

# MySQL 的线程

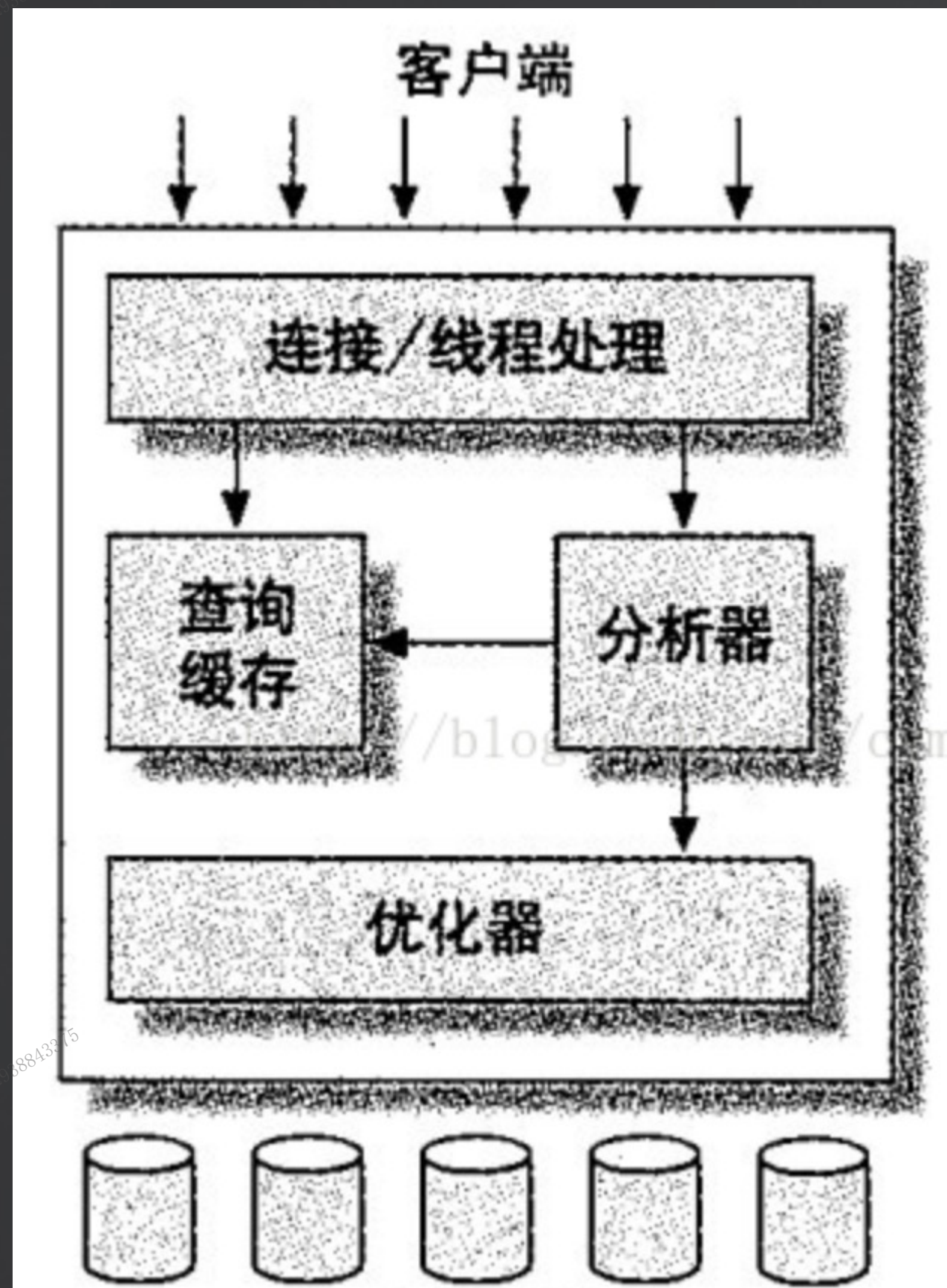
# 目录

1 MySQL 分层模型

2 Server 层线程

3 InnoDB 线程

# MySQL 分层模型



# 目录

1 MySQL 分层模型

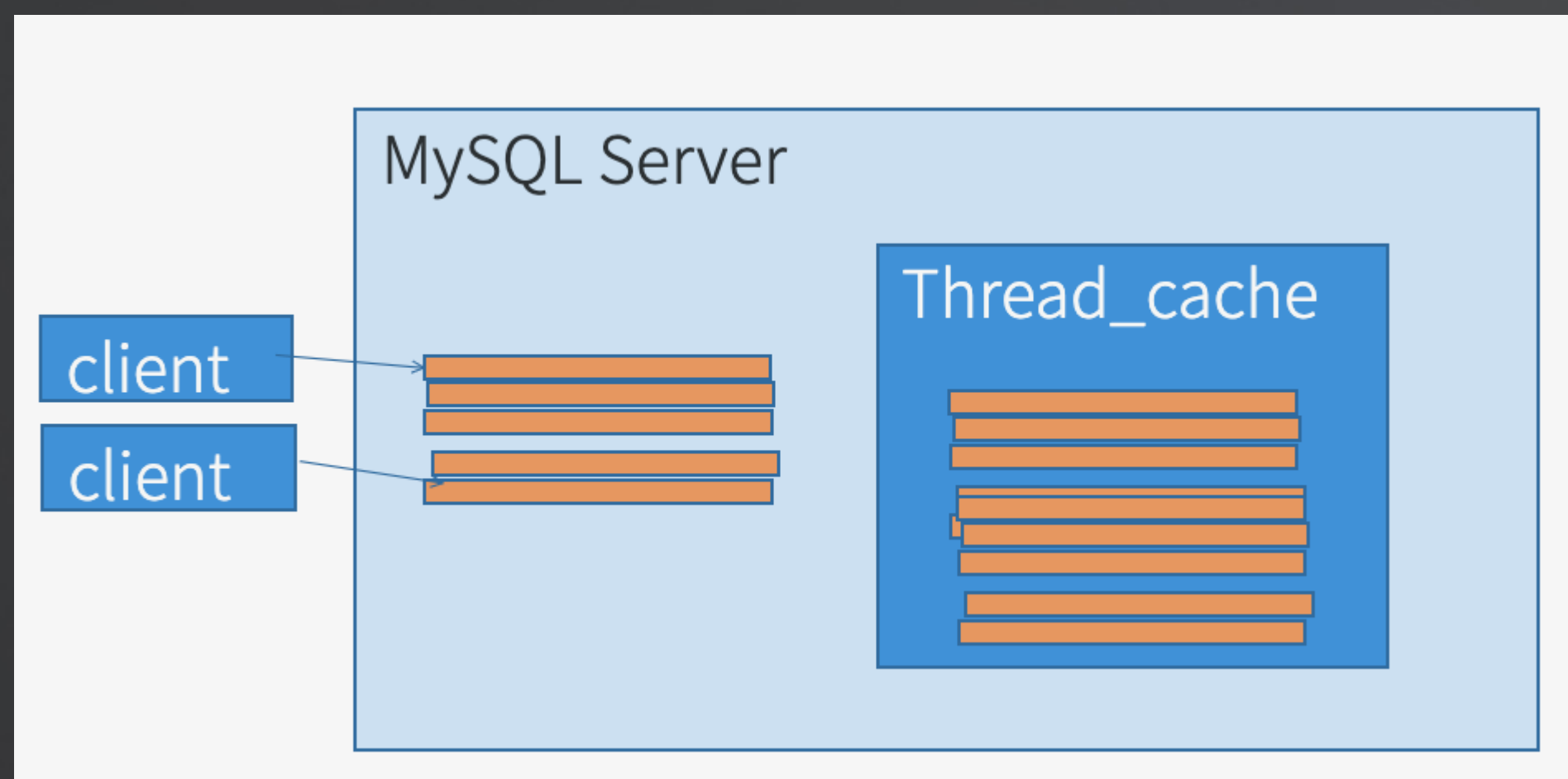
2 Server 层线程

3 InnoDB 线程

# Server 层线程 - 用户线程

参数: **thread\_cache\_size**

默认值:  $\min(8 + \max\_connections / 100, 100)$





# Server 层线程 - 用户线程三种状态

session1	session2	session3
connect quit	connect	connect
	begin; select * from t for update	
		select * from t for update

等连接 : block\_until\_new\_connection  
长链接空闲: Protocol\_classic::read\_packet  
执行状态 : mysql\_execute\_command

