

TDSQL智能运维平台 扁鹊架构与实践

雷海林

腾讯云数据库专家工程师



大纲

- 1.扁鹊的基本介绍
- 2.系统架构
- 3.智能诊断原理与实践
- 4.总结

1.1 扁鹊-TDSQL提供的智能运维平台

- TDSQL (Tencent Distributed MySQL)
 - 腾讯针对金融场景推出的高一致性，分布式数据库集群解决方案
 - 数据库引擎覆盖Percona/Mariadb分支，支持InnoDB和RocksDB引擎
 - 腾讯内部：腾讯 90% 的金融、计费、交易类业务核心
 - 公有云、私有云：覆盖政府、银行、保险、制造业、物流、电商等用户的核心系统

1.2 运营过程面临的挑战

- 数据库节点多，出现故障人工分析效率低下
- DB存在风险没法提前预警和处理
- 用户咨询量非常大，尤其是性能相关的问题
- 专有云的场景下，很多客户不具备很强的故障分析能力

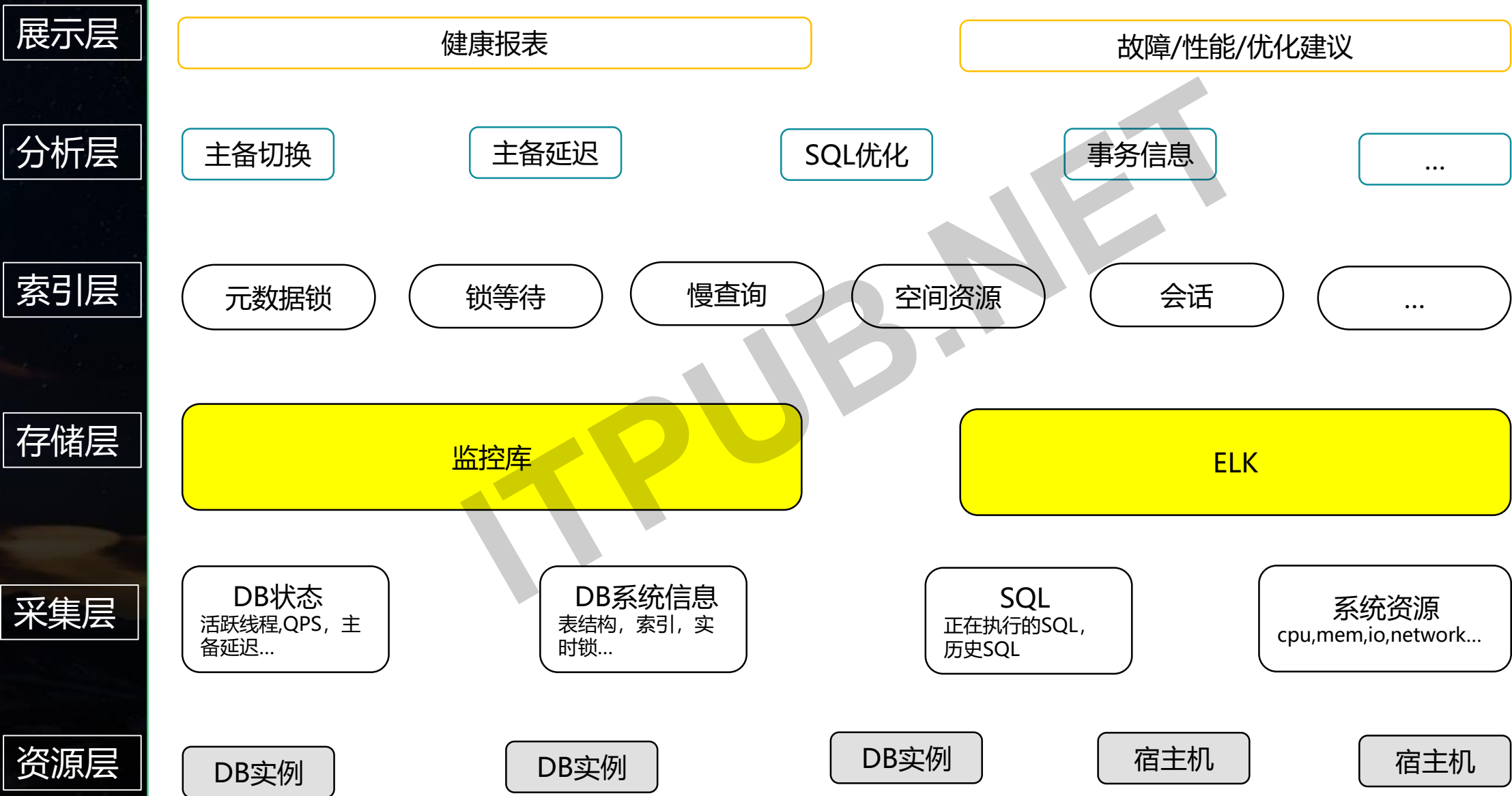
1.3 扁鹊-智能运营平台

- 故障预警
- 故障自动诊断
- 历史事件剖析
- 优化建议
- 操作通过管理台自助化，经验积累
- 降低DBA工作复杂度，提升幸福感

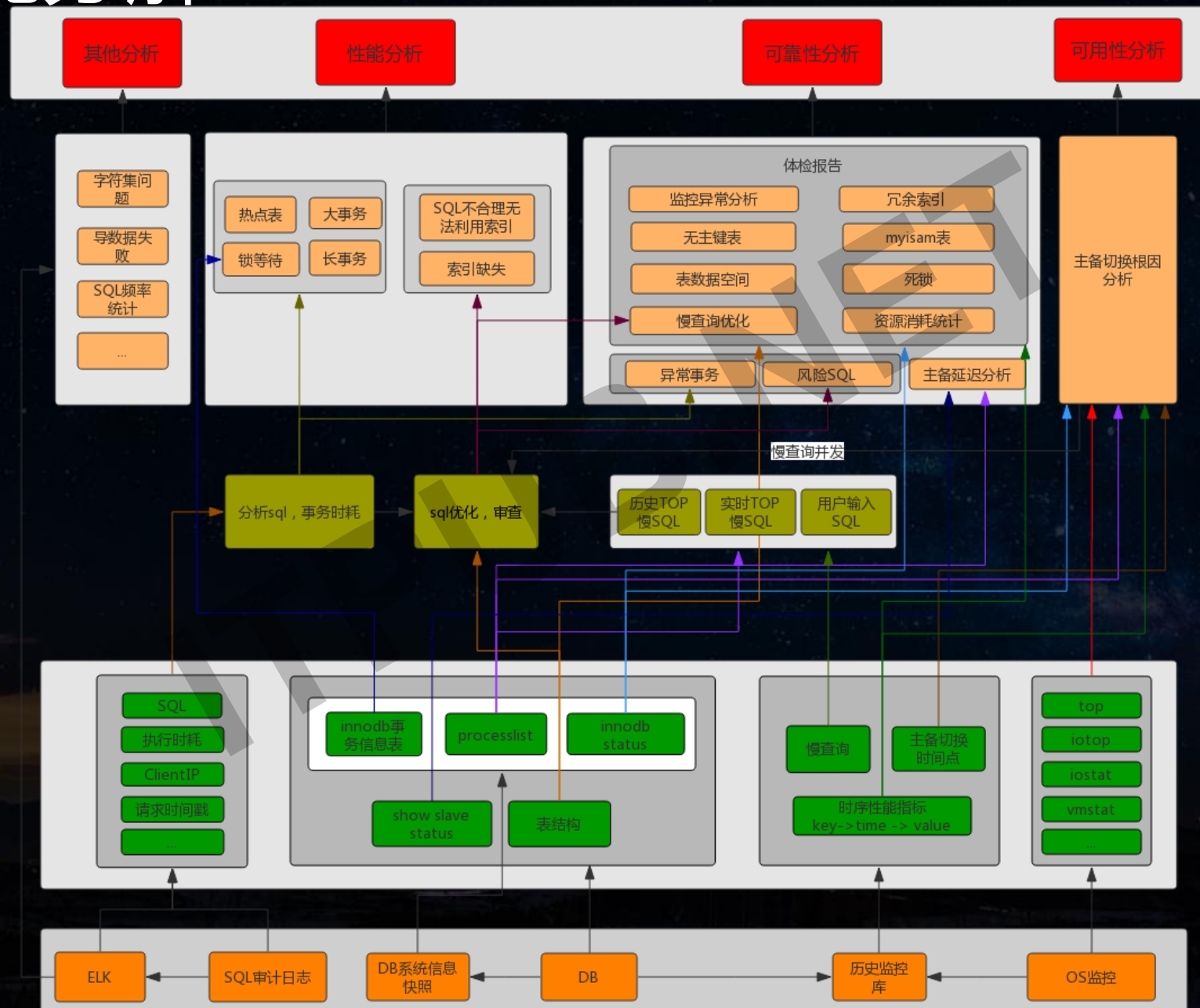
大纲

- 1.扁鹊的基本介绍
- 2.系统架构
- 3.智能诊断原理与实践
- 4.总结

2.1 系统分层结构



2.2 功能分解



大纲

- 1.扁鹊的基本介绍
- 2.系统架构
- 3.智能诊断原理与实践
- 4.总结

3.1.1 主备切换的诊断



3.1.2 触发主备切换常见因素

- DB意外重启
- 内核BUG引起系统hang住等
- 磁盘故障，文件系统故障等
- 资源竞争引起DB异常
 - IO耗尽
 - innodb并发线程耗尽
 - binlog写入竞争

3.1.3 解决方案

- 采集DB的启动时间，监控DB是否重启
- 每分钟采集show processlist信息
- 每秒采集宿主机各种状态信息
- 切换前，保存系统状态信息

```
drwxr-xr-x 2 root root 4.0K Apr 28 00:00 dstatlog
drwxr-xr-x 2 root root 4.0K Apr 28 00:00 iostatlog
drwxr-xr-x 2 root root 4.0K Apr 28 00:00 iotoplog
drwxr-xr-x 2 root root 4.0K Apr 28 00:00 meminfo
drwxr-xr-x 2 root root 4.0K Apr 28 00:01 toplog
drwxr-xr-x 2 root root 4.0K Apr 28 00:01 vmstat
```

```
5100_innodb_status_20190428.log
5100_threadpool_20190428.log
5100_metalock_20190428.log
5100_innodb_trx_20190428.log
5100_innodb_lock_waits_20190428.log
5100_innodb_locks_20190428.log
5100_innodb_processlist_20190428.log
5100_lock_analysis_20190428.log
5100_sysdump_20190428.log
```

