openGauss融合引擎演进

业春林

个人简介

职位：华为内部配套产品数据库规划，openGauss内核产品管理代表

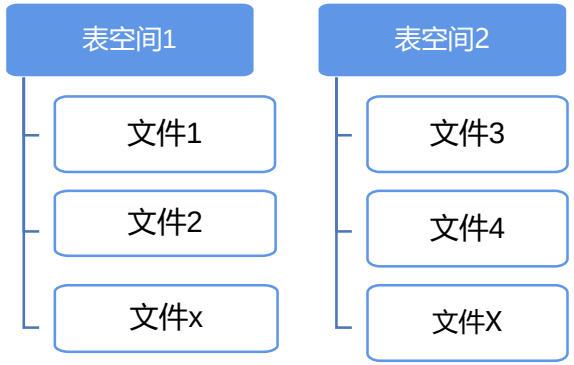
职责：负责华为内部各产品线数据库国产化替代规划，对广泛的各种部署模式（云/ on-premise /消费者终端），各种业务场景（运营商核心系统、网络/网元设备管理、嵌入式终端设备）下的数据库负载/数据库选型/解决方案有充分理解，同时对口openGauss内核产品管理工作，同时对口openGauss内核产品管理工作，希望和广大用户、开发者、DBV伙伴充分交流互动，共同繁荣openGauss生态

