中兴通讯 GoldenDB 数据库 V6 技术白皮书

2025-08-15 发布 2025-08-15 实施

中兴通讯股份有限公司发布

法律声明

若接收**中兴通讯股份有限公司**（以下称为“中兴通讯”）的此份文档，即表示您已同意以下条款。若不同意以下条款，请停止使用本文档。

本产品符合有关环境保护和人身安全方面的设计要求，产品的存放、使用和弃置应遵照产品手册、相关合同或相关国法律、法规的要求进行。

本文档按“现状”和“仅此状态”提供。本文档中的信息随着中兴通讯产品和技术的进步将不断更新，中兴通讯不再通知此类信息的更新。

中兴通讯股份有限公司

目录

1 文档概述 7

1.1 文档使用范围 7

1.2 术语 7

1.3 缩略语 8

2 产品介绍 9

2.1 产品定位 9

2.2 产品特点 10

3 体系结构 12

3.1 架构设计 12

3.1.1 逻辑架构 12

3.1.2 系统数据流向 14

3.1.3 分布式/集中式一体化 16

3.1.4 混合负载（HTAP）能力 17

3.1.5 多语法引擎支持 18

3.2 部署模式 18

3.2.1 集中式部署 18

3.2.2 分布式部署 19

3.2.3 混合部署 20

3.2.4 物理机部署 20

3.2.5 容器化部署 21

3.3 可靠性方案设计 22

3.3.1 单管理节点 22

3.3.2 本地单中心高可用 23

3.3.3 同城三中心 24

3.3.4 本地同城双中心高可用 25

3.3.5 两地三中心高可用 25

3.3.6 两地四中心高可用 26

3.3.7 三地五中心高可用 27

4 功能介绍 27
4.1 分布式事务控制 28
4.1.1 隔离级别 28
4.1.2 分布式事务控制 28
4.2 基础功能 30
4.2.1 表存储组织 30
4.2.2 数据类型 30
4.2.3 常见数据对象 31
4.2.4 数据分片功能 31
4.2.5 函数功能 32
4.2.6 索引机制 33
4.2.7 存储过程 33
4.2.8 触发器功能 33
4.2.9 视图功能 34
4.2.10 分区功能 34
4.2.11 SQL 绑定 34
4.2.12 执行计划 34
4.2.13 外部表 35
4.2.14 临时表 35
4.2.15 字符集 35
4.2.16 国产化适配 35
4.3 高级功能 36
4.3.1 扩展性 36
4.3.2 MVCC 36
4.3.3 数据重分布 37
4.3.4 读写分离 38
4.3.5 导入导出 39
4.3.6 Oracle 兼容性 40
4.3.7 XA 事务 43
4.3.8 压缩功能 43
4.3.9 日切卸数 44
4.3.10 多租户 44

4.3.11 数据库负载回放 45
4.4 产品性能 45
4.4.1 缓存管理能力 45
4.4.2 并发控制机制 45
4.4.3 数据复制优化技术 46
4.4.4 批量协议 46
4.4.5 并行执行 46
4.4.6 流式查询 46
4.4.7 复杂查询并发 46
4.4.8 大容量数据处理能力 47
4.4.9 分布式批处理 47
4.5 安全性 47
4.5.1 安全审计 48
4.5.2 密码安全 49
4.5.3 数据安全 50
4.5.4 运维安全 54
4.6 数据库管理功能 56
4.6.1 安装与升级 57
4.6.2 数据配置 57
4.6.3 运维 58
4.7 GoldenDB 工具集 59
4.7.1 CACTool 迁移评估工具 59
4.7.2 Sloth 数据迁移工具 61
4.7.3 Replay 回放工具 64
4.7.4 InsightTool 巡检诊断工具 65
4.7.5 GDC 可视化开发工具 66

5 应用场景 67
5.1 金融行业 67
5.2 电信行业 69
5.3 政务行业 70
5.4 交通行业 72

5.5 医疗行业 73

6 产品生态 74

1 文档概述

1.1 文档使用范围

本文针对中兴通讯 GoldenDB 数据库的市场定位和特点，帮助用户、合作伙伴等从产品的体系架构、组件基本原理、产品功能和应用场景上了解本产品。

本文档适合初次接触本产品的用户，用于指导用户从宏观上对 GoldenDB 数据库建立初步的了解和认识。

1.2 术语

本章节对 GoldenDB 数据库 V6 技术白皮书中常用的术语、定义以及英文缩写进行汇总说明，便于用户阅读。

表 1-1 术语表

术语/定义	说明
分布式数据库	具备分布式事务处理能力、可平滑扩展、分布于计算机网络且逻辑上统一的数据库。
集中式数据库	一种经典、传统的数据库结构，由一台或少数几台机器联合管理数据，一般不对数据进行分片，没有分布式事务。
分布式事务	指事务的参与者、支持事务的服务器、资源服务器以及事务管理器分别位于不同的分布式系统的不同节点之上。
计算节点	GoldenDB 数据库中负责计算的组件，对外提供数据库实例接入服务；负责 SQL 优化、SQL 路由、数据节点的负载均衡、分布式事务的调度等。
数据节点	GoldenDB 数据库的数据存储组件，实现对数据的存储。提供单分片、多分片两种存储形式。分片包含多副本，保证数据安全。
全局事务管理器	全局事务管理，用于协助计算节点进行分布式事务管理，主要包括生成、释放全局事务 ID（GTID）、维护活跃事务以及当前活跃 GTIDs 的快照。
租户实例	GoldenDB 数据库提供多租户数据库实例管理，在 GoldenDB 中，可以创建多个数据库实例，分配给不同的业务使用。租户实例也称作集群，实例由计算节点、数据节点、全局事务管理器构成，可以是单分片或多分片。租户之间资源、数据相互隔离。
分片	GoldenDB 将数据依据分发策略存储在不同数据节点上，数据节点简称为分片。
DBGroup/安全分	多个数据节点组成一个安全分片组，其中有一个主节点(Master)，一个或多个

术语/定义	说明
片组	备节点(Slave)，主备之间进行数据复制，实现数据的多副本存储。
原子性	事务中的所有操作要么全部执行成功，要么全部失败回滚，没有中间状态。
一致性	事务开始前和结束后，数据库的完整性约束没有被破坏。
隔离性	并发执行的事务之间是相互隔离的，一个事务的执行不能影响其他事务的执行。
持久性	一旦事务提交，其结果应该是永久性的，即使系统发生故障，事务的结果也不应丢失。
Insight	GoldenDB 数据库运维平台的名称，负责数据库实例的创建、运维操作，提供监控、告警等运维功能。
JDBC	Java 数据库连接,是 Java 语言中用来规范客户端程序如何来访问数据库的应用程序接口，提供了诸如查询和更新数据库中数据的方法。
ODBC	开放式数据库连接接口，开发人员能够使用相同的 API 来连接、查询和操作不同类型的数据库。
OCI	Oracle 调用层接口，Oracle 公司提供的由头文件和库函数等组成的一个访问 Oracle 数据库的应用程序编程接口。它允许开发人员在第三代编程语言中通过 SQL 操纵 Oracle 数据库。
SQL	结构化查询语言，一种数据库查询和程序设计语言，用于存取数据以及查询、更新和管理数据库。
DDL	数据库模式定义语言 DDL(Data Definition Language)，用于定义或改变数据库或表的结构等初始化工作。
DML	数据操纵语言 DML(Data Manipulation Language)，用于管理和检索数据库中的数据。
DQL	数据查询语言 DQL(Data Query Language)，用于查询数据库表中记录。
DCL	数据控制语言 DCL(Data Control Language)，用于定义数据库的访问权限和安全级别。
gSync	GoldenDB 使用的数据节点之间的同步复制技术。
CACTool	GoldenDB 迁移评估工具。

1.3 缩略语

本文使用的专用缩略语以及说明参见下表。

表 1-2 缩略语表

缩略语	英文	中文
CN	ComputeNode	计算节点
DN	DataNode	数据节点
GTM	GlobalTransactionManager	全局事务管理节点
MDS	MetaDataServer	元数据管理服务
PM	ProxyManager	计算节点管理服务
CM	ClusterManager	集群管理服务
JDBC	Java Database Connectivity	Java 数据库连接
ODBC	Open Database Connectivity	开放数据库连接
OCI	Oracle Call Interfce	Oracle 调用层接口
SQL	Structured Query Language	结构化查询语言
API	Application Programming Interface	应用程序接口
ACID	Atomicity、Consistency、Isolation、Durability	ACID,指数据库事务正确执行的四个基本要素的缩写。包含： 原子性（Atomicity） 一致性（Consistency） 隔离性（Isolation） 持久性（Durability）
OLTP	On-Line Transaction Processing	联机事务处理
OLAP	On-Line Analytical Processing	联机分析处理
HTAP	Hybrid Transaction/Analytical Processing	混合事务分析处理
DDL	Data Definition Language	数据定义语言
DML	Data Manipulation Language	数据操纵语言
DQL	Data Query Language	数据查询语言
DCL	Data Control Language	数据控制语言

2 产品介绍

2.1 产品定位

GoldenDB 是安全、自主、可控的集中式/分布式数据库产品，产生背景：

- 互联网技术的兴起和发展，给各行各业带来大数据量的冲击，传统数据库难以应对应用层的高并发数据访问
- 主管部门从信息安全等角度对 IT 基础设施提出了安全可靠的要求
- 日趋严重的 IT 成本控制压力对 IT 行业提出更高要求

GoldenDB 是中兴通讯自主研发的集中式/分布式数据库产品，作为成熟稳定商用领先的金融级数据库产品，深耕银行(国有大行、股份制银行、城商行等)、证券、保险、电信运营商等多个关键领域。产品融入中兴通讯多年的数据库研发经验，为用户提供了强一致、高可用、高可靠、高安全、高性能、可扩展的一体化的数据库解决方案；满足 OLTP 和 OLAP 不同类应用要求，提供统一的基础数据服务平台，有利于提升业务创新能力和用户体验。

2.2产品特点

GoldenDB 通过全套数据库解决方案，可对需要关系型数据库的应用系统提供统一的数据库服务，实现数据的集中存放、统一管理和数据能力开放平台解决方案，为客户提供扁平化的基础数据处理平台。GoldenDB 数据库在功能、高可用、安全性、高可用提供的主要特性如下：

- 主要功能特性：

1） 强一致性分布式事务

GoldenDB 数据库从数据库层面实现分布式事务的强一致性，对应用透明，无需应用改造，提升应用迁移效率。采用分布式事务管理器和已提交事务自动回滚相结合的技术，解决分布式事务的强一致性问题。使用全局事务 ID 解决分布式事务的并发控制，严格保证事务 ACID 特性，在实现强一致性的同时确保性能优异。

2） 灵活的数据切片技术

GoldenDB 数据库分布式场景下支持哈希、范围、列表、复制、多级分片等多种数据分片规则，可以根据业务数据特征，选择最适合的分片技术把数据分别存储在多个数据安全组中；通过合理的数据分片规则，发挥分布式数据库的最佳性能。

3） 可扩展性

GoldenDB 软件架构分层设计，基于 Share-Nothing 架构由多个独立且互不共享 CPU、内存、存储等系统资源的逻辑节点组成，采用集群方式部署，支持集中式和分布式混合部署，支持物理机、虚拟机和云化部署方式。其中分布式集群实现各组件的灵活扩展，从而提供高性能的数据库服务。同时结合数据动态重分布和读写分离等技术，实现性能的线性扩展。计算节点、数据节点均可横向线性扩展，满足性能及容量的无限扩展

需求。通过增加计算节点，可以提升系统的业务处理能力；通过增加数据节点和在线数据重分布功能，可以提升系统的存储能力。支持数据节点的扩容、缩容，能高效地将数据均匀分布到数据库集群上；同时保证对在线业务影响小，且可操作性强。

- 高 SQL 兼容性

GoldenDB 支持 SQL92、SQL99、SQL2003 标准语法，兼容 MySQL 语法，兼容常用 Oracle、DB2 语法。支持多级分片、存储过程、全局唯一索引、MVCC、闪回、XA 接口等高级特性。

- 数据备份恢复

GoldenDB 提供在线热备功能，支持一键恢复到任意时刻、指定库表备份恢复，克隆在线数据恢复，并支持分布式场景下的全局一致数据恢复。

- 支持读写分离，提升读写效率

提供灵活的读写分离功能，系统根据负载情况及操作类型，把写操作发送到主库、读操作负载均衡到从库，提高从库的利用率。支持读主节点、读备节点、主备节点权重配置，在线权重调整，支持灵活的负载均衡模式，灵活提升系统读性能。

- 高可靠性

GoldenDB 整个集群无单点故障，数据多副本，具备完善的数据备份恢复机制，支持双活数据中心，支持异地灾备。支持多地多中心（AZ）组网，任何节点不存在单点故障，可以支持多种组网架构，创新研发了 gSync 数据复制技术，针对不同的业务场景灵活配置不同的策略来满足不同的可用性和可靠性要求，提高系统吞吐量的同时，实现同城 RPO 为 0。满足金融级数据高可靠、服务高可用要求。GoldenDB 支持机房级故障自动切换、支持异地带载演练和异地带压演练。

