

openGauss
3.0.0

技术白皮书

文档版本 01
发布日期 2022-03-31



版权所有 © 华为技术有限公司 2022。保留一切权利。

非经本公司书面许可，任何单位和个人不得擅自摘抄、复制本文档内容的部分或全部，并不得以任何形式传播。

商标声明



HUAWEI和其他华为商标均为华为技术有限公司的商标。

本文档提及的其他所有商标或注册商标，由各自的所有人拥有。

注意

您购买的产品、服务或特性等应受华为公司商业合同和条款的约束，本文档中描述的全部或部分产品、服务或特性可能不在您的购买或使用范围之内。除非合同另有约定，华为公司对本文档内容不做任何明示或暗示的声明或保证。

由于产品版本升级或其他原因，本文档内容会不定期进行更新。除非另有约定，本文档仅作为使用指导，本文档中的所有陈述、信息和建议不构成任何明示或暗示的担保。

华为技术有限公司

地址： 深圳市龙岗区坂田华为总部办公楼 邮编： 518129

网址： <https://www.huawei.com>

客户服务邮箱： support@huawei.com

客户服务电话： 4008302118

目 录

| | |
|----------------------|----|
| 1 产品定位..... | 1 |
| 2 应用场景..... | 2 |
| 3 技术特点..... | 3 |
| 4 软件架构..... | 4 |
| 5 部署方案..... | 6 |
| 6 典型组网..... | 10 |
| 7 软硬件配置要求..... | 13 |
| 8 数据库核心技术..... | 15 |
| 8.1 面向应用开发的基本功能..... | 15 |
| 8.2 高性能..... | 17 |
| 8.3 高可用..... | 22 |
| 8.4 可维护性..... | 24 |
| 8.5 数据库安全..... | 26 |
| 8.6 AI 能力..... | 30 |
| 9 技术指标..... | 32 |
| 10 术语表..... | 33 |

1 产品定位

openGauss是一款支持SQL2003标准语法，支持主备部署的高可用关系型数据库。

- 多种存储模式支持复合业务场景，新引入提供原地更新存储引擎。
- NUMA化数据结构支持高性能。
- Paxos一致性日志复制协议，主备模式，CRC校验支持高可用。
- 支持全密态计算，账本数据库等安全特性，提供全方位端到端的数据安全保护。
- 通过Table Access Method接口层支持多存储引擎。

openGauss是一款提供面向多核的极致性能、全链路的业务和数据安全、基于AI的调优和高效运维的能力，全面友好开放，携手伙伴共同打造全球领先的企业级开源关系型数据库，采用木兰宽松许可证v2发行。openGauss深度融合华为在数据库领域多年的研发经验，结合企业级场景需求，持续构建竞争力特性。

2 应用场景

- **交易型应用**
大并发、大数据量、以联机事务处理为主的交易型应用，如电商、金融、O2O、电信CRM/计费等，应用可按需选择不同的主备部署模式。
- **物联网数据**
在工业监控和远程控制、智慧城市的延展、智能家居、车联网等物联网场景下，传感监控设备多，采样率高，数据存储为追加模型，操作和分析并重的场景。

3 技术特点

openGauss相比于其他开源数据库主要有以下几个主要特点：

- 高性能
 - 提供了面向多核架构的并发控制技术结合鲲鹏硬件优化，在两路鲲鹏下TPCC Benchmark达成性能150万tpmc。
 - 针对当前硬件多核numa的架构趋势，在内核关键结构上采用了Numa-Aware的数据结构。
 - 提供Sql-bypass智能快速引擎技术。
 - 针对频繁更新场景，提供ustore存储引擎。
- 高可用
 - 支持主备同步，异步以及级联备机多种部署模式。
 - 数据页CRC校验，损坏数据页通过备机自动修复。
 - 备机并行恢复，10秒内可升主提供服务。
 - 提供基于paxos分布式一致性协议的日志复制及选主框架。
- 高安全
 - 支持全密态计算，访问控制、加密认证、数据库审计、动态数据脱敏等安全特性，提供全方位端到端的数据安全保护。
- 易运维
 - 基于AI的智能参数调优和索引推荐，提供AI自动参数推荐。
 - 慢SQL诊断，多维性能自监控视图，实时掌控系统的性能表现。
 - 提供在线自学习的SQL时间预测。
- 全开放
 - 采用木兰宽松许可证协议，允许对代码自由修改，使用，引用。
 - 数据库内核能力全开放。
 - 提供丰富的伙伴认证，培训体系和高校课程。

4 软件架构

openGauss是单机系统，在这样的系统架构中，业务数据存储在单个物理节点上，数据访问任务被推送到服务节点执行，通过服务器的高并发，实现对数据处理的快速响应。同时通过日志复制可以把数据复制到备机，提供数据的高可靠和读扩展。

openGauss支持主备部署，openGauss逻辑架构如图4-1所示。

图 4-1 openGauss 逻辑架构图

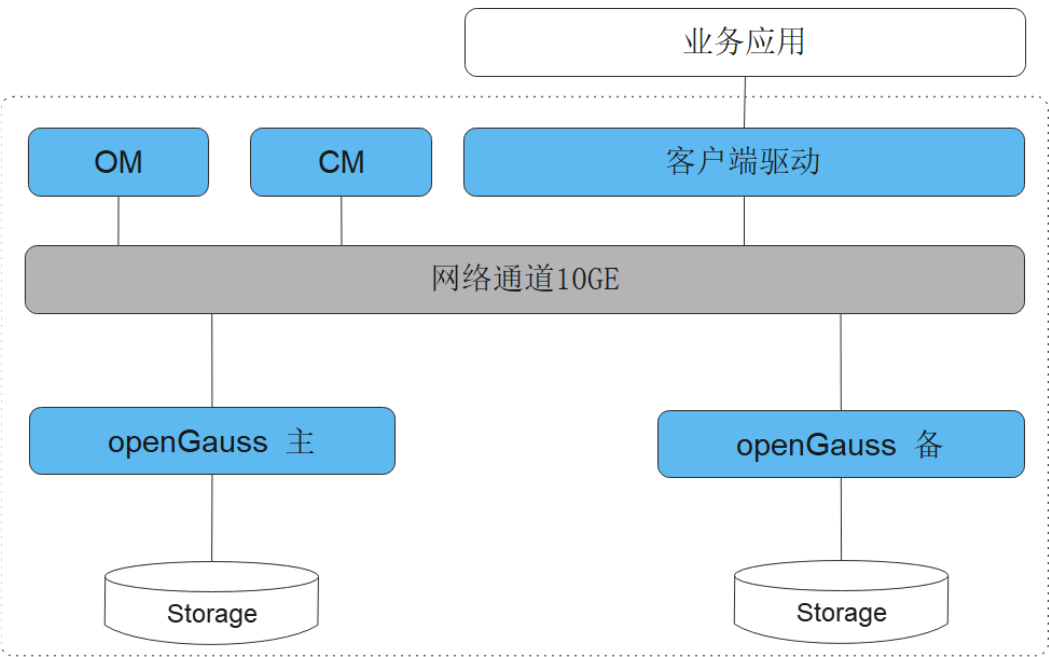


表 4-1 架构说明

| 名称 | 描述 |
|----|---|
| OM | 运维管理模块（Operation Manager）。提供数据库日常运维、配置管理的管理接口、工具。 |

| 名称 | 描述 |
|---------------|---|
| CM | 数据管理模块（Cluster Manager）。管理和监控数据库系统中各个功能单元和物理资源的运行情况，确保整个系统的稳定运行。 |
| 客户端驱动 | 客户端驱动（Client Driver）。负责接收来自应用的访问请求，并向应用返回执行结果。客户端驱动负责与openGauss实例通信，发送应用的SQL命令，接收openGauss实例的执行结果。 |
| openGauss（主备） | openGauss主备（Datanode）。负责存储业务数据、执行数据查询任务以及向客户端返回执行结果。 openGauss实例包含主、备两种类型，支持一主多备。建议将主、备openGauss实例分散部署在不同的物理节点中。 |
| Storage | 服务器的本地存储资源，持久化存储数据。 |

5 部署方案

openGauss主要支持单机部署和一主多备部署两种部署形态。

常用概念

- 单机
单机指的是只有一个数据库实例。
- 双机
双机指的是系统中存在主备数据库实例，主实例支持读写，备实例支持只读。
- 一主多备
一主多备指的是在系统存在一个主机，多个备机。最多支持8个备机。
- 冷热备份
冷备份：是指备份就是一个简单的备份集，不可以提供服务。
热备份：是指备份数据库可以对外提供服务。

部署形态汇总

单机和双机两种部署形态方案介绍请见[表5-1](#)。

表 5-1 openGauss 部署形态汇总表

| 部署形态 | 技术方案 | 高可用 | 基础设置要求 | 业务场景 | 场景特点 | 技术规格 |
|------|------|--------|--------|------|---|--|
| 单机 | 单机 | 无高可用能力 | 单机房 | 物理机 | <ul style="list-style-type: none">• 对系统的可靠性和可用性无任何要求• 主要用于体验试用以及调测场景 | <ul style="list-style-type: none">• 系统RTO和RPO不可控• 无实例级容灾能力，一旦出实例故障，系统不可用• 一旦实例级数据丢失，则数据永久丢失，无法恢复 |

| 部署形态 | 技术方案 | 高可用 | 基础设置要求 | 业务场景 | 场景特点 | 技术规格 |
|------|-----------------------------|---------|--------|------|---|--|
| 主备 | 主机+备机 | 抵御实例级故障 | 单机房 | 物理机 | <ul style="list-style-type: none">节点间无网络延迟要求承受数据库内实例级故障适用于对系统可靠性要求不高的场景 | <ul style="list-style-type: none">RPO=0实例故障RTO<10s无AZ级容灾能力推荐主备最大可用模式 |
| 一主多备 | 主机+多个备机 Quorum/ Paxos | 抵御实例级故障 | 单机房 | 物理机 | <ul style="list-style-type: none">节点间无网络延迟要求承受数据库内实例级故障 | <ul style="list-style-type: none">RPO=0实例故障RTO<10s无AZ级容灾能力推荐主备同步模式最少2个副本，最多4个副本 |

软硬件规格说明

openGauss支持的CPU和OS范围：

表 5-2 openGauss 软硬件规格

| 交付模式 | CPU | OS |
|------|-----|--------------------------------|
| 开源线下 | X86 | CentOS 7.6、openEuler 20.03 LTS |
| | 鲲鹏 | openEuler 20.03 LTS、麒麟V10 |

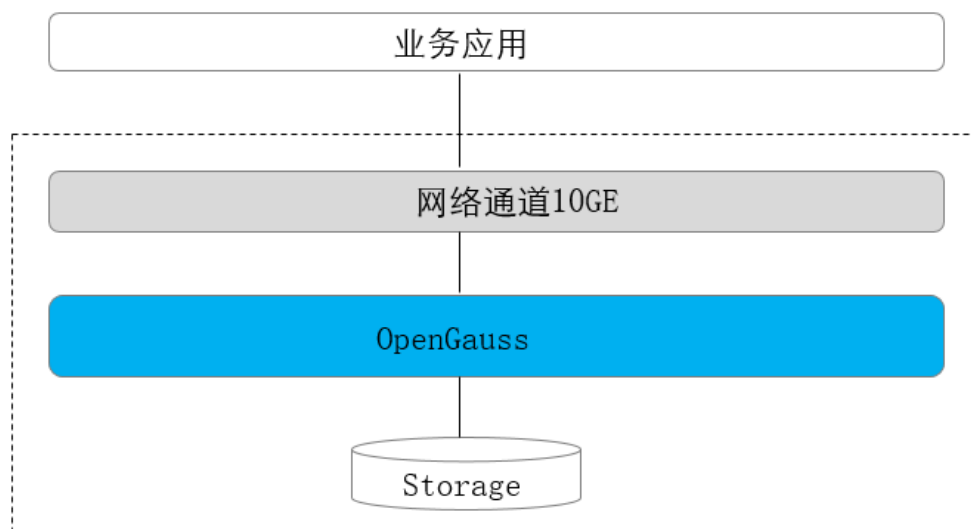
部署方案介绍

整体部署方案可以分为三类：单机部署、一主一备部署、一主多备部署。

- 单机部署

单机部署形态是一种非常特殊的部署形态，这种形态对于可靠性、可用性均无任何保证。由于只有一个数据副本，一旦发生数据损坏、丢失，只能通过物理备份恢复数据。这种部署形态，一般用于数据库体验用户，以及测试环境做语法功能调测等场景。不建议用于商业现网运行。