3.1.4 主备切换: 高并发

- innodb状态日志看到活跃线程有64个（配置值就是64）
- Session状态看到大量查询处于执行或者等待状态

ROW OPERATIONS

64 queries inside InnoDB, 53 queries in queue
64 read views open inside InnoDB
3 RW transactions active inside InnoDB

18956044		b	Query	1696	Sending to client	select * from [redacted] where
((1)) 1695371						
18955360			Query	1692	Sending to client	select * from [redacted] where
((1)) 1692123						
18955416			Query	1686	Sending to client	select * from [redacted] where
((1)) 1685534						
18956147			Query	1675	Sending to client	select * from [redacted] where
((1)) 1675147						
18956174			Query	1671	Sending to client	select * from [redacted] where
((1)) 1670328						
18956178			Query	1661	Sending to client	select * from [redacted] where
((1)) 1660717						

3.1.5 主备切换: 大binlog事务

- 心跳探测线程被阻塞
- 大量事务已经完成了prepared, 等待写入binlog

```
3      localhost      NULL      Query      19      Sending data      show global status where Variable_name in('Com_insert','Com_select',
'Com_update','Com_delete','Com_replace','Com_replace_select','slow_queries') 18397 0 0
12      localhost      sysdb     Query      19      query end         replace into SysDB.StatusTable set ts = from_unixtime(1544778177),ip
='',port=4022      18609 0 0
13      localhost      NULL      Query      19      query end         replace into SysDB.StatusTableForHb set tid='7f3ac4ff9700',ts = from
_unixtime(1544778177),ip='',port=4022 18609 0 0
923989  localhost      sysdb     Query      16      update replace into SysDB.StatusTable set ts = from_unixtime(1544778180),ip=
',port=4022      15599 0 0
21      localhost      NULL      Query      16      query end         replace into SysDB.StatusTableForHb set tid='7f3ab7fff700',ts = from
_unixtime(1544778180),ip='',port=4022 15600 0 0
```

```
---TRANSACTION 215881534, ACTIVE (PREPARED) 17 sec
6 lock struct(s), heap size 1136, 3 row lock(s), undo log entries 1
MySQL thread id 843095, OS thread handle 140179633829632, query id 19009306997 starting
commit
---TRANSACTION 215881533, ACTIVE (PREPARED) 18 sec
mysql tables in use 1, locked 1
2 lock struct(s), heap size 1136, 1 row lock(s), undo log entries 1
MySQL thread id 13, OS thread handle 140184281028352, query id 19009302629 localhost query end
replace into SysDB.StatusTableForHb set tid='7f3ac4ff9700',ts = from_unixtime(1544778177),ip='',port=4022
```


3.1.6 扁鹊自动化诊断样例

- 主机宕机引起切换

The screenshot displays the Biaoque database management system's log management interface. On the left, a sidebar contains navigation options: 实例详情, 数据库管理, DB监控, Proxy监控, 异常会话, 实例监控, 告警查询, 日志管理 (highlighted), 备份&恢复, 数据迁移, and 性能分析. The main panel has tabs for 慢查询, 主备切换, DBA日志, 控制台操作日志, and 网关日志. The '主备切换' tab is active, showing a time range from 2019-03-01 to 2019-04-28. A task ID 'GJID_CONSIST_FAILOVER_000' is visible. A modal window titled '主备切换详情' (Failover Details) is open, showing the cause of the failover: '原因: 原主机切换前发生过不存活' (Cause: The original host had a non-survival event before the switch). The details section describes the process: 1. Check if the original host is alive, noting that the 'process file handle limit' metric dropped to 0 before the switch at 2019-03-27 05:20:42. 2. Check resource status before the switch, listing CPU and IO usage metrics. 3. Analyze the DB state dump logs, showing empty processlist and innob_status logs.

实例详情 慢查询 主备切换 DBA日志 控制台操作日志 网关日志

时间区间: 2019-03-01 to 2019-04-28 查询

任务ID
GJID_CONSIST_FAILOVER_000

主备切换详情

原因:
原主机切换前发生过不存活

详细:
[2019-03-27 05:20:42]发生切换, 由主机降级为备机

1. 开始检查原主机是否发生不存活
监控指标 '进程文件句柄上限' 切换前变为0, 说明原主机切换前发生过不存活
2. 开始检查原主机切换前的资源状态
[2019-03-27 05:19:00 , 2019-03-27 05:21:00]之间
cpu_usage_max最大值为169%
cpu_usage最大值为137%
io_usage_max最大值为48% (未找到io_usage_max最大值所在盘符)
io_usage最大值为30% (所在盘符: nvme3n1)
3. 开始分析切换前DB状态的dumplog
processlist dump log 为空
innob_status dump log 为空