- 云化支持能力

GoldenDB 借助云技术来实现数据库即服务，具备集中管控、快速部署、高效运维，以及弹性扩缩容等优点。GoldenDB 可对接云化平台标准接口，提供网络和存储持久化方案，把云平台和 GoldenDB 高可靠机制相结合，实现云化场景下的数据库多副本，节点自愈、故障隔离等能力。

- 支持混合负载

GoldenDB 支持混合负载（HTAP），实现了分布式并行计算处理能力，具有更快速的关系型数据复杂计算处理速度。应用程序可使用 SQL 语句查询 PB 量级的数据，可在毫秒级或秒级返回 AP 类查询响应，用于数据分析、海量数据聚合和报表生成等联机分析处理的使用场景。GoldenDB 进一步扩展列存存储 VEngine 模式，适用于交互式 SQL 数据仓库应用场景，用于提升业务报表统计和多维分析的处理性能。

- 自主、安全、可控

GoldenDB 数据库自主研发，核心代码 100%掌握，并在金融、电信等多个关键行业核心业务大规模商用。GoldenDB 数据库通过安全可靠测评、EAL4+、商用密码认证，可信数据库安全专项等多项安全和自主可控相关的专项测试。

- 高性能

北京国家金融科技认证中心、国家工业信息安全发展研究中心、建设银行、中信银行、银联数据、中国移动等众多权威机构和重要客户组织的测试中性能排名第一。

3 体系结构

3.1 架构设计

GoldenDB 是自主研发的数据库系统，整体由计算节点、数据节点、全局事务管理器、管理节点四种核心模块组成，整体架构采用高可靠性设计，无单点故障。计算节点为无状态多节点部署，数据集群内由多个安全分片组构成，每个安全分片组内数据节点主备多机部署，全局事务管理器主备多机部署。

3.1.1 逻辑架构

GoldenDB 数据库总体逻辑架构如图 3.1-1 所示

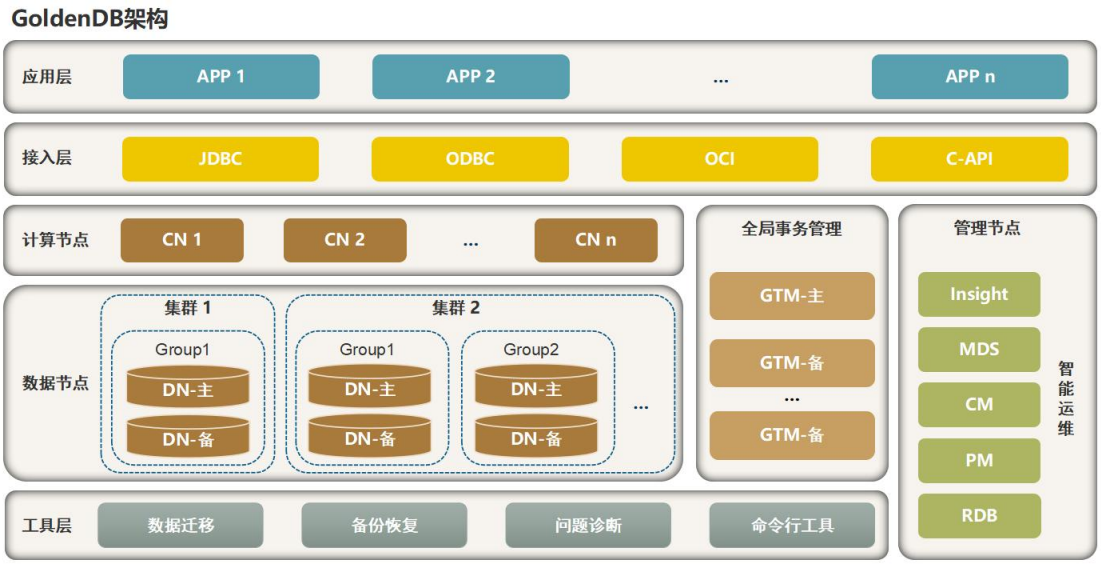


图 3.1-1 GoldenDB 逻辑架构图

GoldenDB 数据库由计算节点、数据节点、管理节点和全局事务管理节点四个部分组成。各个节点无需共享任何资源，都是独立自治的通用计算机节点，之间通过高速互联的网络通

讯，从而完成对应用数据请求的快速处理和响应，系统内部各节点之间负载均衡，使得每个节点资源得到充分利用。

计算节点

计算节点集群是分布式数据库的核心层，由无状态的计算节点组成。计算节点从应用层驱动或者管理节点接收 SQL 请求（结构化查询语言 Structured Query Language），进行逻辑优化和物理优化，生成满足分布式事务一致性的分布式查询计划；在执行分布式查询计划时，通过持续地访问数据节点，完成 SQL 请求的最终计算。计算节点可在线在任何可用域内进行扩展，所有计算节点都是对等的。

数据节点

数据节点集群是数据的最终存储模块，由多个安全组组成，每个表中的数据按照特定策略进行横向水平分片或纵向垂直拆分后存放到对应的安全组中。

安全组是由一个或多个数据节点构成的数据库节点组，组内的数据库节点拥有相同的数据。当安全组中存在多个数据节点时，其中一个数据节点为主用节点，其他数据节点都为备用节点，数据在主备节点之间实时复制。主用节点具备读写能力，备用节点可以提供读能力。支持在线调度，支持集群内在线增加副本，包括异地节点，没有副本数量限制，数据节点副本数量越多，可靠性就越高。

数据分片策略包括复制策略、哈希策略、范围策略、列表策略、多级分片。

- 哈希策略：适用于将数据均匀的分布到预先定义的各安全组上，保证各安全组的数据量大致一致。一般用于不需要关心分发字段的取值范围和具体含义，且对该表的 SQL 操作基本都是等值操作的场景。
- 范围策略：适用于指定一个给定的列值或列值集合应该保存在哪个安全组上，常用于时间、日期、数值等类型的字段上，如数据按照自然月或自然天分布存储。
- 列表策略：适用于含有一系列限定性或枚举性的字段上，如数据按照机构代码、国家代码、地区代码分布存储。
- 复制策略：适用于不常修改，且频繁出现在关联或子查询中的小表。复制策略下，复制表的数据保存在每一个节点，因此所有节点都需要这个表中数据的情况下，可减少节点间网络数据的传输，提高这种查询的性能。
- 多级分片：支持 5 层的分片规则设置，配合表的二级分区能力，支持最多 7 层的数据分片分区能力，适用于需要精细化控制数据在集群中的分布形态的场景，如多法人场景，将不同法人的数据划分到不同数据安全组。

管理节点

管理节点在分布式数据库中负责集群管理流程，不涉及业务的访问流程，无负载压力。管理节点具备如下功能：

- 可视化统一运维管理。GoldenDB 的管理平台（Insight）提供 GoldenDB 数据库产品的统一操作维护入口，用户可以在 Insight 上进行用户和权限管理、元数据管理、计算节点管理、数据节点管理、DDL 执行、节点扩容、备份恢复、系统安装、统计及告警管理等。
- 元数据管理。元数据指数据的元信息，如库、表、视图、触发器、存储过程、函数等数据模型的定义。元数据管理器存放系统的全量元数据，是整个分布式数据库集群的元数据中心。元数据管理器还保存整个集群的拓扑信息，因此是更广义的元数据管理。
- 计算节点管理。管理工作包括计算节点的创建、启用、禁用、删除和高可用调度。
- 数据节点管理，包括数据节点、安全组、数据节点集群的创建、变更、删除和数据节点异常、恢复后的调度管理、数据节点备份恢复的调度、数据重分布等功能的任务调度管理。

全局事务管理节点

全局事务管理器在分布式数据库中维护全局事务的全生命周期，提供申请、释放、查询全局事务的能力，采用集群方式部署。

全局事务管理节点高可用模式和数据节点一样使用一主多备的多数据副本容灾方案，数据副本扩容也是 online。

3.1.2 系统数据流向

GoldenDB 的计算节点和数据节点是多对多关系。一个计算节点可以接入多个数据节点子集群，提高计算节点利用率；一个数据节点子集群也可以通过多个计算节点接入，多个计算节点间可以实现高可用，当某个计算节点异常后，不影响应用访问。

上述多对多关系是通过服务端口来建立的。服务端口是 GoldenDB 的端口访问控制机制，定义了计算节点端口信息至后端数据节点子集群的一组对应关系，绑定了该连接实例的计算节点即提供对应数据节点子集群的访问接入。应用向绑定了该服务端口的计算节点的对应端

口发送 SQL 请求，该 SQL 请求会被计算节点路由到该服务端口定义的数据节点子集群。

当不同应用接入相同数据子节点集群时，可以定义不同的服务端口，达到接入权限、连接资源的隔离。

GoldenDB 应用访问关系如下图所示。

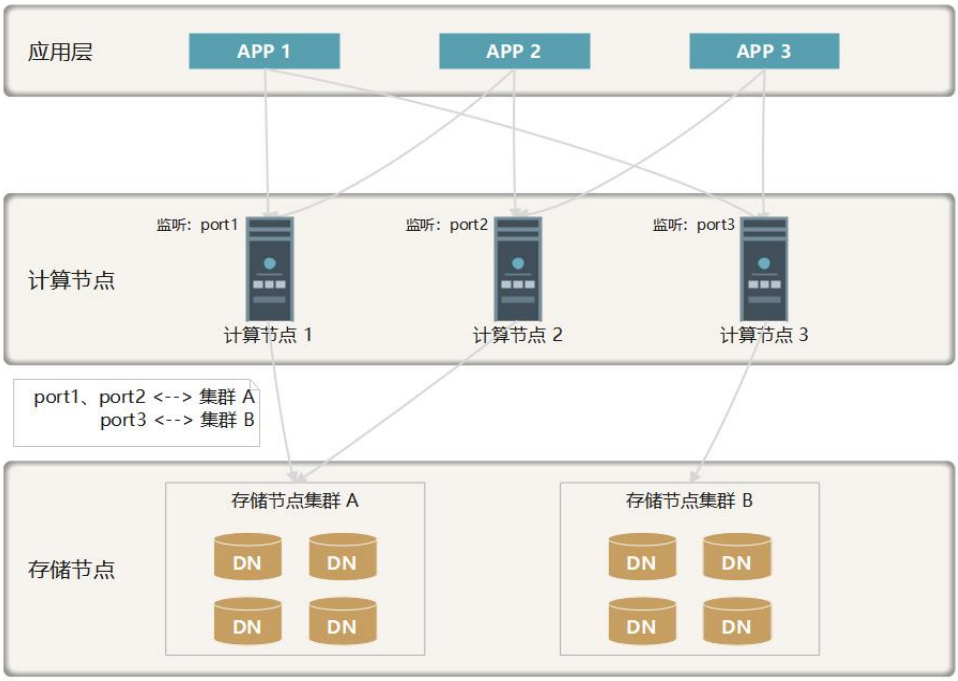


图 3.1-2 应用访问关系图

“图 3.1-2 应用访问关系图”说明：

- App1 和 App2 共享数据节点集群 A，App3 要求数据物理隔离，独享数据节点集群 B。
- 其中 APP1 使用 Port1 通过计算节点 1 和计算节点 3 接入集群 A，APP2 使用 Port2 通过计算节点 1 和计算节点 2 接入集群 A,APP3 使用 Port3 通过计算节点 2 和计算节点 3 接入集群 B。

结合上述对应关系，以 APP1 为例，一次 SQL 请求的数据流向说明如下：

1. 应用 APP1 使向计算节点 1 的 Port1 发起建链请求。
2. 建链成功后，APP1 在该链路上下发 SQL 请求。
3. 计算节点收到 SQL 请求后，进行语法解析、SQL 优化等，生成该 SQL 的分布式执行计划树；同时，通过查找 Port1 的连接实例定义信息，将请求下发至数据节点集群 A 的相关数据安全组。
4. 数据节点完成该请求的本地执行，并将结果集返回至计算节点 1。
5. 计算节点进行结果集汇总、计算，并将结果集返回至 APP1，过程中可能会在计

算节点和数据节点之间产生多次的数据交互。

3.1.3 分布式/集中式一体化

GoldenDB 支持在 IPV4 和 IPV6 环境中进行多种安装模式部署，适应不同的需求和场景。实现分布式/集中式一体化运维管理。

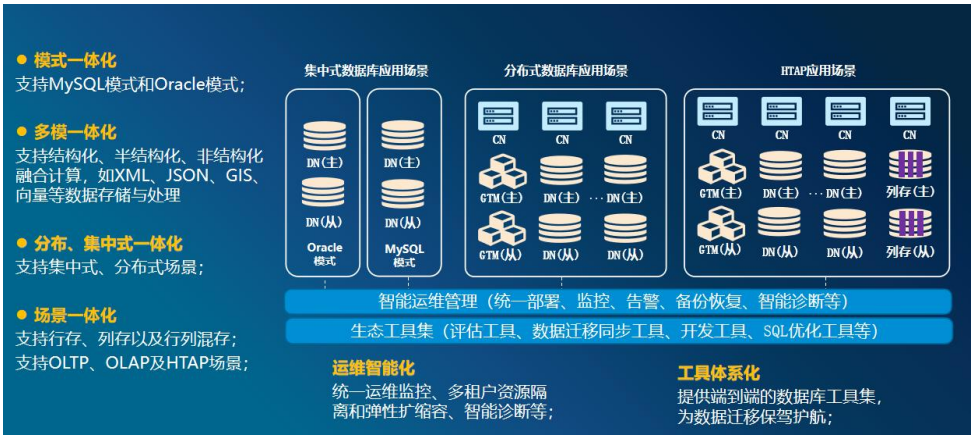


图 3.1-3 GoldenDB 一体化设计

多种部署模式：

- GoldenDB 提供了集中式部署模式。在集中式部署模式下的租户只包含单分片租户(CN/DN)且不带 GTM 节点。这种模式适用于数据规模比较小，不用按分片存储的业务。它只有普通事务，没有分布式事务。
- GoldenDB 支持分布式部署模式。分布式部署模式下的租户只包含多分片租户(CN/DN)且带 GTM 节点。这种模式适用于数据规模比较大，单个分片存储不下的业务。因为数据分布在多个分片，事务执行可能跨多个分片，即存在分布式事务，需要全局事务节点管理分布式事务。
- GoldenDB 还提供了集中式+分布式混合部署模式。在这种模式下，包含多种类型的租户，其中租户可分为单分片租户且不带 GTM 节点以及多分片租户且带 GTM 节点。这种模式适用于不同的数据规模业务要求，较为灵活。