01 **业务场景&痛点**

02 openGauss融合引擎架构

应用使用PG原生数据库遇到的几个典型痛点

- **物理数据文件数量膨胀**



场景：

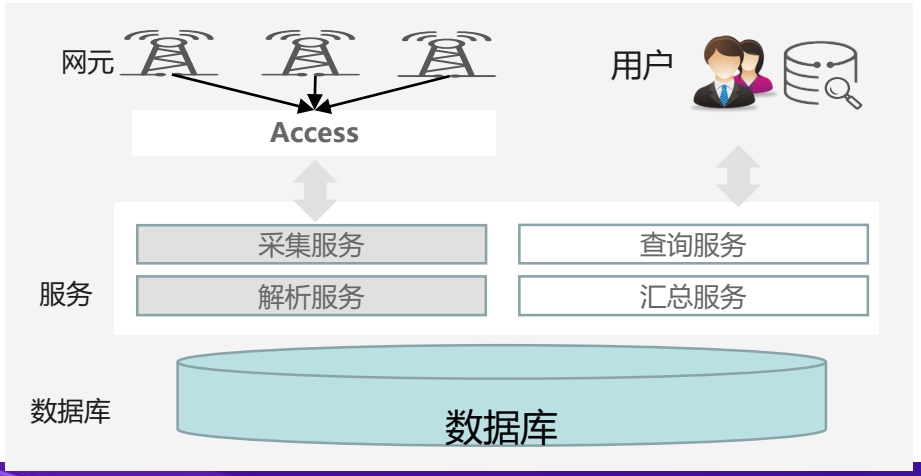
✓部分系统因历史兼容性或设计原因，单实例表数量超过50万+

✓面向中小开发者的Serverless多租户数据库服务，可能在一个节点上多个开发者累计建了很多表

问题&挑战：

✓基于原生PG机制，一个表（同时建索引）会触发创建多个物理文件，导致单个数据库物理文件数量太多，严重影响性能

- **存储空间膨胀**



场景：

✓管理数据量大，最大单节点数据：8T

✓数据入库、更新、删除频繁

问题&挑战：

✓存储空间膨胀严重，大约有7倍膨胀

✓需要开发人员写vacuum操作

✓事务ID回卷

01 业务场景&痛点

02 **openGauss融合引擎架构**

openGauss关键技术：融合存储引擎

SQL引擎

访问抽象

存储引擎

基于统一执行引擎与存储引擎抽象，实现一套架构支持多种存储引擎（多模态）

关键设计原则：

- 统一事务机制
- 统一日志系统
- 统一并发控制系统
- 统一元信息系统
- 统一缓存管理系统

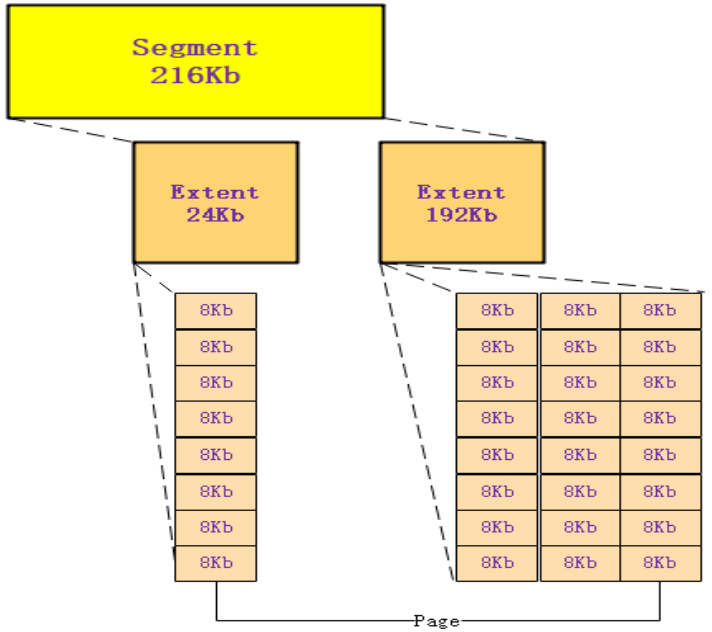
关键技术：