3.1.7 扁鹊自动化诊断样例

- 大量慢SQL引起切换

The screenshot displays the '主备切换详情' (Primary-Slave Switch Details) page in the IT-OPS.NET system. The left sidebar contains navigation options: 实例详情, 数据库管理, DB监控, Proxy监控, 异常会话, 实例监控, 告警查询, 日志管理 (highlighted), 备份&恢复, 数据迁移, and 性能分析. The main content area is divided into a left panel for filters (时间区间: 2019-03-01 to 2019-03-01, 任务ID: GJID_CONSIST_FAILOVER_000) and a right panel for details. The details panel shows the '原因' (Cause) as '切换前运行慢SQL引起' (Slow SQL execution before switch) and the '详细' (Details) as a list of steps: 1. 开始监控指标, 2. 开始分析, 3. 开始切换. The '聚合信息' (Aggregated Information) section provides a summary of the slow SQL queries: MAX_TIME: 79, MIN_TIME: 19, COUNT: 109, and DIGEST_HASH: aaaf68664836a2934cbb2ab46a2595ba. The 'EXAMPLE_QUERY' section shows the SQL query: SELECT orderId, metroUserId, partnerId, channelType, orderType, totalFee, payFee, refundFee, merchantsId, orderStatus, adjustType, payId, inRecordId, lineIdStart, stationIdStart, inDeviceId, inStationId, inStationName, inRecordTime, outRecordId, lineIdEnd, outDeviceId, stationIdEnd, outStationId, outStationName, outRecordTime, affRecordId, affLineId, affStationId, affStationOrgId, affStationName, affRecordTime, affDeviceId, mobile, logicId, tradeDate, createTime, updateTime, payLineId, payStationId FROM ... WHERE ... ORDER BY ... LIMIT ?, ?. The 'INFO' section shows the query execution details: ID: 13710153, USER: ar frm, HOST: ..., DB: ..., COMMAND: Query, TIME: 79, STATE: Sending data, and INFO: SELECT orderId, metroUserId, partnerId, channelType, orderType, totalFee, payFee, refundFee, merchantsId, orderStatus, adjustType, payId, inRecordId, lineIdStart, stationIdStart, inDeviceId, inStationId, inStationName, inRecordTime, outRecordId, lineIdEnd, outDeviceId, stationIdEnd, outStationId, outStationName, outRecordTime, affRecordId, affLineId, affStationId, affStationOrgId, affStationName, affRecordTime, affDeviceId, mobile, logicId, tradeDate, createTime, updateTime, payLineId, payStationId FROM ... WHERE ... ORDER BY ... LIMIT ?, ?.

主备切换详情

-运行时间超过15s的processlist聚合信息:

MAX_TIME: 79
MIN_TIME: 19
COUNT: 109
DIGEST_HASH: aaaf68664836a2934cbb2ab46a2595ba

QUERY_DIGEST: SELECT orderId, metroUserId, partnerId, channelType, orderType
\t, totalFee, payFee, refundFee, merchantsId, orderStatus
\t, adjustType, payId, inRecordId, lineIdStart, stationIdStart
\t, inDeviceId, inStationId, inStationName, inRecordTime, outRecordId
\t, lineIdEnd, outDeviceId, stationIdEnd, outStationId, outStationName
\t, outRecordTime, affRecordId, affLineId, affStationId, affStationOrgId
\t, affStationName, affRecordTime, affDeviceId, mobile, logicId
\t, tradeDate, createTime, updateTime, payLineId, payStationId
FROM
WHERE
ORDER
LIMIT ?, ?

EXAMPLE_QUERY:

ID: 13710153
USER: ar frm
HOST:
DB:
COMMAND: Query
TIME: 79
STATE: Sending data
INFO: SELECT orderId, metroUserId, partnerId, channelType, orderType, totalFee, payFee, refundFee, merchantsId, orderStatus, adjustType, payId, inRecordId, lineIdStart, stationIdStart, inDeviceId, inStationId, inStationName, inRecordTime, outRecordId, lineIdEnd, outDeviceId, stationIdEnd, outStationId, outStationName, outRecordTime, affRecordId, affLineId, affStationId, affStationOrgId, affStationName, affRecordTime, affDeviceId, mobile, logicId, tradeDate, createTime, updateTime, payLineId, payStationId FROM ... WHERE ... ORDER BY ... LIMIT ?, ?