关键部署特性：

- GoldenDB 支持同一集群的多租户特性，同一集群支持 MySQL 和 Oracle 两种兼容模式租户，兼容不同存量业务的快速对接和迁移。从而可以有效地支持多租户环境下的

数据库服务。租户资源模板有 CPU、内存、IOPS 三种资源分配维度，租户间资源相互隔离。租户内用户之间可通过创建资源组进行 CPU 和 IOPS 隔离。

- 在部署场景方面，GoldenDB 支持多种配置。可以选择单管理节点部署，适用于较小规模的应用；也可以选择本地单中心高可用部署，提高系统的可用性和容错性；还可以选择本地同城双中心高可用、两地三中心高可用、两地四中心高可用等复杂场景，满足多地域、多数据中心的需求。并且支持数据库租户的可用资源的灵活可配，包括 CPU、内存等单机资源和可用数据库节点数等。

- GoldenDB 可以部署在物理机，也可以进行容器化部署。容器化部署可以提供更高的灵活性、可移植性和资源利用率，使得 GoldenDB 能够更好地适应云计算环境和容器编排平台。

总之，GoldenDB 提供了多种安装部署模式、适应不同的场景和平台，为用户提供了可靠、高效的数据管理解决方案。

3.1.4 混合负载（HTAP）能力

GoldenDB 支持混合负载处理能力，同时满足交易类应用系统和分析类应用系统的数据库要求。其中 TP 部分由计算节点(CN)处理，并可增加 AP 组件(Coordinator 和 Worker)进行 SQL 统计查询，具有更快速的关系型数据复杂计算处理速度，适用于在毫秒级、秒级或分钟级返回查询响应，用于比较复杂的联机数据分析、海量数据聚合查询和报表生成等多表关联和多层嵌套查询的联机分析处理的使用场景。

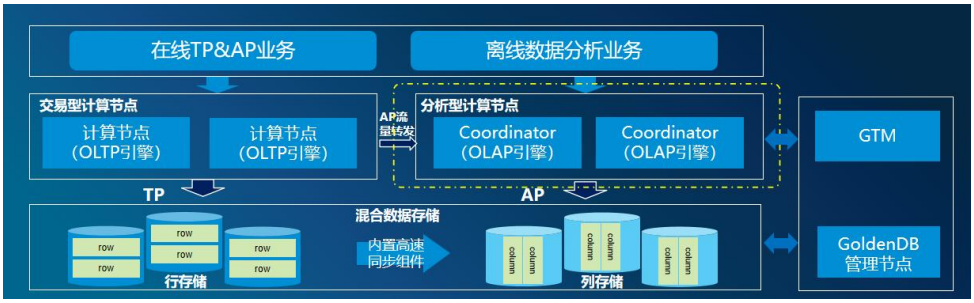


图 3.1-4 GoldenDB HTAP 架构

GoldenDB 可选装列存存储 VEngine 节点，适用于交互式 SQL 数据仓库应用场景，用于提升业务报表统计和多维分析的处理性能。通过 VEngine 节点，可创建行存模式表、列存模式表和行列混合表三种表类型。GoldenDB 对行列混合表同时在行存存储节点和列存存储节

点创建表，并通过日志同步组件自动进行数据准实时同步。

应用的 SQL 请求统一通过 CN 接入，CN 根据 SQL 所访问表的存储类型进行自动路由：CN 可直接访问行式存储，或者通过 AP 的 Coordinator/Worker 调度，对行式表或列式表进行查询计算。

支持根据业务场景要求创建纯行存表、纯列存表以及适应 HTAP 场景的行存列存冗余表，也可根据业务特性要求为行存表创建列存索引或是为列存表创建行存索引。支持 online 将行存表转换为列存表或行列混存表。

同一租户内可同时支持行存数据副本和列存数据副本，即可让同一行存表具备一个或多个列存副本。

3.1.5 多语法引擎支持

为适配更多存量业务平滑迁移，GoldenDB 支持一体化架构设计，支持在同一数据库集群内融合多种语法引擎，在同一数据库内支持 oracle 和 mysql 两种语法引擎。配合多租户能力可以在同一个数据库集群和同一个数据库引擎下同时运行 ORACLE 和 MYSQL 两种数据库租户。

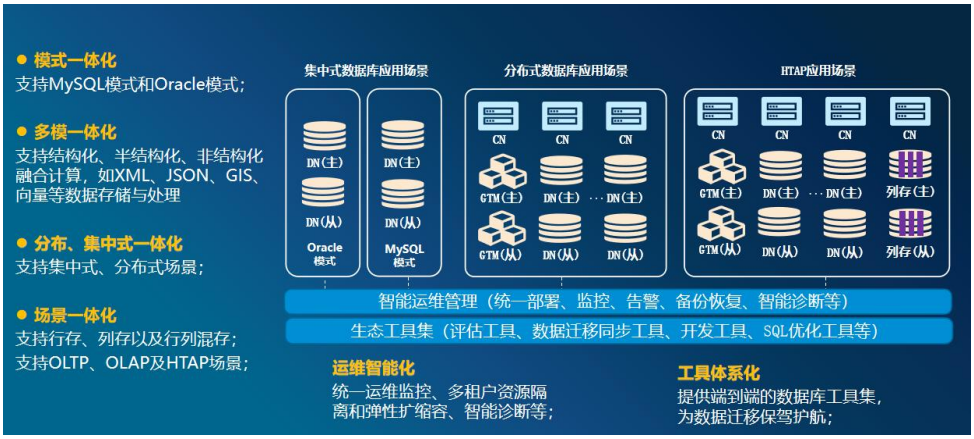


图 3.1-5 GoldenDB 多语法引擎设计

3.2 部署模式

3.2.1 集中式部署

集中式部署指的是单分片租户，且不带 GTM 节点。GoldenDB 的集中式部署模式适用于

数据规模较小，不需要按分片存储的业务。在这种模式下，仅需支持单机事务即可。

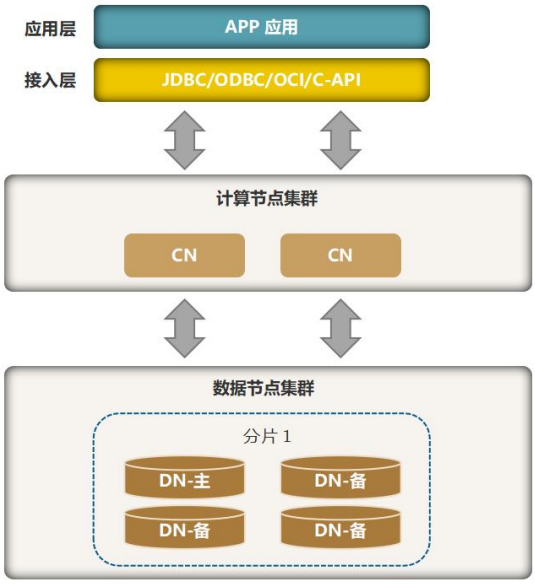


图 3.2-1 集中式部署

3.2.2 分布式部署

分式部署指的是多分片租户，且带 GTM 节点。GoldenDB 的分布式部署模式适用于数据规模较大，单个分片无法满足存储需求的业务。在这种模式下，数据分布在多个分片上，事务可能涉及多个分片，因此支持分布式事务，并通过全局事务节点管理分布式事务。

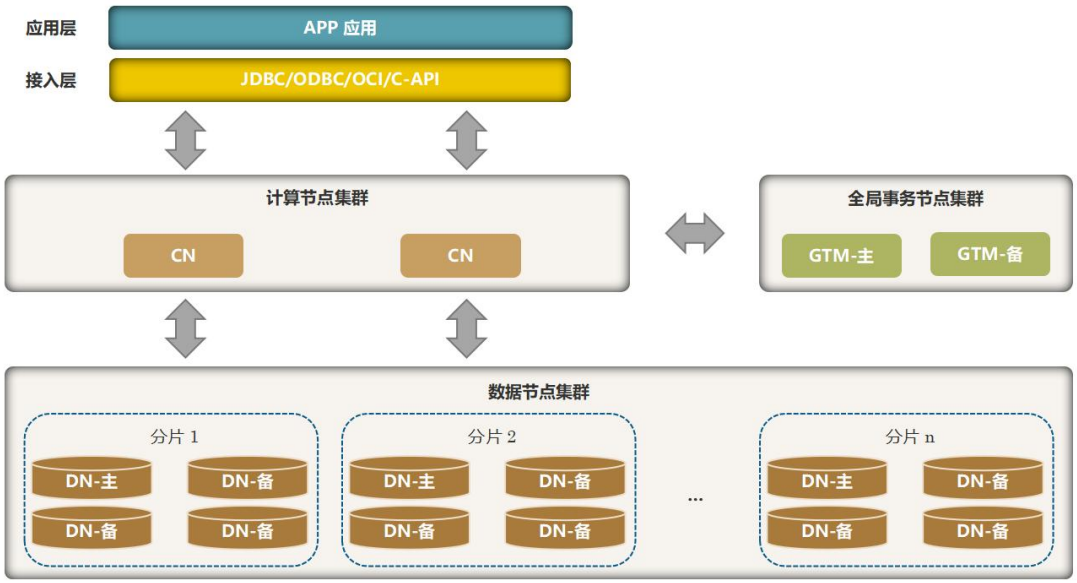


图 3.2-2 分布式部署

3.2.3 混合部署

混合部署指的是既有单分片租户，也有多分片租户。GoldenDB 的集中式+分布式混合部署模式提供了更大的灵活性，适用于不同规模和需求的业务。

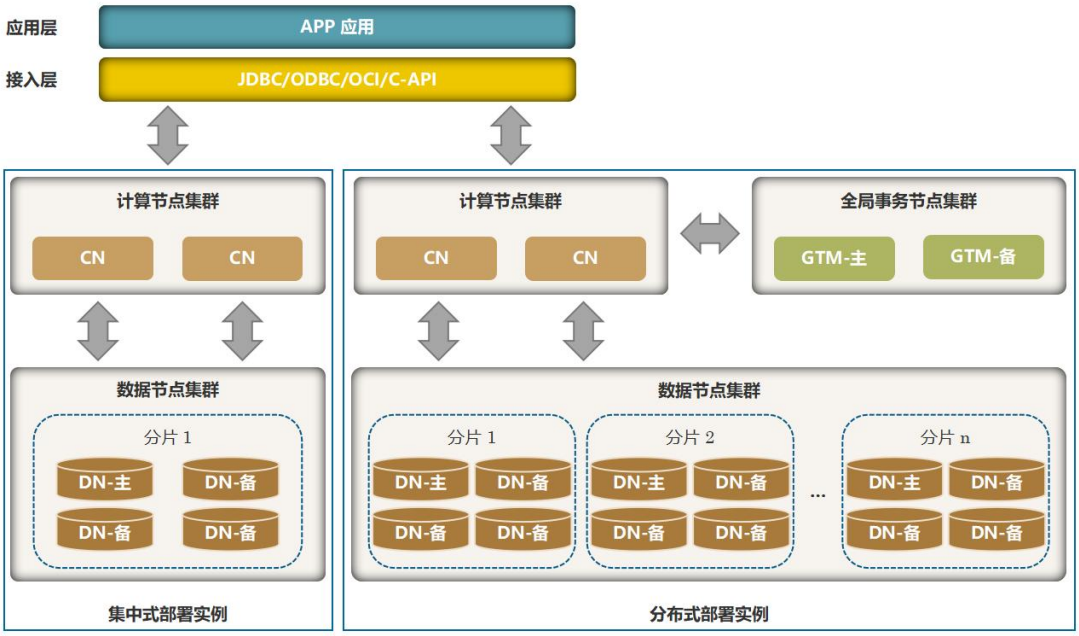


图 3.2-3 混合部署

3.2.4 物理机部署

GoldenDB 部署载体分为物理机部署和容器化部署。物理机部署是指 GoldenDB 管理节点(CM/PM/MDS/InsightServer 等)和非管理节点(CN/DN/GTM/LDS 等)都部署在物理机。GoldenDB 物理机部署的优势在于提供了灵活性、可靠性和性能优势。