- 通过分离数据页面与索引页面回收机制，降低页面回收CPU负载
- 通过在日志中记录数据增量修改，降低日志空间占用

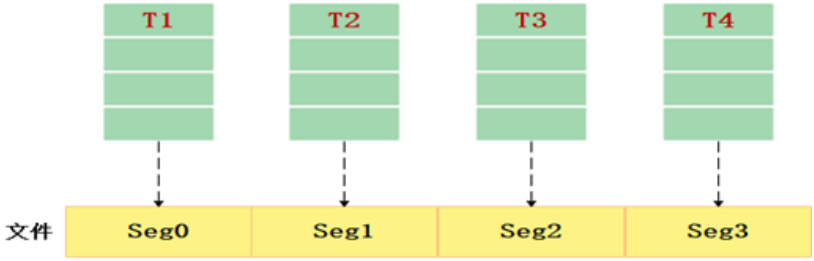
架构优势：

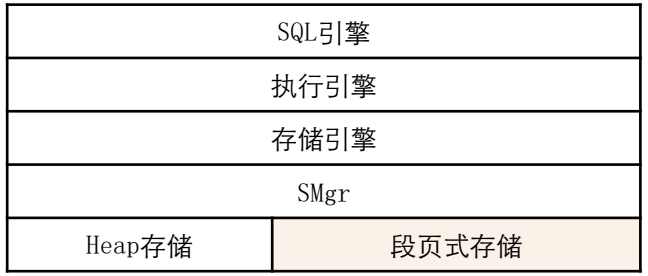
- 采用分层/隔离设计原则，组件化，可替换性，可独立演进
- 可适应计算存储分离等后续架构的演进

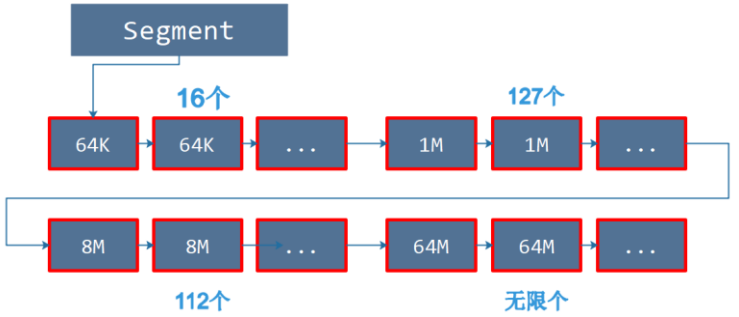
openGauss关键技术: 段页式存储



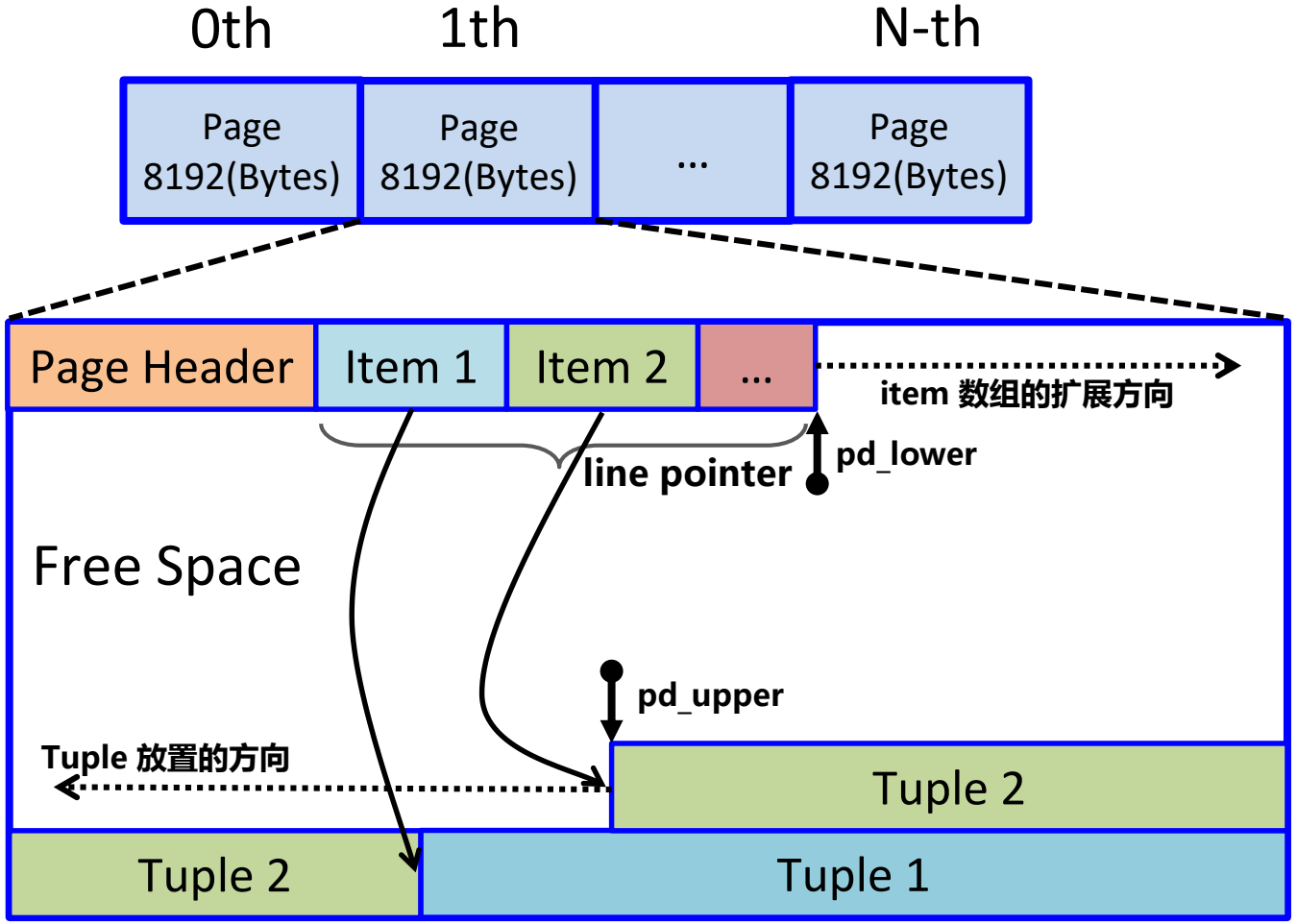
1) 不同table的数据存储在同一个物理文件上，可以控制系统的物理文件数量，避免在分区表过多时对文件系统的压力。
2) 抽象出逻辑页号，统一使用 SMGR 接口，向上层同时提供段页式存储和heap存储能力。代码模块划分清晰。
3) 段页式组织上，采用静态扩展策略。即前16个extent大小固定为64K，接下来127个extent大小为1M，依次类推。扩展粒度逐渐增大，控制段页式造成的存储空间浪费比例。
4) Extent的地址是存储在二级页表中。静态扩展策略允许直接通过逻辑页号计算出extent编号，再通过二级页表查询找到对应的物理页号。
5) Segment Header对应的buffer，在事务周期内会被常pin在内存，避免额外的IO开销。