3.1.8 扁鹊自动化诊断样例

- 由大事务（binlog写入竞争）引发的主备切换

[实例详情](#)[慢查询](#)[主备切换](#)[DBA日志](#)[控制台操作日志](#)[网关日志](#)

主备切换详情

0.00 inserts/s, 0.00 updates/s, 0.00 deletes/s, 3.80 reads/s
切换前innodb中的活跃线程数为: 0, 等待线程数为: 0
切换前运行时间超过15s的查询有3条,未达到innodb最大并发线程的限制

-分析是否有SQL阻塞了TDSQL探活语句binlog的写入
切换前3条TDSQL探活语句被阻塞在写入binlog状态
切换前1条commit语句被阻塞在写入binlog状态
可能是有TDSQL探活语句在写入binlog阶段被阻塞至超时引起心跳检测失败引发主备切换
引起binlog阻塞的SQL可能是

ID: 173291050
USER: [REDACTED]
HOST: [REDACTED]
DB: [REDACTED]
COMMAND: [REDACTED]
TIME: 392
STATE: query end
INFO: UPDATE [REDACTED] c SET c.SUB [REDACTED] 3.08*c .PAYMI

-运行时间超过15s的processlist聚合信息:

MAX_TIME: 18
MIN_TIME: 18
COUNT: 1
DIGEST_HASH: 09a9c81f7e6ab41133cfe0c7dc5a1616
QUERY_DIGEST: UPDATE [REDACTED] SET LAS [REDACTED] E = ? WHERE [REDACTED] ? AND [REDACTED] ?
EXAMPLE_QUERY:
ID: 172344113

3.2 DB性能问题诊断

- 通常表现为用户请求耗时较长
 - 网络因素，如延迟，丢包等
 - SQL自身执行较慢
 - 系统资源被其他慢SQL占用
 - 锁等待

3.2.1 SQL执行慢的诊断与优化

- 索引不合理，SQL写法效率低等原因导致SQL执行时耗较高
- 对某个慢查询使用扁鹊的SQL优化分析，得优化建议
- 如果是SELECT也可以建议采用读写分离

创建时间	诊断数据库	诊断SQL	建议
2019-03-07 15:36:05 [查看]	test_xa	select * from sbtest1 where value = 10	alter table test_xa.sbtest1 add index idx_value(value)

3.2.2 资源饱和

- 当前CPU/IO等系统资源利用率彪升
- 利用扁鹊的会话诊断功能
- 将processlist汇总，按照SQL摘要进行聚合，统计出当前会话执行的TOP SQL
- 然后对TOP SQL给出优化建议

实例详情

Set管理

数据库管理

DB监控

Proxy监控

异常会话

实例监控

告警查询

日志管理

备份&恢复

性能分析

检测报告SQL优化实时诊断会话检查表空间分布故障诊断SQL审计

SetId: set_1556160811_3 Host: 重新加载

processlist汇总自定义字段导出

set_host	sql_digest	db	count	tc_min	tc_max	user@host	example_sql	advice
	SELECT * FROM `activity` WHERE VALUE < ? ORDER BY VALUE	test	322	0 s	3 s	test@	select * from activity where value < 1000 order by value	alter table test.activity add index idx_value(value)
	SELECT * FROM `information_schema`.`processlist` WHERE `INFO` IS NOT NULL ORDER BY TIME DESC	information_schema	1	0 s	0 s		select * from information_schema.processlist where INFO is not NULL order by time desc	

总共 2 条记录

processlist自定义字段导出

ID	USER	HOST	DB	COMMAND	TIME	STATE	INFO
653101			information_schema	Query	0 s	executing	select * from information_schema.processlist where INFO is not NULL order by time desc
652815			test	Query	0 s	Creating sort index	select * from activity where value < 1000 order by value
652814			test	Query	2 s	Creating sort index	select * from activity where value < 1000 order by value
652813			test	Query	1 s	Creating sort index	select * from activity where value < 1000 order by value

3.2.3 锁等待

- 常见现象
 - SQL单独执行很快
 - 负载低的环境中时快时慢
 - 偶尔还会出现Lock wait timeout错误
- 常见原因
 - 事务未提交
 - 事务时耗长