# Server 层线程 - 用户线程三种状态

```
Thread 41 (Thread 0x7fec284f2700 (LWP 1171631)):  
#0  0x00007fec4aadf3fc in pthread_cond_wait@@GLIBC_2.3.2 () from /lib64/libpthread.so.0  
#1  0x000000000499a90f in native_cond_wait (cond=0x86b68e0 <Per_thread_connection_handler::COND_thread_cache>, mutex=0xc31f178)  
d.h:109  
#2  0x000000000499aa7a in safe_cond_wait (cond=0x86b68e0 <Per_thread_connection_handler::COND_thread_cache>, mp=0xc31f150, file=  
n_handler/connection_handler_per_thread.cc", line=162) at /root/dq/mysql-8.0.39/mysys/thr_cond.cc:72  
#3  0x000000000358cfe0 in my_cond_wait (cond=0x86b68e0 <Per_thread_connection_handler::COND_thread_cache>, mp=0x86b68a0 <Per_thr  
e>, file=0x5f97dc8 "/root/dq/mysql-8.0.39/sql/conn_handler/connection_handler_per_thread.cc", line=162) at /root/dq/mysql-8.0.39  
#4  0x000000000358d124 in inline_mysql_cond_wait (that=0x86b68e0 <Per_thread_connection_handler::COND_thread_cache>, mutex=0x86b  
K_thread_cache>, src_file=0x5f97dc8 "/root/dq/mysql-8.0.39/sql/conn_handler/connection_handler_per_thread.cc", src_line=162) at  
mysql_cond.h:181  
#5  0x000000000358d5c8 in Per_thread_connection_handler::block_until_new_connection () at /root/dq/mysql-8.0.39/sql/conn_handler  
#6  0x000000000358dad7 in handle_connection (arg=0xc6baca0) at /root/dq/mysql-8.0.39/sql/conn_handler/connection_handler_per_thr  
#7  0x00000000051b8909 in pfs_spawn_thread (arg=0xc499870) at /root/dq/mysql-8.0.39/storage/perfschema/pfs.cc:3050  
#8  0x00007fec4aad917a in start_thread () from /lib64/libpthread.so.0  
#9  0x00007fec48e7ddc3 in clone () from /lib64/libc.so.6
```

# Server 层线程 - 用户线程三种状态

```
Thread 39 (Thread 0x7fec286f6700 (LWP 1161427)):  
#0 0x00007fec48e72b36 in ppoll () from /lib64/libc.so.6  
#1 0x000000000550c22c in vio_io_wait (vio=0x7febcc000ef0, event=VIO_IO_EVENT_READ, timeout=28800000) at /:  
#2 0x000000000550ab1f in vio_socket_io_wait (vio=0x7febcc000ef0, event=VIO_IO_EVENT_READ) at /root/dq/mys:  
#3 0x000000000550cc3b in vio_ssl_read (vio=0x7febcc000ef0, buf=0x7febcc006dd0 "\001", size=4) at /root/dq:  
#4 0x0000000003564147 in net_read_raw_loop (net=0x7febcc003128, count=4) at /root/dq/mysql-8.0.39/sql-com:  
#5 0x000000000356471c in net_read_packet_header (net=0x7febcc003128) at /root/dq/mysql-8.0.39/sql-common/  
#6 0x0000000003565a52 in net_read_packet (net=0x7febcc003128, complen=0x7fec286f52c8) at /root/dq/mysql-8.  
#7 0x0000000003565d08 in net_read_uncompressed_packet (net=0x7febcc003128, len=@0x7fec286f5308: 52159872)  
#8 0x0000000003565ff6 in my_net_read (net=0x7febcc003128) at /root/dq/mysql-8.0.39/sql-common/net_serv.cc:  
#9 0x0000000003a04e9c in Protocol_classic::read_packet (this=0x7febcc0054f0) at /root/dq/mysql-8.0.39/sql.  
#10 0x0000000003a06149 in Protocol_classic::get_command (this=0x7febcc0054f0, com_data=0x7fec286f5aa0, cmd:  
.cc:2992  
#11 0x0000000003376da3 in do_command (thd=0x7febcc001330) at /root/dq/mysql-8.0.39/sql/sql_parse.cc:1381  
#12 0x000000000358da25 in handle_connection (arg=0xc6fc5b0) at /root/dq/mysql-8.0.39/sql/conn_handler/conn:  
#13 0x00000000051b8909 in pfs_spawn_thread (arg=0xc6b7af0) at /root/dq/mysql-8.0.39/storage/perfschema/pfs.  
#14 0x00007fec4aad917a in start_thread () from /lib64/libpthread.so.0  
#15 0x00007fec48e7ddc3 in clone () from /lib64/libc.so.6
```



# Server 层线程 - 用户线程三种状态

```
Thread 40 (Thread 0x7fec285f4700 (LWP 1162954)):  
#0 0x00007fec4aadf3fc in pthread_cond_wait@@GLIBC_2.3.2 () from /lib64/libpthread.so.0  
#1 0x0000000004bcdcdc in os_event::wait (this=0x7fec8029f10) at /root/dq/mysql-8.0.39/storage/innobase/os/os0event.cc:186  
#2 0x0000000004bccbf8 in os_event::wait_low (this=0x7fec8029f10, reset_sig_count=3) at /root/dq/mysql-8.0.39/storage/innobase/os/os0event.cc:366  
#3 0x0000000004bcd1cf in os_event_wait_low (event=0x7fec8029f10, reset_sig_count=0) at /root/dq/mysql-8.0.39/storage/innobase/os/os0event.cc:590  
#4 0x0000000004b19d0a in os_event_wait (e=0x7fec8029f10) at /root/dq/mysql-8.0.39/storage/innobase/include/os0event.h:104  
#5 0x0000000004b1b202 in lock_wait_suspend_thread (thr=0x7fec80ad7f8) at /root/dq/mysql-8.0.39/storage/innobase/lock/lock0wait.cc:297  
#6 0x0000000004c409d7 in row_mysql_handle_errors (new_err=0x7fec285efdbc, trx=0x7fec406f13e8, thr=0x7fec80ad7f8, savept=0x0) at /root/dq/mysql-8.0.39/row/row0mysql.cc:711  
#7 0x0000000004ca7300 in row_search_mvcc (buf=0x7fec8013e20 "\377", mode=PAGE_CUR_G, prebuilt=0x7fec80acde8, match_mode=0, direction=0) at /root/dq/mysql-8.0.39/storage/innobase/row/row0sel.cc:5918  
#8 0x0000000004a08c72 in ha_innobase::index_read (this=0x7fec80ab690, buf=0x7fec8013e20 "\377", key_ptr=0x0, key_len=0, find_flag=HA_READ_AFTER_KEY) at /root/dq/mysql-8.0.39/storage/innobase/handler/ha_innobase.cc:10289  
#9 0x0000000004a09dad in ha_innobase::index_first (this=0x7fec80ab690, buf=0x7fec8013e20 "\377") at /root/dq/mysql-8.0.39/storage/innobase/handler/ha_innobase.cc:10300  
#10 0x0000000004a0a54f in ha_innobase::rnd_next (this=0x7fec80ab690, buf=0x7fec8013e20 "\377") at /root/dq/mysql-8.0.39/storage/innobase/handler/ha_innobase.cc:10311  
#11 0x000000000373b91c in handler::ha_rnd_next (this=0x7fec80ab690, buf=0x7fec8013e20 "\377") at /root/dq/mysql-8.0.39/sql/handler.cc:2969  
#12 0x00000000038e7f02 in TableScanIterator::Read (this=0x7fec8036da0) at /root/dq/mysql-8.0.39/sql/iterators/basic_row_iterators.cc:220  
#13 0x00000000034b58bc in Sql_cmd_update::update_single_table (this=0x7fec8036328, thd=0x7fec8007040) at /root/dq/mysql-8.0.39/sql/sql_update.cc:87  
#14 0x00000000034b8942 in Sql_cmd_update::execute_inner (this=0x7fec8036328, thd=0x7fec8007040) at /root/dq/mysql-8.0.39/sql/sql_update.cc:1818  
#15 0x000000000340131c in Sql_cmd_dml::execute (this=0x7fec8036328, thd=0x7fec8007040) at /root/dq/mysql-8.0.39/sql/sql_select.cc:794  
#16 0x000000000337deb9 in mysql_execute_command (thd=0x7fec8007040, first_level=true) at /root/dq/mysql-8.0.39/sql/sql_parse.cc:3689  
#17 0x00000000033835d7 in dispatch_sql_command (thd=0x7fec8007040, parser_state=0x7fec285f29a0) at /root/dq/mysql-8.0.39/sql/sql_parse.cc:5371  
#18 0x000000000337926f in dispatch_command (thd=0x7fec8007040, com_data=0x7fec285f3aa0, command=COM_QUERY) at /root/dq/mysql-8.0.39/sql/sql_parse.cc:1440  
#19 0x0000000003377153 in do_command (thd=0x7fec8007040) at /root/dq/mysql-8.0.39/sql/sql_parse.cc:1440  
#20 0x000000000358da25 in handle_connection (arg=0xc6baa80) at /root/dq/mysql-8.0.39/sql/conn_handler/connection_handler_per_thread.cc:303  
#21 0x00000000051b8909 in pfs_spawn_thread (arg=0xc6b6870) at /root/dq/mysql-8.0.39/storage/perfschema/pfs.cc:3050
```

