CHINA POSTGRESQL ASSOCIATION

浅析腾讯云 PG 自研特性与实践经验

施博文 腾讯云数据库 内核研发工程师

目录

/01

实践经验分享

腾讯云 PG 近年来的一些“踩坑”经验

实践经验分享

发布订阅复制槽被占满

- **问题：**只建立 1 个发布订阅，设置 max_sync_workers_per_subscription=1, max_replication_slots=92，发布订阅在全量同步阶段卡住（无并发）。

- 发布端报错：复制槽全部被名字叫 pg_xxx_sync_xxx 的 slot 占满
- 订阅端报错：origin 被占满 *（origin 用来标记逻辑复制来源）*

发布端报错

```
[redacted] UTC [13953] STATEMENT:  CREATE_REPLICATION_SLOT "pg_25390_sync_19203_732420
1423076974973" LOGICAL pgoutput (SNAPSHOT 'use')
[redacted] UTC [13955] ERROR:  replication slot "pg_25390_sync_23037_73242014230769749
73" does not exist
[redacted] UTC [13955] STATEMENT:  DROP_REPLICATION_SLOT pg_25390_sync_23037_732420142
3076974973 WAIT
[redacted] UTC [13955] ERROR:  all replication slots are in use
[redacted] UTC [13955] HINT:  Free one or increase max_replication_slots.
```

订阅端报错

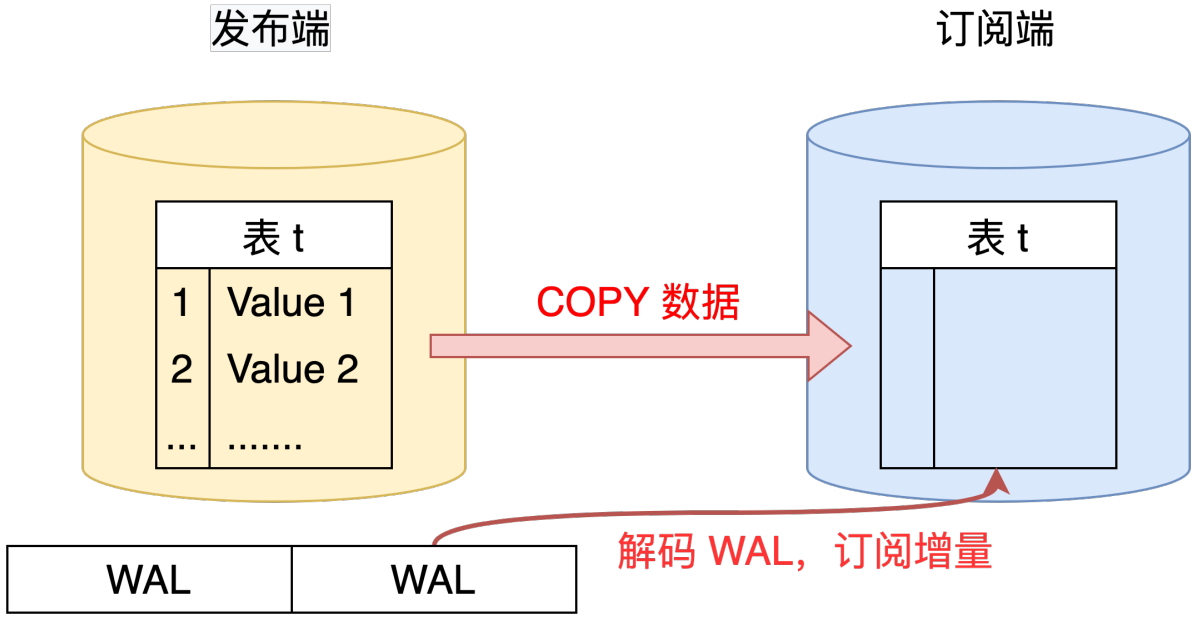
```
[redacted] CST,,,59664,,65a5f416.e910,2,,[redacted] CST,[redacted],ERROR,5
3400,"could not find free replication state slot for replication origin with OID 11",,"Increase max_replic
ation_slots and try again.",,,,,,,"","logical replication worker",,0
```

发布订阅复制槽被占满

➢ 原理介绍

针对表 t 的订阅，分为全量和增量两阶段：

1. 全量阶段：将发布端中 t 的数据 COPY 到订阅端；
2. 增量阶段：全量阶段结束后，解码 WAL，将关于表 t 的 WAL 解码成逻辑日志，发送给订阅端；