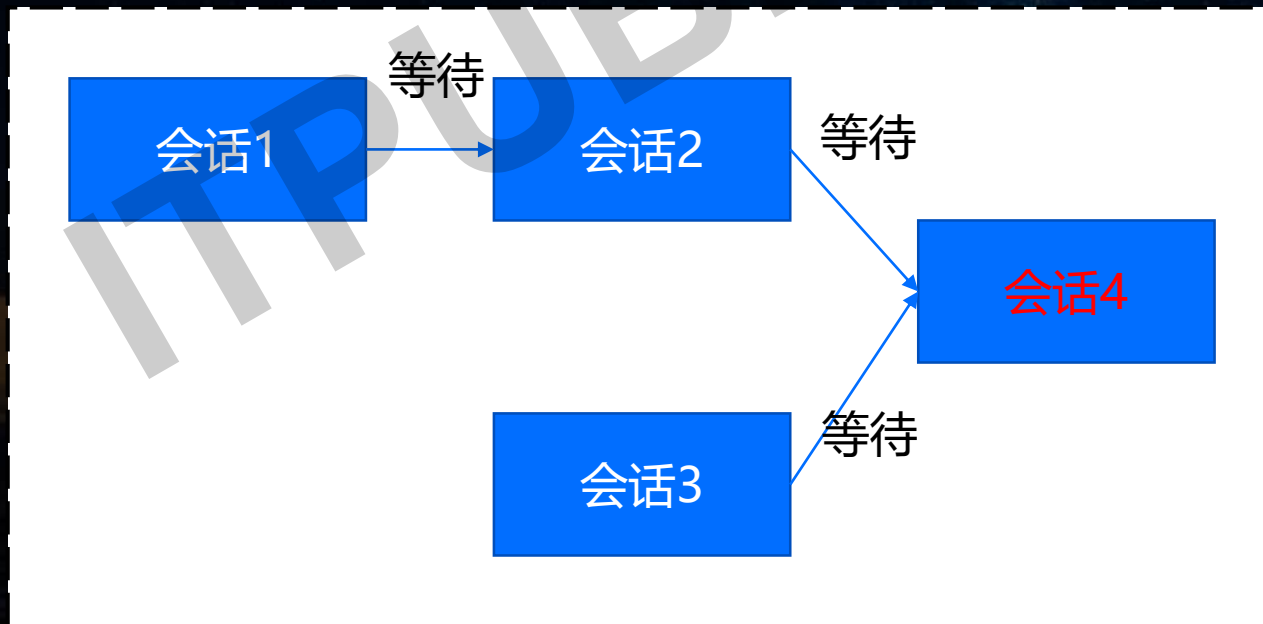
锁: 事务未提交

- 请求响应时间
 - 执行时间
 - 锁等待时间
- 执行时间短
- 等待事务锁的时间长

时间	session1	session2
10:00:00	begin	begin
10:00:01	Update t1 set value=1 where id=1	
10:00:10		Update t1 set value=2 where id=1 (阻塞)
10:10:00	(事务一直没结束, 行锁 一直没释放, 阻塞了 session2)	Lock wait timeout exceeded, try restarting transaction

锁: 事务未提交 (续)

- MySQL的information schema库下有三张表 (innodb_trx, innodb_lock_waits, innodb_locks) 记录了会话之间锁等待的依赖关系
- 分析这三张表的信息分析锁等待关系中并找出当前持有锁的领头的会话



锁: 事务未提交 (续)

- 扁鹊自动化诊断样例

锁等待诊断 (lock_wait) 分数: -10

未提交事务 (uncommitted_trx)

trx_id	trx_state	trx_started	trx_requested_lock_id	trx_wait_started	trx_weight	trx_mysql_thread_id	trx_query	trx_operation_state	trx_tables_in_use	trx_tables_locked	trx_lock_structs	trx_lock_memory_bytes	trx_rows_locked	trx_rows_modified	trx_concurrency_tickets	trx_is
1477448	RUNNING	2018-08-29 11:24:34	NULL	NULL	3	2149	NULL	NULL	0	1	2	1160	1	1	0	REPEA

未提交事务线程 (uncommitted_trx_processlist)

ID	USER	HOST	DB	COMMAND	TIME	STATE	INFO
2149	agent	localhost	test1	Sleep	85		NULL

等待线程 (request_trx_processlist)

ID	USER	HOST	DB	COMMAND	TIME	STATE	INFO
2150	agent	localhost	test1	Query	19	updating	update sbtest1 set k = 10 where id = 1

innodb锁等待 (innodb_lock_waits)

requesting_trx_id	requested_lock_id	blocking_trx_id	blocking_lock_id	requesting_thd_id	blocking_thd_id
1477450	1477450:93860:5:2	1477448	1477448:93860:5:2	2150	2149

innodb锁 (innodb_locks)

lock_id	lock_trx_id	lock_mode	lock_type	lock_table	lock_index	lock_space	lock_page	lock_rec	lock_data
1477450:93860:5:2	1477450	X	RECORD	`test1`.`sbtest1`	PRIMARY	93860	5	2	1
1477448:93860:5:2	1477448	X	RECORD	`test1`.`sbtest1`	PRIMARY	93860	5	2	1

建议

未提交事务会导致后续相关事务陷入锁等待，建议根据业务情况kill会话 2149

锁: 事务时耗长

- 事务持锁时间长
- 其他会话等锁时间长, 执行时间并不长
- 等锁时间超过innodb_lock_wait_timeout会返回锁超时错误

时间	session1	session2
10:00:00	begin	begin
10:00:01	Update t1 set value=1 where id=1	
10:00:10	间隔50s	Update t1 set value=2 where id=1 (阻塞)
10:10:00	commit	1. 执行结束, 返回, 时耗50s 2. Lock wait timeout exceeded, try restarting transaction

锁: 事务时耗长 (续)