- GoldenDB 物理机部署提供了较高的可靠性。由于物理机部署能够将数据库系统与硬件资源进行隔离，一旦出现故障或者需要进行维护，只需对相应的物理机进行操作，而不会影响其他部署在其他物理机上的数据库系统。这种隔离性能够提供更高的可靠性，确保数据库系统的稳定运行。

- GoldenDB 物理机部署还能够提供更好的性能。通过合理配置和管理物理机的硬件资源，可以充分利用处理器、内存和存储等资源，提高数据库系统的运行效率和性能。同时，物理机部署还能够根据实际需求进行横向和纵向扩展，以满足不同规模和负载的数据库管理需求。

综上所述，GoldenDB 物理机部署的优势在于可靠性和性能优势。通过合理配置和管理硬件资源，GoldenDB 能够提供稳定、高效和可靠的数据库管理解决方案，满足不同场景下的需求，并提供优质的数据库服务。

3.2.5 容器化部署

容器化是指 GoldenDB 管理节点(CM/PM/MDS/InsightServer 等)、非管理节点(CN/DN/GTM/LDS 等)都部署在容器中。容器化部署带来了许多优势，使得数据库管理更加灵活、高效和可靠。

- 容器化部署提供了更高的灵活性。GoldenDB 容器可以在不同的环境中轻松部署，无论是在本地服务器、云平台还是私有数据中心。容器化部署使得 GoldenDB 可以与其他容器化应用程序或服务一起运行，并且可以根据需要进行快速扩展或缩减，以适应不同规模和需求的工作负载。
- 容器化部署提供了更高的可移植性。GoldenDB 容器可以在不同的操作系统和硬件平台上运行，而无需重新编写或调整代码。这意味着 GoldenDB 可以轻松地在不同的环境中迁移和部署，不受特定操作系统或硬件的限制，从而提供更大的灵活性和便捷性。
- 容器化部署还提供了更高的效率和资源利用率。GoldenDB 容器可以以轻量级的方式运行，减少了资源的消耗，同时也提供了更快的启动时间和更快的部署速度。容器化部署还可以通过自动化和集中化的管理，简化了 GoldenDB 的部署和维护过程，减少了人工操作的需求，提高了运维效率。

综上所述，容器化部署提供了更高的灵活性和可靠性，同时可以确保安全性，GoldenDB 容器可以通过隔离机制确保应用程序和数据的安全性，避免了因为一个容器的故障或安全问题而影响整个系统。此外，容器化部署还可以实现快速的回滚和恢复，保证数据库服务的高可用性和业务连续性。

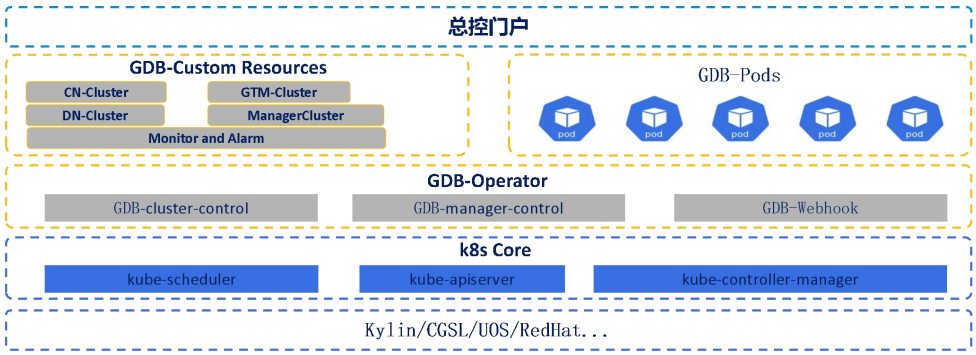


图 3.2-4 容器化部署架构

通过 CRD+Operator 的方式对 K8S 提供的能力进行编排，具备自动部署、切换管理的能力。持久化数据本地存储：不采用云存储服务，保证高 IO 吞吐性能，避免云存储引入数据库性能/可靠性因素，减少私有云部署的环境要求。

容器节点使用固定 IP：由数据库自身多副本技术保证数据高可靠和服务高可用，保持 GoldenDB 基础架构一致，简化私有云部署的环境要求。

3.3 可靠性方案设计

GoldenDB 的数据节点、计算节点、全局事务节点以及管理节点，都具备容忍硬件包括计算、存储、管理、软件进程、网络等方面的单点故障，可保证系统在高负载压力下容错各类异常场景，在任意节点发生故障时，数据库具备自动切换并保证数据不丢失，故障自愈时间小于 8 秒（RTO<8S，RPO=0）。

GoldenDB 支持单数据中心高可用、同城三中心、同城双中心、两地三中心、多地多中心等容灾架构模式。GoldenDB 基于多副本冗余技术、一致性复制技术和灵活高可用策略实现多种容灾方案，集群中任意小于副本半数的数据损坏不会导致用户数据丢失，不会影响系统其它副本的数据访问。单台服务器故障自动切换到本数据中心其他机器的副本上；双中心以上的机房故障时，可快速切换到其他机房或者城市灾难中心，最大程度保证业务连续性。

3.3.1 单管理节点

单个 AZ 且管理节点数量 1。这种部署模式有着简单的组网结构，适合一些小规模或者测试环境的应用。优点：组网简单，只需要在单个可用区中部署一个管理节点，即可满足基本的数据库需求。缺点：由于只有一个管理节点，一旦该节点发生故障，服务将会中断，无

法提供连续的服务。

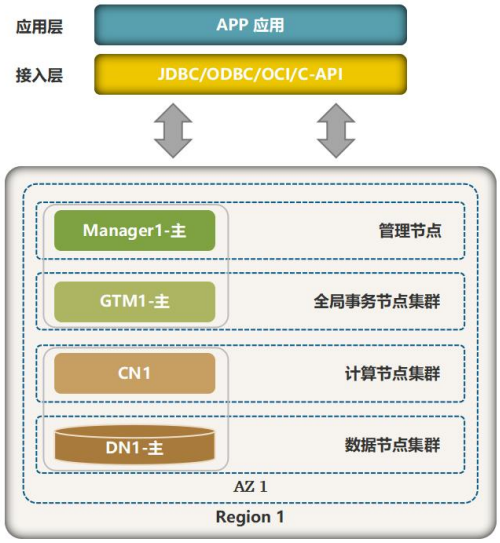


图 3.3-1 单管理节点组网图

3.3.2 本地单中心高可用

单个 AZ 且管理节点数量大于 1。这种部署模式可以提供更高的可用性，适用于对高可用性有一定要求的应用场景。优点：可以实现高可用性，通过在同一个可用区内部署多个管理节点，当其中一个节点发生故障时，其他节点可以接管服务，保证数据库的连续性和可用性。这样可以减少单点故障带来的影响，提高应用的可靠性。缺点：由于所有的管理节点都部署在同一个可用区内，一旦整个可用区发生故障，所有的管理节点都会受到影响，导致服务无法提供。这样就会对应用的可用性产生风险，特别是在面临自然灾害或者硬件故障等情况下。

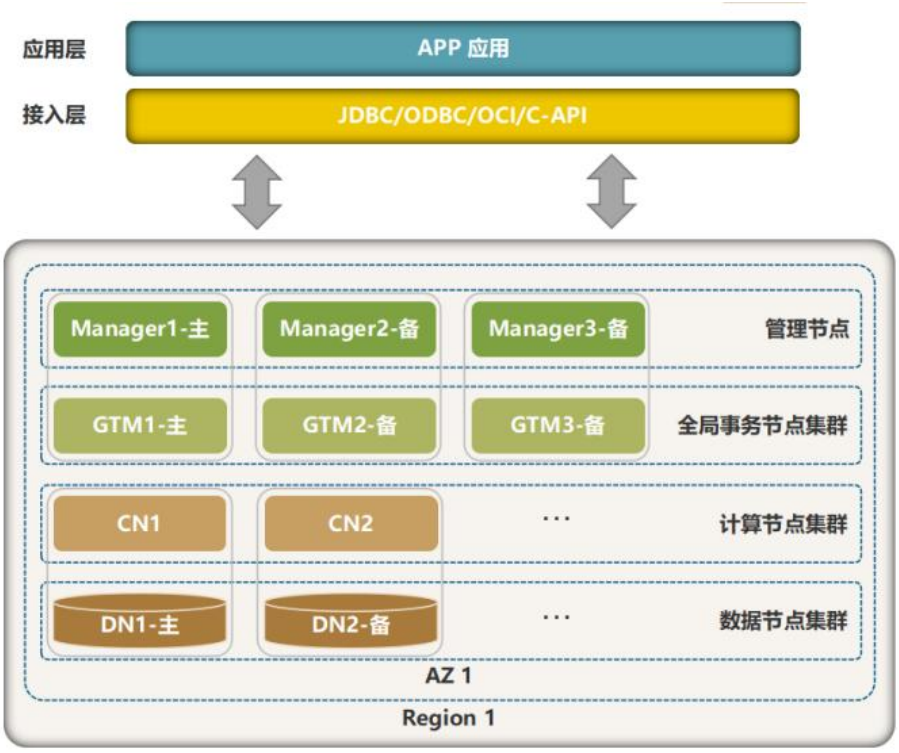


图 3.3-2 本地单中心高可用组网图

3.3.3 同城三中心

同城 3AZ 且管理节点数量大于 1，分布在同一城市。这种部署模式可以提供更高的可用性，适用于对高可用性有较高要求的应用场景。优点：可以实现更高的可用性。通过将管理节点分布在不同的可用区中，当其中一个可用区发生故障时，其他可用区的节点可以接管服务，保证数据库的连续性和可用性。这样可以有效降低单点故障的风险，提高应用的可靠性。缺点：由于所有的管理节点都部署在同一城市内，一旦整个城市的可用区由于自然灾害或者硬件发生故障，所有的管理节点都会受到影响，导致服务无法提供。

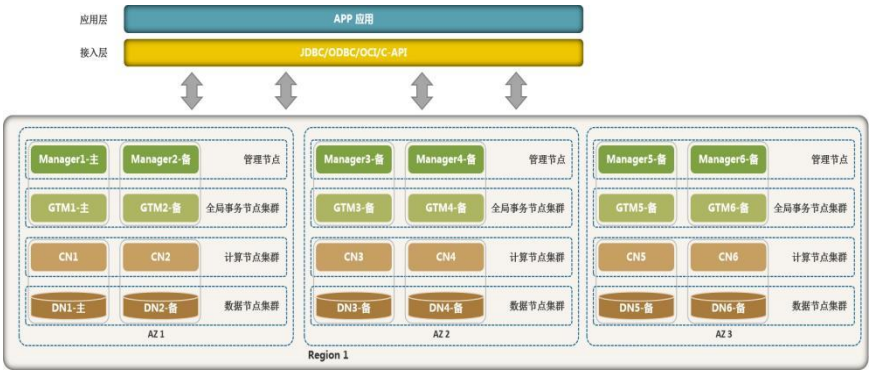


图 3.3-3 同城三中心高可用组网图

3.3.4 本地同城双中心高可用

双 AZ 且管理节点数量大于 1，分布在同一城市。这种部署模式可以提供更高的可用性，适用于对高可用性有较高要求的应用场景。优点：可以实现更高的可用性。通过将管理节点分布在不同的可用区中，当其中一个可用区发生故障时，其他可用区的节点可以接管服务，保证数据库的连续性和可用性。这样可以有效降低单点故障的风险，提高应用的可靠性。缺点：由于所有的管理节点都部署在同一城市内，一旦整个城市的可用区由于自然灾害或者硬件发生故障，所有的管理节点都会受到影响，导致服务无法提供。

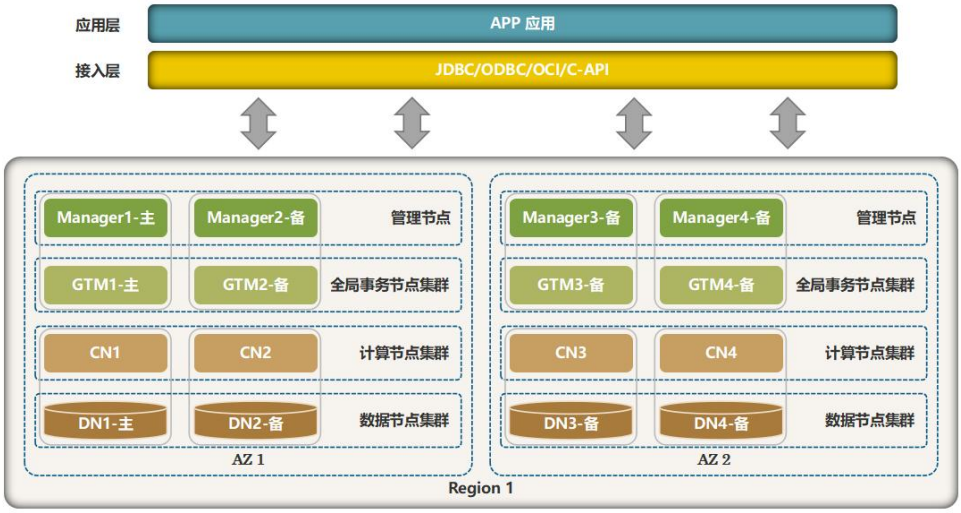


图 3.3-4 本地同城双中心高可用组网图

3.3.5 两地三中心高可用

3 个 AZ 且管理节点数量大于 1，分布在两个不同的城市。这种部署方式具有高可用性的优点，即使其中一个城市或可用区发生故障，系统仍然可以正常运行。然而，这种部署方式需要更多的资源，包括服务器和网络设备等，以支持多个节点的运行。因此，需要仔细考虑资源投入和成本效益之间的平衡。同时，还需要确保两个城市之间的网络连接稳定和可靠，以确保数据同步和节点之间的通信正常运行。