# 关于线程的基础知识回顾

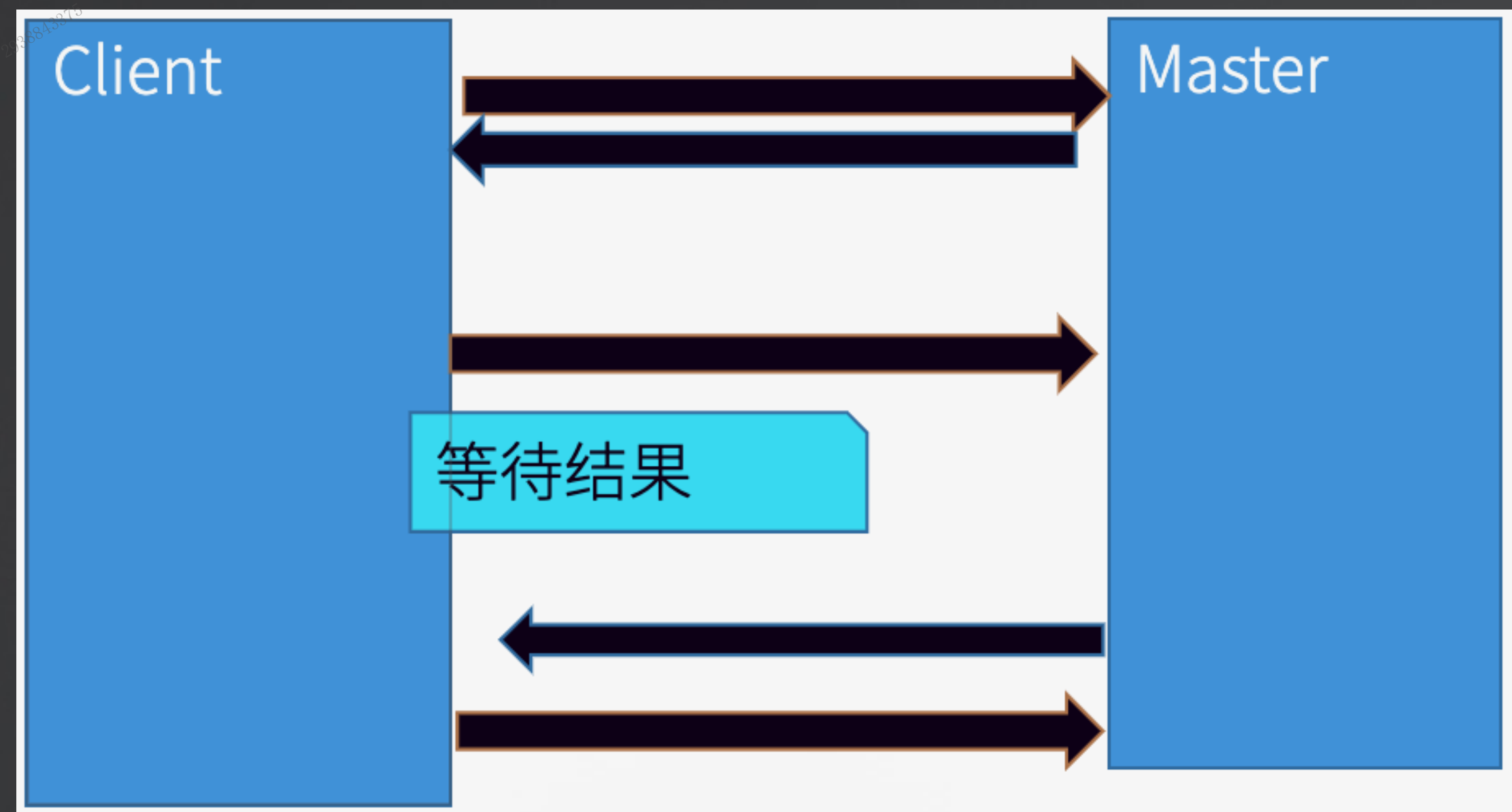
1. MySQL 的线程，跟操作系统线程对应的。
2. 可以在 shell 对进程/线程发 kill 信号。
3. 可以在 MySQL 客户端执行 `kill query/connection <threadid>` 的方式，停止一个查询，或者断开连接。

kill query/connection id

**课堂练习：指出下面这个描述的两处错误**

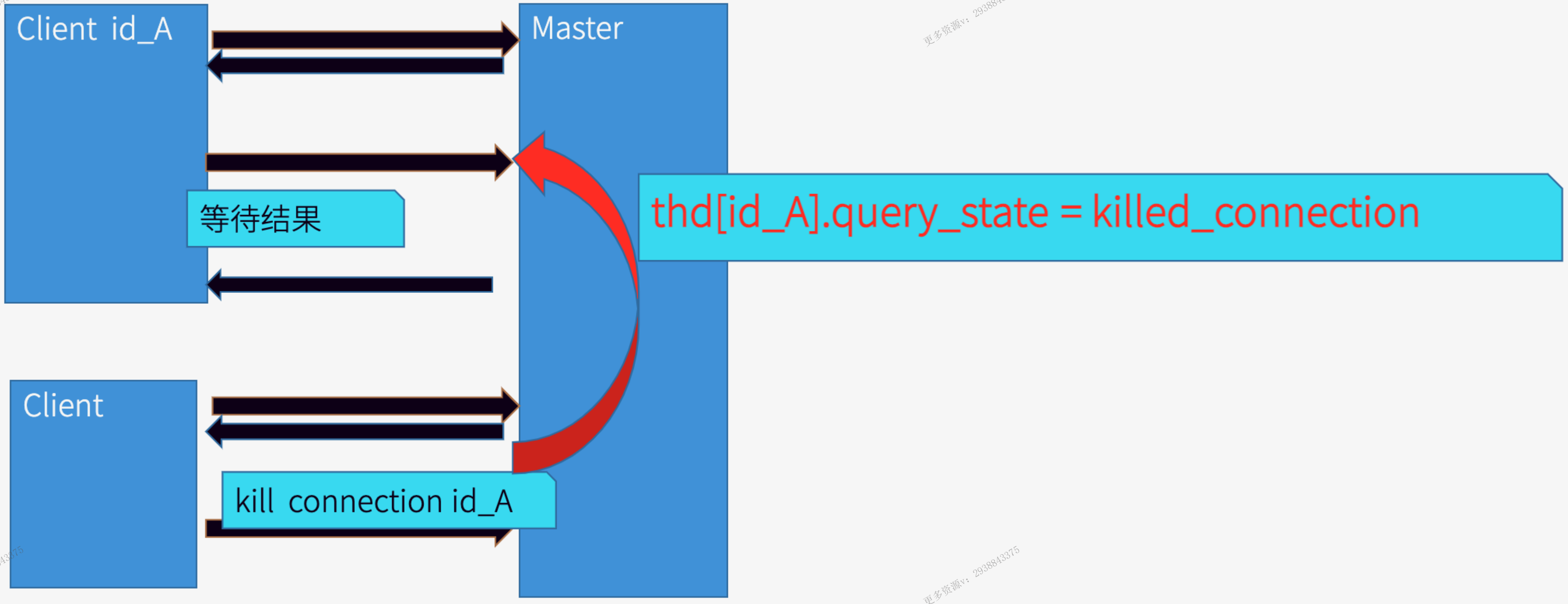
客户端发了一个update语句后，长时间不返回，这时候从这个客户端执行ctrl+c，  
是直接从本客户端线程发一个命令，直接终止服务端对应的线程执行

# MySQL 客户端协议： 停等协议





# MySQL 客户端协议： 停等协议



# 课堂练习：一个线程在 I/O 排队的时候，对它 kill query 是什么现象？

**情况1： 在 MySQL InnoDB 内排队**

- A. 马上终止语句，直接退出
- B. 马上终止语句，进入回滚事务
- C. 完成io请再在开始终止语句逻辑

**情况2： 在操作系统排队**

- A. 马上终止语句，直接退出
- B. 马上终止语句，进入回滚事务
- C. 完成io请再在开始终止语句逻辑

# Server 层线程 - 监听线程

负责监听新连接请求

从 thread\_cache 或 新建 thread 接管

Mysqld\_socket\_listener::listen\_for\_connection\_event

课堂问答：密码验证这个逻辑，是在监听线程做的，还是用户线程做的？

# Server 层线程 - 监听线程 验证方法

gdb 挂在 b my\_error 函数

mysql -h -u -pERRORPWD

```
(gdb) bt
#0  my_error (nr=1045, MyFlags=0) at /root/dq/mysql-8.0.39/mysys/my_error.cc:219
#1  0x0000000003612ff9 in login_failed_error (thd=0x7f63e8000ed0, mpvio=0x7f64483f1060, passwd_used=0) at /root/dq/mysql-8.0.39/sql/auth/sql_authentication.cc:1474
#2  0x0000000003619d95 in acl_authenticate (thd=0x7f63e8000ed0, command=COM_CONNECT) at /root/dq/mysql-8.0.39/sql/auth/sql_authentication.cc:3981
#3  0x00000000032f30a9 in check_connection (thd=0x7f63e8000ed0) at /root/dq/mysql-8.0.39/sql/sql_connect.cc:652
#4  0x00000000032f324c in login_connection (thd=0x7f63e8000ed0) at /root/dq/mysql-8.0.39/sql/sql_connect.cc:706
#5  0x00000000032f4219 in thd_prepare_connection (thd=0x7f63e8000ed0) at /root/dq/mysql-8.0.39/sql/sql_connect.cc:893
#6  0x000000000358d9f7 in handle_connection (arg=0xbbe4090) at /root/dq/mysql-8.0.39/sql/conn_handler/connection_handler_per_thread.cc:299
#7  0x00000000051b8909 in pfs_spawn_thread (arg=0xbb9e0a0) at /root/dq/mysql-8.0.39/storage/perfschema/pfs.cc:3050
#8  0x00007f646586817a in start_thread () from /lib64/libpthread.so.0
#9  0x00007f6463c0cdc3 in clone () from /lib64/libc.so.6
```

```
(gdb) info threads
```

```
38  Thread 0x7f64484f4700 (LWP 1110076) "connection" 0x00007f6463c01b30 in pfs_spawn_thread () from /lib64/libc.so.6
* 39  Thread 0x7f64483f2700 (LWP 1110087) "connection" my_error (nr=1045, MyFlags=0) at /root/dq/mysql-8.0.39/mysys/my_error.cc:219
( " )
```