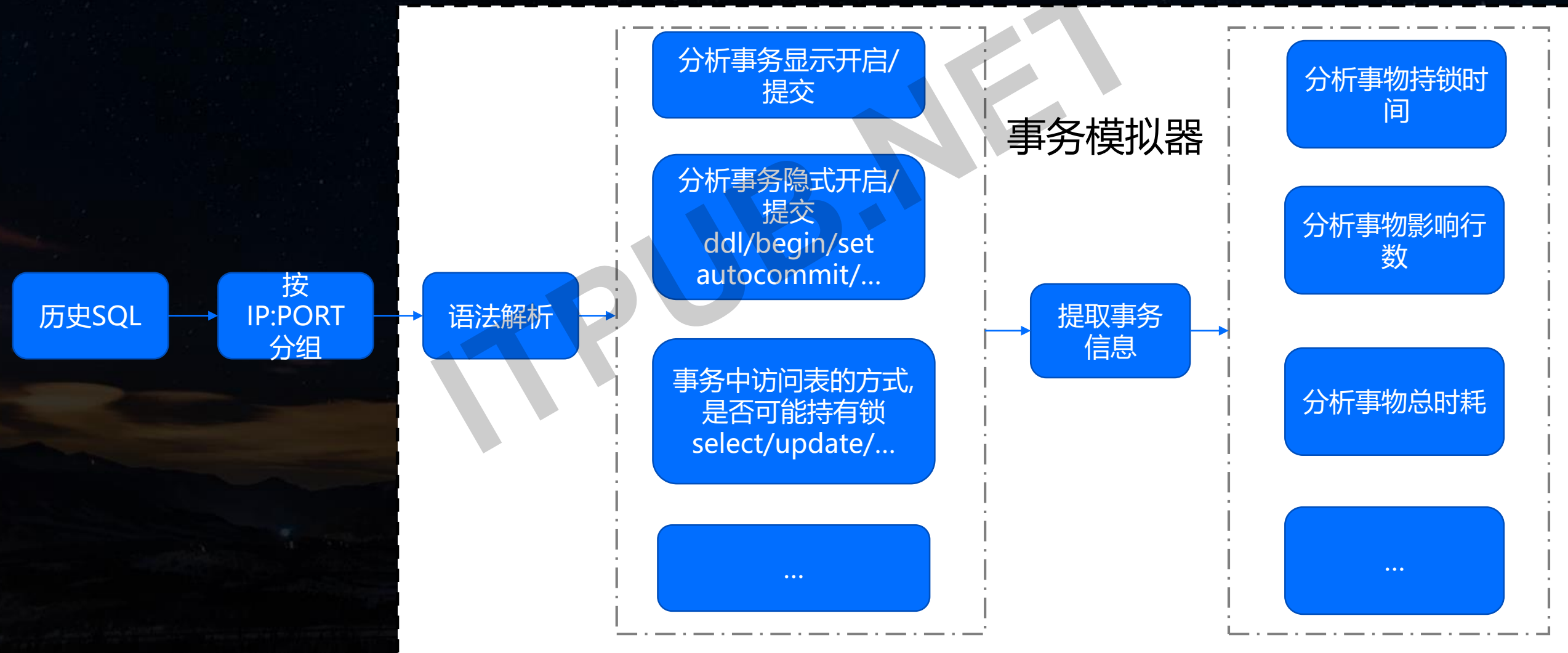
- 审计日志

- 日志记录了SQL, 时耗, 源IP:PORT等信息
- 通过分析审计日志可以提取会话的事务信息

```
[2019-04-30 15:17:05 882071] INFO topic=group_1534163729_3647346&tid=14361&con=0x7f2f843d5000&clientIP=xxxx:xxxx&sql_size=695&sql_type=3&sub_sql_type=5&sql=begin&db=xx&user=xx&backend=xxx:xx&autocommit=1&new_connum=0&conn_tc=0&select_result_sum=0&affect_num=1&hold_conns=2&resultcode=0&timecost=1&join_createtable_table=0&join_filling_table=0&join_do_query=0
[2019-04-30 15:17:05 882071] INFO topic=group_1534163729_3647346&tid=14361&con=0x7f2f843d5000&clientIP=xxxx:xxxx&sql_size=695&sql_type=3&sub_sql_type=5&sql=insert into xxxx values('xxxx')&db=xx&user=xx&backend=xxx:xx&autocommit=1&new_connum=0&conn_tc=0&select_result_sum=0&affect_num=1&hold_conns=2&resultcode=0&timecost=1&join_createtable_table=0&join_filling_table=0&join_do_query=0
[2019-04-30 15:17:05 882071] INFO topic=group_1534163729_3647346&tid=14361&con=0x7f2f843d5000&clientIP=xxxx:xxxx&sql_size=695&sql_type=3&sub_sql_type=5&sql=commit&db=xx&user=xx&backend=xxx:xx&autocommit=1&new_connum=0&conn_tc=0&select_result_sum=0&affect_num=1&hold_conns=2&resultcode=0&timecost=1&join_createtable_table=0&join_filling_table=0&join_do_query=0
```

锁: 事务时耗长 (续)

- 扁鹊历史会话分析



锁: 事务时耗长 (续)

- 扁鹊的历史会话分析功能可以根据
 - 22:00:37
 - T01_NOR_CUST_INFO

两个条件找出begin...commit
包含22:00:37且执行过对表
T01_NOR_CUST_INFO有写操
作的事务信息

时间	session1	session2
	begin	
	Insert/update/delete /select ... for update T01_NOR_CUST_INF O	
22:00:37		update T01_NOR_CUST_INFO... WHERE ... ERROR: Lock wait timeout exceeded, try restarting transaction
	commit	

锁: 事务时耗长 (续)