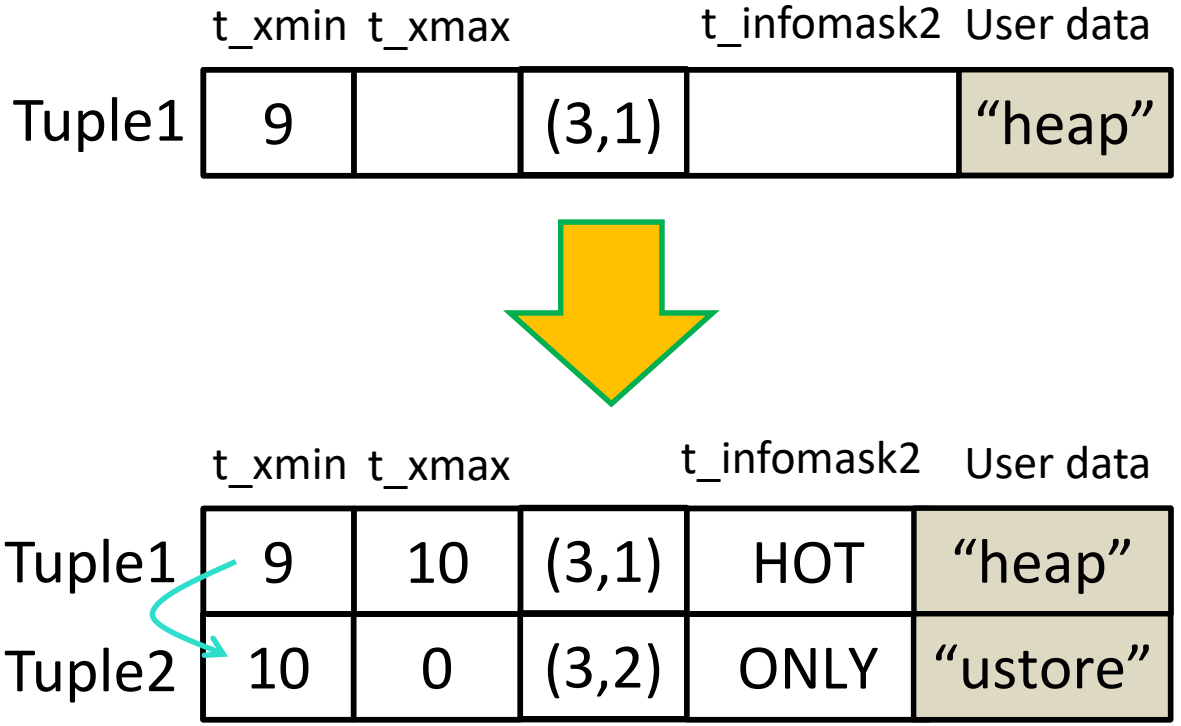






openGauss关键技术：追加写引擎页面结构 （HEAP）



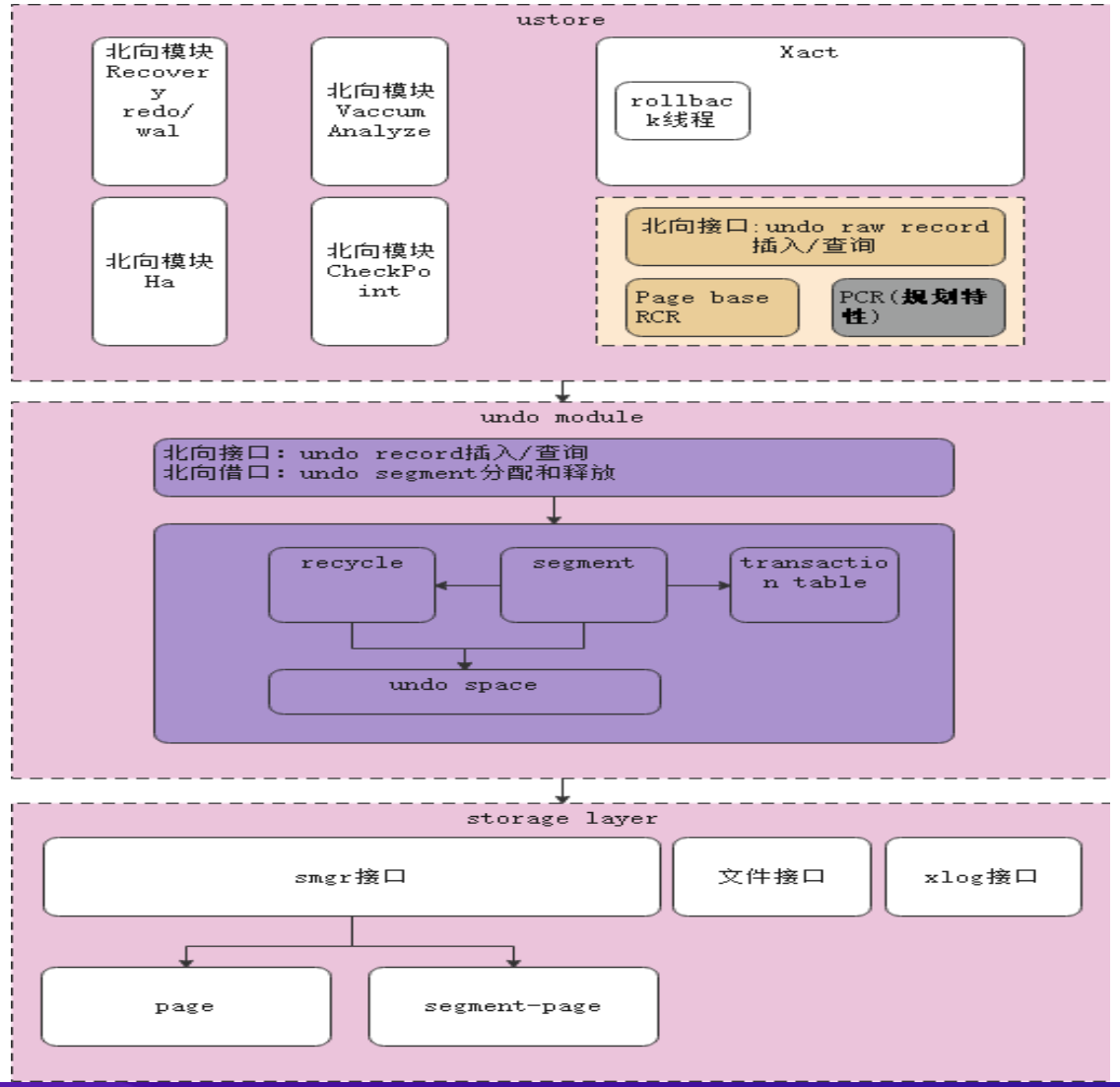


磁盘上存储的页有很多种类，例如：堆页(heap relation)，索引页(index relation)，clog页(提交日志页)，vm页(可见性映射页面)，fsm页(空闲空间管理页)等等。