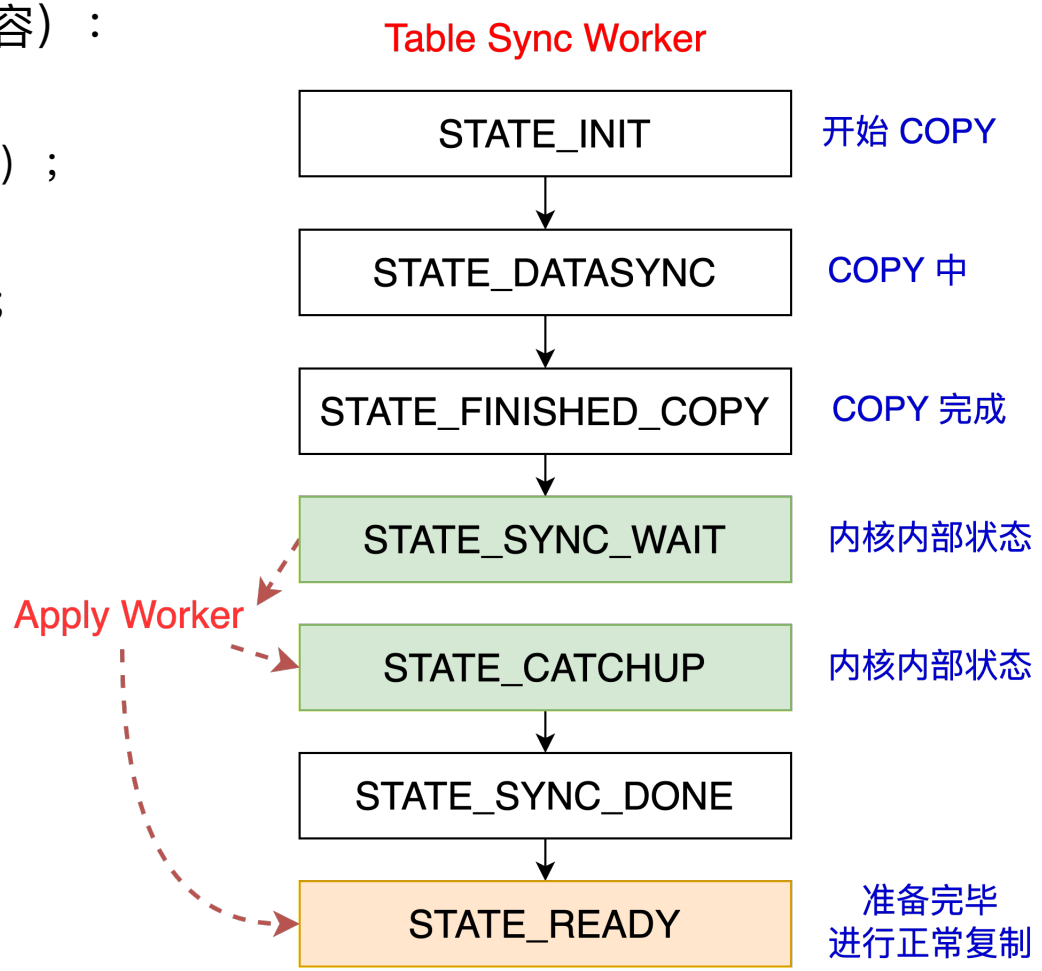


实践经验分享

发布订阅复制槽被占满

➢ 全量阶段订阅端分为以下步骤处理（省略无关内容）：

1. apply 进程启动 table sync worker（表同步进程）；
2. table sync 进程在发布端创建 pg_sync_xxx slot;
3. table sync 进程在订阅端创建 origin；
4. 全量同步（COPY）数据；
5. 同步完成，table sync 进程删除发布端 slot；
6. apply 进程删除 origin；
7. 全量阶段结束，进入普通逻辑复制状态（r）