实例详情

Set管理

数据库管理

DB监控

Proxy监控

异常会话

实例监控

告警查询

日志管理

备份&恢复

性能分析

检测报告SQL优化实时诊断会话检查表空间分布故障诊断SQL审计

实例Idgroup_1556160454_88

查询时间区间2018-10-25 21:00:00to2018-10-25 23:00:54

事务包含的时间点2018-10-25 22:00:37

事务持有过行锁的表名T01_NOR_CUST_INFO

事务执行时耗(ms)0~200000000

是否只输出异常事务否

开始检查

审计结果

[transaction hash: 007a3457c8612a5e0b76c000430353]
[exception]:
该事务在2018-10-25 22:00:12 724548对T01_NOR_CUST_INFO有update操作, 经过60589ms才提交事务, 在这期间可能引起其他对T01_NOR_CUST_INFO有写操作会话阻塞
2018-10-25 22:01:13 314048: 事务中最后两条SQL的执行间隔(60463ms)过长, 建议在事务中执行完所有业务SQL后及时执行commit/rollback释放锁资源, 减少锁冲突带来的性能损耗。
事务总执行时耗(60624ms)过长不利于锁资源的释放, 可能引起锁等待的增加降低并发性影响性能。

[statistics]:
sum_query_interval: 60600, sum_query_timecost: 2, max_query_interval: 60463, max_query_timecost: 1, trans_query_count: 43, trans_timecost: 60624, trans_affect_rows: 10]
[table_with_write_lock]:
2018-10-25 22:00:12 712652: ecif.T01_NOR_CUST_INFO_HIS, release_interval: 60601
2018-10-25 22:00:12 724548: ecif.T01_NOR_CUST_INFO, release_interval: 60589
2018-10-25 22:00:12 737103: ecif.T01_NOR_CUST_EXTEND_INFO_HIS, release_interval: 60576
2018-10-25 22:00:12 746094: ecif.T01_NOR_CUST_EXTEND_INFO, release_interval: 60567
2018-10-25 22:00:12 751470: ecif.T03_NOR_PHYSIC_ADDRESS_HIS, release_interval: 60562
2018-10-25 22:00:12 753843: ecif.T03_NOR_PHYSIC_ADDRESS, release_interval: 60560
2018-10-25 22:00:12 759778: ecif.T03_NOR_PHONE_INFO_HIS, release_interval: 60554
2018-10-25 22:00:12 761279: ecif.T03_NOR_PHONE_INFO, release_interval: 60552
2018-10-25 22:00:12 769837: ecif.T00_NOR_PARTY_RESOLVE_HIS, release_interval: 60544
2018-10-25 22:00:12 774061: ecif.T00_NOR_PARTY_RESOLVE, release_interval: 60539
[query]:
timestamp: 2018-10-25 22:00:12 689846, cliendIP: [redacted] sql_type: Query, db: ecif, user: dba, select_result_sum: 0, conn_tc: 0, autocommit: 1, affect_num: 0, timecost: 0, sql: SET autocommit=0 (interval: 0)
timestamp: 2018-10-25 22:00:12 691687, cliendIP: [redacted] sql_type: Query, db: ecif, user: dba, select_result_sum: 1, conn_tc: 0, autocommit: 0, affect_num: 0, timecost: 0, sql: SELECT
[redacted] ID, PARTY_ID, CUST_NO, LAST_UPDATED_TS, LAST_UPDATED_ORG, CREATED_TS, UPDATED_TS,
INIT_SYSTEM_ID, INIT_CREATED_TS, LAST_SYSTEM_ID, LAST_UPDATED_TS

FR [redacted] CROSS REF
WH [redacted]

and cust_no='2031124331' (interval: 1)
timestamp: 2018-10-25 22:00:12 705235, cliendIP: [redacted] sql_type: Query, db: ecif, user: dba, select_result_sum: 1, conn_tc: 0, autocommit: 0, affect_num: 0, timecost: 0, sql: select

PA [redacted] CUST_NAME, CUST_ENG_NAME, SEX, NATIVE_PLACE, ETHNIC,
NATION, BIRTH_DATE, POLITICAL [redacted] NUM OF BRINGUP,
EDUC_LEVEL, INDIVIDUAL_DIPLOMA [redacted] TE_INSTITUT, GRADUAT [redacted]
DEPARTME [redacted] _DATE,

3.3可靠性问题

- DB当前运行良好，检查潜在风险

- 主备延迟
- 空间不足，碎片空间过大
- 冗余索引
- ...

- DB体检，风险预估

- 系统状态
- 表空间分布
- 冗余索引
- 死锁诊断
- 锁等待诊断
- 慢查询分析
- DB状态检查
- 表检查

检测报告

- 综合分析一段时间内DB统计信息，监控信息等对DB的健康状态评分
- 低评分通常表示DB可能处于风险状态



4. 总结

- 扁鹊的基本情况
- 扁鹊的基本架构
- 诊断原理与实践
 - 主备切换
 - 性能问题
 - 锁等待
 - 可靠性问题
 - 诊断报告

腾讯云数据库客户案例





关注“**腾讯云数据库**”官方微信

体验**移动端**一键管理数据库

获取数据库技术干货和最新资讯

立享**10元**腾讯云代金券