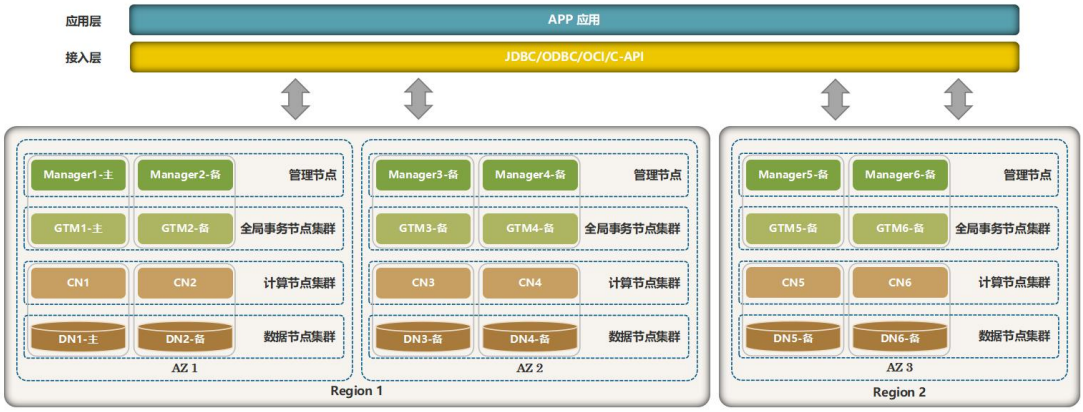


图 3.3-5 两地三中心高可用组网图

3.3.6 两地四中心高可用

4 个 AZ 且管理节点数量大于 1，分布在两个不同的城市。可以提供高可用性，即使一个城市或可用区发生故障，仍可以确保数据库的正常运行。然而，这种部署方式需要更多的资源，因为需要维护多个管理节点，并且需要在不同的城市之间进行数据同步和通信。尽管有这些缺点，但高可用性的好处通常是值得的，特别是对于对数据库可用性要求较高的应用场景。

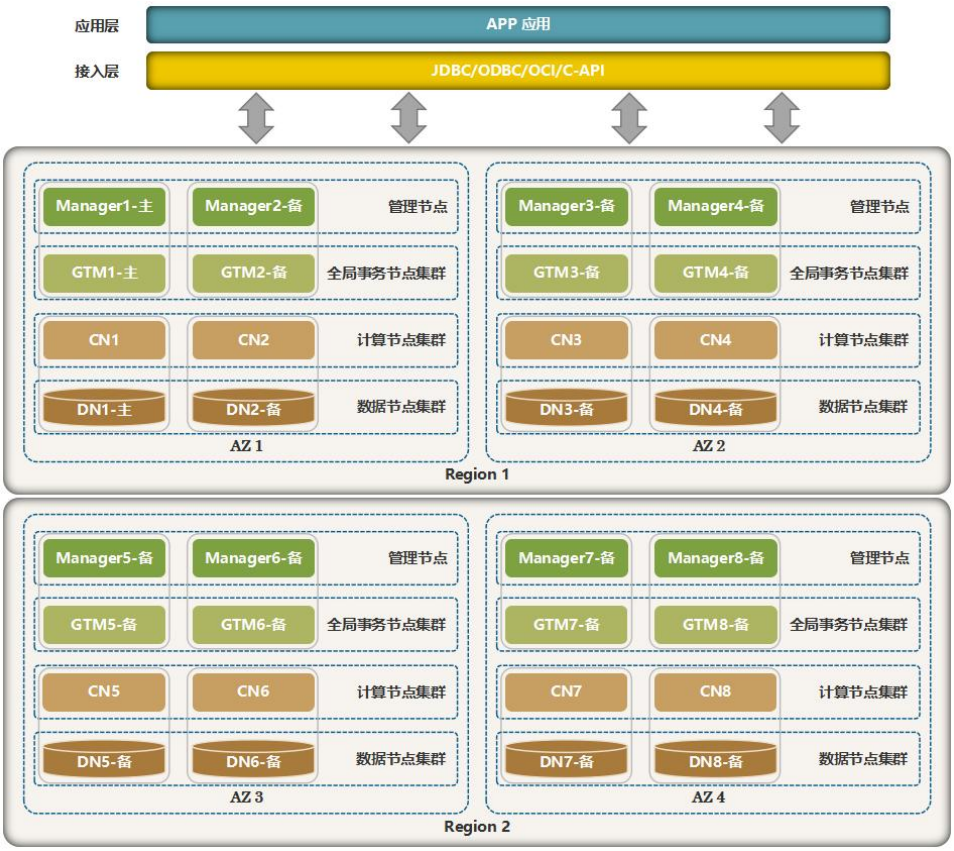


图 3.3-6 两地四中心高可用组网图

3.3.7 三地五中心高可用

这种部署意味着在三个不同的地理位置建立五个数据中心来确保数据库系统的高可用性。这种部署架构在一个地区或数据中心出现故障时，仍能保持数据库系统的正常运行，从而确保业务的连续性。每个数据中心都有多个冗余节点，包括管理节点、数据库节点以及存储节点。这种多重冗余性可以降低系统发生单点故障的风险。数据中心之间通过数据同步和复制机制实现数据的异地备份。这确保了即使某个地区或数据中心发生灾难，仍能够恢复数据并保持业务的连续性。尽管需要更多的资源来维护多个数据中心和节点，但通过合理的资源管理和负载均衡策略，可以最大限度地利用资源，并确保系统的性能和可用性。

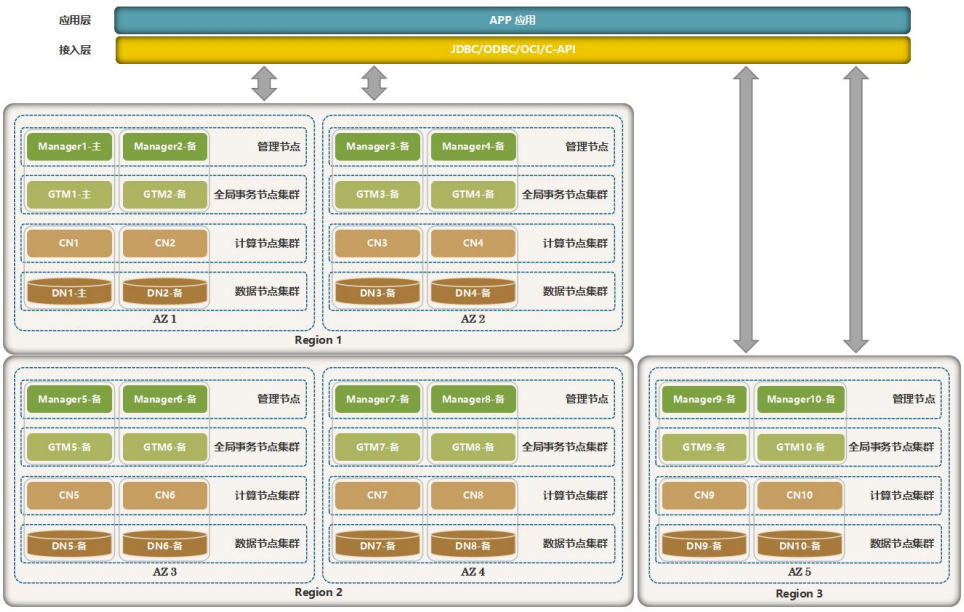


图 3.3-7 三地五中心高可用组网图

4 功能介绍

GoldenDB 是一款适用于 OLTP 场景，同时兼顾 OLAP 的关系型数据库。

GoldenDB 兼容 SQL92、SQL99、SQL2003 标准，同时兼容常用 Oracle、DB2、MySQL 语法，其支持的语法的类别全面覆盖数据库语法类别，包括 DDL 数据定义语言、DML 数据操纵语言、DQL 数据查询语言、DCL 数据控制语言、TCL 事务控制语言、常用函数、程序控制语言、批处理语法、序列等。同时，GoldenDB 对跨节点的复杂 SQL 操作支持全面，使得业务人员的开发工作量大幅降低，无需考虑大量的 SQL 改造。

数据库集群中节点数目和种类很多，为了方便管理，GoldenDB 各项产品化功能完善，

统一运维管理方便用户进行数据库集群的操作管理。

4.1 分布式事务控制

GoldenDB 作为事务型数据库，支持集群所有节点间事务的 ACID 特性，保证故障可恢复，以及恢复后满足数据的 ACID 要求，并负责节点的并发控制。

4.1.1 隔离级别

为保证在数据一致性前提下实现高并发访问，GoldenDB 数据库支持事务隐式启动、默认非自动提交 auto_committed=0、在事务中设置保存点 savepoint，控制不同粒度的数据对象的并发访问控制。

GoldenDB 支持读未提交（Read Uncommitted）、读已提交（Read Committed）、可重复读（Repeatable Read）、串行化（Serializable）的隔离级别，推荐使用 Read Committed 的隔离级别。

4.1.2 分布式事务控制

数据库的数据一致性解决方案可分为强一致性、弱一致性和最终一致性。GoldenDB 实现了强一致性解决方案，即当用户完成数据更新后，后续所有读操作都能且只能读到更新以后的值。

GoldenDB 分布式事务的核心思想是引入全局事务管理器(GTM：Global Transaction Manager)，记录当前所有正在执行的全局事务标识(GTID)及其状态,作为分布式事务的全局标识，通过全局事务机制实现全局事务的隔离性，并协助实现全局事务的原子性。GoldenDB 分布式事务是借鉴单机事务实现，与单机事务的区别在于 GoldenDB 会将所有的表中增加 GTID 隐含列，该列在分布式事务中存放该全局事务 ID (每个分布式写事务会都得到一个全局唯一且递增的整形值),在执行完所有 SQL 后,所有 DN 节点统一进行 COMMIT。如果有 DN 节点 COMMIT 提交失败，所有节点的 DBAgent 通过解析 binlog 中是否含有该异常事务的全局事务 ID 来构造反向回滚语句，并向 DN 下发该反向回滚语句，进行事务回滚，从而保证分布式事务的数据原子性。故 GoldenDB 分布式事务是具有 ACID 特性的。

图 4.1-1 GoldenDB 分布式事务控制关键设计

计算节点作为分布式事务协调者基于改进的两阶段提交事务控制方法，协调多个资源域内的本地事务完成整个分布式事务控制，同时将全局事务状态实时记录在全局事务管理器中，从而保证分布式事务在跨资源域上达成全局的一致性。

- 全局事务管理器为每一个分布式写事务分配一个全局唯一的全局事务 ID，并根据事务的存活情况维护对应 ID 的生命周期。
- 全局事务 ID 加到每条数据记录中，更新数据时同时更新全局事务 ID，将其作为分布式环境下表的全局锁。

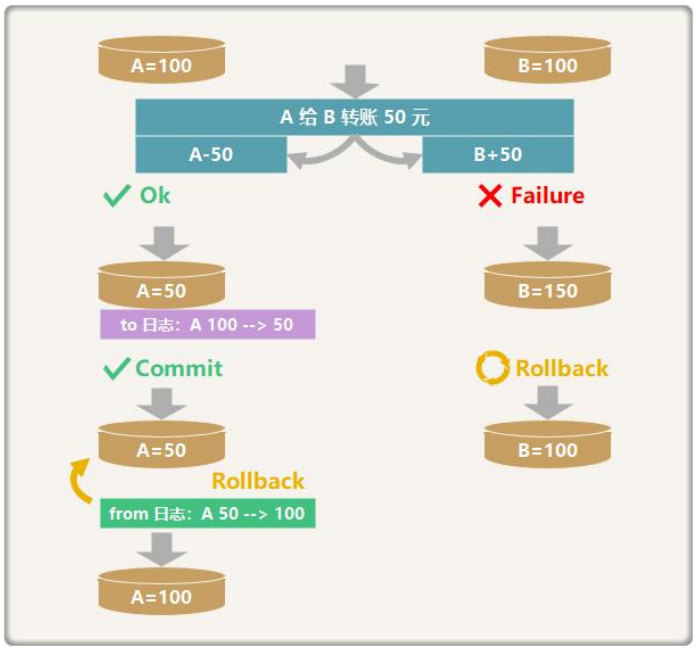


图 4.1-2 转账交易示意图

通过已提交事务回滚，解决分布式一致性问题。以转账交易为例，如图 4-1，假设 A、B 账户原来各自拥有 100 元，A 给 B 转账 50 元。A、B 属于不同节点，各自执行对应更新事务，

A 执行成功，B 执行失败。B 失败后，进入单机事务异常回滚流程，A 成功后，进入已提交事务回滚流程，保证了分布式事务的一致性。

另外，为了提升效率，减少计算节点和全局事务管理器交互次数，将申请单个全局事务 ID 优化为批量申请全局事务 ID，释放单个全局事务 ID 优化为批量释放全局事务 ID。