# Server 层线程 - 监听线程 验证方法

```
Thread 39 (Thread 0x7f64483f2700 (LWP 1110087)):  
#0 0x00007f646586e3fc in pthread_cond_wait@@GLIBC_2.3.2 () from /lib64/libpthread.so.0  
#1 0x000000000499a90f in native_cond_wait (cond=0x86b68e0 <Per_thread_connection_handler::COND_thread_cache>, mutex=0xb807178) at /root/dq/d.h:109  
#2 0x000000000499aa7a in safe_cond_wait (cond=0x86b68e0 <Per_thread_connection_handler::COND_thread_cache>, mp=0xb807150, file=0x5f97dc8 "/n_handler/connection_handler_per_thread.cc", line=162) at /root/dq/mysql-8.0.39/mysys/thr_cond.cc:72  
#3 0x000000000358cfe0 in my_cond_wait (cond=0x86b68e0 <Per_thread_connection_handler::COND_thread_cache>, mp=0x86b68a0 <Per_thread_connection_handler::COND_thread_cache>, file=0x5f97dc8 "/root/dq/mysql-8.0.39/sql/conn_handler/connection_handler_per_thread.cc", line=162) at /root/dq/mysql-8.0.39/include/thr_cond.h:181  
#4 0x000000000358d124 in inline_mysql_cond_wait (that=0x86b68e0 <Per_thread_connection_handler::COND_thread_cache>, mutex=0x86b68a0 <Per_thread_connection_handler::COND_thread_cache>, src_file=0x5f97dc8 "/root/dq/mysql-8.0.39/sql/conn_handler/connection_handler_per_thread.cc", src_line=162) at /root/dq/mysql-8.0.39/sql/conn_handler/connection_handler_per_thread.cc:181  
#5 0x000000000358d5c8 in Per_thread_connection_handler::block_until_new_connection () at /root/dq/mysql-8.0.39/sql/conn_handler/connection_handler_per_thread.cc:335  
#6 0x000000000358dad7 in handle_connection (arg=0xbbe4090) at /root/dq/mysql-8.0.39/sql/conn_handler/connection_handler_per_thread.cc:305  
#7 0x00000000051b8909 in pfs_spawn_thread (arg=0xbb9e0a0) at /root/dq/mysql-8.0.39/storage/perfschema/pfs.cc:3050  
#8 0x00007f646586817a in start_thread () from /lib64/libpthread.so.0  
#9 0x00007f6463c0cdc3 in clone () from /lib64/libc.so.6
```

# Server 层线程 - 两个 GTID 相关的线程

~~compress\_gtid\_table~~

gtid\_executed\_compression\_period

Clone\_persist\_gtid::periodic\_write

s\_time\_threshold=100

s\_compression\_threshold=50

# Server 层线程 - 两个 GTID 相关的线程

```
mysql> select * from mysql.gtid_executed;
```

source_uuid	interval_start	interval_end
e72b91ff-61a8-11ef-9d42-fa163e0574c5	1	2
e72b91ff-61a8-11ef-9d42-fa163e0574c5	3	3
e72b91ff-61a8-11ef-9d42-fa163e0574c5	4	4
e72b91ff-61a8-11ef-9d42-fa163e0574c5	5	5
e72b91ff-61a8-11ef-9d42-fa163e0574c5	6	6
e72b91ff-61a8-11ef-9d42-fa163e0574c5	7	7
e72b91ff-61a8-11ef-9d42-fa163e0574c5	8	8
e72b91ff-61a8-11ef-9d42-fa163e0574c5	9	9
e72b91ff-61a8-11ef-9d42-fa163e0574c5	10	10
e72b91ff-61a8-11ef-9d42-fa163e0574c5	11	11
e72b91ff-61a8-11ef-9d42-fa163e0574c5	12	12
e72b91ff-61a8-11ef-9d42-fa163e0574c5	13	13
e72b91ff-61a8-11ef-9d42-fa163e0574c5	14	14
e72b91ff-61a8-11ef-9d42-fa163e0574c5	15	15
e72b91ff-61a8-11ef-9d42-fa163e0574c5	16	16
e72b91ff-61a8-11ef-9d42-fa163e0574c5	17	17
e72b91ff-61a8-11ef-9d42-fa163e0574c5	18	18

```
mysql> select * from mysql.gtid_executed;
```

source_uuid	interval_start	interval_end
e72b91ff-61a8-11ef-9d42-fa163e0574c5	1	51

1 row in set (0.00 sec)



# Server 层线程 - 事件调度器线程

## Event\_scheduler::run

```
#0 0x00007f95a38f074a in pthread_cond_timedwait@@GLIBC_2.3.2 () from /lib64/libpthread.so.0
#1 0x000000000499a8ea in native_cond_timedwait (cond=0xad41550, mutex=0xb051938, abstime=0x7f95903e8600, f
#2 0x000000000499ac5b in safe_cond_timedwait (cond=0xad41550, mp=0xb051910, abstime=0x7f95903e8600, f
line=721) at /root/dq/mysql-8.0.39/mysys/thr_cond.cc:114
#3 0x0000000003ad2110 in my_cond_timedwait (cond=0xad41550, mp=0xad41520, abstime=0x7f95903e8600, fil
at /root/dq/mysql-8.0.39/include/thr_cond.h:147
#4 0x0000000003ad2332 in inline_mysql_cond_timedwait (that=0xad41550, mutex=0xad41520, abstime=0x7f95
src_file=0x6145038 "/root/dq/mysql-8.0.39/sql/event_queue.cc", src_line=721) at /root/dq/mysql-8.0.
#5 0x0000000003ad474a in Event_queue::cond_wait (this=0xad41520, thd=0x7f9520132680, abstime=0x7f9590
src_func=0x61467a0 <Event_queue::get_top_for_execution_if_time(THD*, Event_queue_element_for_exec*
src_file=0x6145038 "/root/dq/mysql-8.0.39/sql/event_queue.cc", src_line=580) at /root/dq/mysql-8.0.
#6 0x0000000003ad3f07 in Event_queue::get_top_for_execution_if_time (this=0xad41520, thd=0x7f95201326
at /root/dq/mysql-8.0.39/sql/event_queue.cc:579
#7 0x0000000003ad9ae6 in Event_scheduler::run (this=0xaf8ad10, thd=0x7f9520132680) at /root/dq/mysql-
#8 0x0000000003ad8902 in event_scheduler_thread (arg=0x7f95200c7e90) at /root/dq/mysql-8.0.39/sql/ever
#9 0x00000000051b8909 in pfs_spawn_thread (arg=0x7f95200265d0) at /root/dq/mysql-8.0.39/storage/perfs
#10 0x00007f95a38ea17a in start_thread () from /lib64/libpthread.so.0
#11 0x00007f95a1c8edc3 in clone () from /lib64/libc.so.6
#10 0x00007f95a38ea17a in start_thread () from /lib64/libpthread.so.0
#11 0x00007f95a1c8edc3 in clone () from /lib64/libc.so.6
```



# Server 层线程 - 事件调度器线程

```
#0  mysql_execute_command (thd=0x7f9564013540, first_level=false) at /root/dq/mysql-8.0.39/sql/sql_parse.c:151
#1  0x0000000003252b53 in sp_instr_stmt::exec_core (this=0x7f956c00ba58, thd=0x7f9564013540, nextp=0x7f95902e6970) at /root/dq/mysql-8.0.39/sql/sp_instr.cc:101
#2  0x00000000032517ec in sp_lex_instr::reset_lex_and_exec_core (this=0x7f956c00ba58, thd=0x7f9564013540, nextp=0x7f95902e6970) at /root/dq/mysql-8.0.39/sql/sp_instr.cc:462
#3  0x0000000003252272 in sp_lex_instr::validate_lex_and_execute_core (this=0x7f956c00ba58, thd=0x7f9564013540, nextp=0x7f95902e6970) at /root/dq/mysql-8.0.39/sql/sp_instr.cc:746
#4  0x0000000003252848 in sp_instr_stmt::execute (this=0x7f956c00ba58, thd=0x7f9564013540, nextp=0x7f95902e6970) at /root/dq/mysql-8.0.39/sql/sp_instr.cc:101
#5  0x000000000324269a in sp_head::execute (this=0x7f956c007b00, thd=0x7f9564013540, merge_da_on_success=true) at /root/dq/mysql-8.0.39/sql/sp_head.cc:101
#6  0x0000000003244579 in sp_head::execute_procedure (this=0x7f956c007b00, thd=0x7f9564013540, args=0x7f95640066e0) at /root/dq/mysql-8.0.39/sql/sp_head.cc:101
#7  0x000000000359cc87 in Event_job_data::execute (this=0x7f95902e6970, thd=0x7f9564013540, drop=false) at /root/dq/mysql-8.0.39/sql/event_scheduler.cc:101
#8  0x0000000003ad8f70 in Event_worker_thread::run (this=0x7f95902e6b27, thd=0x7f9564013540, event=0x7f95640066e0) at /root/dq/mysql-8.0.39/sql/event_scheduler.cc:101
#9  0x0000000003ad8aaa in event_worker_thread (arg=0x7f95640066e0) at /root/dq/mysql-8.0.39/sql/event_scheduler.cc:101
#10 0x00000000051b8909 in pfs_spawn_thread (arg=0x7f9564001850) at /root/dq/mysql-8.0.39/storage/perfschema/pfs.cc:101
#11 0x00007f95a38ea17a in start_thread () from /lib64/libpthread.so.0
#12 0x00007f95a1c8edc3 in clone () from /lib64/libc.so.6
```

# Server 层线程 - 示例:自动踢掉长事务

```
delimiter ;;
create event kill_long_idle on schedule every 3 second do
begin
    select trx_mysql_thread_id into @long_idle_tid from
information_schema.innodb_trx where timediff(now(),trx_started) >100 limit 1;
    kill connection @long_idle_tid ;
end;;
delimiter ;
```