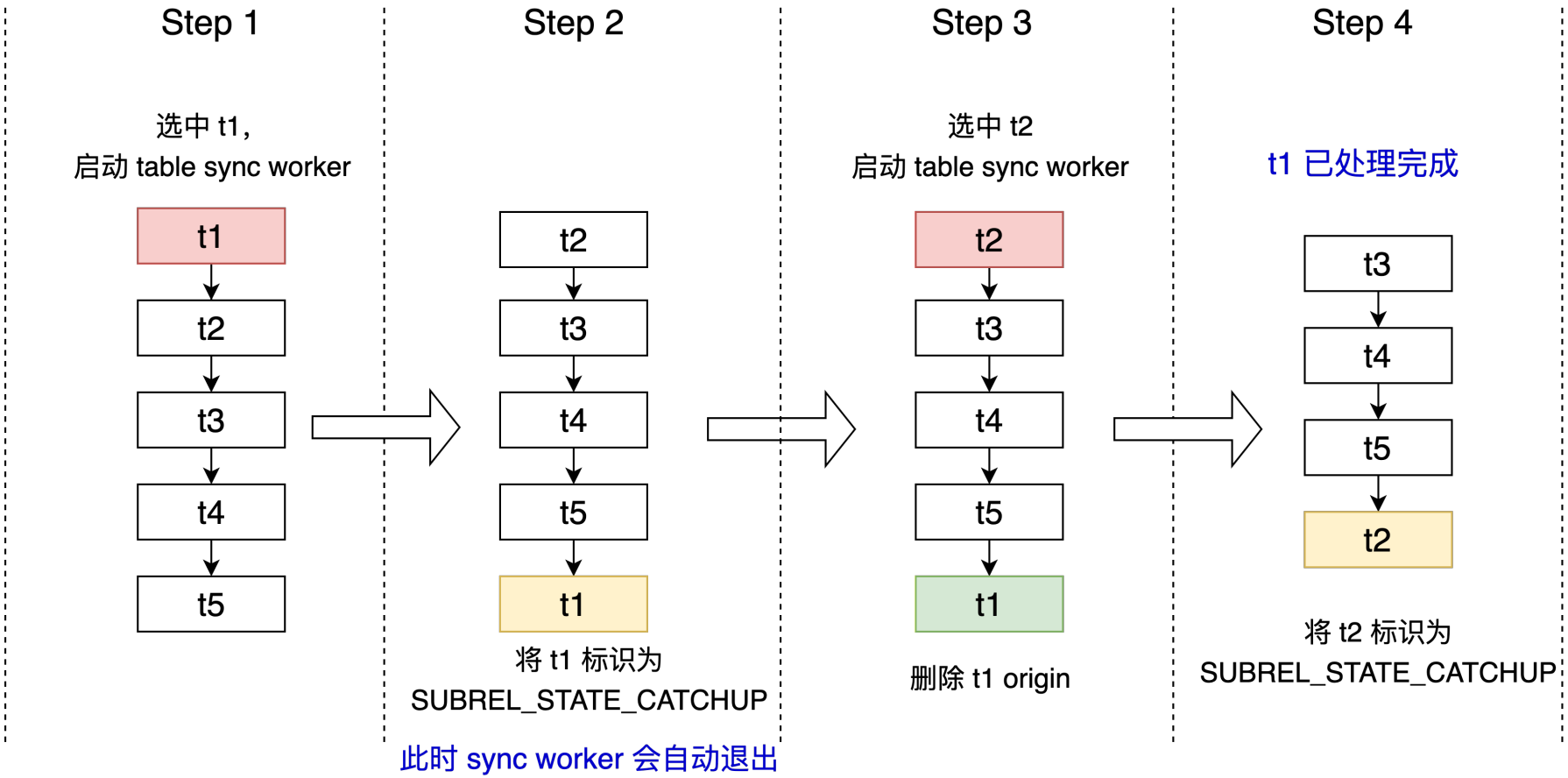


实践经验分享

发布订阅复制槽被占满

➢ **问题分析：**一个发布订阅包含多张表时，正常处理的场景如下 （ max_sync_workers_per_subscription=1）

apply 进程视角

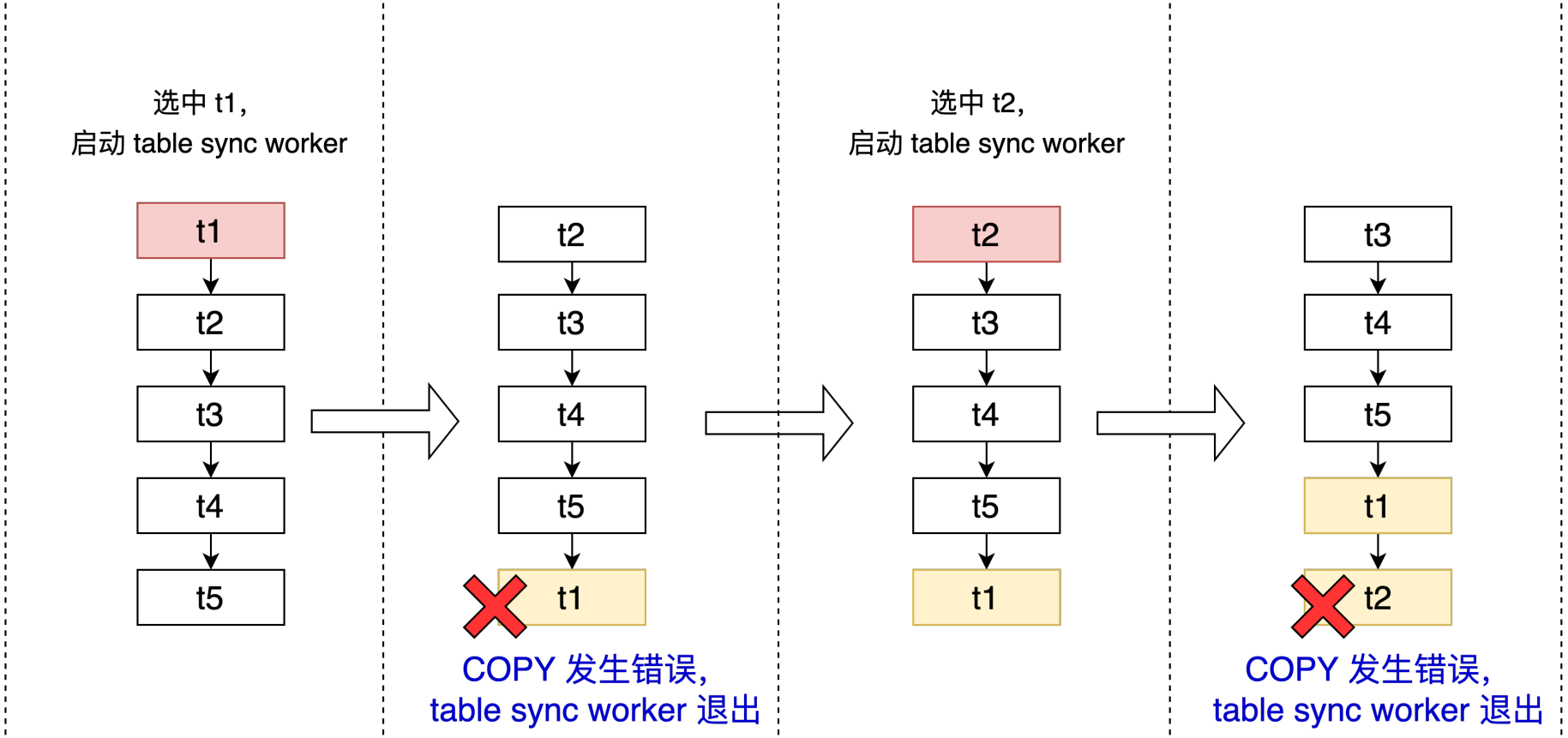


实践经验分享

发布订阅复制槽被占满

- **问题分析：**异常场景下，t1、t2... 在 COPY 阶段出错, 订阅端 origin 未被删除

apply 进程视角



实践经验分享

发布订阅复制槽被占满

- **问题分析：**

1. 表同步阶段出错，订阅端 origin 未释放，最终占满
2. 新启动的 table sync worker 会先在发布端创建 pg_xxx_sync_xxxxx slot，然后发现订阅端 origin 被占满，报错退出，此时发布端的 pg_xxx_sync_xxxxxx slot 并不会被删除（PG 14 特性）；
3. 发布端 slot 被占满；
4. 发布端和订阅端均开始报错，发布订阅卡住，无法继续进行。

- **影响版本：**

PG 14、15（ pg_xxx_sync_xxxxxx slot 从 temporary slot 变为 permanent slot）

实践经验分享

发布订阅复制槽被占满

➢ **问题修复：**

PG 16 已完成部分修复，分为两个 commit 提交（f6c5edb、88f4883），但仍有发生问题的可能

- 社区修复：将 origin 的删除动作交由 table sync worker 完成，让其在退出前先删除订阅端的 origin。
- 异常修复：apply 进程择机将 table sync worker 因为异常退出留下的 origin 删掉
- 临时规避手段：订阅端 max_replication_slot 调大一点

实践经验分享

逻辑复制 walsender 进程卡住

- **背景**：用户在使用逻辑复制时，发现 walsender 进程卡在同一个位点很久（超过 1h），始终不继续向前推进
- **现场**：perf 图显示 CPU 卡在 hash_seq_search 函数上，pg_waldump 对应的 lsn 是一条 drop publication 语句。

进程堆栈

```
#0  hash_seq_search ()
#1  rel_sync_cache_publication_cb () from pgoutput.so
#2  CallSyscacheCallbacks ()
#3  ReorderBufferCommit ()
#4  DecodeXactOp ()
#5  LogicalDecodingProcessRecord ()
#6  XLogSendLogical ()
#7  WalSndLoop ()
#8  exec_replication_command ()
#9  PostgresMain ()
#10 ServerLoop ()
#11 PostmasterMain ()
#12 main ()
```