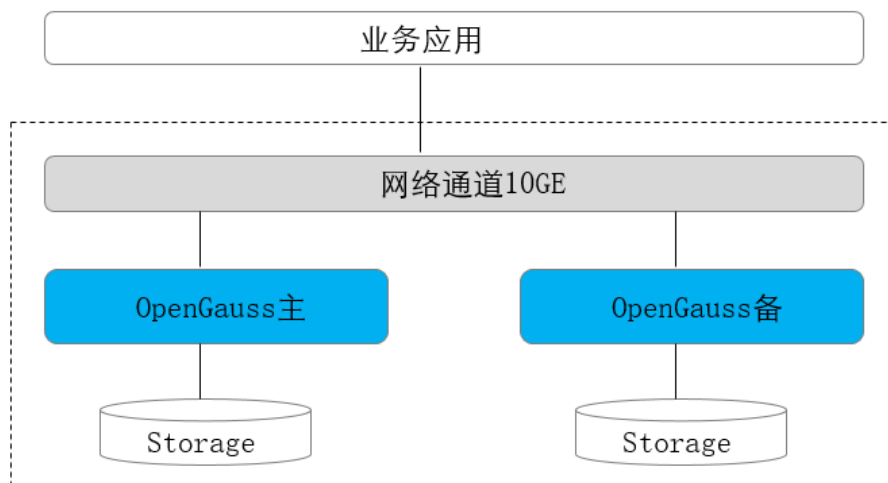
图 5-1 单机部署形态图



- 主备部署

主备模式相当于两个数据副本，主机和备机各一个数据副本，备机接受日志、执行日志回放；

图 5-2 主备部署形态图



- 一主多备部署

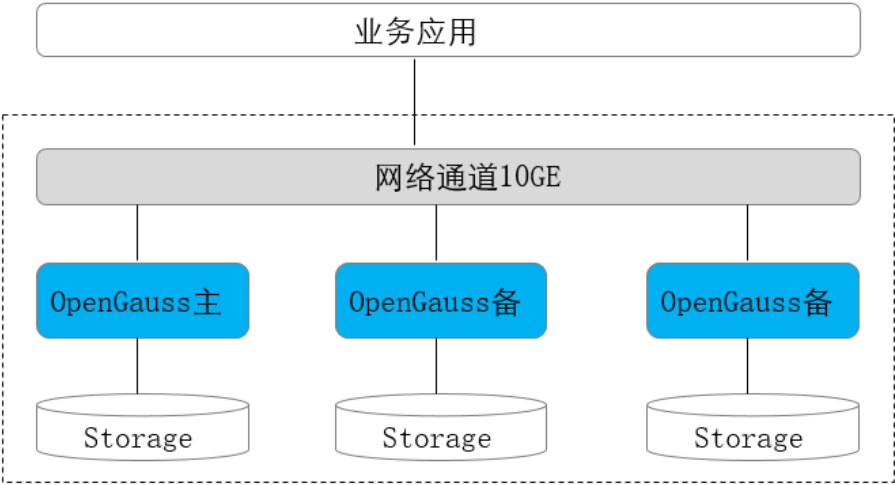
多副本的部署形态，提供了抵御实例级故障的能力。适用于不要求机房级别容灾，但是需要抵御个别硬件故障的应用场景。

一般多副本部署时使用1主2备模式，总共3个副本，3个副本的可靠性为4个9，可以满足大多数应用的可靠性要求。

- 主备间Quorum复制，至少同步到一台备机，保证最大性能。
- 主备任意一个节点故障，不影响业务的进行。
- 数据有三份，任何一个节点故障，系统仍然有双份数据确保继续运行。任何一个备份都可以升主。

- 主备实例之间不可部署在同一台物理机上。

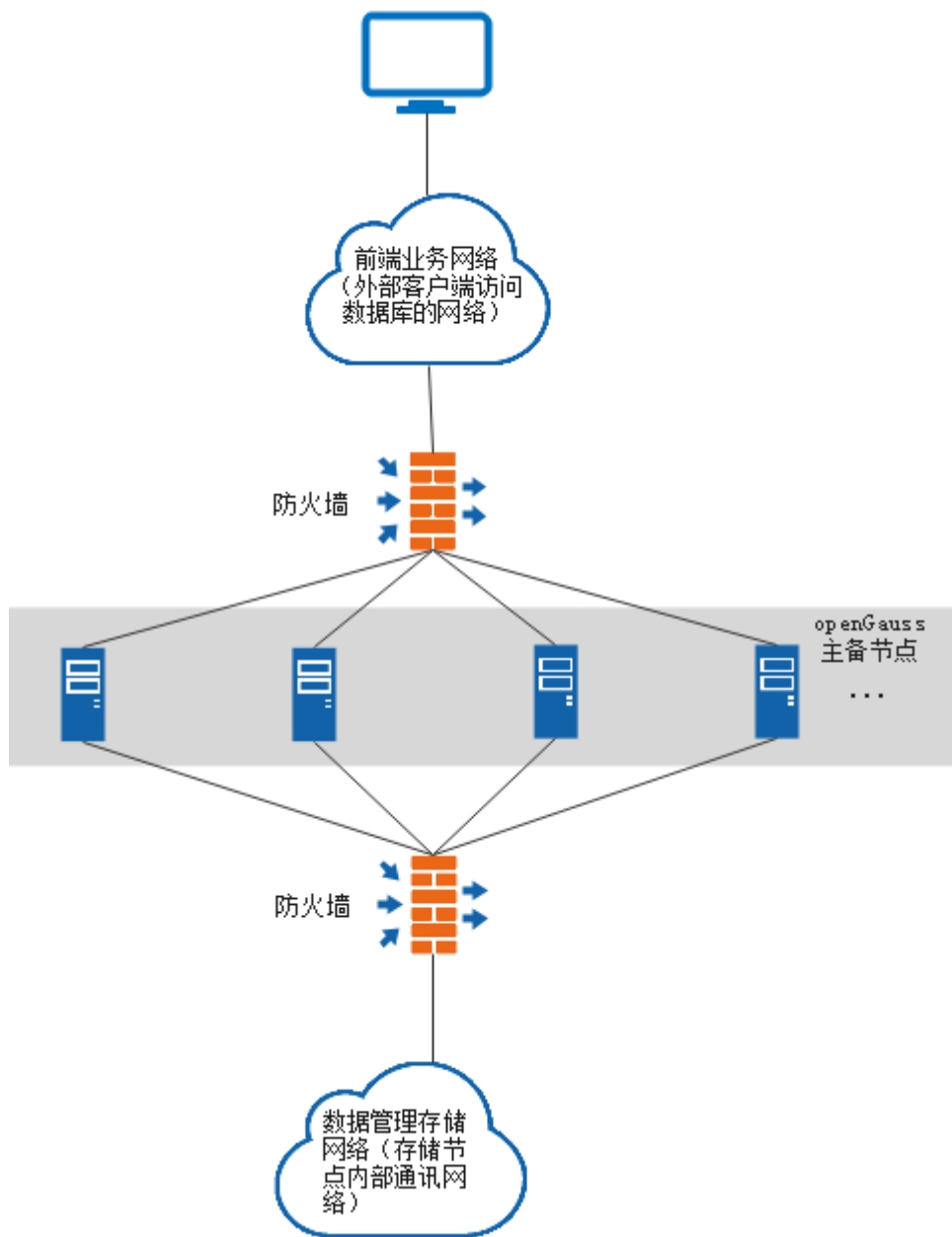
图 5-3 一主多备部署形态图



6 典型组网

为了保证整个应用数据的安全性，建议将openGauss的典型组网划分为两个独立网络：前端业务网络和数据管理存储网络。

图 6-1 典型组网



网络划分说明如表6-1所示。

表 6-1 网络划分

| 类型 | 描述 |
|-----------|---|
| 数据库管理存储网络 | DBA通过此网络调用OM脚本管理和维护openGauss实例。同时，用于openGauss主备通信组网。数据库管理存储网络也是应用执行系统监控的网络。 |
| 前端业务网络 | 外部客户端通过此网络访问openGauss数据库。 |

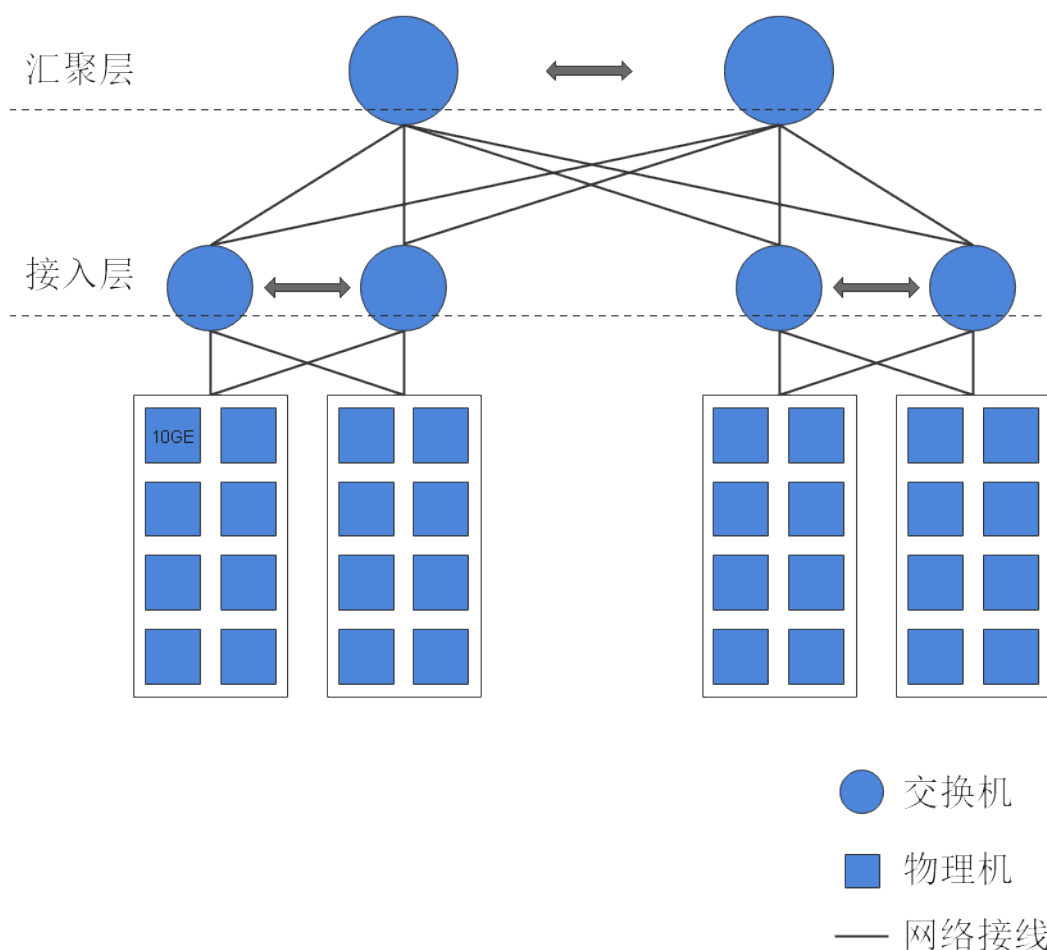
该典型组网有如下优点：

- 业务网络与数据库管理存储网络的隔离，有效保护了后端存储数据的安全。
- 业务网络和数据库管理存储网络的隔离，可以防止攻击者通过互联网试图对数据库服务器进行管理操作，增加了系统安全性。

网络独占性及1:1的带宽收敛比是openGauss数据库网络性能的基本要求。因此，在生产系统中，对图6-1中的后端存储网络，需满足独占性及至少1:1收敛比的要求。例如，图6-2中，其本质是Fattree组网方式。为实现收敛比1:1，交换网络层级每提高一层，带宽增加一倍。图中每根加粗连接线代表80GE带宽，即8台物理机带宽上限之和。接入层每单台交换机下行带宽160GE，上行带宽160GE，收敛比1:1；汇聚层每单台交换机接入带宽320GE。