事务调度器的实践使用建议

# Server 层线程 - 信号处理线程

kill -9 <pid>

kill -15 <pid>

kill -6 <pid>    **innodb\_buffer\_pool\_in\_core\_file (>= 8.0.14)**

# 课堂练习: kill 信号发给线程会怎么样?

## pidstat -t -p <pid>

```
root      2972681  0.7  0.5 1729880 175444 pts/7  S1   17:45   0:00 /mnt/nfs/dq/mysql-5.7.44/sql/mys
44/mysql-test/var/my.cnf --user=root --log-output=file --loose-debug-sync-timeout=600 --loose-skip-
ce_gtid_consistency=ON --innodb_buffer_pool_chunk_size=1048576 --innodb_page_cleaners=4 --core-file
root      2973434  0.0  0.0  12136   1100 pts/6   S+   17:47   0:00 grep --color=auto mysqld
[root@yuqing-4 dq]# pidstat -t -p 2972681
Linux 4.18.0-348.7.1.el8_5.x86_64 (yuqing-4)      09/01/2024      _x86_64_      (4 CPU)

05:47:21 PM  UID      TGID      TID      %usr %system %guest  %wait  %CPU  CPU  Command
05:47:21 PM    0      2972681      -      0.00  0.00  0.00  0.00  0.00   2  mysqld
05:47:21 PM    0      -      2972681      0.00  0.00  0.00  0.00  0.00   2  __mysqld
05:47:21 PM    0      -      2972682      0.00  0.00  0.00  0.00  0.00   1  __mysqld
05:47:21 PM    0      -      2972683      0.00  0.00  0.00  0.00  0.00   1  __mysqld
05:47:21 PM    0      -      2972684      0.00  0.00  0.00  0.00  0.00   1  __mysqld
05:47:21 PM    0      -      2972685      0.00  0.00  0.00  0.00  0.00   1  __mysqld
05:47:21 PM    0      -      2972686      0.00  0.00  0.00  0.00  0.00   1  __mysqld
05:47:21 PM    0      -      2972687      0.00  0.00  0.00  0.00  0.00   1  __mysqld
05:47:21 PM    0      -      2972688      0.00  0.00  0.00  0.00  0.00   2  __mysqld
05:47:21 PM    0      -      2972689      0.00  0.00  0.00  0.00  0.00   0  __mysqld
05:47:21 PM    0      -      2972692      0.00  0.00  0.00  0.00  0.00   1  __mysqld
05:47:21 PM    0      -      2972693      0.00  0.00  0.00  0.00  0.00   3  __mysqld
05:47:21 PM    0      -      2972694      0.00  0.00  0.00  0.00  0.00   1  __mysqld
```

## kill -15 2972694

```
2024-09-01T09:47:37.588971Z 0 [Note] InnoDB: Starting shutdown...
2024-09-01T09:47:37.689846Z 0 [Note] InnoDB: Dumping buffer pool(s) to /mnt/nfs/dq/mysql-5.7.44/mysql-test/var/mysqld.1/data/ib_buffer_pool
2024-09-01T09:47:37.699365Z 0 [Note] InnoDB: Buffer pool(s) dump completed at 240901 12:47:37
2024-09-01T09:47:38.303585Z 0 [Note] InnoDB: Shutdown completed; log sequence number 1350972
2024-09-01T09:47:38.307680Z 0 [Note] InnoDB: Removed temporary tablespace data file: "ibtmp1"
2024-09-01T09:47:38.308981Z 0 [Note] Debug sync points hit: 11522
2024-09-01T09:47:38.308992Z 0 [Note] Debug sync points executed: 0
2024-09-01T09:47:38.308995Z 0 [Note] Debug sync points max active per thread: 0
2024-09-01T09:47:38.312592Z 0 [Note] /mnt/nfs/dq/mysql-5.7.44/sql/mysqld: Shutdown complete
```



# 课堂练习:内存消耗

binlog\_cache\_size 和 binlog\_stmt\_cache\_size 默认值 32k，库里有空表 t

session1

connect  
T0:  
truncate table t;  
T1:  
set global binlog\_cache\_size=3G  
set global binlog\_stmt\_cache\_size=3G  
T2:  
truncate table t;  
T3:  
disconnect  
T4:  
connect  
T5:  
truncate table t;  
T6:

//问题：假设只考察这一个线程的消耗，在上述7个时间点，binlog 的cache占用内存分别是多少

A. 0K B. 32K C. 64K D. 3G E. 6G

# 用户线程中的吃内存大户 --binlog cache

binlog\_cache\_size

binlog\_stmt\_cache\_size

内存申请时机      会话中第一次要写 binlog 时,两个一起申请, 申请定义值

内存回收时机      回收连接到thread\_cache或销毁时, 一起回收

# 课堂练习:内存消耗

binlog\_cache\_size 和 binlog\_stmt\_cache\_size 默认值 32k， 库里有个空表t

```
session1
connect
T0:      OK
truncate table t;
T1:      64K
set global binlog_cache_size=3G
set global binlog_stmt_cache_size=3G
T2:      64k
truncate table t;
T3:      64k
disconnect
T4:      OK
connect
T5:      OK
truncate table t;
T6:      6G
```

更多资源v: 2938843375

# 用户线程中的吃内存大户 -- sort buffer

## 课堂练习：看代码推策略

```
void SortingIterator::CleanupAfterQuery() {  
    m_fs_info.free_sort_buffer();  
    my_free(m_fs_info.merge_chunks.array());  
}
```

### session1

connect

set sort\_buffer\_size=32k;

T0:

select ... order //一个能够吃满sort\_buffer 的SQL

T1:

set sort\_buffer\_size=3G

select ... order //一个能够吃满sort\_buffer 的SQL

T2

disconnect

//**问题**：假设只考察这一个线程的消耗，在上述3个时间点，sort\_buffer占用内存分别是多少

A. 0K B. 32K C. 64K D. 3G E. 6G

更多资源v: 2938843375



# 应用端/中间层连接池跟MySQL线程的关系

join buffer、tmp\_table 策略与 sort buffer 类似

# 用户线程中的吃内存大户

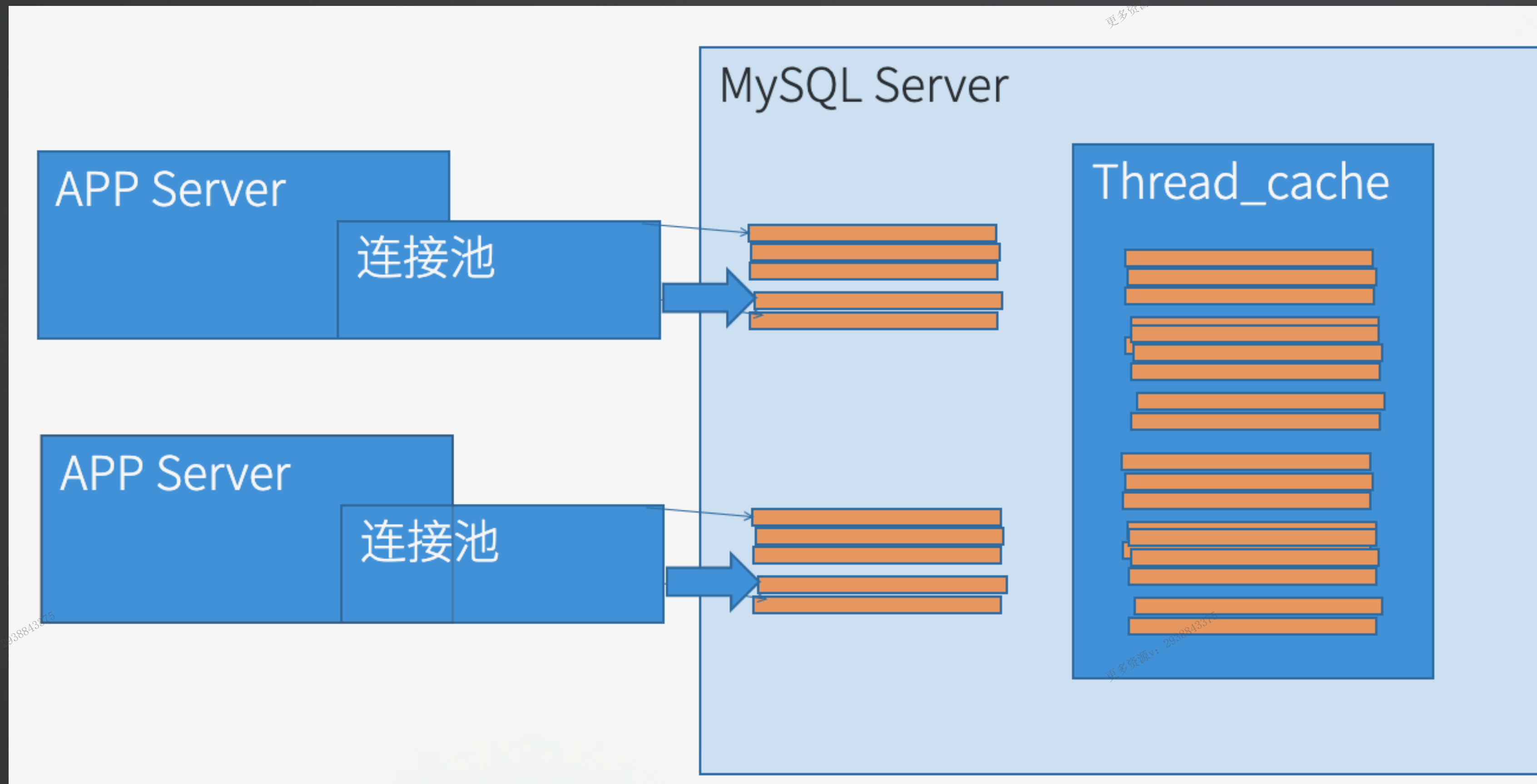
如何处理 binlog\_cache\_cache和客户端长连接的矛盾

研发要的是什么

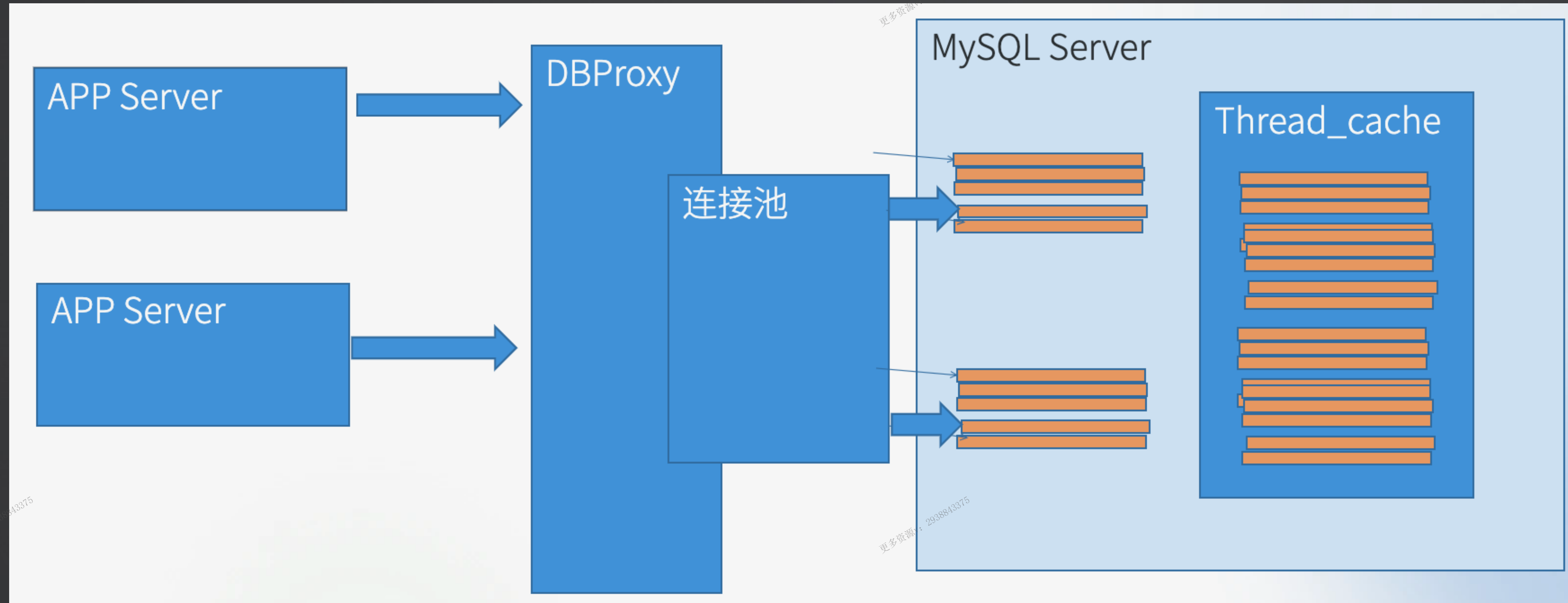
resetconnection; X

disconnect + connect

# 应用端/中间层连接池跟MySQL线程的关系

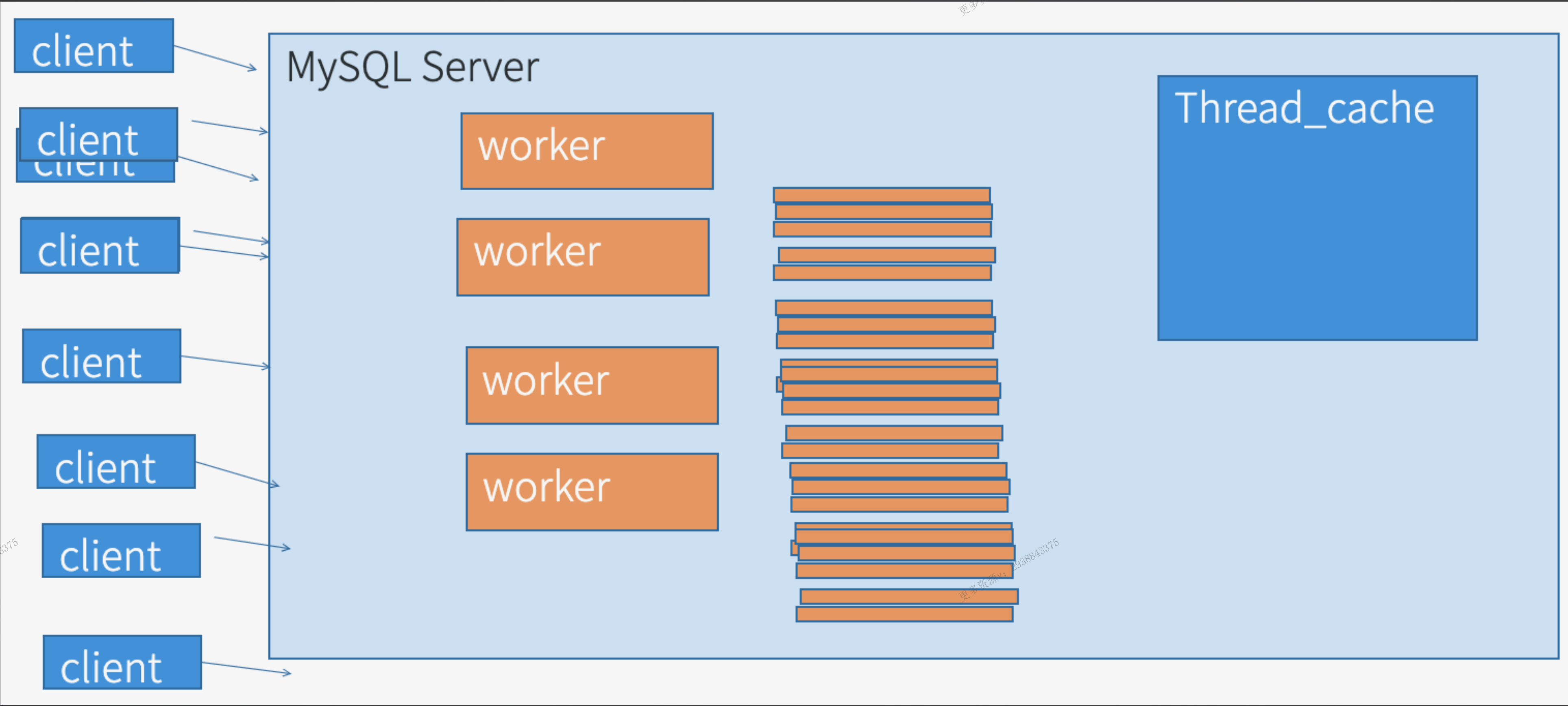


# 连接池工作原理

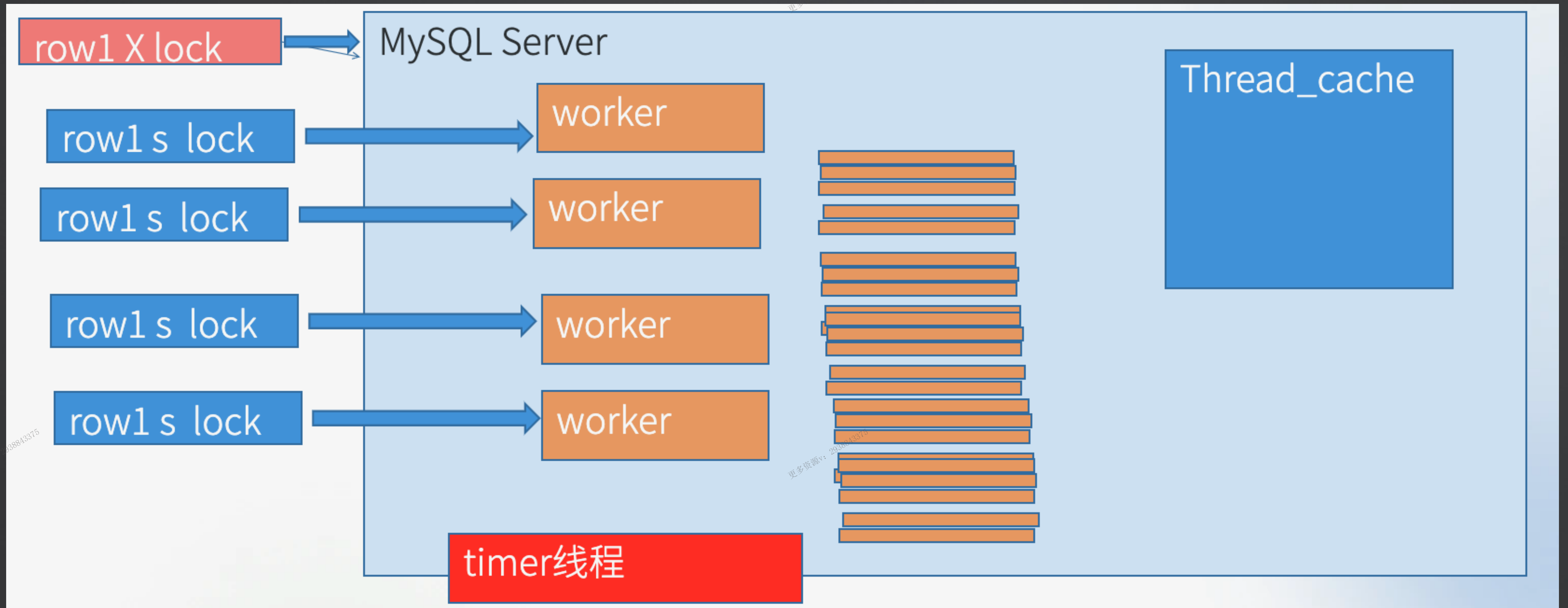




# 线程池工作原理



# 线程池工作原理



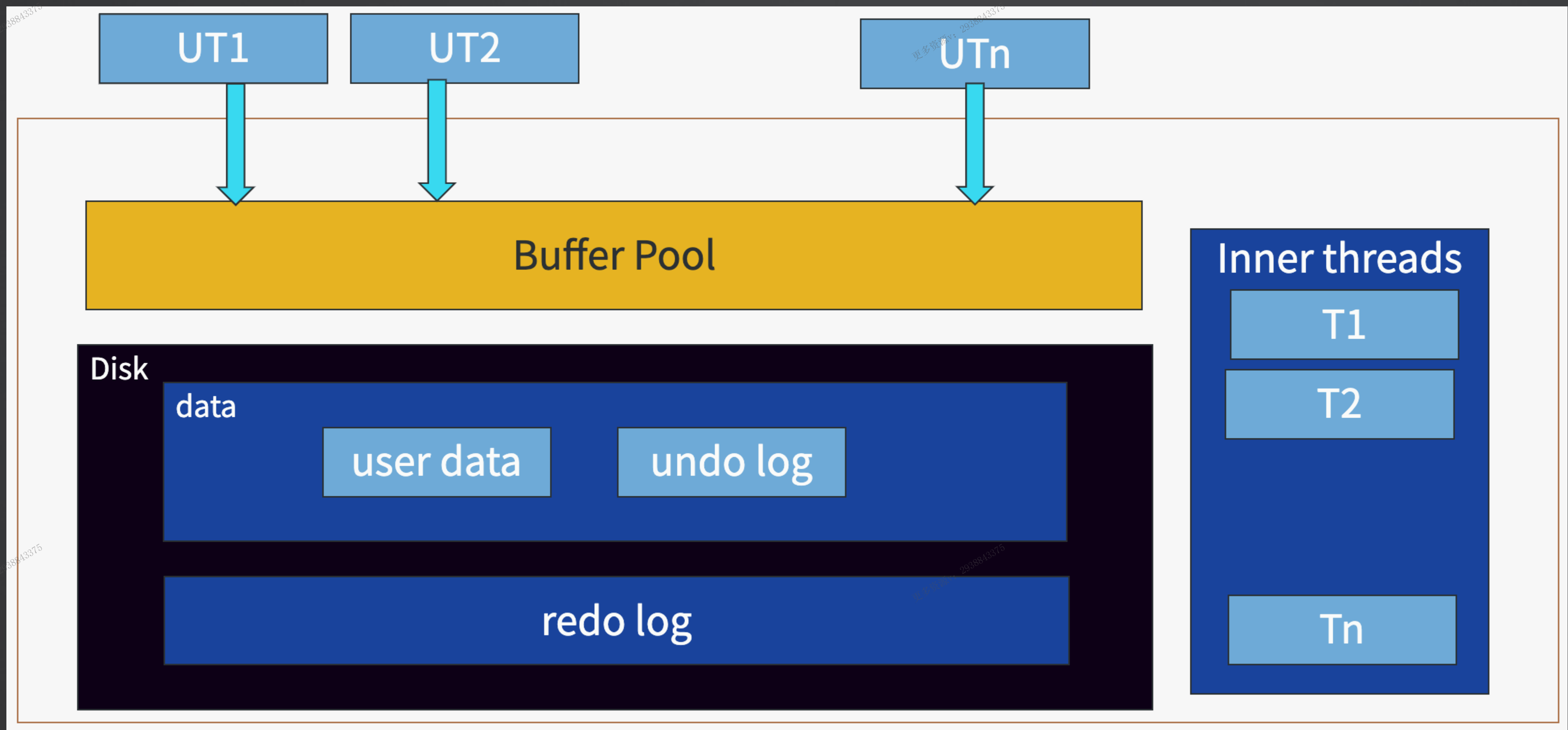
# 目录

1 MySQL 分层模型

2 Server 层线程

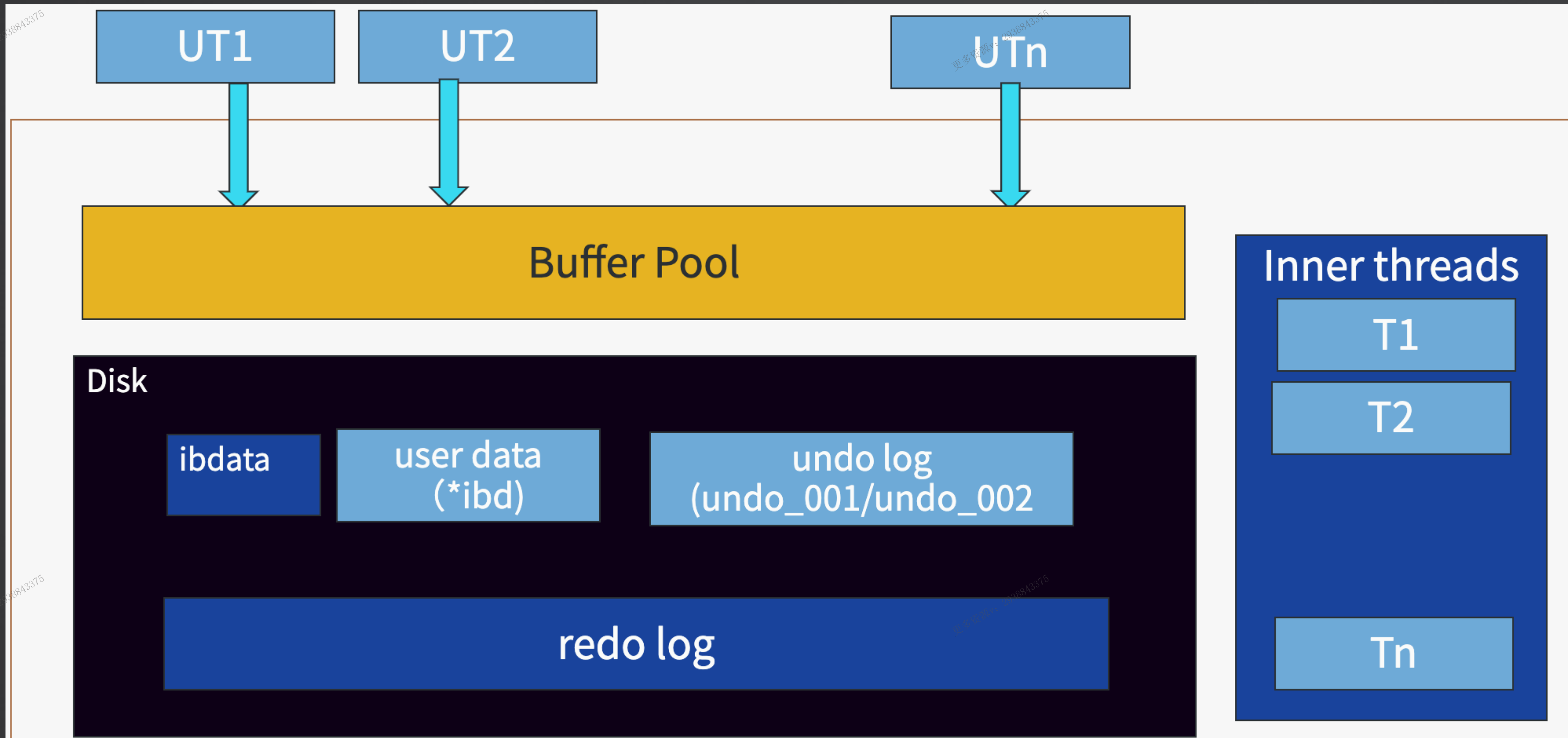
3 InnoDB 线程

# 基础知识回顾





# 基础知识回顾



# 基础知识回顾

```
mysql> select thread_id,name from performance_schema.threads where name like '%innodb%' order by name;
```

thread_id	name
33	thread/innodb/buf_dump_thread
26	thread/innodb/buf_resize_thread
34	thread/innodb/clone_gtid_thread
28	thread/innodb/dict_stats_thread
29	thread/innodb/fts_optimize_thread
3	thread/innodb/io_ibuf_thread
4	thread/innodb/io_read_thread
7	thread/innodb/io_read_thread
6	thread/innodb/io_read_thread
5	thread/innodb/io_read_thread
10	thread/innodb/io_write_thread
11	thread/innodb/io_write_thread
9	thread/innodb/io_write_thread
8	thread/innodb/io_write_thread
13	thread/innodb/log_checkpoint_thread
18	thread/innodb/log_files_governor_thread
14	thread/innodb/log_flush_notifier_thread
15	thread/innodb/log_flusher_thread
16	thread/innodb/log_write_notifier_thread
17	thread/innodb/log_writer_thread
12	thread/innodb/page_flush_coordinator_thread
24	thread/innodb/srv_error_monitor_thread
23	thread/innodb/srv_lock_timeout_thread
27	thread/innodb/srv_master_thread
25	thread/innodb/srv_monitor_thread
35	thread/innodb/srv_purge_thread
36	thread/innodb/srv_worker_thread
37	thread/innodb/srv_worker_thread
38	thread/innodb/srv_worker_thread