WAL 信息

rmgr: Transaction len (rec/tot): 475877/475877, tx: 1511906902, lsn: 1F627/6ADC7DB8, prev 1F627/6ADC7D80, desc: COMMIT 2022-12-08 11:22:54.989016 CST; inval msgs:

catcache 45 catcache 44 catcache 47 catcache 46 catcache 47 catcache 46 catcache 47 catcache 46 后面省略很多个 catcache

实践经验分享

逻辑复制 walsender 进程卡住

- **问题复现 （版本 ≤ PG 13)**：

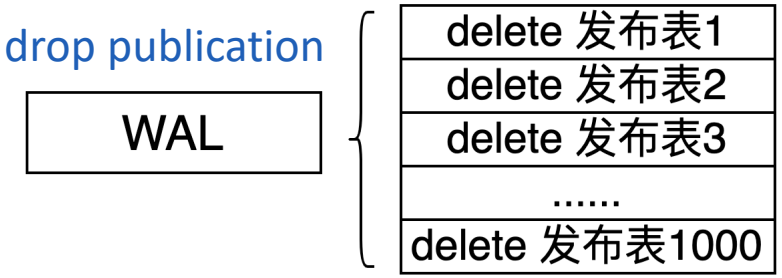
1. 创建两个 PG 实例，一个作为发布端，一个作为订阅端，创建一张表 t，并建立一个只包含表 t 的发布订阅；
2. 发布端创建 10w 张表；
3. 发布端创建一个发布 pub_large，只包含 1000 张表；
4. 发布端向 10w 张表每张表里插入一条数据；
5. 发布端运行 insert into t values(1); ，此时订阅端能很快接收到此条数据；
6. 发布端运行 drop publication pub_large;
7. 发布端运行 insert into t values(1); ，此时订阅端无法接收到此条数据；

实践经验分享

逻辑复制 walsender 进程卡住

- **原因：重复遍历哈希表**
  1. pgoutput invalid cache 回调函数多次重复遍历哈希表（无修改情况）；
  2. drop publication → 从系统表中删除 1000 条元组 → 生成 4000 条 invalid cache 消息

     walsender 进程解析到该条 WAL 日志时，针对每条 invalid cache 消息都会从哈希表中删除对应元素；



RelfilenodeMap 哈希表：通过 value 找 key，顺序遍历哈希表所有元素并删除对应的

时间复杂度： O (1000 * 4000 * 100000) ≈ O (10^{11})

command-id 数量　　invalid cache 消息数量　　哈希表元素个数

实践经验分享

逻辑复制 walsender 进程卡住

- **TencentDB for PG 优化（版本 ≤ PG 13 ）**
  1. pgoutput 增加 lazy flag，避免无效重复遍历哈希表；
  2. 建立反向哈希表 (relid → relfilenode)

     时间复杂度：O (1000 * 4000 * 1) ≈ O(10^6)

- **社区优化（版本 ≥ PG 14）**
  1. 新增一种 XLOG_XACT_INVALIDATIONS 日志类型，记录 invalid cache 消息

     时间复杂度：O (1000 * 4 * 100000) ≈ O(10^8)