对于测试系统，上述要求可以适当降低。

图 6-2 数据库管理存储网络组网示例



7 软硬件配置要求

软件配置要求

表 7-1 软件配置要求

| 软件类型 | 配置描述 |
|--------|---|
| 操作系统 | <ul style="list-style-type: none">● ARM：<ul style="list-style-type: none">- openEuler 20.3LTS（推荐采用此操作系统）- 麒麟V10● x86：<ul style="list-style-type: none">- openEuler 20.3LTS- CentOS 7.6 |
| 文件系统 | 在EulerOS操作系统下建议首选使用Ext4格式文件系统。 剩余inode个数 > 15亿（推荐）。 |
| 工具 | bzip2 |
| Python | <ul style="list-style-type: none">● openEuler：支持Python 3.7.X● CentOS：支持Python 3.6.X● 麒麟：支持Python 3.7.X |

硬件配置要求

表 7-2 硬件配置要求

| 项目 | 配置描述 |
|------|---|
| 最小内存 | 功能调试32GB以上。 性能测试和商业部署时，单实例部署，建议128GB以上。 复杂的查询对内存的需求量比较高，在高并发场景下，可能出现内存不足。此时建议使用大内存的机器，或使用负载管理限制系统的并发。 |
| CPU | 功能调试最小1×8 核 2.0GHz。 性能测试和商业部署时，单实例部署，建议1×16核 2.0GHz。 CPU超线程和非超线程两种模式都支持。但是，主备各节点的设置需保持一致。 说明 目前，openGauss仅支持鲲鹏服务器和基于X86_64通用PC服务器的CPU。 |
| 硬盘 | 用于安装实例的硬盘需最少满足如下要求： <ul style="list-style-type: none">至少1GB用于安装实例的应用程序包。每个主机需大约300MB用于元数据存储。预留70%以上的磁盘剩余空间用于数据存储。 建议系统盘配置为Raid1，数据盘配置为Raid5，且规划4组Raid5数据盘用于安装openGauss实例。有关Raid的配置方法在本手册中不做介绍。请参考硬件厂家的手册或互联网上的方法进行配置，其中Disk Cache Policy一项需要设置为Disabled，否则机器异常掉电后有数据丢失的风险。 openGauss支持使用SSD盘作为数据库的主存储设备，但必须是SAS接口的SSD盘，以RAID的方式部署使用。 |
| 网络要求 | 300兆以上以太网。 建议网卡设置为双网卡冗余bond。有关网卡冗余bond的配置方法在本手册中不做介绍。请参考硬件厂商的手册或互联网上的方法进行配置。 主备网络如果配置bond，请保证bond模式一致，不一致的bond配置可能导致主备通信工作异常。 |

8 数据库核心技术

8.1 面向应用开发的基本功能

8.2 高性能

8.3 高可用

8.4 可维护性

8.5 数据库安全

8.6 AI能力

8.1 面向应用开发的基本功能

- 支持标准SQL

openGauss数据库支持标准的SQL。SQL标准是一个国际性的标准，定期会进行更新。SQL标准的定义分成核心特性以及可选特性，绝大部分的数据库都没有100%支撑SQL标准。遗憾的是，SQL特性的构筑成为了数据库厂商吸引用户和提高应用迁移成本的手段，新的SQL特性在厂商之间差异越来越大，目前还没有机构来进行权威的SQL标准度的测试。

openGauss数据库支持SQL:2011大部分的核心特性，同时还支持部分的可选特性，具体的特性列表可以参考《开发者指南》中“SQL参考>SQL语法”章节。

标准SQL的引入为所有的数据库厂商提供统一的SQL界面，减少使用者的学习成本和应用openGauss程序的迁移代价。

- 支持标准开发接口

提供业界标准的ODBC及JDBC接口，保证用户业务快速迁移至openGauss。

目前支持标准的ODBC 3.5及JDBC 4.0接口，其中ODBC支持SUSE、Win32、Win64平台，JDBC无平台差异。

- 支持多存储引擎

openGauss基于统一的事务机制，统一的日志系统，统一的并发控制系统，统一的元信息信息，统一缓存管理提供Table Access Method接口，支持不同的存储引擎。

目前支持Astore和Ustore存储引擎。

- 事务支持

事务支持指的就是系统提供事务的能力，支持全局事务的ACID，保证事务的原子性、一致性、隔离性和持久性。

事务支持及数据一致性保证是绝大多数数据库的基本功能，只有支持了事务，才能满足事务化的应用需求。

- A: Atomicity 原子性

整个事务中的所有操作，要么全部完成，要么全部不完成，不可能停滞在中间某个环节。

- C: Consistency 一致性

事务必须始终保持系统处于一致的状态，不管在任何给定的时间并发事务的数量。

- I: Isolation 隔离性

隔离状态执行事务，使它们好像是系统在给定时间内执行的唯一操作。如果有两个事务，运行在相同的时间内，执行相同的功能，事务的隔离性将确保每一事务在系统中认为只有该事务在使用系统。

- D: Durability 持久性

在事务完成以后，该事务对数据库所作的更改便持久的保存在数据库之中，并不会被回滚。

支持事务的默认隔离级别是读已提交。保证不会读到脏数据。

事务分为单语句事务和事务块，相关基础接口：

- Start transaction: 事务开启

- Commit: 事务提交

- Rollback: 事务回滚

另有Set transaction可设置隔离级别、读写模式或可推迟模式。详细语法参见《开发者指南》。

- 函数及存储过程支持

函数是数据库中的一种重要对象，主要功能将用户特定功能的SQL语句集进行封装，并方便调用。

存储过程是SQL、PL/SQL的组合。存储过程可以使执行商业规则的代码从应用程序中移动到数据库。从而，代码存储一次能够被多个程序使用。

- 允许客户模块化程序设计，对SQL语句集进行封装，调用方便。
- 存储过程会进行编译缓存，可以提升用户执行SQL语句集的速度。
- 系统管理员通过对执行某一存储过程的权限进行限制，能够实现对相应数据访问权限的限制，避免了非授权用户对数据的访问，保证了数据的安全。
- 为了处理SQL语句，存储过程进程分配一段内存区域来保存上下文联系。游标是指向上下文区域的句柄或指针。借助游标，存储过程可以控制上下文区域的变化。
- 支持6种异常信息级别方便客户对存储过程进行调试。支持存储过程调试，存储过程调试是一种调试手段，可以在存储过程开发中，一步一步跟踪存储过程执行的流程，根据变量的值，找到错误的原因或者程序的bug，提高问题定位效率。支持设置断点和单步调试。

openGauss支持SQL标准中的函数及存储过程，增强了存储过程的易用性。存储过程具体的使用方式可以参考《开发者指南》。

- PG接口兼容

兼容PSQL客户端，兼容PostgreSQL标准接口。

- 支持SQL hint

支持SQL hint影响执行计划生成、提升SQL查询性能。

Plan Hint为用户提供了直接影响执行计划生成的手段，用户可以通过指定join顺序，join、stream、scan方法，指定结果行数，等多个手段来进行执行计划的调优，以提升查询的性能。

- Copy接口支持容错机制

openGauss提供用户封装好的Copy错误表创建函数，并允许用户在使用Copy From指令时指定容错选项，使得Copy From语句在执行过程中部分解析、数据格式、字符集等相关的报错不会报错中断事务、而是被记录至错误表中，使得在Copy From的目标文件即使有少量数据错误也可以完成入库操作。用户随后可以在错误表中对相关的错误进行定位以及进一步排查。

8.2 高性能

CBO 优化器

openGauss优化器是典型的基于代价的优化（Cost-Based Optimization，简称CBO）。在这种优化器模型下，数据库根据表的元组数、字段宽度、NULL记录比率、distinct值、MCV值、HB值等表的特征值，以及一定的代价计算模型，计算出每一个执行步骤的不同执行方式的输出元组数和执行代价（cost），进而选出整体执行代价最小/首元组返回代价最小的执行方式进行执行。

CBO优化器能够在众多计划中依据代价选出最高效的执行计划，最大限度的满足客户业务要求。

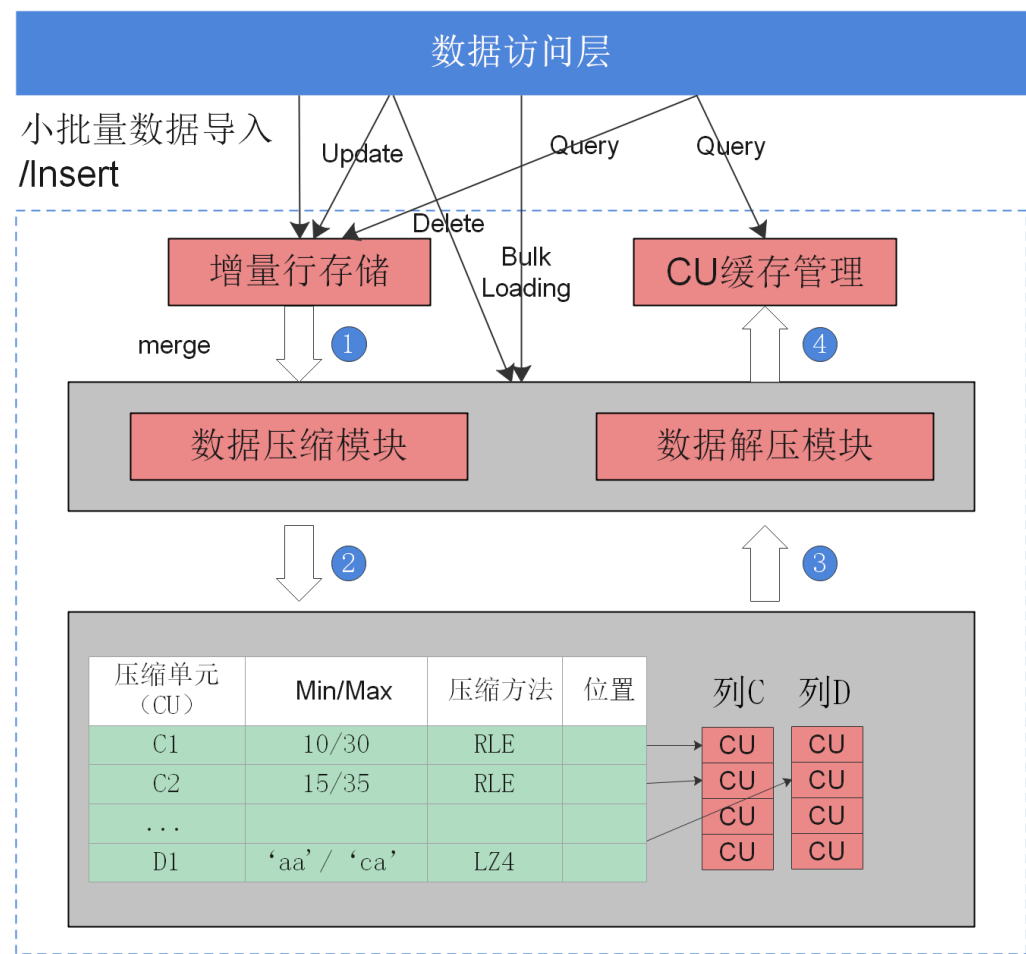
行列混合存储

openGauss支持行存储和列存储两种存储模型，用户可以根据应用场景，建表的时候选择行存储还是列存储表。

一般情况下，如果表的字段比较多（大宽表），查询中涉及到的列不很多的情况下，适合列存储。如果表的字段个数比较少，查询大部分字段，那么选择行存储比较好。

列存储方式如[图8-1](#)所示。

图 8-1 列存储示意图



在大宽表、数据量比较大的场景中，查询经常关注某些列，行存储引擎查询性能比较差。例如气象局的场景，单表有200~800个列，查询经常访问10个列，在类似这样的场景下，向量化执行技术和列存储引擎可以极大的提升性能和减少存储空间。

行存表和列存表各有优劣，建议根据实际情况选择。

- **行存表**
默认创建表的类型。数据按行进行存储，即一行数据紧挨着存储。行存表支持完整的增删改查。适用于对数据需要经常更新的场景。
- **列存表**
数据按列进行存储，即一列所有数据紧挨着存储。单列查询IO小，比行存表占用更少的存储空间。适合数据批量插入、更新较少和以查询为主统计分析类的场景。列存表不适合点查询，insert插入单条记录性能差。