为了避免全局事务管理器异常导致数据库不可用，全局事务管理器采用主备架构，主出现异常时，自动切换到备上继续提供服务。

为了降低全局事务管理器异常影响到数据库所有租户的风险，全局事务管理器扩展为全局事务管理器集群架构，多个全局事务管理器和多个租户绑定，某个全局事务管理器出现异常后发生切换，那么切换过程中影响的也仅仅是故障全局事务管理器绑定的租户，其它租户依旧可以正常运行。

4.2 基础功能

GoldenDB 支持通用数据库标准协议，支持 JDBC、ODBC、OCI、Native CAPI、.NET、Pro*C 等常见开发语言所使用的接口驱动，包括 C、C++、Java、.Net、Python 等，支持 JDBC、ODBC、OCI 内置负载均衡能力。

GoldenDB 兼容 SQL92、SQL99、SQL2003 标准语法，兼容 MySQL 语法，兼容常用 Oracle、DB2 语法，支持视图、序列、同义词、函数、大对象、索引、存储过程、包和包体、触发器、分区等功能，帮助用户从 Oracle、DB2、MySQL 等数据库快速地迁移到 GoldenDB 数据库，节省应用改造的成本，提升迁移效率。

4.2.1 表存储组织

GoldenDB 支持堆表（HEAP TABLE）和索引组织表（INDEX ORGANIZED TABLE）来适配更多的业务场景。可以通过 organization 全局参数指定创建表组织方式为堆表或索引组织表。

4.2.2 数据类型

GoldenDB 支持日常应用所能涉及的各种数据类型，可以满足企业数据管理的需要，部分数据类型如下：

- 数值类型：BOOL、BOOLEAN、BIT、TINYINT、SAMLLINT、MEDIUMINT、INT、INTERGER、BIGINT、DECIMAL、DEC、NUMERIC、FLOAT、DOUBLE、DOUBLE PRECISION。
- 字符类型：CHAR、VARCHAR、BINARY、VARBINARY、TINYBLOB、TINYTEXT、BLOB、TEXT、MEDIUMTEXT、LONGBLOB、LONGTEXT、ENUM、SET。
- 日期类型：DATE、DATETIME、TIMESTAMP、TIME、YEAR。
- JSON 数据类型。
- 空间数据类型。

4.2.3 常见数据对象

GoldenDB 数据库支持常见的数据库对象：

- 视图：GoldenDB 中视图不会保存数据，是虚拟的表，建立在已有的基础表之上，数据保存在基础表中，视图可以查看基础表中的数据，部分视图还可以添加更新和删除，对应基础表中的数据也会发生变化。支持视图的创建、删除、查看和修改操作。
- 同义词：GoldenDB 支持同义词，用于简化对其他数据库对象的引用，并提供更灵活的数据库设计。使用同义词来隐藏数据库对象的实际名称和位置，以减少代码中的硬编码，从而提高代码的可维护性和可移植性。谨慎使用同义词，避免引起混淆或命名冲突，确保同义词的命名清晰且易于理解。GoldenDB 支持表、视图等数据对象同义词。
- 序列：GoldenDB 支持序列（SEQUENCE）,序列全局唯一，在 GTM 节点上进行维护，提供全局序列的创建、修改、申请。
- LOB 大对象：支持 BLOB/CLOB 等大对象的创建

4.2.4 数据分片功能

GoldenDB 支持数据的水平和垂直分片技术，通过分片技术可以控制数据在集群内的分布形态，适配业务逻辑，提升系统性能，满足企业数据管理的需要，相关能力：

- 哈希策略：适用于将数据均匀的分布到预先定义的各安全组上，保证各安全组的数据量大致一致。一般用于不需要关心分发字段的取值范围和具体含义，且对该表的 SQL 操作基本都是等值操作的场景。
- 范围策略：适用于指定一个给定的列值或列值集合应该保存在哪个安全组上，常用

于时间、日期、数值等类型的字段上，如数据按照自然月或自然天分布存储。

- 列表策略：适用于含有一系列限定性或枚举性的字段上，如数据按照机构代码、国家代码、地区代码分布存储。
- 复制策略：适用于不常修改，且频繁出现在关联或子查询中的小表。复制策略下，复制表的数据保存在每一个节点，因此所有节点都需要这个表中数据的情况下，可减少节点间网络数据的传输，提高这种查询的性能。
- 多级分片：支持 5 层的分片规则设置，配合表的二级分区能力，支持最多 7 层的数据分片分区能力，适用于需要精细化控制数据在集群中的分布形态的场景，如多法人场景，将不同法人的数据划分到不同数据安全组。
- 分区和分片键支持持主键、唯一索引、非唯一索引、全局分区索引、普通字段等多类型，索引可创建在表中的任意列，该列可以是分片列也可以是非分片列。
- 支持更换分片和分区键、UPDATE 分片和分区键。可以在线调整分片键，通过在线重分布功能，可以将表的分发策略调整一致，提升表关联性，降低跨节点通信，提升关联查询效率，达到优化分发策略以及故障恢复能力。

4.2.5 函数功能

GoldenDB 支持的函数包括控制流函数、字符串函数、数学函数、日期和时间函数、转换函数等，部分函数如下：

- 控制流函数包括 CASE WHEN 控制、IF（expr1,expr2,expr3）、IFNULL(expr1,expr2)等。
- 字符串函数包括 CHAR、CHAR_LENGTH、COALESCE、CONCAT、CONVERT、IN、INSTR、LENGTH、LOWER、REPLACE、LTRIM、RTRIM、TRIM、UPPER、LEFT、RIGHT 等。
- 数学函数包括 ABS、CEIL/CEILING、FLOOR、ROUND、MOD、POW 等。
- 日期和时间函数包括 DATE、DATE_ADD、DATE_FORMAT、DATE_SUB、DATEDIFF、DAY、DAYNAME、DAYOFMONTH、DAYOFWEEK、DAYOFYEAR、CURDATE、CURRENT_DATE、CURRENT_TIMESTAMP、CURTIME、LAST_DAY、MINUTE、MONTH、NOW、TIME、TIMEDIFF、TIMESTAMPADD、TIMESTAMPDIFF、TO_DAYS、TO_SECONDS、WEEK、YEAR 等。
- 转换函数包括 CAST、CONVERT 等。
- 聚合函数包括 AVG、COUNT、MIN、MAX、SUM 等。
- 位运算函数包括 &（按位与）、|（按位或）、!（按位非）、^（按位异或）<<（按

位左移）、>>（按位右移）等。

4.2.6 索引机制

GoldenDB 支持多种索引类型，包括唯一索引、复合索引、函数索引、全局唯一索引以及全局二级分区索引等，通过使用索引可以快速定位数据，提高查询性能。GoldenDB 支持全局索引，在分布式数据库中，要在多个分片间实现索引功能，需要建立分布式全局索引。全局索引能够快速定位记录所在的分片信息，根据分片信息下发查询语句到正确的分片上，然后利用单数据节点索引进行数据快速查询。创建索引为 online 操作，创建索引时不会阻塞 insert、update、delete 等 DML 语句并发执行，不影响在线业务的执行。

支持全局二级分区索引，可以对分布式的分区表在不依赖主键和分区键的情况下创建全局唯一索引和全局二级分区索引。实现非分区键 SQL 性能提升，对业务友好。对分区进行 truncate 和 drop 操作时，全局（唯一）索引以及全局分区二级索引索引不失效，保持索引有效，数据库会自动维护统计信息。

GoldenDB 支持索引跳跃扫描技术（SKIP SCAN），可以绕过低选择性前导列直接利用高效的非前导列过滤，但需严格满足前导列基数低、非前导列选择性高的条件合理使用时，可显著减少 I/O 负载，反之则可能引发性能风险。

4.2.7 存储过程

GoldenDB 支持存储过程。存储过程是在数据库中定义一些 SQL 语句的集合，将特定功能的 SQL 语句集通过存储过程存储在数据库中，用户通过指定存储过程的名字并给出参数来执行即可，方便 SQL 语句调用，提升语句执行效率，同时通过限制存储过程权限保证数据安全，存储过程既可以有返回值，也可以没有返回值，主要用于执行操作。GoldenDB 的存储过程是在整个集群上执行，存储过程定义和解析放在计算节点上，由计算节点统筹判断存储过程体中的语句是否跨数据分片，是否需要在计算节点处理数据，是否可以将语句下压到存储节点执行。

4.2.8 触发器功能

GoldenDB 支持触发器功能。

- 触发时间支持操作前触发（BEFORE）与操作后（AFTER）触发。
- 触发事件支持表的插入（INSERT）触发，表的更新（UPDATE）触发，表的删除（DELETE）

触发等。

4.2.9 视图功能

GoldenDB 支持视图功能，视图保存复杂查询，简化用户操作，支持单表和多表视图。GoldenDB 支持创建视图、删除视图、修改视图等视图基本操作。

GoldenDB 支持物化视图功能，物化视图是数据库中的一种特殊视图。普通视图仅保存查询语句,物化视图将查询的结果存储在数据库中。物化视图可以显著提高复杂查询的性能，避免了每次查询时需要重新计算结果。

支持实时刷新物化视图，物化视图可以支持基于外部表、普通视图和物化 视图创建。

4.2.10 分区功能

GoldenDB 支持在分片的基础上再进行分区操作，支持 RANGE、LIST、HASH 等多种分区策略，支持以上 3 种策略的组合的 9 种两级分区模式。分区表按照哈希算法进行分区，将数据均匀地分布在多个分区中，从而实现并行查询，查询操作只需要处理特定的分区，从而减少扫描的数据量，提高查询效率。分区表也可以按照时间范围进行分区，每个分区包含特定时间段内的数据，这使得按时间范围查询变得更高效。支持创建分区表，在线新增、删除、修改分区类型（如 range→list）、支持手工分区，支持分区自动分裂等分区操作，支持单表 online 修改为分区表，支持 SUBPARTITION TEMPLATE 分区模板。

4.2.11 SQL 绑定

GoldenDB 支持 SQL 绑定变量功能，实现 SQL 预解析，减少硬解析，降低锁存争抢。实现执行计划复用，减少重复解析开销，提升性能、安全性和可维护性。

4.2.12 执行计划

GoldenDB 支持查看数据查询、数据插入、数据更新和数据删除等操作的执行计划，并且可以根据不同的需要输出不同的内容，可以仅输出计算节点的执行计划，也可以仅输出数据节点的执行计划、还可以同时输出计算节点和数据节点的执行计划等。通过查看执行计划可以了解 SQL 的执行顺序、执行瓶颈和进行辅助优化。

支持 explian 指令查看和分析对应 SQL 语句的执行计划，支持 hint 执行计划功能，支

持 outline 执行计划功能（Oracle 的 sql profile），可以实现绑定执行计划，保证 SQL 语句在各种场景下执行计划按照最优的方式执行。

4.2.13 外部表

GoldenDB 支持外部表功能，外部表支持创建分区，可直接在 HDFS、OSS/S3 等存储上创建 CSV、Parquet、ORC 格式的外部表，支持数据导入指定分区，支持 Roaring bitmap、Array、Map 数据类型。

4.2.14 临时表

临时表查询是指在数据库中对已创建的临时表进行数据检索、更新、删除等操作。临时表查询与普通表的查询类似，允许用户对临时存储的数据进行各种 SQL 操作，以满足特定的业务需求或数据处理需求。GoldenDB 支持事务级和会话级的临时表，可用于存储处理中间结果。会话级的临时表会在会话结束后自动清理数据；事务级则是在事务结束后自动清理临时表数据。

4.2.15 字符集

GoldenDB 当前支持 latin1、gbk、utf8、utf8mb4 和 gb18030-2022 等字符集。

4.2.16 国产化适配

GoldenDB 具备全栈国产化适配能力，完成适配包括但不限于：

- **CPU**：鲲鹏、飞腾、海光、兆芯、龙芯、申威、珠峰等。
- **操作系统**：麒麟、统信、新支点、凝思、深度、龙蜥、BC-Linux 等。
- **服务器**：中兴、浪潮、曙光、中科可控、超聚变、新华三、华鲲振宇、宝德、联想、五舟、鲲泰、四川虹信、河南昆仑、黄河、武汉长江、神码杭研、同方、湖南湘江、中科、紫光等。
- **同步备份工具**：英方、迪思杰、鼎甲科技、爱数科技、安钛飞、云信达、华为 OceanProtect、中电长城、航天壹进制、中安数联等。
- **中间件**：宝兰德、东方通、中创、普元、金蝶等。
- **业务**：华为、亚信、思特奇、新大陆、新炬、神州泰岳、浩鲸等。
- **云化**：移动云（含磐基 PaaS 平台）、华为云、阿里云、腾讯云、天宫云、天翼云、金山云等。