堆页和索引页的结构图如上图所示，8k的空间包括：页头+数据部分

更新操作采用追加写的方式，新增新元组，将旧元组标记删除

openGauss关键技术: Ustore（原地更新）存储引擎



历史记录版本统一管理和回收，适合更新频繁的业务场景

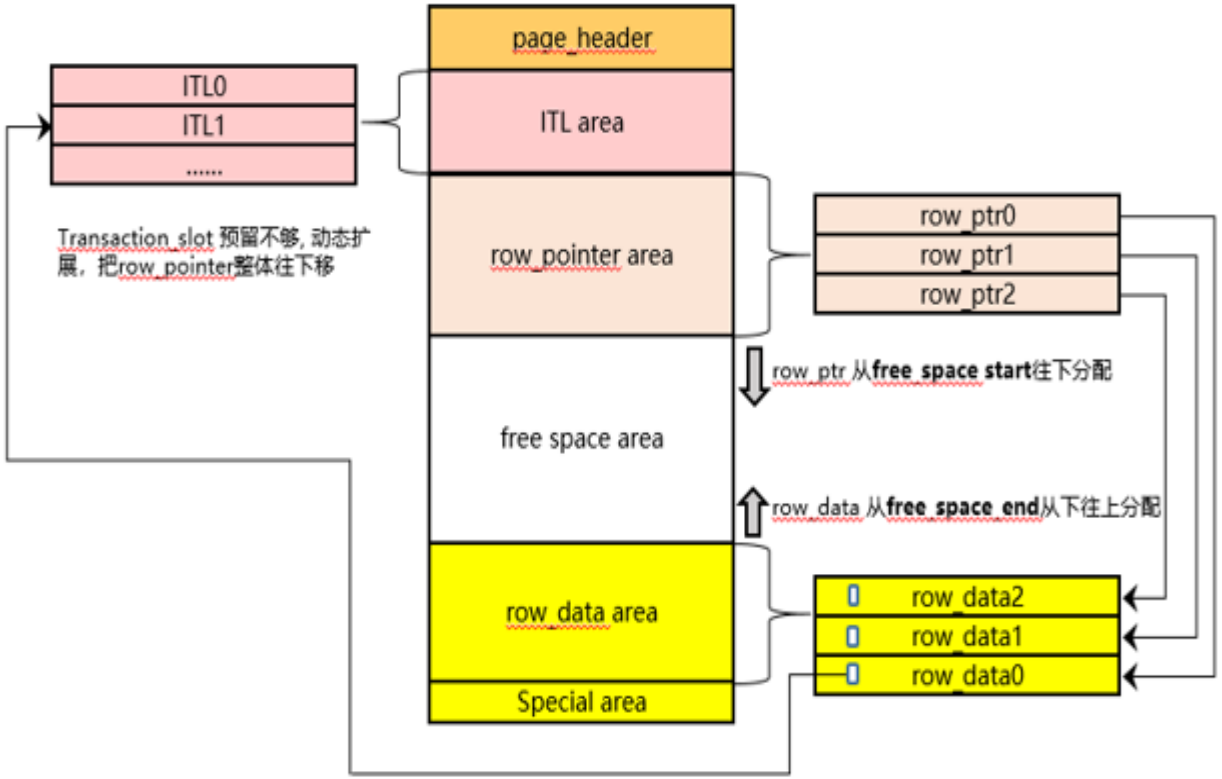
NUMA-Aware 设计

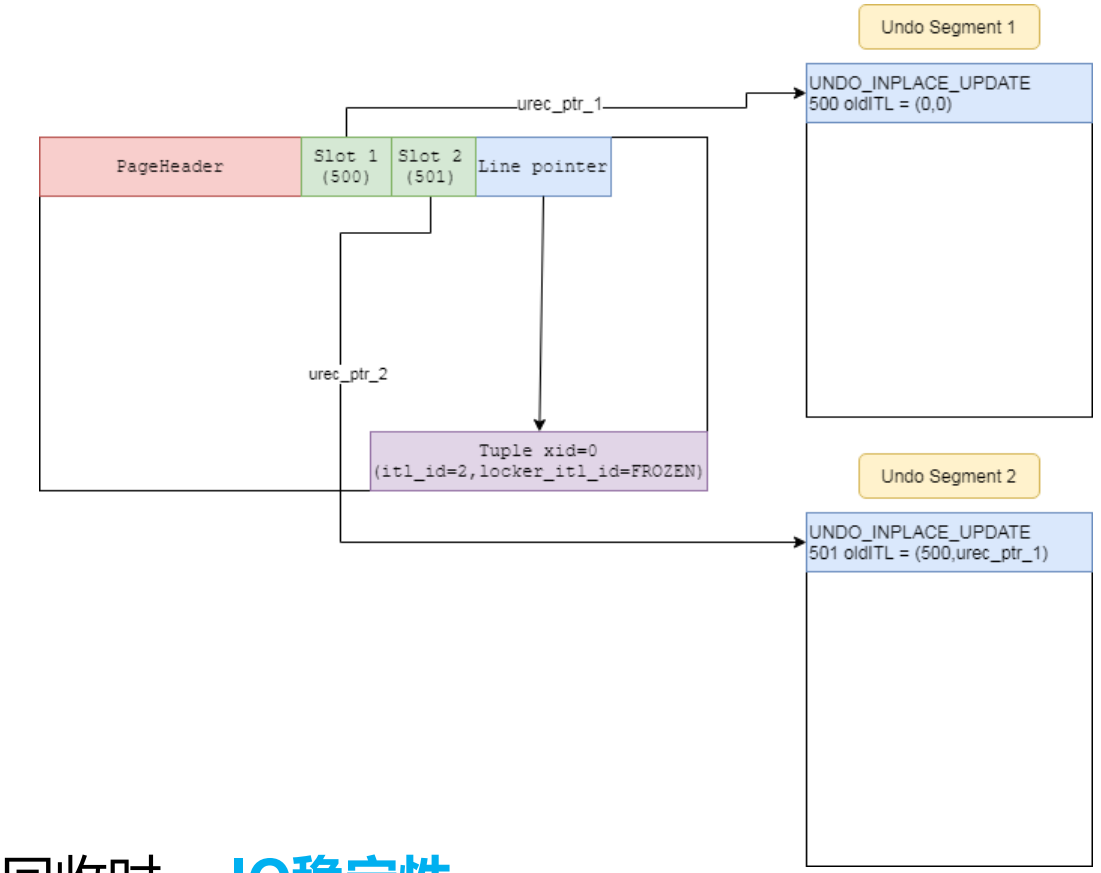
1) Undo空间就近NUMA核分配

2) 按NUMA核粒度进行并发回收

3）去除vacuum

openGauss关键技术: Ustore（原地更新）存储引擎页面结构





- Index支持多版本, 去除vacuum, 保持数据回收时， **IO稳定性**
- index多版本, 去除数据回收时data page与index page关联
- **xlog空间/data空间缩减**

Ustore（原地更新）引擎，性能曲线平顺、时延稳定

目标：存储引擎支持inplace-update，性能稳定性达到TPCC标准

中期进展：

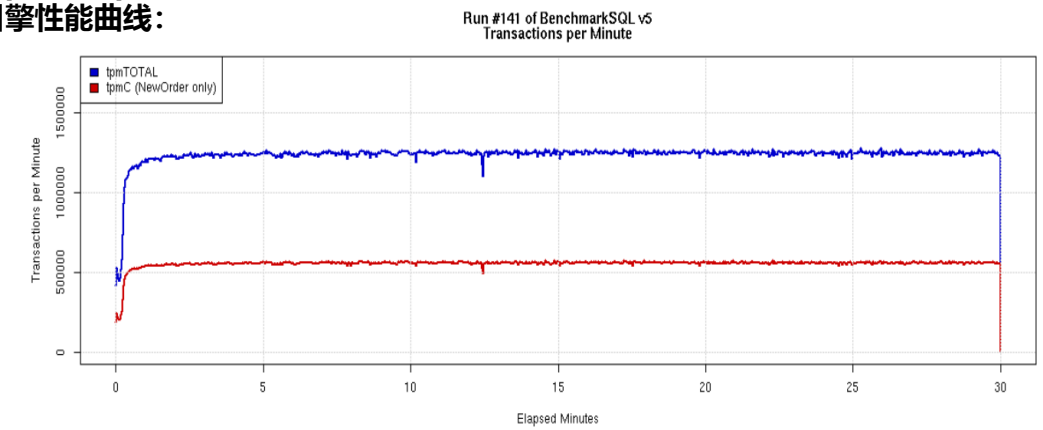
- **原地更新引擎性能和时延输出非常稳定；**
- **运行tpc-c测试模型，性能稳定在50w+ TpmC**