行存表和列存表的选择原则如下：

- **更新频繁程度**
数据如果频繁更新，选择行存表。
- **插入频繁程度**
频繁的少量插入，选择行存表。一次插入大批量数据，选择列存表。

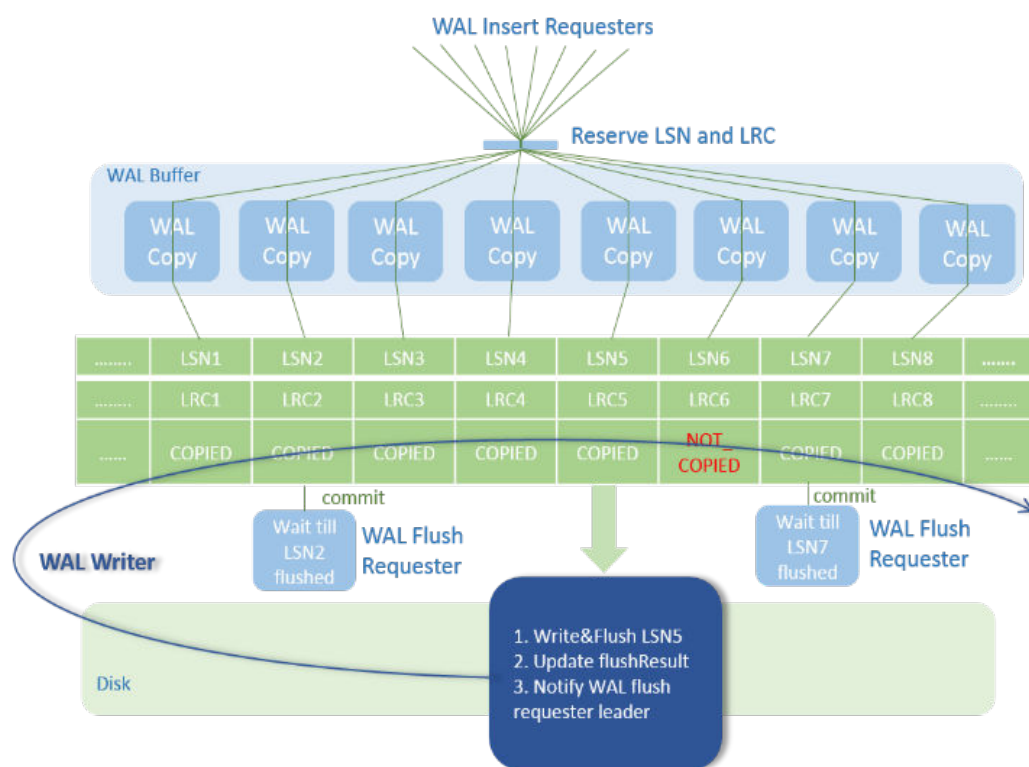
- 表的列数
表的列数很多，选择列存表。
- 查询的列数
如果每次查询时，只涉及了表的少数（<50%总列数）几个列，选择列存表。
- 压缩率
列存表比行存表压缩率高。但高压缩率会消耗更多的CPU资源。

In-place update 存储

新增的In-place update存储引擎很好的解决了Append update存储引擎空间膨胀，元组较大的劣势，高效回滚段的设计是In-place update存储引擎的基础。

Xlog 无锁刷新与并行 Page 回放

图 8-2 Xlog lock less Design



本特性对WalInsertLock进行优化，利用LSN（Log Sequence Number）及LRC（Log Record Count）记录了每个backend的拷贝进度，取消WalInsertLock机制。在backend将日志拷贝至WalBuffer时，不用对WalInsertLock进行争抢，可直接进行日志拷贝操作。并利用专用的WalWriter写日志线程，不需要backend线程自身来保证xlog的Flush。通过以上优化，取消WalInsertLock争抢及WalWriter专用磁盘写入线程，在保持原有XLog功能不变的基础上，可进一步提升系统性能。针对Ustore Inplace update WAL log写入，Ustore DML operation并行回放分发进行优化。通过利用Prefix和suffix来减少update WAL log的写入。通过把回放线程分多个类型来解决Ustore DML WAL大多都是多页面回放问题。同时把Ustore的数据页面回放分为按照blkno去分发里更好的提高并行回放的并行程度。

自适应压缩

当前主流数据库通常都会采用数据压缩技术。数据类型不同，适用于它的压缩算法不同。对于相同类型的数据，其数据特征不同，采用不同的压缩算法达到的效果也不相同。自适应压缩正是从数据类型和数据特征出发，采用相应的压缩算法，实现了良好的压缩比、快速的入库性能以及良好的查询性能。

数据入库和频繁的海量数据查询是用户的主要应用场景。在数据入库场景中，自适应压缩可以大幅度地减少数据量，成倍提高IO操作效率，将数据簇集存储，从而获得快速的入库性能。当用户进行数据查询时，少量的IO操作和快速的数据解压可以加快数据获取的速率，从而在更短的时间内得到查询结果。

目前，数据库已实现了RLE、DELTA、BYTEPACK/BITPACK、LZ4、ZLIB、LOCAL DICTIONARY等多种压缩算法。数据库支持的数据类型与压缩算法的映射关系如下表所示。

| - | RLE | DELTA | BITPACK/ BYTEPACK | LZ4 | ZLIB | LOCAL DICTIONARY |
|---|-----|-------|----------------------|-----|------|---------------------|
| Smallint/int/bigint/Oid Decimal/real/double Money/time/date/ timestamp | ✓ | ✓ | ✓ | ✓ | ✓ | - |
| Tinterval/interval/Time with time zone/ | - | - | - | - | ✓ | - |
| Numeric/char/varchar/ text/nvarchar2 以及其他支持数据类型 | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |

例如，支持类手机号字符串的大整数压缩、支持numeric类型的大整数压缩、支持对压缩算法进行不同压缩水平的调整。

分区

在openGauss系统中，数据分区是在一个实例内部按照用户指定的策略对数据做进一步的水平分表，将表按照指定范围划分为多个数据互不重叠的部分。

对于大多数用户使用场景，分区表和普通表相比具有以下优点：

- 改善查询性能：对分区对象的查询可以仅搜索自己关心的分区，提高检索效率。
- 增强可用性：如果分区表的某个分区出现故障，表在其他分区的数据仍然可用。
- 方便维护：如果分区表的某个分区出现故障，需要修复数据，只修复该分区即可。
- 均衡I/O：可以把不同的分区映射到不同的磁盘以平衡I/O，改善整个系统性能。

目前openGauss数据库支持的分区表为范围分区表、列表分区表、哈希分区表。

- 范围分区表：将数据基于范围映射到每一个分区，这个范围是由创建分区表时指定的分区键决定的。这种分区方式是最为常用的。

范围分区功能，即根据表的一列或者多列，将要插入表的记录分为若干个范围（这些范围在不同的分区里没有重叠），然后为每个范围创建一个分区，用来存储相应的数据。

- 列表分区表：将数据基于各个分区内包含的键值映射到每一个分区，分区包含的键值在创建分区时指定。

列表分区功能，即根据表的一列，将要插入表的记录中出现的键值分为若干个列表（这些列表在不同的分区里没有重叠），然后为每个列表创建一个分区，用来存储相应的数据。

- 哈希分区表：将数据通过哈希映射到每一个分区，每一个分区中存储了具有相同哈希值的记录。

哈希分区功能，即根据表的一列，通过内部哈希算法将要插入表的记录划分到对应的分区中。

用户在CREATE TABLE时增加PARTITION参数，即表示针对此表应用数据分区功能。

用户可以在实际使用中根据需要调整建表时的分区键，使每次查询结果尽可能存储在相同或者最少的分区内（称为“分区剪枝”），通过获取连续I/O大幅度提升查询性能。

实际业务中，时间经常被作为查询对象的过滤条件。因此，用户可考虑选择时间列为分区键，键值范围可根据总数据量、一次查询数据量调整。

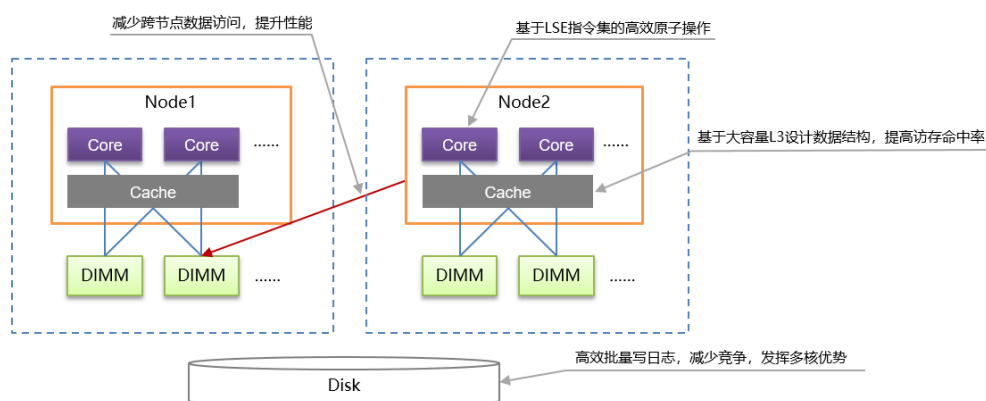
SQL by pass

在典型的OLTP场景中，简单查询占了很大一部分比例。这种查询的特征是只涉及单表和简单表达式的查询，因此为了加速这类查询，提出了SQL-BY-PASS框架，在parse层对这类查询做简单的模式判别后，进入到特殊的执行路径里，跳过经典的执行器执行框架，包括算子的初始化与执行、表达式与投影等经典框架，直接重写一套简洁的执行路径，并且直接调用存储接口，这样可以大大加速简单查询的执行速度。

鲲鹏 NUMA 架构优化

鲲鹏NUMA架构优化图如下。

图 8-3 鲲鹏 NUMA 架构优化图



1. openGauss根据鲲鹏处理器的多核NUMA架构特点，进行针对性一系列NUMA架构相关优化，一方面尽量减少跨核内存访问的时延问题，另一方面充分发挥鲲鹏多核算力优势，所提供的关键技术包括重做日志批插，热点数据NUMA分布，CLog分区等，大幅提升TP系统的处理性能。

2. openGauss基于鲲鹏芯片所使用的ARMv8.1架构，利用LSE扩展指令集实现高效的原子操作，有效提升CPU利用率，从而提升多线程间同步性能，XLog写入性能等。
3. openGauss基于鲲鹏芯片提供的更宽的L3缓存cacheline，针对热点数据访问进行优化，有效提高缓存访问命中率，降低Cache缓存一致性维护开销，大幅提升系统整体的数据访问性能。

线程池高并发

在OLTP领域中，数据库需要处理大量的客户端连接。因此，高并发场景的处理能力是数据库的重要能力之一。

对于外部连接最简单的处理模式是per-thread-per-connection模式，即来一个用户连接产生一个线程。这个模式好处是架构上处理简单，但是高并发下，由于线程太多，线程切换和数据库轻量级锁区域的冲突过大导致性能急剧下降，使得系统性能（吞吐量）严重下降，无法满足用户性能的SLA。

因此，需要通过线程资源池化复用的技术来解决该问题。线程池技术的整体设计思想是线程资源池化、并且在不同连接之间复用。系统在启动之后会根据当前核数或者用户配置启动固定一批数量的工作线程，一个工作线程会服务一到多个连接session，这样把session和thread进行了解耦。因为工作线程数是固定的，因此在高并发下不会导致线程的频繁切换，而由数据库层来进行session的调度管理。

并行查询

openGauss的SMP并行技术是一种利用计算机多核CPU架构来实现多线程并行计算，以充分利用CPU资源来提高查询性能的技术。在复杂查询场景中，单个查询的执行较长，系统并发度低，通过SMP并行执行技术实现算子级的并行，能够有效减少查询执行时间，提升查询性能及资源利用率。SMP并行技术的整体实现思想是对于能够并行的查询算子，将数据分片，启动若干个工作线程分别计算，最后将结果汇总，返回前端。SMP并行执行增加数据交互算子（Stream），实现多个工作线程之间的数据交互，确保查询的正确性，完成整体的查询。

动态编译执行

openGauss借助LLVM提供的库函数，依据查询执行计划树，将原本在执行器阶段才会确定查询实际执行路径的过程提前到执行初始化阶段，从而规避原本查询执行时候伴随的函数调用、逻辑条件分支判断以及大量的数据读取等问题，以达到提升查询性能的目的。