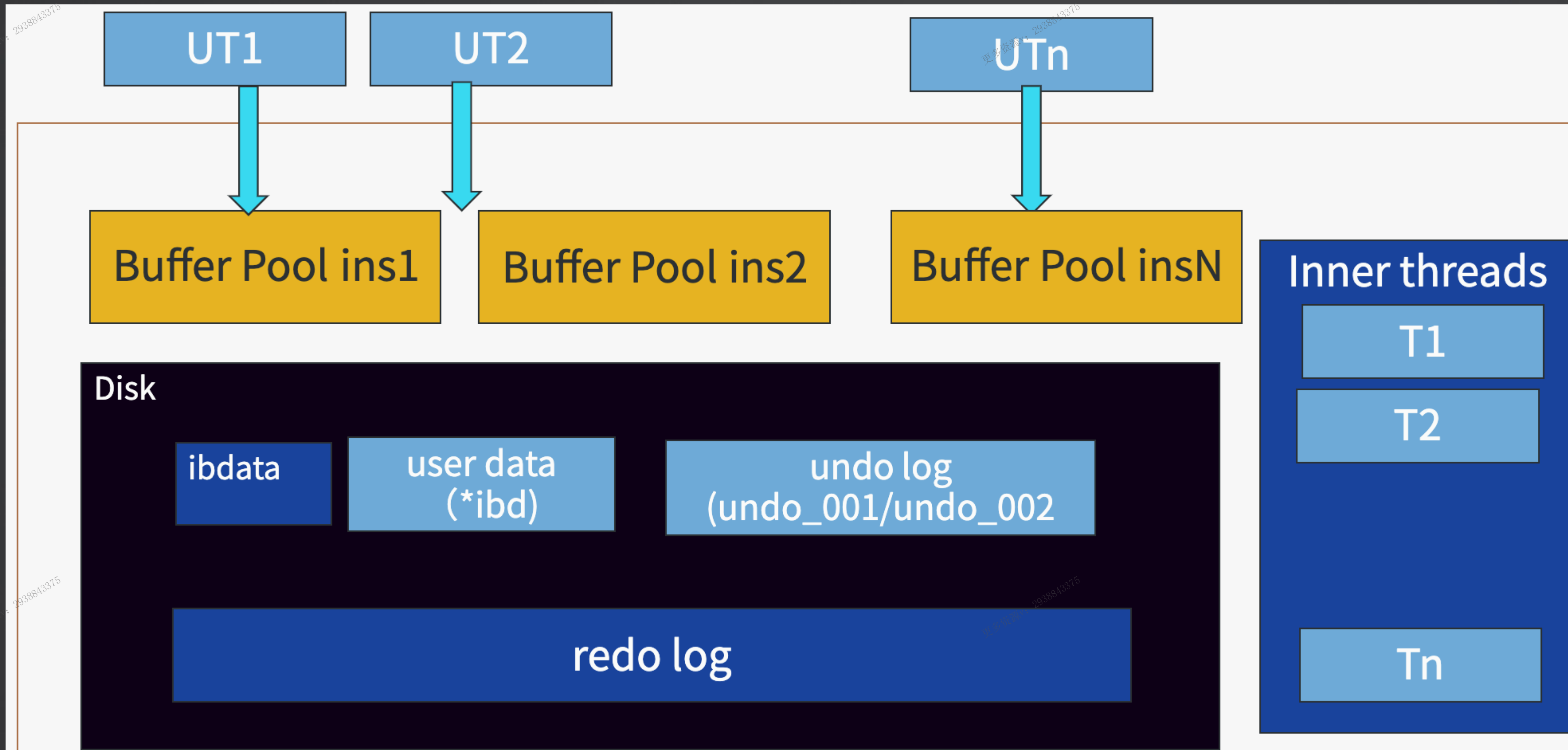
29 rows in set (0.00 sec)

# buf\_dump\_thread

```
mysql> set global innodb_buffer_pool_dump_now=on;
```

```
Query OK, 0 rows affected (0.00 sec)
```

# buf\_dump\_thread





## buf\_dump\_thread:innodb\_buffer\_pool\_dump\_pct

4294967279, 109  
4294967279, 227  
4294967279, 226  
4294967279, 225  
4294967279, 223  
4294967279, 106  
4294967279, 221  
4294967279, 220  
4294967279, 219  
4294967279, 218  
4294967279, 276  
4294967279, 215

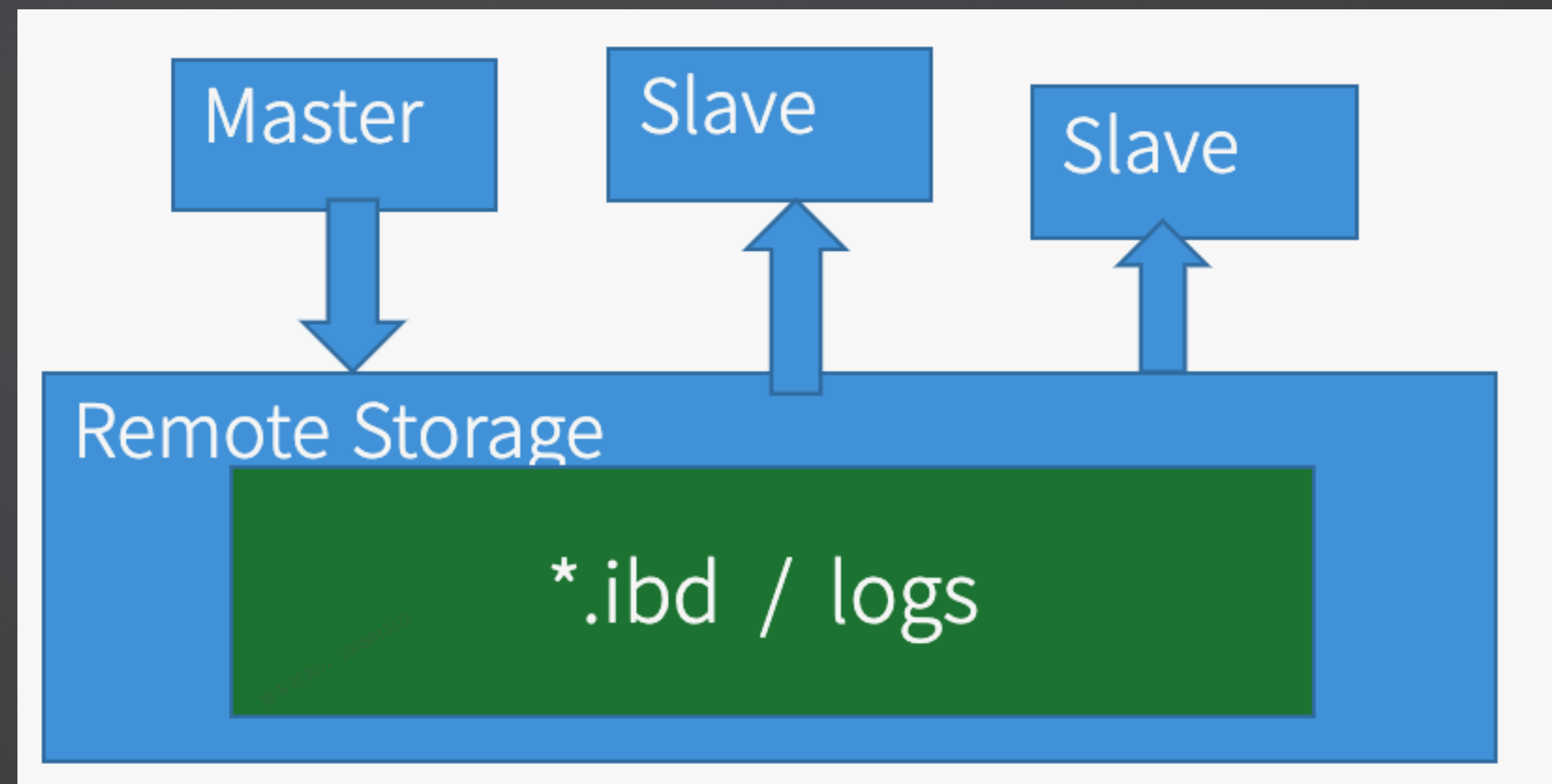
```
/* walk through each buffer pool */
```

```
for (i = 0; i < srv_buf_pool_instances && !SHOULD_QUIT(); i++) {  
    buf_pool_t *buf_pool;
```

```
    for (auto bpage : buf_pool->LRU) {  
        if (n_pages <= j) break;  
        ut_a(buf_page_in_file(bpage));  
  
        dump[j++] = BUF_DUMP_CREATE(bpage->id.space(), bpage->id.page_no());  
    }  
  
    ut_a(j == n_pages);
```

buf\_dump\_thread:innodb\_buffer\_pool\_dump\_pct

课堂练习：半同步复制的主备架构，主库的 buf\_dump 拿到读库去 load，有作用吗？



# Q&A

THANKS