/02

内核自研特性介绍

腾讯云 PG 做了哪些事情

内核自研特性介绍

CHINA POSTGRESQL ASSOCIATION

数据库审计日志

- 支持极速版和精细版两种模式的审计
- 针对开启审计后的场景进行性能优化
- 对接 dbbrain，增强分析能力

实例进程级监控

- 支持进程级别的 CPU/内存监控

全方位数据安全能力

- 提供TDE数据加密能力，实现数据存储透明加密
- tencentdb_superuser 云上管理员权限体系
- 多方面内核安全加固措施

cos_fdw 冷热分离插件

- 将不常用数据落冷到 COS，使用 cos_fdw 可查询存储在 COS 上的数据，最大程度帮助用户降本增效

逻辑复制槽故障转移

- 支持 failover slot，HA 时用户逻辑复制不受影响

pg_basebackup 优化

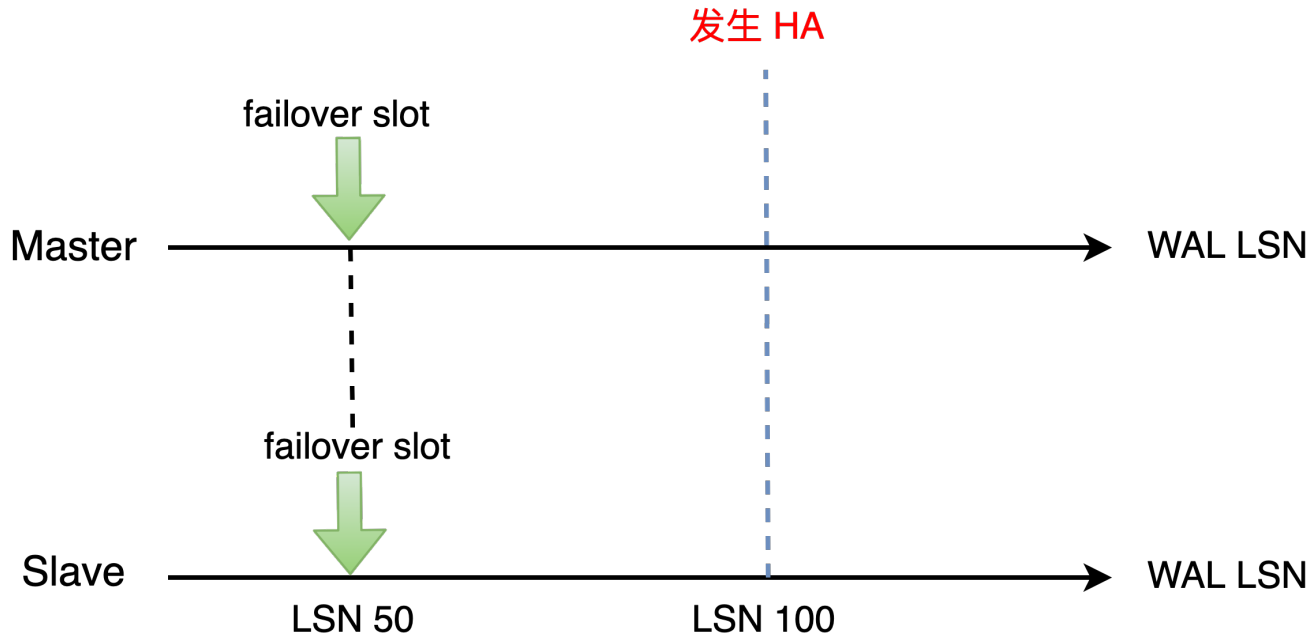
- 支持 wal 和 data 双流合并
- 支持 pg_basebackup 时过滤指定文件或文件夹

内核自研特性介绍

逻辑复制槽故障转移 （Failover Slot)

- 实例 HA 后用户可以无感知的继续使用原有的逻辑复制

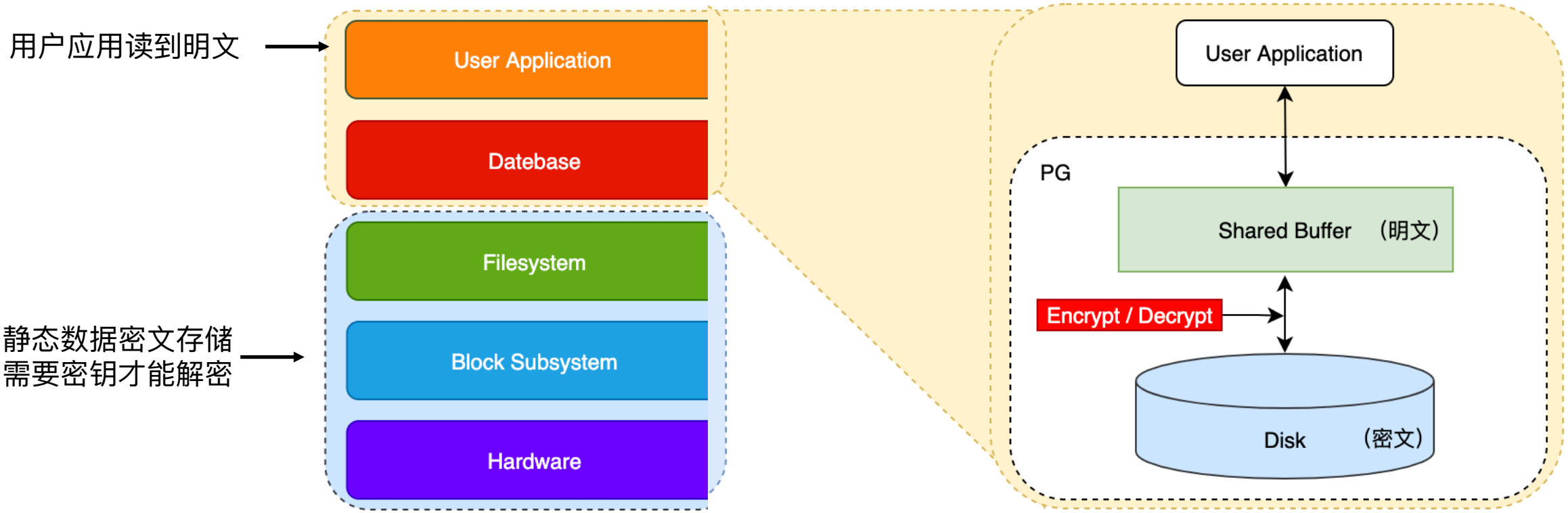
- 功能介绍
  - tencentdb_failover_slot 插件
    (创建/删除/查询 failover slot)
  - WAL 日志记录 slot 信息，主备同步；
  - 与社区 PG 同一大版本兼容
  - 针对异常情况做特殊处理