8.3 高可用

主备机

为了保证故障的可恢复，需要将数据写多份，设置主备多个副本，通过日志进行数据同步，可以实现节点故障、停止后重启等情况下，openGauss能够保证故障之前的数据无丢失，满足ACID特性。主备环境可以支持主备和一主多备两种模式。主备模式下，备机需要重做日志，可以升主。在一主多备模式下，所有的备机都需要重做日志，都可以升主。主备主要用于一般可靠性的OLTP系统能够节省一定的存储资源。而一主多备提供更高的容灾能力，适合于更高可靠性事务处理的OLTP系统。

主备之间可以通过switchover进行角色切换，主机故障后可以通过failover对备机进行升主。

为保证failover的时间可控，可以开启日志流控功能，控制日志发往备机的速率，保证备机堆积的日志会在小于流控配置的目标时间内回放完。开启流控后因为发送给备机日志的速率被动态调整，从而整体的事务的性能会有相应的降低。

初始化安装或者备份恢复等场景中，需要根据主机重建备机的数据，此时需要build功能，将主机的数据和WAL日志发送到备机。主机故障后重新以备机的角色加入时，也需要build功能将其数据和日志与新主机拉齐。Build包含全量build和增量build，全量build要全部依赖主机数据进行重建，拷贝的数据量比较大，耗时比较长，而增量build只拷贝差异文件，拷贝的数据量比较小，耗时比较短。一般情况下，优先选择增量build来进行故障恢复，如果增量build失败，再继续执行全量build，直至故障恢复。

openGauss除了流复制主备双机外，还支持逻辑复制。在逻辑复制中把主库称为源端数据库，备库称为目标端数据库，源端数据库根据预先指定好的逻辑解析规则对WAL文件进行解析，把DML操作解析成一定的逻辑变化信息（标准SQL语句），源端数据库把标准SQL语句发给目标端数据库，目标端数据库收到后进行应用，从而实现数据同步。逻辑复制只有DML操作。逻辑复制可以实现跨版本复制，异构数据库复制，双写数据库复制，表级别复制。

逻辑备份

openGauss提供逻辑备份能力，可以将用户表的数据以通用的text或者csv格式备份到本地磁盘文件，并在同构/异构数据库中恢复该用户表的数据。

物理备份

openGauss提供物理备份能力，可以将整个实例的数据以数据库内部格式备份到本地磁盘文件中，并在同构数据库中恢复整个实例的数据。

物理备份主要分为全量备份和增量备份，区别如下：全量备份包含备份时刻点上数据库的全量数据，耗时时间长（和数据库数据总量成正比），自身即可恢复出完整的数据库；增量备份只包含从指定时刻点之后的增量修改数据，耗时间短（和增量数据成正比，和数据总量无关），但是必须要和全量备份数据一起才能恢复出完整的数据库。openGauss支持全量备份和增量备份。

闪回恢复

利用回收站的闪回恢复删除的表。数据库的回收站功能类似于windows系统的回收站，将删除的表信息保存到回收站中。利用MVCC机制闪回恢复到指定时间点或者SCN点。

极致 RTO

极致RTO开关开启后，xlog日志回放建立多级流水线，提高并发度，提升日志回放速度。

当业务压力过大时，备机的回放速度跟不上主机的速度。在系统长时间的运行后，备机上会出现日志累积。当主机故障后，数据恢复需要很长时间，数据库不可用，严重影响系统可用性。开启极致RTO（Recovery Time Object，恢复时间目标），减少了主机故障后数据的恢复时间，提高了可用性。

逻辑复制

openGauss提供逻辑解码功能，将物理日志反解析为逻辑日志。通过DRS等逻辑复制工具将逻辑日志转化为SQL语句，到对端数据库回放，达到异构数据库同步数据的功能。目前支持openGauss数据库与MySQL数据库、Oracle数据库之间的单向、双向逻辑复制。DN通过物理日志反解析为逻辑日志，DRS等逻辑复制工具从DN抽取逻辑日志转换为SQL语句，到对端数据库（MySQL）回放。逻辑复制工具同时从MySQL数据库抽取逻辑日志，反解析为SQL语句之后回放至openGauss，达到异构数据库同步数据的目的。

恢复到指定时间点（PITR）

时间点恢复（Point In Time Recovery）基本原理是通过基础热备 + WAL预写日志 + WAL归档日志进行备份恢复。重放WAL记录的时候可以在任意点停止重放，这样就有在一个在任意时间的数据库一致的快照。即可以把数据库恢复到自开始备份以来的任意时刻的状态。openGauss在恢复时可以指定恢复的停止点位置为TID，时间和LSN。

基于 Paxos 协议的高可用（DCF）

DCF开关开启后，DN支持基于Paxos协议的复制与仲裁，实现高可用和容灾的能力。DN支持自选主及日志复制，复制过程支持压缩和流控能力，防止带宽占用过高。提供基于Paxos多种角色的节点类型，并能够进行调整。

8.4 可维护性

支持 WDR 诊断报告

WDR（Workload Diagnosis Report）基于两次不同时间点系统的性能快照数据，生成这两个时间点之间的性能表现报表，用于诊断数据库内核的性能故障。

WDR主要依赖两个组件：

- SNAPSHOT性能快照：性能快照可以配置成按一定时间间隔从内核采集一定量的性能数据，持久化在用户表空间。任何一个SNAPSHOT可以作为一个性能基线，其他SNAPSHOT与之比较的结果，可以分析出与基线的性能表现。
- WDR Reporter：报表生成工具基于两个SNAPSHOT，分析系统总体性能表现，并能计算出更多项具体的性能指标在这两个时间段之间的变化量，生成SUMMARY和DETAIL两个不同级别的性能数据。如表8-1、表8-2所示。

表 8-1 SUMMARY 级别诊断报告

| 诊断类别 | 描述 |
|---------------|---|
| Database Stat | 主要用于评估当前数据库上的负载，IO状况，负载和IO是衡量TP系统最最重要的特性 包含当前连接到该数据库的session，提交、回滚的事务数，读取的磁盘块的数量，高速缓存中已经发现的磁盘块的次数，通过数据库查询返回、抓取、插入、更新、删除的行数，冲突、死锁发生的次数，临时文件的使用量，IO读写时间等 |

| 诊断类别 | 描述 |
|---------------------------------|--|
| Load Profile | 从时间，IO，事务，SQL几个维度评估当前系统负载的表现 包含作业运行elapsed time、CPU time，事务日质量，逻辑和物理读的量，读写IO次数、大小，登入登出次数，SQL、事务执行量，SQL P85、P90响应时间等 |
| Instance Efficiency Percentages | 用于评估当前系统的缓存的效率。 主要包含数据库缓存命中率 |
| Events | 用于评估当前系统内核关键资源，关键事件的性能 主要包含数据库内核关键时间的发生次数，时间的等待时间 |
| Wait Classes | 用于评估当前系统关键事件类型的性能 主要包含数据内核在主要的等待事件种类上的发布： STATUS、LWLOCK_EVENT、LOCK_EVENT、IO_EVENT |
| CPU | 主要包含CPU在用户态、内核态、Wait IO、空闲状态下的时间发布 |
| IO Profile | 主要包含数据库Database IO次数、Database IO数据量、Redo IO次数、Redo IO量 |
| Memory Statistics | 包含最大进程内存、进程已经使用内存、最大共享内存、已经使用共享内存大小等 |

表 8-2 DETAIL 级别诊断报告

| 诊断类别 | 描述 |
|----------------|--|
| Time Model | 主要用于评估当前系统在时间维度的性能表现 包含系统在各个阶段上消耗的时间：内核时间、CPU时间、执行时间、解析时间、编译时间、查询重写时间、计划生成时间、网络时间、IO时间 |
| SQL Statistics | 主要用于SQL语句性能问题的诊断 包含归一化的SQL的性能指标在多个维度上的排序：Elapsed Time、CPU Time、Rows Returned、Tuples Reads、Executions、Physical Reads、Logical Reads。这些指标的种类包括：执行时间，执行次数、行活动、Cache IO等 |
| Wait Events | 主要用于系统关键资源，关键时间的详细性能诊断 包含所有关键事件在一段时间内的表现，主要是事件发生的次数，消耗的时间 |
| Cache IO Stats | 用于诊断用户表和索引的性能 包含所有用户表、索引上的文件读写，缓存命中 |
| Utility status | 用于诊断后端作业性能 包含页面操作，复制等后端操作的性能 |

| 诊断类别 | 描述 |
|------------------------|---|
| Object stats | 用于诊断数据库对象的性能 包含用户表、索引上的表、索引扫描活动，insert、update、delete活动，有效行数量，表维护操作的状态等 |
| Configuration settings | 用于判断配置是否有变更 包含当前所有配置参数的快照 |

应用价值：

- WDR报表是长期性能问题最主要的诊断手段。基于SNAPSHOT的性能基线，从多维度做性能分析，能帮助DBA 掌握系统负载繁忙程度，各个组件的性能表现，性能瓶颈。
- SNAPSHOT也是后续性能问题自诊断和自优化建议的重要数据来源。

慢 SQL 诊断

慢SQL能根据用户提供的执行时间阈值，记录所有超过阈值的执行完毕的作业信息。

历史慢SQL提供表和函数两种维度的查询接口，用户从接口中能查询到作业的执行计划，开始、结束执行时间，执行查询的语句，行活动，内核时间，CPU时间，执行时间，解析时间，编译时间，查询重写时间，计划生成时间，网络时间，IO时间，网络开销，锁开销等。所有信息都是脱敏的。

慢SQL提供给用户对于慢SQL诊断所需的详细信息，用户无需通过复现就能离线诊断特定慢SQL的性能问题。表和函数接口方便用户统计慢SQL指标，对接第三方平台。

8.5 数据库安全

访问控制

管理用户对数据库的访问控制权限，涵盖数据库系统权限和对象权限。

支持基于角色的访问控制机制，将角色和权限关联起来，通过将权限赋予给对应的角色，再将角色授予给用户，可实现用户访问控制权限管理。其中登录访问控制通过用户标识和认证技术来共同实现，而对象访问控制则基于用户在对象上的权限，通过对对象权限检查实现对象访问控制。用户为相关的数据库用户分配完成任务所需要的最小权限从而将数据库使用风险降到最低。

支持三权分立权限访问控制模型，数据库角色可分为系统管理员、安全管理员和审计管理员。其中安全管理员负责创建和管理用户，系统管理员负责授予和撤销用户权限，审计管理员负责审计所有用户的行为。

默认情况下，使用基于角色的访问控制模型。客户可通过设置参数来选择是否开启三权分立控制模型。

控制权和访问权分离

针对系统管理员用户，实现表对象的控制权和访问权分离，提高普通用户数据安全性，限制管理员对象访问权限。

该特性适用于如下场景，即对于有多个业务部门的企业，各部门间使用不同的数据库用户进行业务操作，同时存在同级别的数据库维护部门使用数据库管理员进行运维操作，业务部门希望在未经授权的情况下，管理员用户只能对各部门的数据进行控制操作（DROP、ALTER、TRUNCATE），但是不能进行访问操作（INSERT、DELETE、UPDATE、SELECT、COPY）。即针对管理员用户，表对象的控制权和访问权分离，提高用户数据的安全性。

系统管理员可以在创建用户时指定INDEPENDENT属性，表示该用户为私有用户。针对该用户的对象，数据库管理员（包含初始用户和其他管理员用户）在未经其授权前，只能进行控制操作（DROP、ALTER、TRUNCATE），无权进行INSERT、DELETE、SELECT、UPDATE、COPY、GRANT、REVOKE、ALTER OWNER操作。

数据库内置角色权限管理

openGauss提供了一组默认角色，以gs_role_开头命名。它们提供对特定的、通常需要高权限的操作的访问，可以将这些角色GRANT给数据库内的其他用户或角色，让这些用户能够使用特定的功能。在授予这些角色时应当非常小心，以确保它们被用在需要的地方。[表3](#)描述了内置角色允许的权限范围：