- 运维管理平台：新炬网络、云和恩墨、新数科技、中国移动 BOMC、溪数科技、上海上讯等。

4.3 高级功能

4.3.1 扩展性

GoldenDB 的架构设计使得其具备高良好的扩展性：

- 计算节点集群是无状态的，可以动态任意扩展。
- 数据库集群可以根据需要进行如下几个维度的扩展：

– 在安全组内部添加数据节点或增加安全组数目，增加数据副本数。可以获得更高的可靠性，同时也可以通过读写分离功能达到读能力的线性扩展；

– 增加分片。获得数据节点存储和计算能力的线性扩展，在增加数据的分片时需要考虑如何将数据重新迁移到新分片上，即表数据的重分布，见下一节描述；

– 增加数据节点集群数目，扩展组建新的子集群。可以在处理能力扩展的同时做到不同业务间数据的隔离性，用户可以根据要求选择。

GoldenDB 计算节点和数据分片的数目配比推荐为 1:1，性能可以随计算节点和分片数目的增加线性扩展，损耗控制在 5%内。

若业务为 CPU 密集型，则计算节点数目要适当增加；若业务为 IO 密集型，计算节点数目可以适当减少。

4.3.2 MVCC

MVCC(Multi Version Concurrency Control 的简称)，即多版本并发控制，提供高度并发的数据访问方式，允许用户读取其所应该看见的版本。该技术最大的优势：读不加锁，读写不冲突。在读多写少的 OLTP 应用中，可以极大的提升系统吞吐量，读写不冲突。

GoldenDB 支持分布式的 MVCC 功能，在分布式场景下，SELECT 语句一致性读的整体过程：

1. 数据是全局可见，直接查询当前数据；
2. 数据非全局可见，查看前像数据，如果前像数据已经全局提交，返回该数据；
3. 数据非全局可见，查看前像数据，如果前像数据未全局提交（下发 select 时候的全局快照），继续查找前像数据；

将全局活跃事务列表带到 DB 层进行可见性判断。可见性判断规则：

- 使用 proxy 上收到的 active_gtm_gtid_list 进行数据活跃判断；
- 如果当前数据行上的 GTID 判断不活跃，直接返回数据；
- 如果当前数据行上的 GTID 判断活跃，通过 undo 构造全局的前像数据。

4.3.3 数据重分布

业务的增长不可避免的需要对资源进行扩容，由于使用了分片技术，数据被切分成细小的分片分布在数据节点集群中。集群扩容后，原有的数据分片就面临着被打散重新分配的过程，这个过程就是数据重分布。

GoldenDB 数据重分布有多种数据应用场景，如当业务系统需要根据业务场景进行扩容和缩容时，可以使用重分布技术，在线进行扩容和缩容，在扩容和缩容的过程中，数据自动进行迁移，业务不中断。如当关联 SQL 语句执行效率低的原因是关联表数据需要在在计算层进行计算时，可以将关联表的分发策略和分发字段调整一致，表的重分布策略调整可以使用数据重分布，重分布过程不影响原有业务系统性能，不阻塞业务。

GoldenDB 采用后台数据迁移和日志回放相结合的方法，通过智能过滤，减少数据迁移量，并通过多表并发提升迁移效率，通过极限逼近和循环回放技术，减少新旧数据切换时间，实现不停机的在线数据重分布，达到数据库扩容、缩容对业务无感知。GoldenDB 的数据重分布对在线业务影响小、且可操作性强，具备如下特点：

- **执行时间可控制，对业务影响秒级**

从前面的原理介绍中可以看出，重分布过程中数据的迁出迁入是个循序渐进的过程，仅在原分片和目标分片数据存在少量差距时才会禁写，因此对在线业务影响秒级，且不影响读服务。

- **支持多表并发重分布**

在分布式数据库的数据模型中，会将有关联关系的表采用相同的分片策略。在数据重分布过程中，这些相关表可以选择同步进行处理，以避免在数据重分布过程中由于相关表的分片策略临时不同而导致该 SQL 语句无法直接下压，影响 SQL 执行性能。

- **重分布数据智能识别**

数据重分布过程中会涉及到数据从源分片至目的分片的移动，开销很可观，GoldenDB 支持对表的源分片信息和目标分片信息进行详尽的分析比对，最大程度的减少数据在分片间的移动，提升重分布效率。

- **可操作性好**

GoldenDB 在统一运维界面 Insight 上提供重分布操作界面，全程可视化控制，包括执行、暂停、继续、取消、异常情况下的重试等。

- **产品通用性好**

对比需要预先做好分片规划的预 Sharding 方案，GoldenDB 的重分布方案对设计人员的要求相对较低，GoldenDB 支持任意分片策略间的重分布，彻底与业务模型解耦，无需在系统设计初期就要精确规划好未来的数据分布情况。

4.3.4 读写分离

读写分离是指利用数据节点集群安全组多副本，将部分读请求发往备用节点，提升系统的读能力。

在启动读写分离时，GoldenDB 的计算节点在收到应用 SQL 请求时，根据当前的语句类型和负载策略选择 SQL 下发的数据节点，将写操作发往主节点，将读操作发往备用数据节点。

在多个应用接入一个数据节点集群时，为了满足不同应用的需求，GoldenDB 支持对同一集群不同的服务端口，设置不同的读写分离模式，包括以下几种：

- 本区域/同区域查询

读操作在本区域和同区域的主、备机间根据配置的权重做读负载均衡；

- 异区域查询

读操作仅在异区域备机间做读负载均衡；

- AZ 级查询

读操作在区域间根据配置的权重做读负载均衡；

- 最新副本

读操作在指定机房的最新副本间做读负载均衡。

除了上述服务端口级别的读写分离模式设置，GoldenDB 还支持：

- **SQL 语句级别的读写分离**

应用可以在 SQL 语句后面添加 hint 信息强制发往主或备用节点，SQL 级别的优先级高于服务端口级别。

- **事务级读写分离**

只读事务内的第一条读语句根据读写分离配置的规则选取备机，后续读语句绑定该备机执行；

- **会话（连接）级读写分离**

当前会话设置为只读后，事务内只支持读语句，并根据事务级读写分离规则做读负载均衡。

常见的使用场景如下：

- 由于数据在安全组内部的节点间同步存在时延，因此对实时性要求较高的 SQL 请求，应用希望将其发往主节点；
- 对于一些 SQL 如分析聚合类 SQL，应用希望将其发往备节点，减少对主节点的影响。

结合上述服务端口和 SQL 级别的读写分离模式设置，应用可以根据自己的希望设计出合理的读写分离策略。

4.3.5 导入导出

数据导入导出一般用于系统间的数据迁移，小到导入导出一部分数据，大到数据割接、数据库迁移式升级、数据的分库、海量数据的迁移等，使用场景十分丰富，是数据库常用功能。

GoldenDB 支持将集群中符合查询条件的记录导出到指定的数据文件中，也支持将外部数据文件导入至集群合适的数据分片中。

导入导出功能介绍如下：

- 外部接口。

GoldenDB 导入导出的外部接口为文件，待导入文件的格式要符合要求，一般为字段间用分隔符隔离的文本文件。导出结果也同样为文本文件。

- 工具插件。

导入导出可以认为是系统的一个功能插件，部署灵活，与在线交易流分离，可以降低对联机交易影响。

- 原始文件自动导入数据库集群。

GoldenDB 的导入导出工具，能自动根据导入表的分片规则，将数据下发至对应数据节点，最终完成导入。其中的各环节均支持并发处理，提升整体导入效率。工具也支持带条件导入，即仅将满足条件的记录导入至系统中。

- 多分片数据统一出口导出。

GoldenDB 的导入导出工具，能自动控制各个分片并发进行数据导出，并可选择将导出文件上传至同一服务器；进一步，也可以将多个导出文件汇总成单个大文件，用户可以灵活选择控制导出过程。

- 容错处理。

导入操作耗时长，对系统资源占用多，且通常是将外部系统的数据文件导入到分布式系统中，而外部的文件往往会有异常数据。因此导入操作经常会遇到错误，GoldenDB 充分考虑导入异常的错误处理。

– 举例如下：

– 导入过程中遇到异常数据时，能将异常数据分拣出来，待后续处理。

– 流程处理的各阶段均有重试机制，最大程度保证导入成功。

– 系统异常时，能从异常时刻位置继续执行导入流程。

– 支持对原始文件记录末尾的空值进行处理。

4.3.6 Oracle 兼容性

GoldenDB 兼容常用 ORACLE 语法，大大降低业务从 Oracle 迁移至 GoldenDB 的工作量。

- **数据对象兼容**

– 支持序列 SEQUENCE 的创建、删除、获取；

– 支持 DBLINK 的创建、删除、使用；支持远程访问 Oracle、MySQL、Postgres 等其他异构数据库；支持远程访问同构数据库 GoldeDB。

– 支持 BEFORE、AFTER、INSTEAD-OF 等触发器的创建、删除、触发；

- 支持包 Packages、自定义函数、存储过程的创建、删除、调用等；
- 支持 REF Data Types、VARRAYS 等自定义类型的创建、删除、使用；

- **SQL 语法兼容**

- 支持 ROWID、ROWNUM、DUAL 等；
- 支持 FULL JOIN、(+) JOIN、MINUS、START WITH、MERGE INTO 等；
- 支持 FLASHBACK BEFORE DROP、FLASHBACK AS OF TIMESTAMP 等；

- **内置函数兼容**

- 支持字符串函数，包括：LPAD、RPAD、SUBSTR、TO_CHAR、INSTR、TRIM、LTRIM/RTRIM、||拼接等；
- 支持数值函数，包括：ROUND、TRUNC 等；
- 支持日期和时间函数，包括：SYSDATE、TRUNC、MONTHS_BETWEEN、ADD_MONTHS 等；
- 支持转换函数，包括：TO_CHAR、TO_NUMBER、TO_DATE、TO_TIMESTAMP、TO_CLOB、ROWTOHEX 等；
- 其他函数，包括 NVL、NVL2、DECODE、TO_DATE、ADD_MONTHS、ROW_NUMBER、TO_CHAR、TO_NUMBER、MONTHS_BETWEEN、SUBSTR、REGEXP_REPLACE、WM_CONCAT、DECODE、LISTAGG、NVL2、INSTR、RANK、REGEXP_SUBSTR、REGEXP_LIKE、ABS、TRUNC、SYSTIMESTAMP、SYS_CONTEXT。

- **PL/SQL 兼容**

- 支持%TYPE、%ROWTYPE 等数据类型；
- 支持 LOOP、GOTO、FOR LOOP、CASE WHEN... END CASE 等流程控制语法；
- 支持 FORALL、INTO、:=变量赋值等特有语法；
- 其他语法：游标 fetch 操作、connect by、left_join、right_join、merge into、start with 、with as、（+） 、||、cast_to_raw、forall 循环、bulk collect 操作、select for update skip locked、case when、select for update nowait、select for update wait（或 select for update 且默认为 wait）、exception when others then。

- **系统视图兼容**

- 支持 V$DATABASE、V$INSTANCE、V$SESSION、V$MYSTAT 等常用系统视图；
- 支持 DBA|ALL|USER_TABLES、DBA|ALL|USERS_USERS、DBA|ALL|USER_VIEWS、DBA|ALL|USER_INDEXES、DBA|ALL|USER_PART_TABLES、DBA|ALL|USER_TAB_PARTITIONS、DBA|ALL|USER_TAB_SUBPARTITION、

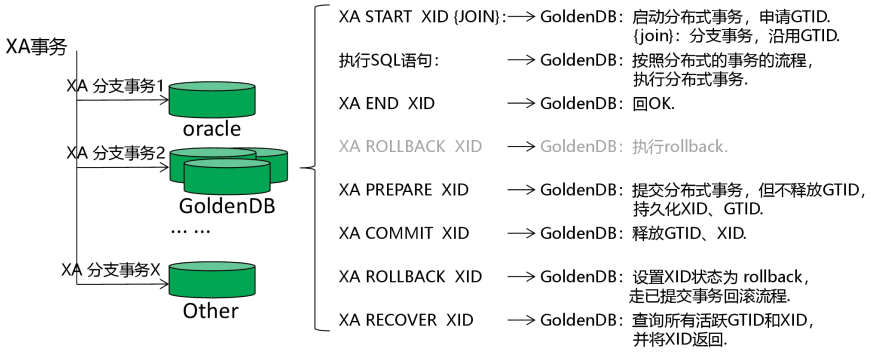
ALL_TAB_COLUMNS、 ALL_CONS_COLUMNS、ALL_SOURCE、ALL_TAB_PRIVS、 V$SESSTAT, V$INSTANCE、 V$LOCK、 V$SQL_WORKAREA、 V$OPEN_CURSOR。

- **高级包兼容**

– 支持 BMS_OUTPUT.PUT_LINE、DBMS_RANDOM.VALUE、DBMS_JOB.SUBMIT、DBMS_JOB.RUN、DBMS_LOB.GETLENGTH 、 DBMS_UTILITY.GET_TIME 、 DBMS_DEBUG.INITIALIZE 、 DBMS_LOB.APPEND、DBMS_SQL.OPEN_CURSOR、DBMS_SQL.PARSE、DBMS_SQL.EXECUTE、DBMS_SQL.CLOSE_CURSOR 、 DBMS_STATS.GATHER_TABLE_STATS 、 DBMS_METADATA.GET_DDL。DBMS_OUTPUT、DBMS_SQL、DBMS_RANDOM、DBMS_UTILITY、DBMS_JOB、DBMS_CRYPTO、DBMS_STATS 、DBMS_LOB 等常用高级包；

- **XA 事务兼容**

– GoldenDB 提供的 JDBC 驱动中支持标准 XA 接口，兼容 Oracle 的 XA 事务，支持 XA START、XA END、XA PREPARE、XA COMMIT、XA ROLLBACK、XA COMMIT ... ONE PHASE、XA RECOVER 等常见 XA 指令。



