamp}$

主备切换之：切换 - 多读



主备切换之：切换 - 多读

```
gtid-set uuid_A:[1-3]
```

```
gtid-no uuid_A:1; uuid_A:2;uuid_A:3
```

```
uuid_B:[1-10000];
```

```
update uuid_A:[1-4]
```

```
set gtid_next='uuid_B:50';
```

```
begin;
```

```
commit;
```

```
gtid-set uuid_A:[1-4];uuid_B[50]
```

```
gtid-set uuid_A:[1-4];uuid_B[1-50]
```

```
relay_log uuid_B:[31-100]
```

```
gtid-set uuid_A:[1-4];uuid_B[1-2]
```

```
skip_slave_error=1062,1032
```

```
gtid-set uuid_A:[1-4];uuid_B[1-9980]
```


Q&A

THANKS