表 8-3 内置角色权限说明

| 角色 | 权限描述 |
|--------------------------|---|
| gs_role_copy_files | 具有执行copy ... to/from filename 的权限，但需要先打开GUC参数enable_copy_server_files。 |
| gs_role_signal_backend | 具有调用函数pg_cancel_backend、pg_terminate_backend和pg_terminate_session来取消或终止其他会话的权限，但不能操作属于初始用户和PERSISTENCE用户的会话。 |
| gs_role_tablespace | 具有创建表空间（tablespace）的权限。 |
| gs_role_replication | 具有调用逻辑复制相关函数的权限，例如kill_snapshot、pg_create_logical_replication_slot、pg_create_physical_replication_slot、pg_drop_replication_slot、pg_replication_slot_advance、pg_create_physical_replication_slot_external、pg_logical_slot_get_changes、pg_logical_slot_peek_changes、pg_logical_slot_get_binary_changes、pg_logical_slot_peek_binary_changes。 |
| gs_role_account_lock | 具有加解锁用户的权限，但不能加解锁初始用户和PERSISTENCE用户。 |
| gs_role_directory_create | 具有执行创建directory对象的权限，但需要先打开GUC参数enable_access_server_directory。 |
| gs_role_directory_drop | 具有执行删除directory对象的权限，但需要先打开GUC参数enable_access_server_directory。 |

数据库加密认证

采用基于RFC5802机制的口令加密认证方法。

加密认证过程中采用单向Hash不可逆加密算法PBKDF2，可有效防止彩虹攻击。

创建用户所设置的口令被加密存储在系统表中。整个认证过程中口令加密存储和传输，通过计算相应的hash值并与服务端存储的值比较来进行正确性校验。

统一加密认证过程中的消息处理流程，可有效防止攻击者通过抓取报文猜解用户名或者口令的正确性。

数据库审计

审计日志记录用户对数据库的启停、连接、DDL、DML、DCL等操作。审计日志机制主要增强数据库系统对非法操作的追溯及举证能力。

用户可以通过参数配置对哪些语句或操作记录审计日志。

审计日志记录事件的时间、类型、执行结果、用户名、数据库、连接信息、数据库对象、数据库实例名称和端口号以及详细信息。支持按起止时间段查询审计日志，并根据记录的字段进行筛选。

数据库安全管理员可以利用这些日志信息，重现导致数据库现状的一系列事件，找出非法操作的用户、时间和内容等。

网络通信安全特性

支持通过SSL加密客户端和服务端之间的通信数据，保证客户的客户端与服务端通信安全。

采用TLS 1.2协议标准，并使用安全强度较高的加密算法套件，支持的加密算法套件如表8-4所示。

表 8-4 加密算法套件

| OpenSSL套件名 | IANA套件名 | 安全程度 |
|-------------------------------|---|------|
| ECDHE-RSA-AES128-GCM-SHA256 | TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 | HIGH |
| ECDHE-RSA-AES256-GCM-SHA384 | TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 | HIGH |
| ECDHE-ECDSA-AES128-GCM-SHA256 | TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256 | HIGH |
| ECDHE-ECDSA-AES256-GCM-SHA384 | TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 | HIGH |

行级访问控制

行级访问控制特性将数据库访问粒度控制到数据表行级别，使数据库达到行级访问控制的能力。不同用户执行相同的SQL查询操作，按照行访问控制策略，读取到的结果可能是不同的。

用户可以在数据表创建行访问控制（Row Level Security）策略，该策略是指针对特定数据库用户、特定SQL操作生效的表达式。当数据库用户对数据表访问时，若SQL满足数据表特定的Row Level Security策略，在查询优化阶段将满足条件的表达式，按照属

性（PERMISSIVE | RESTRICTIVE）类型，通过AND或OR方式拼接，应用到执行计划上。

行级访问控制的目的是控制表中行级数据可见性，通过在数据表上预定义Filter，在查询优化阶段将满足条件的表达式应用到执行计划上，影响最终的执行结果。当前行级访问控制支持的SQL语句包括SELECT，UPDATE，DELETE。

资源标签

资源标签（Resource Label）特性通过将数据库资源按照用户自定义的方式划分，实现资源分类管理的目的。管理员可以通过配置资源标签统一地为一组数据库资源进行安全策略的配置如审计或数据脱敏。

资源标签能够将数据库资源按照“特征”、“作用场景”等分组归类，对指定资源标签的管理操作即对标签范围下所有的数据库资源的管理操作，能够大大降低策略配置的复杂度和信息冗余度，提高管理效率。

当前资源标签所支持的数据库资源类型包括：SCHEMA、TABLE、COLUMN、VIEW、FUNCTION。

动态数据脱敏

为了在一定程度上限制非授权用户对隐私数据的窥探，可以利用动态数据脱敏（Dynamic Data Masking）特性保护用户隐私数据。在非授权用户访问配置了动态数据脱敏策略的数据时，数据库将返回脱敏后的数据而达到对隐私数据保护的目的。

管理员可以在数据列上创建动态数据脱敏策略，该策略指出针对特定用户场景应采取何种数据脱敏方式。在开启动态数据脱敏功能后，当用户访问敏感列数据时，系统将用户身份信息例如：访问IP、客户端工具、用户名来匹配相应的脱敏策略，在匹配成功后将根据脱敏策略对访问列的查询结果实施数据脱敏。

动态数据脱敏的目的是在不改变源数据的前提下，通过在脱敏策略上配置针对的用户场景（FILTER）、指定的敏感列标签（LABEL）和对应的脱敏方式（MASKING FUNCTION）来灵活地进行隐私数据保护。

统一审计

统一审计（Unified Auditing）利用策略和条件在数据库内部有选择地进行审计，管理员可以对数据库资源或资源标签统一地配置审计策略，从而达到简化管理、针对性地生成审计日志、减少审计日志冗余、提高管理效率的目的。

管理员可以定制化的为操作行为或数据库资源配置审计策略，该策略针对特定的用户场景、用户行为或数据库资源进行审计。在开启了统一审计功能后，当用户访问数据库时，系统将根据用户身份信息如：访问IP、客户端工具、用户名来匹配相应的统一审计策略，之后根据策略信息对用户行为按照访问资源（LABEL）和用户操作类型（DML | DDL）进行统一审计。

统一审计的目的是将现有的传统审计行为转变为针对性的跟踪审计行为，将目标之外的行为排除在审计之外，从而简化了管理，提高了数据库生成审计数据的安全性。

用户口令强度校验机制

为了加固客户账户和数据的安全，禁止设置过低强度的口令，当初始化数据库、创建用户、修改用户时需要指定密码。密码必须满足强度校验，否则会提示用户重新输入密码。

账户密码复杂度对用户密码大小写字母、数字、特殊字符的最少个数，最大最小长度，不能和用户名、用户名倒写相同，不能是弱口令等进行了限制，从而增强了用户账户的安全性。

其中弱口令指的是强度较低，容易被破解的密码，对于不同的用户或群体，弱口令的定义可能会有所区别，用户需要自己添加定制化的弱口令。

用户口令强度校验机制是否开启由参数password_policy控制，当该参数设置为1时表示采用密码复杂度校验，默认值为1。

数据加密存储

提供对插入数据进行加密存储。

为用户提供数据加解密接口，针对用户识别的敏感信息列使用加密函数，使数据加密后再存储在表内。

当用户需要对整张表进行加密存储处理时，则需要为每一列单独书写加密函数，不同的属性列可使用不同的入参。

当具有对应权限的用户需要查看具体数据时，可通过解密函数接口对相应的属性列进行解密处理。

账本数据库

为了防止数据库运维人员可能存在监守自盗、篡改数据库并擦除痕迹等行为，可以利用账本数据库特性来进行更加全面的审计和追溯历史。在修改防篡改用户表数据时，数据库会将修改行为记录至只可追加数据的历史表中，从而达到记录操作历史和操作溯源的能力。

账本数据库通过生成数据hash摘要的方式来保存和验证历史操作。账本指的是用户历史表和全局区块表，对于表级别数据修改操作，系统将操作信息以及hash摘要记录至全局区块表中，同时每个防篡改用户表对应一个用户历史表来记录行级数据变更的hash摘要。用户可以通过重新计算hash摘要、验证hash摘要一致等方式判断防篡改用户表是否被篡改。

账本中的每条记录代表一条已经发生了的既定操作事实，其内容只可追加不可修改，通过对防篡改用户表和相应历史表进行一致性校验可以实现对篡改行为的识别和追溯溯源。此外，账本数据库提供了防篡改用户表一致性校验接口、历史表恢复和归档接口以满足篡改识别、数据膨胀消减、历史数据修复归档等需求。

8.6 AI 能力

AI4DB

包括参数智能调优与诊断、慢SQL发现、索引推荐、时序预测、异常检测等，能够为用户提供更便捷的运维操作和性能提升，实现自调优、自监控、自诊断等功能。

DB4AI

兼容MADlib生态，支持70+算法，性能相比MADlib on PostgreSQL 具有数倍提升。新增XGBoost、prophet、GBDT等高级且常用的算法套件，补充MADlib生态的不足。统一SQL到机器学习的技术栈，实现从数据管理到模型训练的SQL语句“一键驱动”。

提供fenced UDF能力;提供数据库原生DB4AI库内算法能力,包括库内执行计划、库内算子及SQL语法。

9 技术指标

| 技术指标 | 最大值 |
|----------------------|---|
| 数据库容量 | 受限于操作系统与硬件 |
| 单表大小 | 32TB |
| 单行数据大小 | 1GB |
| 每条记录单个字段的大小 | 1GB |
| 单表记录数 | 最大为 $2^{32} * (8k/\text{行宽})$ 。代码层面的限制是单表最多 2^{32} 个页面，每个页面大小为8k。假设当前数据行宽是1k，则单表记录数为 $2^{32} * 8 = 2^{35}$ 行（当前页面大小是8k，每个页面包含8行数据）。 |
| 单表列数 | 250~1600（随字段类型不同会有变化） |
| 单表中的索引个数 | 无限制 |
| 复合索引包含列数 | 32 |
| 数据库名长度 | 64 |
| 对象名长度（除数据库名以外的其他对象名） | 64 |
| 单表约束个数 | 无限制 |
| 并发连接数 | 10000 |
| 分区表的分区个数 | 32767 |
| 分区表的单个分区大小 | 32TB |
| 分区表的单个分区记录数 | 2^{55} |
| LOB最大容量 | (1G - 8203)B |
| SQL文本最大长度 | 1048576B |

10 术语表

表 10-1 术语表

| 术语 | 解释 |
|--------------|---|
| A - E | |
| ACID | 在可靠数据库管理系统（DBMS）中，事务（transaction）所应该具有四个特性：原子性（Atomicity）、一致性（Consistency）、隔离性（Isolation）和持久性（Durability）。 |
| AZ | Available Zone，通常指一个机房。 |
| Bgwriter | 数据库启动时创建的一个后台写线程，此线程用于将数据库中脏页面写入到持久性设备（例如磁盘）中。 |
| bit | 比特。计算机处理的最小的信息单位。比特用来表示二进制数字1或0，或者一种逻辑条件真或假。在物理上，比特表示一个电路上高或低的电压点或者磁盘上的磁化单程或其它。一个单独的比特位所传达的信息很少有意义的。然而，一个8位组却构成了一个字节，可用于表示如一个英文字母，十进制数字，或其它字符等多种类型的信息。 |
| Bloom Filter | 布隆过滤器。由Howard Bloom在1970年提出的二进制向量数据结构，它具有很好的空间和时间效率，被用来检测一个元素是不是集合中的一个成员，这种检测只会对在集合内的数据错判，而不会对不是集合内的数据进行错判，这样每个检测请求返回有“在集合内（可能错误）”和“不在集合内（绝对不在集合内）”两种情况，可见Bloom filter是牺牲了正确率换取时间和空间。 |
| CEK | Column Encryption Key，列加密密钥。 |
| CIDR | Classless Inter-Domain Routing，无类域间路由IP编址方案。CIDR摒弃传统的基于类（A类：8，B类：16，C类：24）的地址分配方式，允许使用任意长度的地址前缀，有效提高地址空间的利用率。CIDR表示方法：IP地址/网络ID的位数。比如192.168.23.35/21，其中“21”表示前面地址中的前21位代表网络部分，其余位代表主机部分。 |