环境：2288v3：2*E5 2650V4、512G内存、2*1600G NVME SSD卡，10GE网络

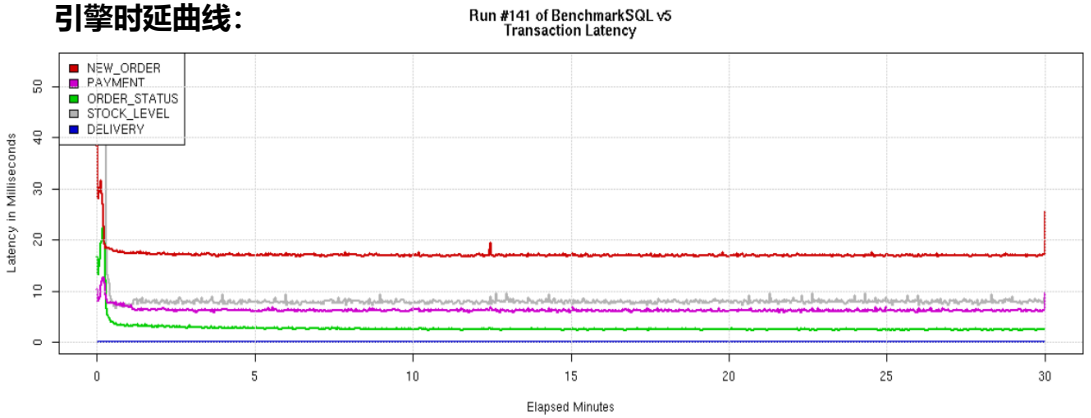
TPCC参数：1000仓（100G）/812并发/30min

存储引擎	tpmc	吞吐量标准差（稳定期）	磁盘占用
ustore	556559.3	7906.42	204G

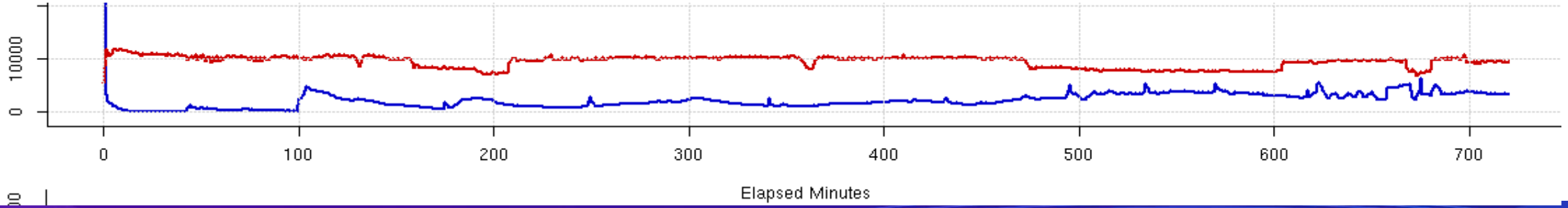
Inplace-update 引擎性能曲线：



Inplace-update 引擎时延曲线：



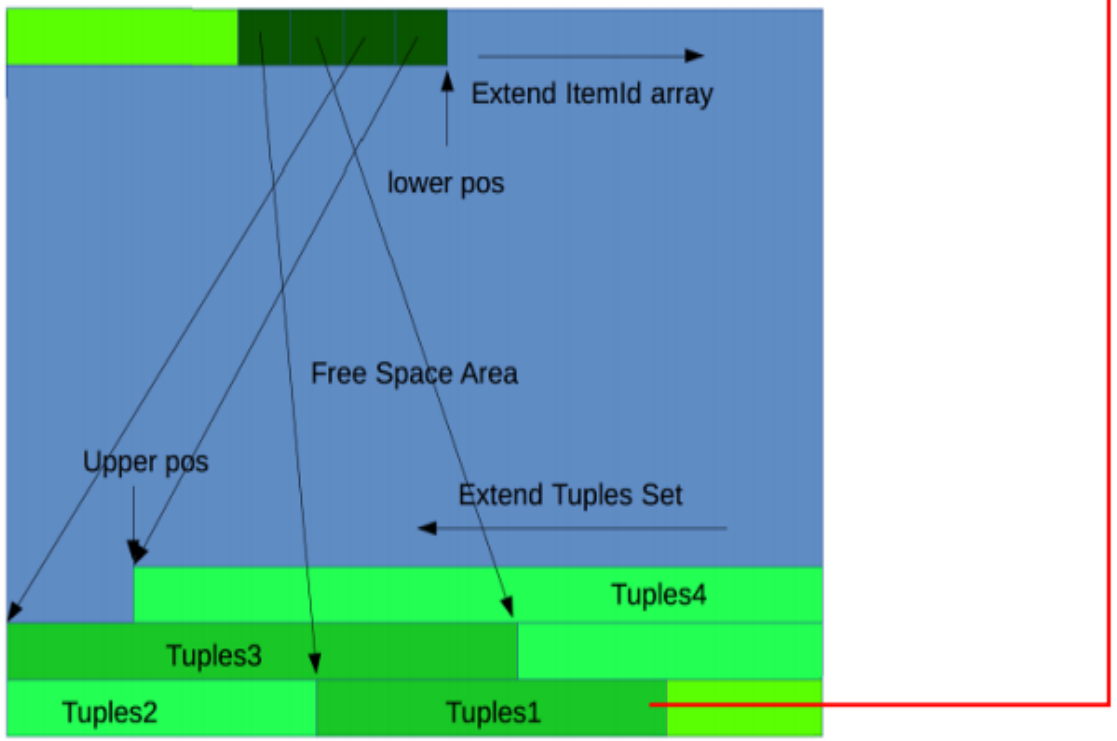
Inplace-update 引擎12小时IO曲线：



openGauss关键技术:列存存储模型

行存储模型- 整行读取

a	b	c	d	e
1	tom	5	2001-03-20	30.5
2	jerry	1	2002-05-16	56.7
3	jeff	9	1990-04-21	13.6



列存储模型 - 按列读取

a	b	c	d	e
1	tom	5	2001-03-20	30.5
2	jerry	1	2002-05-16	56.7
3	jeff	9	1990-04-21	13.6