内核自研特性介绍

TDE 透明数据加密

- 数据在硬盘上读写时进行加/解密，用户层使用无感知

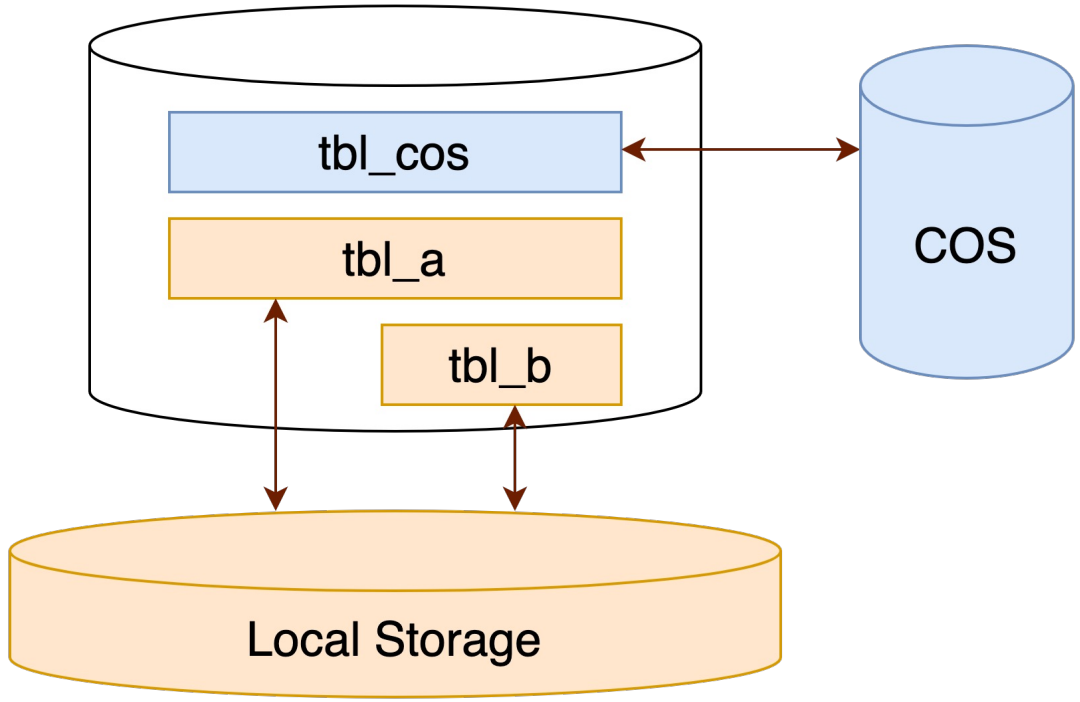


内核自研特性介绍

冷热存储分离

- **cos_fdw 插件：**将不常用的数据放到 COS（对象存储） 上，使用 cos_fdw 可直接将对象存储中的数据当作外部表来访问

```
SELECT
        a.id,b.name,c.value
FROM
        tbl_a a,tbl_b b,tbl_cos cos
WHERE
        a.id = b.id .....
```