| 术语 | 解释 |
|-----------|---|
| CLI | Command-line Interface，命令行界面。应用程序和用户交互的一种方式，完全基于文本输入和输出。命令通过键盘或类似装置输入，由程序编译并执行。结果是以文本或图形的方式呈现在终端界面。 |
| CM | Cluster Manager，数据库管理模块。管理和监控系统中各个功能单元和物理资源的运行情况，确保整个系统的稳定运行。 |
| CMK | 全密态场景：Client Master Key，客户端加密主密钥。 |
| CU | Compression Unit，压缩单元。列存表的最小存储单位。 |
| core文件 | 当程序出现内存越界、断言失败或者访问非法内存时，操作系统会中止进程，并将当前内存状态导出到core文件中，以便进一步分析。 core文件包含内存转储，支持全二进制和指定端口格式。core文件名称由字符串core以及操作系统进程ID组成。 core文件不依赖于任何平台。 |
| Core Dump | 通常在程序异常终止时，核心转储（Core Dump）、内存转储或系统转储用于记录特定时间计算机程序工作内存的状态。实际上，其它关键程序的状态经常在同一时间进行转储，例如处理器寄存器，包括程序指标和栈指针、内存管理信息、其它处理器和操作系统标记及信息。Core Dump经常用于辅助诊断和纠错计算机程序问题。 |
| DBA | Database Administrator，数据库管理员。指导或执行所有和维护数据库环境相关的操作。 |
| DBLINK | DBLINK是定义一个数据库到另一个数据库路径的对象，通过它可以查询远程数据库对象。 |
| DBMS | Database Management System，数据库管理系统。数据库管理系统是为了访问数据库中的信息而使用的一个管理系统软件。它包含一组程序使用户可以进入、管理、查询数据库中数据。基于真实数据的位置，可以分为内存数据库管理系统和磁盘数据库管理系统。 |
| DCF | Distributed Consensus Framework，分布式共识框架，基于Paxos算法实现数据同步强一致。 |
| DCL | Data Control Language，数据控制语言。 |
| DDL | Data Definition Language，数据定义语言。 |
| DEK | Data Encryption Key，数据加密密钥。 |
| DML | Data Manipulation Language，数据操纵语言。 |
| 备份 | 备份件或者备份过程。指复制并归档计算机数据，当发生数据丢失事件时，可以用该复制并归档的数据来恢复原始数据。 |
| 备份和恢复 | 保护数据库防止由于媒介失效或人为错误造成的数据丢失过程中涉及的一组概念、过程及策略。 |
| 备机 | openGauss双机方案中的一个节点，用于作为主机的备份，在主机异常时，备机会切换到主机状态，以确保能正常提供数据服务。 |

| 术语 | 解释 |
|-------|--|
| 崩溃 | 崩溃（或系统崩溃）指计算机或程序（例如软件应用程序或操作系统）异常终止的事件。出现错误后，通常会自动退出。有时出现恶意程序冻结或挂起直到崩溃上报服务记录崩溃的详细信息。对于操作系统内核关键部分的程序，整个计算机可能瘫痪（可能造成致命的系统错误）。 |
| 编码 | 编码是指用代码来表示各组数据资料，使其成为可利用计算机进行处理和分析的信息。用预先规定的方法将文字、数字或其它对象编成数码，或将信息、数据转换成规定的电脉冲信号。 |
| 编码技术 | 呈现计算机软硬件识别的特定字符集数据的技术。 |
| 表 | 表是由行与列组合成的。每一列被当作是一个字段。每个字段中的值代表一种类型的数据。例如，一个表可能有3个字段：姓名、城市和国家。这个表就会有3列：一列代表姓名，一列代表城市，一列代表国家。表中的每一行包含3个字段的内容，姓名字段包含姓名，城市字段包含城市，国家字段包含国家。 |
| 表空间 | 包含表、索引、大对象、长数据等数据的逻辑存储结构。表空间在物理数据和逻辑数据间提供了抽象的一层，为所有的数据库对象分配存储空间。表空间创建好后，创建数据库对象时可以指定该对象所属的表空间。 |
| 并发控制 | 在多用户环境下同时执行多个事务并保证数据完整性的一个DBMS服务。并发控制是openGauss提供的一种多线程管理机制，用来保证多线程环境下在数据库中执行的操作是安全的和一致的。 |
| 查询 | 向数据库发出的信息请求，包含更新、修改、查询或删除信息的请求。 |
| 查询操作符 | Query Operator，也称为查询迭代算子（Iterator）或查询节点（Query Tree Node）。一个查询的执行可以分解为一个或多个查询操作符，是构成一个查询执行的最基本单位。常见的查询操作符包括表扫描（Scan），表关联（Join），表聚集（Aggregation）等。 |
| 持久性 | 数据库事务的ACID特性之一。在事务完成以后，该事务对数据库所作的更改便持久的保存在数据库之中，并不会被回滚。 |
| 存储过程 | 存储过程（Stored Procedure）是在大型数据库系统中，一组为了完成特定功能的SQL语句集，经编译后存储在数据库中，用户通过指定存储过程的名称并设置参数（如果该存储过程带有参数）来执行它。 |
| 操作系统 | 操作系统OS（operating system）由引导程序加载到计算中，对计算机中其它程序进行管理。其它程序叫做应用或应用程序。 |
| 大对象 | 大对象（Blob）在数据库中指使用二进制方式存储的数据。它通常可以用于存储视频、音频和图像等多媒体数据。 |
| 段 | 数据库中，一段指包含一个或多个区域的数据库中的一部分。区域是数据库的最小范围，由单元调用块组成。一个或多个段组成一个表空间。 |

| 术语 | 解释 |
|----------|---|
| F - J | |
| Failover | 指当某个节点出现故障时，自动切换到备节点上的过程。反之，从备节点上切换回来的过程称为Failback。 |
| FDW | Foreign Data Wrapper，外部数据封装器。是openGauss提供的一个SQL接口，用于访问远程数据存储中的大数据对象，使DBA可以整合来自不相关数据源的数据，将它们存入数据库中的一个公共模型。 |
| Freeze | 在事务ID耗尽时由AutoVacuum Worker进程自动执行的操作。openGauss会把事务ID记在行头，在一个事务取得一行时，通过比较行头的事务ID和事务本身的ID判断这行是否可见，而事务ID是一个无符号整数，如果事务ID耗尽，事务ID会跨过整数的界限重新计算，此时原先可见的行就会变成不可见的行，为了避免这个问题，Freeze操作会将行头的事务标记为一个特殊的事务ID，标记了这个特殊的事务ID的行将对所有事务可见，以此避免事务ID耗尽产生的问题。 |
| GDB | GNU工程调试器，可以监控其它程序运行时的内部情况，或者其它程序要崩溃时发生了什么。GDB支持如下四种主要操作（使PDK功能更加强大），辅助查找缺陷。 <ul style="list-style-type: none">• 启动程序，指定可能影响行为的任何因素。• 特定条件下，停止程序。• 程序停止时，检查发生了什么。• 修改程序内容，尝试纠正一个缺陷并继续下一个。 |
| GIN索引 | Generalized Inverted Index，通用倒排索引。作用为处理索引项为组合值的情况，查询时需要通过索引搜索出出现在组合值中的特定元素值。 |
| GNU | GNU计划，又称革奴计划，是由RichardStallman在1983年9月27日公开发起的。它的目标是创建一套完全自由的操作系统。GNU是“GNU's NotUnix”的递归缩写。Stallman宣布GNU应当发音为Guh-NOO以避免与new这个单词混淆（注：Gnu在英文中原意为非洲牛羚，发音与new相同）。Unix是一种广泛使用的商业操作系统的名称。技术上讲，GNU类似Unix。但是GNU却给了用户自由。 |
| gsql | openGauss交互终端。通过gsql能够以交互的方式输入查询，下发查询到openGauss，然后查看查询结果。或者，也可以从文件中输入。此外，gsql还提供许多元命令和各种类似shell命令，协助脚本编写及自动化各种任务。 |
| GUC | Grand Unified Configuration，数据库运行参数。配置这些参数可以影响数据库系统的行为。 |
| HA | 高可用性（HighAvailability），通过尽量缩短因日常维护操作（计划）和突发的系统崩溃（非计划）所导致的停机时间，以提高系统和应用的可用性。 |

| 术语 | 解释 |
|--------------|---|
| HBA | host-based authentication, 主机认证。主机鉴权允许主机鉴权部分或全部系统用户。适用于系统所有用户或者使用Match指令的子集。该类型鉴权对于管理计算以及其它完全同质设备非常有用。总之, 服务器上的三个文件以及客户端上的一个文件必须修改, 为主机鉴权做准备。 |
| IV | Initialization Vector, 初始向量。初始向量是许多加密模式中用于随机化加密的一块数据, 因此可以由相同的明文、相同的密钥产生不同的密文。 |
| 服务器 | 为客户端提供服务的软硬件的组合。单独使用时, 指运行服务器操作系统的计算机, 也可以指提供服务的软件或者专用硬件。 |
| 隔离性 | 数据库事务的ACID特性之一。它是指一个事务内部的操作及使用的数据对其它并发事务是隔离的, 并发执行的各个事务之间不能互相干扰。 |
| 关系型数据库 | 创建在关系模型基础上的数据库。关系型数据库借助于集合代数等数学概念和方法来处理数据库中的数据。 |
| 归档线程 | 数据库打开归档功能时启动的一个线程, 此线程用于将数据库日志归档到指定的路径。 |
| 故障接管 | 功能对等的系统部件对于故障部件的自动替换过程。系统部件包含处理器、服务器、网络、数据库等。 |
| 环境变量 | 定义进程操作环境某一方面的变量。例如, 环境变量可以为主目录, 命令搜索路径, 使用终端或当前时区。 |
| 检查点 | 将数据库内存中某一时刻的数据存到磁盘的机制。openGauss定期将已提交的事务数据和未提交的事务数据存到磁盘, 这些数据用来和Redo日志一起在数据库重启和崩溃时恢复数据库。 |
| 加密 | 用于传输数据的功能。通过该功能, 可以隐藏信息内容, 防止非法使用。 |
| 节点 | 将构成openGauss数据库环境的各台服务器(物理机或虚拟机)称为数据库节点, 简称节点。 |
| 纠错 | 系统自动识别软件和数据流上的错误并自动修正错误的能力, 提升系统的稳定性和可靠性。 |
| 进程 | 在单个计算机上执行程序的实例。一个进程由一个或多个线程组成。其它进程不能接入某个进程已占用的线程。 |
| 基于时间点恢复 | PITR (Point-In-Time Recovery), 基于时间点恢复是openGauss备份恢复的一个特性, 是指在备份数据和WAL日志正常的情况下, 数据可以恢复到指定时间点。 |
| 记录 | 在关系型数据库中, 每一条记录对应表中的每一行数据。 |
| K - O | |
| KMC | Key Management Component, 密钥管理组件。 |
| KMS | Key Management Service, 密钥管理服务。 |

| 术语 | 解释 |
|--------------|--|
| KSF | Key Store File，密钥存储文件。 |
| 逻辑复制 | 数据库主备或两个数据库间的数据同步方式。区别于通过物理日志回放方式的物理复制，逻辑复制在两个数据库间传输逻辑日志或通过逻辑日志对应的SQL语句实现数据同步。 |
| 逻辑日志 | 数据库修改的日志记录，可直接对应为SQL语句，一般为行级记录。区别于物理日志，物理日志是记录物理页面修改的日志。 |
| 逻辑解码 | 逻辑解码是一种通过对xlog日志的反解实现将数据库表的所有持久更改抽取到一种清晰、易于理解的格式的处理过程。 |
| 逻辑复制槽 | 在逻辑复制的环境下，逻辑复制槽用以防止Xlog被系统或Vaccum回收。openGauss中用于记录逻辑解码位置的对象，提供创建、删除、读取、推进等多个SQL接口函数。 |
| MVCC | Multi-Version Concurrency Control，多版本并发控制。数据库并发控制协议的一种，它的基本算法是一个元组可以有多个版本，不同的查询可以工作在不同的版本上。一个基本的好处是读和写可以不冲突。 |
| NameNode | NameNode是Hadoop系统中的一个中心服务器，负责管理文件系统的名称空间（namespace）以及客户端对文件的访问。 |
| OM | Operations Management，运维管理模块。提供数据库日常运维、配置管理的管理接口、工具。 |
| 客户端 | 连接或请求其它计算机或程序服务的计算机或程序。 |
| 空闲空间管理 | 管理表内空闲空间的机制，通过记录每个表内空闲空间信息，并建立易于查找的数据结构，可以加速对空闲空间进行的操作（例如INSERT）。 |
| 垃圾元组 | 是指使用DELETE和UPDATE语句删除的元组，openGauss在删除元组时只是打个删除标记，由Vacuum线程清理这种垃圾元组。 |
| 列 | 字段的等效概念。在数据库中，表由一列或多列组成。 |
| 逻辑节点 | 一个物理节点上可以安装多个逻辑节点。一个逻辑节点是一个数据库实例。 |
| 模式 | 数据库对象集，包括逻辑结构，例如表、视图、序、存储过程、同义名、索引及数据库链接。 |
| 模式文件 | 用于决定数据库结构的SQL文件。 |
| P - T | |
| Page | openGauss数据库关系对象结构中行存的最小内存单元。一个Page大小默认为8KB。 |
| Paxos | 分布式一致性协议。 |
| PostgreSQL | PostgreSQL是一个开源的关系数据库管理系统（DBMS），由全球志愿者团队开发。PostgreSQL不受任何公司或个体所控制，源代码免费使用。 |