对于查询SELECT b FROM tbl WHERE c > 2实际对查询结果有影响的列为b、c列

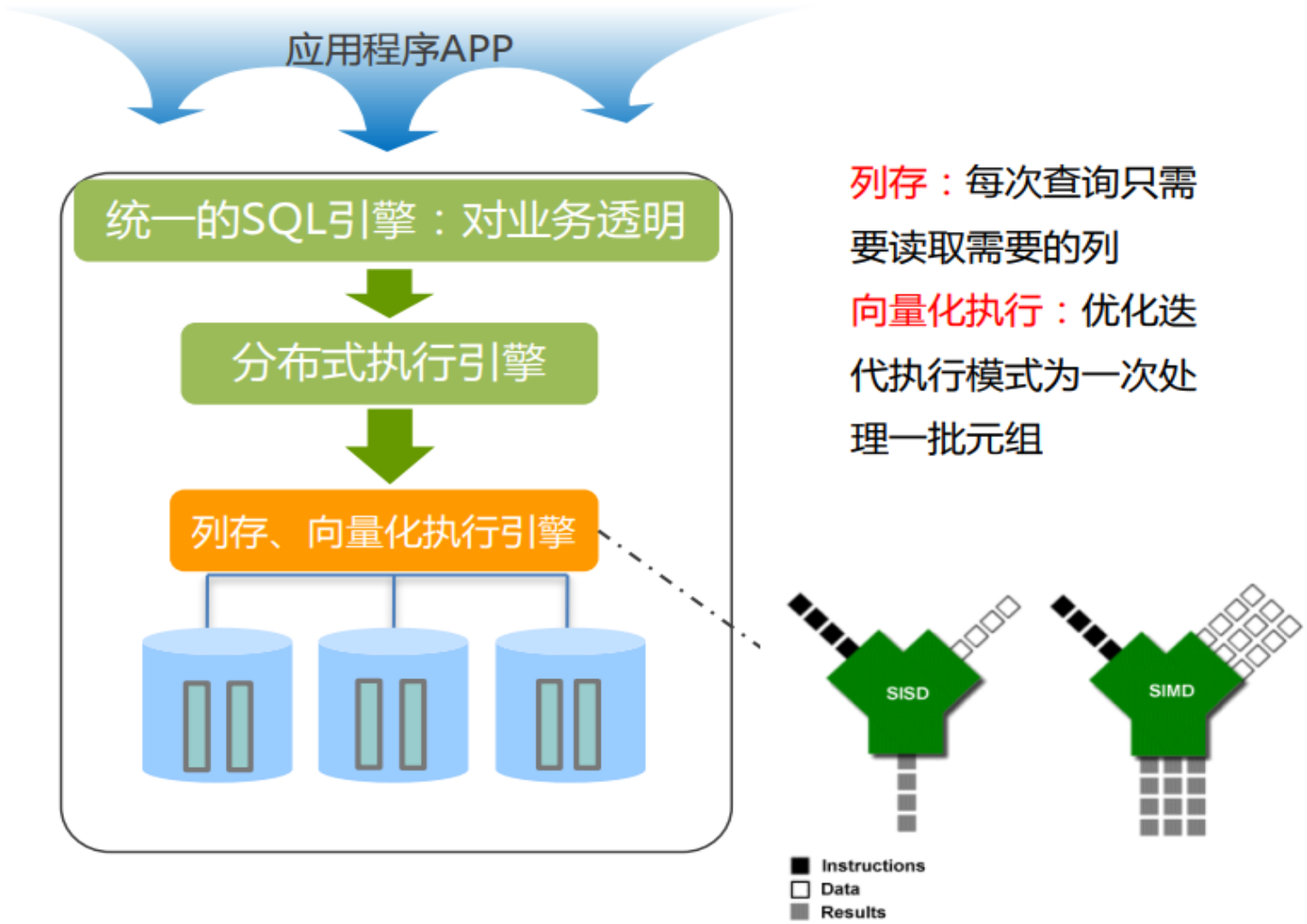
- 在行存储模型下需要读取所有列
- 在列存储模型下仅需要读取需要访问的数据列c，b列

当表tbl很宽时行存储模型相比列存储需要读取大量的无效数据

openGauss关键技术:向量化执行引擎

核心思想，将查询处理过程批量化的元组批量化处理

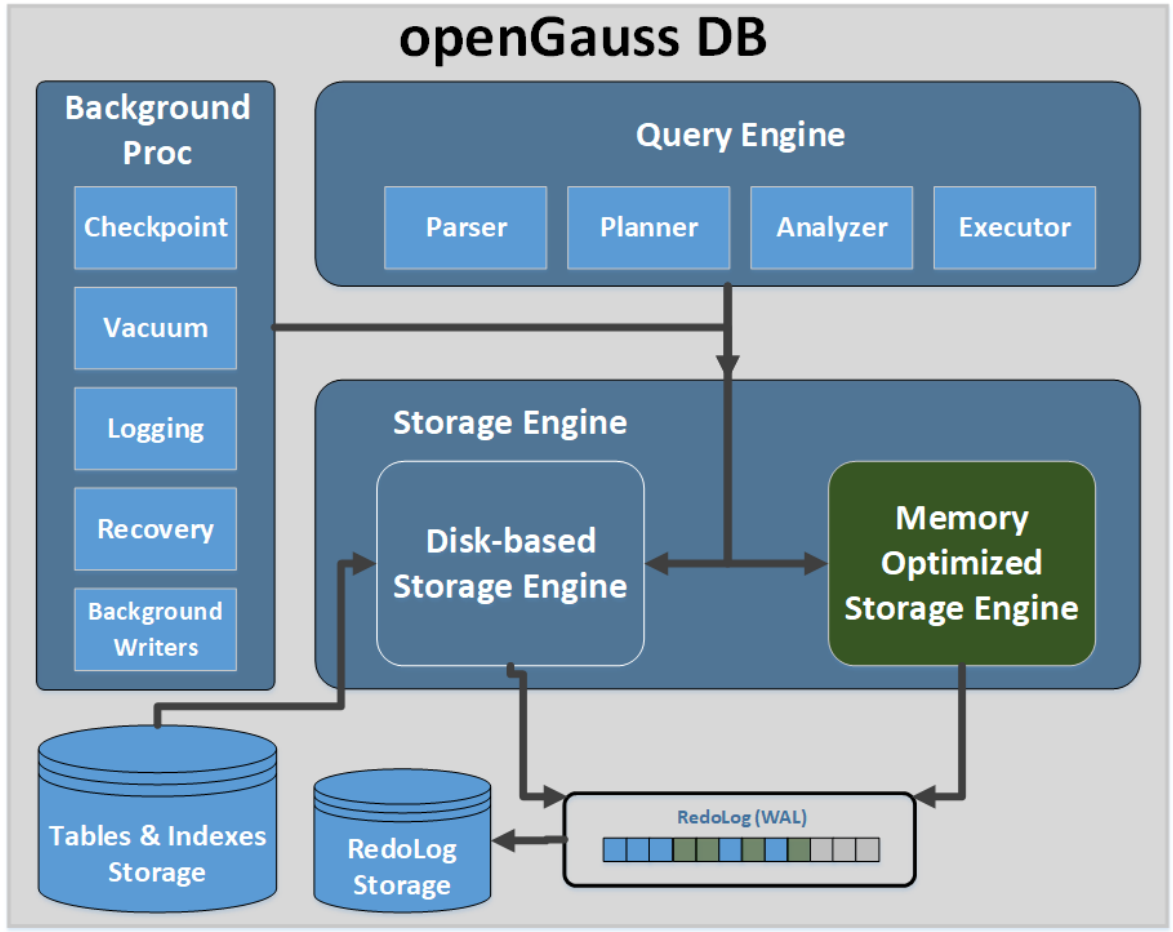
- 数据在行存存引擎中处理的单位是Tuple
- 数据在向量引擎中处理的单位是VectorBatch