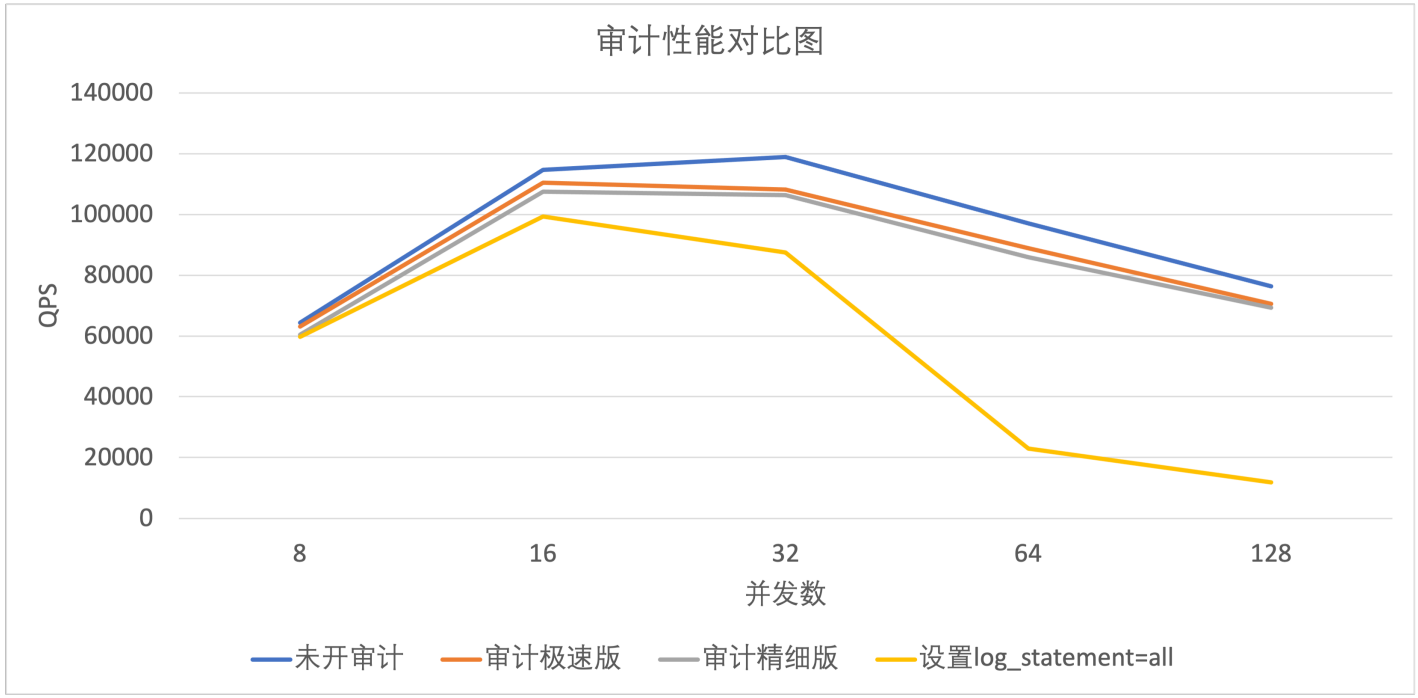
内核自研特性介绍

审计日志功能

- **支持 “极速版” 和 “精细版” 两种模式：**用户可根据自身业务场景动态调整
  - 极速版：对标 PG 原生的全 log_statement = all
  - 精细版：对标 pgaudit 插件的能力，支持审计 SQL 类型、对象名
  - 以上两种模式，都额外新增审计影响行数、执行时长的能力

- **审计日志性能优化**
  - 现网 8C 32G 测试结果如右图所示
  - 高并发场景下性能提升明显

- **对接 DBbrain 平台**
  - AI4DB，帮助用户分析业务场景



内核自研特性介绍

CHINA POSTGRESQL ASSOCIATION

实例进程监控

- 支持进程级别的 CPU/内存 使用率监控
  - 插件的方式提供监控视图，对内核无侵入
  - 读取操作系统 /proc/pid/stat 提供的统计信息并计算，保证准确性和及时性
  - 易用性高，支持按照 database、进程类型进行分类统计，适用于 SaaS 场景

```
postgres=# select pid,query,cpu_usage,memory_bytes from tencentdb_process_system_usage ;
  pid  |                                    query                                    | cpu_usage | memory_bytes
-------+-----------------------------------------------------------------------------+-----------+--------------
 30744 |                                                                             |         0 |      7921664
 30741 |                                                                             |         0 |      4300800
 30742 |                                                                             |         0 |      8736768
 30767 | select sleep_and_loop2(0.66);                                               |      0.66 |     15556608
 30740 |                                                                             |         0 |      4300800
 30764 | select pid,query,cpu_usage,memory_bytes from tencentdb_process_system_usage ; |         0 |     16990208
 30739 |                                                                             |         0 |      4300800
(7 rows)
```

/03

未来与展望

腾讯云 PG 的未来

未来与展望

CHINA POSTGRESQL ASSOCIATION

TencentDB for PG 未来

➢ 云原生化

- 打造基于云原生的操作系统
- 一云多芯

➢ 智能化

- AI4DB：数据库自治，协助 DBA 简化运维；
- DB4AI：在数据库中使用机器学习的算法处理、分析数据；

➢ Serveless 化

- 极致弹性，从卖资源转向卖能力，释放云计算资源池化潜力，实现客户与云厂商的双赢；
- 多级存储机制，冷数据高效查询分析，进一步帮助用户降本增效；

THANKS

总想玩世不恭

江苏 南通

扫一扫上面的二维码图案，加我为朋友。