| 术语 | 解释 |
|-------------|---|
| Postmaster | 数据库服务启动时启动的一个线程。用于侦听来自数据库其它节点或客户端的连接请求。 主机上侦听到备机连接请求，并接受后，就会创建一个WAL Sender线程，用于处理与备机的交互。 |
| publication | 发布可以被定义在任何物理复制的主服务器上。定义有发布的节点被称为发布者。发布是从一个表或者一组表生成的改变的集合，也可以被描述为更改集合或者复制集合。每个发布都只存在于一个数据库中。 |
| RHEL | Red Hat Enterprise Linux，红帽企业Linux。 |
| REDO日志 | 记录对数据库进行操作的日志，这些日志包含重新执行这些操作所需要的信息。当数据库故障时，可以利用REDO日志将数据库恢复到故障前的状态。 |
| RK | Root Key，加密根密钥。 |
| SCTP | Stream Control Transmission Protocol，流控制传输协议。是IETF于2000年新定义的一个传输层协议。是提供基于不可靠传输业务的协议之上的可靠的数据报传输协议。SCTP的设计用于通过IP网传输SCN窄带信令消息。 |
| Savepoint | 保存点。是一种在关系数据库管理系统中实现子事务（也称为嵌套事务）的方法。在一个长事务中，可以把操作过程分成几部分，前面部分执行成功后，可以建一个保存点，若后面的执行失败，则回滚到这个保存点即可，无需回滚整个事务。保存点对于在数据库应用程序中实现复杂错误恢复很有用。如果在多语句事务中发生错误，则应用程序可能能够从错误中恢复（通过回滚到保存点）而无需中止整个事务。 |
| Session | 数据库系统在接收到应用程序的连接请求时，为该连接创建的一个任务。它被Session Manager管理，完成一些初始化任务，执行用户的所有操作。 |
| SMP | Symmetric Multi-Processing，对称多处理技术，是指在一台计算机上汇集了一组处理器（多CPU），各CPU之间共享内存子系统以及总线结构。操作系统必须支持多任务和多线程处理，以使得SMP系统发挥高效的性能。数据库领域的SMP并行技术，一般指利用多线程技术实现查询的并行执行，以充分利用CPU资源，从而提升查询性能。 |
| SQL | Structure Query Language，结构化查询语言。数据库的标准查询语言。它可以分为数据定义语言（DDL），数据操纵语言（DML）和数据控制语言（DCL）。 |
| SSL | Secure Socket Layer，安全套接层。SSL是Netscape公司率先采用的网络安全协议。它是在传输通信协议（TCP/IP）上实现的一种安全协议，采用公开密钥技术。SSL广泛支持各种类型的网络，同时提供三种基本的安全服务，它们都使用公开密钥技术。SSL支持服务通过网络进行通信而不损害安全性。它在客户端和服务端之间创建一个安全连接。然后通过该连接安全地发送任意数据量。 |

| 术语 | 解释 |
|---------------------|--|
| subscription | 订阅是逻辑复制的下游端。订阅被定义在其中的节点被称为订阅者。一个订阅会定义到另一个数据库的连接以及它想要订阅的发布集合（一个或者多个）。 |
| 收敛比 | 交换机下行带宽与上行带宽的比值。收敛比越高，流量收敛程度越大，丢包越严重。 |
| Table Access Method | 表访问方法层，对执行引擎和存储引擎进行解耦，实现存储引擎的可插拔能力。 |
| TCP | Transmission Control Protocol，传输控制协议。用于将数据信息分解成信息包，使之经过IP协议发送；并对利用IP协议接收来的信息包进行校验并将其重新装配成完整的信息。TCP是面向连接的可靠协议，能够确保信息的无误发送。 |
| trace | 一种特殊的日志记录方法，用来记录程序执行的信息。程序员使用该信息进行纠错。另外，根据trace日志中信息的类型和内容，有经验的系统管理员或技术支持人员以及软件监控工具诊断软件常见问题。 |
| 强一致性 | 任何查询不会瞬时的看到一个事务的中间状态。 |
| 全备份 | 备份整个数据库。 |
| 全量同步 | openGauss双机方案中的一种数据同步机制，是指把主机中的所有数据同步给备机。 |
| 日志文件 | 计算机记录自身活动的记录。 |
| 事务 | 数据库管理系统执行过程中的一个逻辑单位，由一个有限的数据库操作序列构成，事务必须满足ACID原则。 |
| 数据 | 事实或指令的一种表达形式，适用于人为或自动的通信、解释或处理。数据包含常量、变量、阵列和字符串。 |
| 数据分区 | 数据分区是指在一个数据库实例内部，将表按照指定范围划分为多个数据互不重叠的部分（Partition）。具体的分区方式可以有：范围分区（Range），它根据元组中指定字段的取值所处的范围映射到目标存储位置。 |
| 数据库 | 数据库是存储在一起的相关数据的集合，这些数据可以被访问，管理以及更新。同一视图中，数据库可以根据存储内容类型分为以下几类：数目类、全文本类、数字类及图像类。 |
| 数据库实例 | 一个数据库实例是一个openGauss进程以及它控制的数据库文件。openGauss在一个物理节点上安装多个数据库实例。一个数据库实例也被称为一个逻辑节点。 |
| 数据库双机 | openGauss提供的高可靠性双机方案。在此方案中，每个openGauss逻辑节点标识为主机或备机。在同一时间内，只有一个openGauss被标识为主机。双机初次建立时，主机会对每个备机数据做全量同步，然后做增量同步。双机建立之后的运行过程中，主机能接受数据读和写的操作请求，备机只做日志同步。 |
| 数据库文件 | 保存用户数据和数据库系统内部数据的二进制文件。 |

| 术语 | 解释 |
|-----------------|--|
| 数据字典 | 数据字典是一系列只读的表，用来提供数据库的信息。这些信息包括：数据库设计信息、存储过程信息、用户权限、用户统计数据、数据库进程信息、数据库增长统计数据和数据库性能统计数据。 |
| 死锁 | 为使用同一资源而产生的无法解决的争用状态。 |
| 索引 | 数据库索引，是数据库管理系统中一个排序的数据结构，以协助快速查询、更新数据库表中数据。 |
| 统计信息 | 数据库使用统计信息估算查询代价，以查找代价最小的执行计划，统计信息一般是数据库自动收集的，包括表级信息（元组数、页面数等）和列级信息（列的值域分布直方图）。 |
| 停用词 | 在信息检索中，为节省存储空间和提高搜索效率，在处理自然语言数据（或文本）之前或之后会自动过滤掉某些字或词，这些字或词即被称为Stop Words（停用词）。 |
| U - Z | |
| Ustore | In-place Update存储引擎别称，很好的解决了Append update存储引擎空间膨胀，元组较大的劣势，高效回滚段的设计是In-place update存储引擎的基础。 |
| Undo Record | 撤销记录，用于undo记录的插入、查询以及组织，北向对接ustore，南向对接buffer pool。 |
| Undo Space | 管理Undo记录的物理资源，包括增删undo文件等。 |
| Undo Zone | 与业务线程绑定，管理每个业务线程的Undo逻辑资源。 |
| TransactionSlot | 按照事务粒度记录Undo Record，用于事务回滚及Undo记录回收。 |
| TIMECAPSULE | 闪回的关键字，使用闪回技术后，恢复已提交的数据库修改前的数据，只需要秒级，而且恢复时间和数据库大小无关。 |
| RECYCLE BIN | 回收站，开启回收站开关后，DROP TABLE可以将表及其子对象放入回收站中。 |
| PURGE | 对回收站对象进行清理。 |
| Vacuum | 数据库定期启动的清理垃圾元组的线程，根据配置参数可以同时启动多个。 |
| verbose | verbose选项指定显示在屏幕上的处理信息。 |
| WAL | Write-Ahead Logging，预写日志系统。实现事务日志的标准方法，是指对数据文件（表和索引的载体）持久化修改之前必须先持久化相应的日志。 |
| WAL Receiver | 数据库复制时备机创建的一个线程的名称。此线程用于从主机接收数据、命令，并反馈确认信息至主机。一个备机只有一个WAL Receiver线程。 |

| 术语 | 解释 |
|------------|--|
| WAL Sender | 数据库复制过程中，主机接受到备机的连接请求后创建的一个线程的名称。此线程用于发送命令、数据到备机，并从备机接收信息。一个主机可能会有多个WAL Sender线程，每一个WAL Sender线程对应一个备机的一个连接请求。 |
| WAL Writer | 数据库启动时创建的一个写Redo日志的线程，用于将内存中的日志写入到持久性设备（如：磁盘）。 |
| Xlog | 表示事务日志，一个逻辑节点中只有一个，不允许创建多个Xlog文件。 |
| xDR | 详单。用户面和信令面详单的统称，包括CDR和UFDR、TDR和SDR。 |
| 物理节点 | 一个物理机器称为一个物理节点。 |
| 系统表 | 存储数据库元信息的表，元信息包括数据库中的用户表、索引、列、函数和数据类型等。 |
| 下推 | openGauss可以利用多DN并行执行查询计划，即将数据库主节点中的查询计划下发到各DN中并行执行。这种行为称为下推。与将数据抽取到数据库主节点上执行查询的方式相比，下推可以大幅提升查询性能。 |
| 压缩 | 数据压缩，信源编码，或比特率降低涉及使用相比原来较少比特的编码信息。压缩可以是有损或无损。无损压缩通过识别和消除统计冗余降低比特位。无损压缩中没有信息丢失。有损压缩识别并删除次要信息，减少了比特位。减少数据文件大小的方法被普遍称为数据压缩，尽管其正式名称为源编码（数据源的编码，然后将其存储或传输）。 |
| 一致性 | 数据库事务的ACID特性之一。在事务开始之前和事务结束以后，数据库的完整性约束没有被破坏。 |
| 元数据 | 用来定义数据的数据。主要是描述数据自身信息，包含源、大小、格式或其它数据特征。数据库字段中，元数据用于理解以及诠释数据仓库的内容。 |
| 原子性 | 数据库事务的ACID特性之一。整个事务中的所有操作，要么全部完成，要么全部不完成，不可能停滞在中间某个环节。事务在执行过程中发生错误，会被回滚到事务开始前的状态，就像这个事务从来没有执行过一样。 |
| 脏页面 | 已经被修改且未写入持久性设备的页面。 |
| 增量备份 | 基于上次有效备份之后对文件修改的备份。 |
| 增量同步 | openGauss双机方案中的一种数据同步机制，是指把主机中数据增量同步给备机，即只同步主备间有差异的数据。 |
| 主机 | openGauss数据库双机系统中接受数据读写操作的节点，和所有备机一起协同工作。在同一时间内，双机系统中只有一个节点被标识为主机。 |
| 主题词 | 在标引和检索中用以表达文献主题的规范化的词或词组。 |

| 术语 | 解释 |
|-------|---|
| 转储文件 | 转储文件是一种特定类型的trace文件。转储文件为响应事件过程中一次性输出的诊断数据，trace文件指诊断数据的连续输出。 |
| 最小恢复点 | 最小恢复点是openGauss提供的数据库一致性保障手段之一。最小恢复点特性可以在openGauss启动时检查出WAL日志和持久化到磁盘的数据的不一致性，并提示用户进行处理。 |