列存：每次查询只需要读取需要的列

向量化执行：优化迭代执行模式为一次处理一批元组

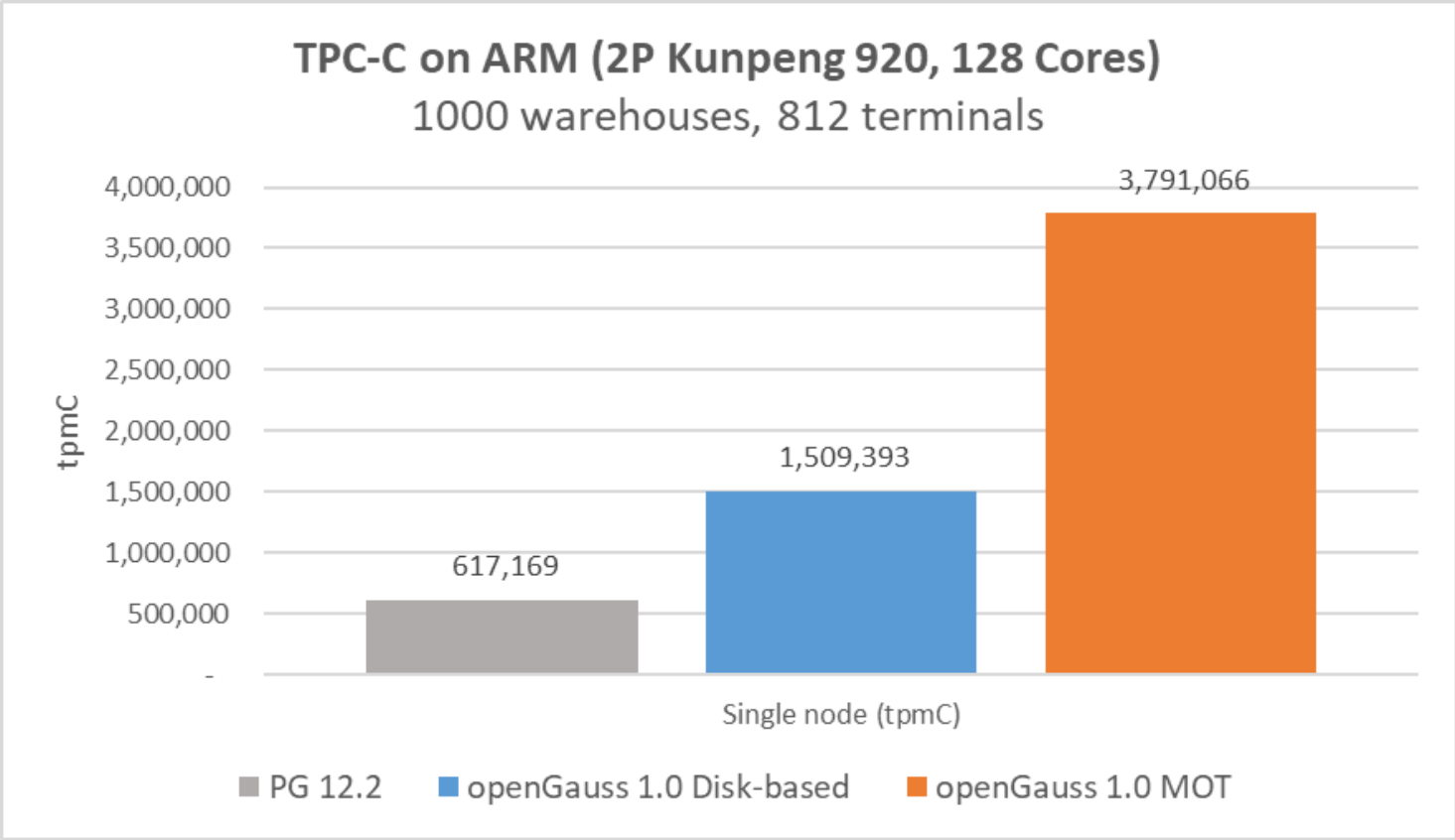
openGauss关键技术: MOT内存引擎介绍



关键能力:

- 行存储
- 基于多核及大内存服务器优化
- ACID compliant + Strict durability + High Availability
- 与openGauss磁盘库完全继承, 支持大部分特性 (including stored procedures, functions ...)
- x86 and ARM64 Kunpeng
- 高性能: 3x vs. DISK-table 6x vs. PG 12.2

openGauss关键技术: MOT内存引擎性能



- MOT 性能高于than PG 12.2倍
- 测试环境: Taishan 2280 v2 2P, 2 socket Kunpeng 920(total 128 Cores), 800GB RAM, 1TB NVMe.