– 在 XA 事务执行中，确保读写/写写事务不冲突，保证一致性读。

- **OCI 接口**

– 支持 OCI 接口：OCIStmtExecute, OCIBindByName, OCITransCommit, OCIStmtPrepare, OCIStmtFetch, OCITransStart, OCIAttrSet, OCIAttrGet, OCILobRead, OCIDescriptorAlloc, OCILobWrite, OCIBindByPos, OCITransRollback, OCIInitialize, OCIEnvCreate, OCIEnvInit, OCIErrorGet, OCITerminate, OCIHandleAlloc, OCIServerAttach,OCIPing, OCIStmtFetch2, OCISessionBegin, OCIHandleFree, OCISessionEnd, OCIServerDetach, OCIStmtPrepare2, OCIDefineByPos, OCIStmtRelease, OCILobLocatorIsInit, OCINumberFromInt, OCINumberFromReal, OCINumberIsInt, OCINumberToReal, OCINumberIsZero, OCINumberSetZero, OCINumberTrunc, OCINumberAbs, OCINumberAdd, OCINumberCmp, OCINumberToInt, OCINumberAssign, OCITransDetach, OCIDateToText, OCIDateSysDate, OCIDateFromText,

OCIStringResize, OCIStringAssignText, OCIStringSize, OCIStringAllocSize, OCIStmtSetPieceInfo, OCIStmtGetPieceInfo

4.3.7 XA 事务

GoldenDB 的 XA 实现使 GoldenDB 服务器能够充当资源管理器，用于处理全局 XA 事务,连接到 GoldenDB 服务器的客户端程序充当事务管理器。当全局事务进入到提交阶段后，事务管理器采用两阶段提交协议保证该全局事务的原子性：

- 准备阶段：事务管理器向所有参与者发送准备请求，并等待它们的响应；参与者在收到准备请求后，将事务的状态（准备就绪或不准备）通知事务管理器。
- 提交阶段：如果所有参与者都准备就绪，事务管理器向所有参与者发送提交请求，参与者在收到提交请求后，执行事务的最终提交，并通知事务管理器；如果任何一个参与者在准备阶段表示不准备或在提交阶段发生错误,事务管理器会向所有参与者发送回滚请求。

XA 协议中没有描述如何实现分布式事务的隔离性,但是 XA 协议要求 DTP 模型中的每个 RM 都要实现,基于 XA 协议实现的分布式事务的隔离性是由每个 RM 本地事务的隔离性来保证的，当一个分布式事务的所有子事务都是隔离的，那么这个分布式事务天然的就实现了隔离性。

4.3.8 压缩功能

GoldenDB 支持表数据 压缩和备份压缩，对增删改查性能无影响。

表数据压缩，通过对表或表分区进行数据压缩来减少磁盘存储空间；普通表支持透明页压缩，分区表支持行格式压缩；压缩算法支持 Zlib 和 LZ4，压缩率在 40%到 90%之间。

备份压缩，是指对备份出来的数据进行压缩来减少空间占用，GoldenDB 支持 Quicklz 和 LZ4 算法对备份文件进行压缩，默认使用 Quicklz 压缩算法。

GoldenDB 同时支持对表的数据空间进行归并整理，可以实现对通过数据压缩和归并整理优化出来的空间进行释放到表空间，从而实现最大限度的存储优化能力。

4.3.9 日切卸数

有些核心业务场景，需要能准确地将会计日当天的业务数据卸载出来，不受会计日切换的影响。使用传统数据库存在卸数丢失跨天事务、分析报告不准确的问题，需要事后进行手工修复数据。GoldenDB 提供日切卸数功能，可以卸载准确的会计日当日业务数据，更快得到正确的分析报告，无需在事后手工修复数据。以卸载日切时刻的数据快照为例，使用 GoldenDB 日切卸数功能导出数据无需经过数据补录或清洗处理，既能满足事务全局一致性，又能满足具有业务含义的数据一致性（在日切前发生的事务操作都保留在快照数据中，日切后发生的事务操作都不体现在快照数据上）。

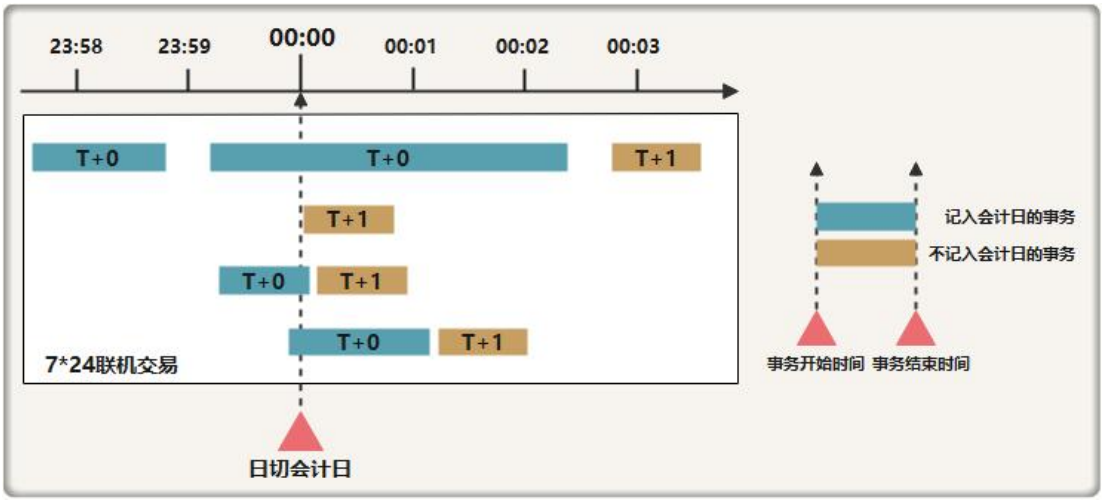


图 4.3-1 日切卸数示意图

4.3.10 多租户

多租户技术可以实现多个租户之间共享系统实例，同时又可以实现租户的个性化定制。通过使用多租户技术可以保证系统共性的部分被共享，个性的部分被单独隔离。通过在多个租户之间的资源复用，运营管理维护资源，有效节省开发应用的成本。

数据库多租户也满足 IT 行业多租户架构的定义：以单一系统架构与服务提供多数客户端相同甚至可定制化的服务，并且仍可以保障客户的数据隔离。多租户数据库提供的是数据库服务，它可以满足多个不同业务对数据库的需求。在多租户数据库中，租户是数据库对象管理和资源管理的基础，也即租户是各类数据库对象的容器，又是资源（ CPU、Memory、IOPS 等）的容器。租户之间数据是隔离的，资源是隔离的，就像独自使用数据库一样，不知道其他租户的存在，也不受其他租户的影响。

多租户 GoldenDB 数据库具备以下优势：降低成本、提升资源使用率、按需创建、快速部署、统一运维。

GoldenDB 支持多租户，支持在一个数据库集群内同时运行两种数据库语法引擎（Oracle、MySQL），支持在同一个数据库集群和同一个数据库引擎同时运行两种数据库租户（Oracle、MySQL），支持不同租户下的统一运维管理。在 GoldenDB 中，可以创建租户、扩缩容租户、删除租户等。GoldenDB 租户包含计算节点 CN、存储节点 DN 和全局事务节点 GTM，租户之间资源隔离，数据隔离。租户内数据实现多副本存储，保障数据的安全。租户内计算节点可以扩缩容，存储节点可以扩缩容。租户内用户之间 CPU、内存、IO 和存储空间的资源隔离，租户的 MySQL/Oracle 兼容模式在创建租户时选择，一旦确定兼容模式，不允许修改，只能通过删除租户再新建的方式变更。

4.3.11 数据库负载回放

支持 DBreplay 功能，可以从 GoldenDB/MySQL/ORACLE 等常见数据库录制数据库负载流量（Oracle 的 WCR 文件或 GoldenDB、MySQL 的 General Log），通过解析这些负载文件或日志，在 GoldenDB 客户端回放，实现语句的兼容性以及执行效率的评估。该功能支持多线程并行回放，可在支持单点或多客户端多压力的分布式回放，支持负载倍增回放，提供回放报告评估 SQL 执行效率以及 ORACLE、MySQL 等数据库的语法兼容性。

4.4产品性能

GoldenDB 通过执行计划缓存、并发事务控制、数据复制优化、批量协议、并行执行、流式查询等手段，对整体性能进行提升，满足银行业务高并发低时延的特性。

4.4.1 缓存管理能力

GoldenDB 通过构建执行计划缓存、SQL 缓存，提高 SQL 解析效率、提高数据读取效率，在高频度读写系统中可减少磁盘 I/O 负担，提升整体系统效率。

4.4.2 并发控制机制

GoldenDB 在事务处理上通过采用成熟的主流技术来实现高效的事务处理，这些技术主

要包括：以行级为主表级为辅的锁技术、多版本并行控制技术（multiversioning）、全局乐观锁+自动补偿机制。这些技术在保证事务 ACID 特征的前提下大大提高了事务的并发处理能力。

4.4.3 数据复制优化技术

分布式数据库通过增加副本数来提高系统可用性，为了数据的安全可靠，数据必须在满足拥有一定数量的副本之后，才返回处理结果给客户端。GoldenDB 采用自研的 gSync 复制技术，在配置的副本同步策略满足后，主库返回操作结果给客户端；通过线程池、非阻塞式同步、并行复制等关键技术可实现在确保数据 RPO 为 0 的同时保证系统的吞吐量。

4.4.4 批量协议

针对 PREPARE 模式,GoldenDB 引入批量协议,支持在 PREPARE 时只需要携带一组 VALUE，而在 EXECUTE 时可以携带任意多组 VALUE。批量协议有效提高 PREPARE 模式使用灵活性和成功率，大大提升 PREPARE 模式下 GoldenDB 的处理性能。

4.4.5 并行执行

SQL 语句在 GoldenDB 计算节点到各个数据节点间、计算节点到单个数据节点内，都支持并行执行，且保证分布式事务一致性。这种并行执行模式，降低查询过程中节点之间的数据流动，能有效提高计算节点执行性能。同时 GoldenDB 数据节点支持并行查询和并行 DDL，进一步提高对应场景下的处理性能。

4.4.6 流式查询

GoldenDB 支持流式查询，数据节点和计算节点之间、计算节点和客户端之间，都不需要等待结果集全部处理完成才回响应，而是按照设置分批回响应，且结果集不落盘。在大结果集查询场景下，全流程“execute-fetch”有效减少响应等待时间，整体性能大幅提升，并能降低系统 CPU、内存资源占用。

4.4.7 复杂查询并发

GoldenDB 自研 OLAP 引擎可以有效提升复杂规模查询计算效率，充分利用集群中多台服

务器的 CPU 以及内存资源，对海量数据进行快速查询分析。计算节点统一接收业务请求，判断涉及 OLAP 类查询或者列存查询时，自动转发给 AP 节点，由其承担复杂查询和复杂分析的任务。自研 OLAP 引擎基于分布式执行架构，实现多机并行执行；使用更丰富的数据统计信息，应用维度更科学的代价模型，在处理复杂查询时生成更优的执行计划；采用列式存储，实现全面向量化，充分利用单机多核计算能力并发执行，全面提升查询速度。

4.4.8 大容量数据处理能力

GoldenDB 具备完备的海量数据管理功能，提供 PB 级的数据容量支持，具有高效索引和查询优化技术，具备海量数据处理能力并支持大数量用户的并发访问。

4.4.9 分布式批处理

GoldenDB 提供分布式架构下批处理功能，满足大数据量批处理需求；支持流式处理和存储过程对数据进行批处理，减少客户端与数据库的访问次数，批量返回数据集并进行批量处理。

4.5 安全性

GoldenDB 作为一款安全、自主、可控的数据库产品，在安全性方面取得了显著成就，并得到了众多权威机构的广泛认证与认可。

全面的安全性保障

- **安全审计：**GoldenDB 支持界面化的安全审计功能，能够详尽记录并检测数据库管理操作及用户数据库对象操作，为数据库的安全监控提供了坚实保障。
- **密码安全：**GoldenDB 具备密钥管理与标准密码运算能力，支持符合国家商用密码标准的加密算法，确保密码的安全可靠。支持 SSL 连接加密认证。
- **数据安全：**GoldenDB 通过精细的授权管理员角色、数据脱敏查询、数据传输加密、数据加密存储、数据运行安全等功能，以及强大的备份和恢复机制，全面确保数据的完整性和可恢复性。
- **运维安全：**GoldenDB 通过适合分布式架构的一致性元数据管理能力、异常问题诊

断能力、监控告警能力和数据一致性管理能力等，确保 GoldenDB 的运维安全，全面保障数据库的稳定可靠。

权威机构认证

- 安全可靠测评：于 2024 年 9 月 30 日，首批通过由中国信息安全测评中心、国家保密科技测评中心、国家知识产权组织的分布式数据库的安全可靠测评，充分证明了 GoldenDB 在安全可靠方面的卓越表现，为用户提供了有力的安全保障。
- 信息技术产品安全分级评估 EAL4+认证：由公安部第三研究所于 2023 年 7 月 9 日颁发，证明了 GoldenDB 在信息安全方面的卓越能力和高标准。
- GM/T0028 商用密码产品认证证书：由国家密码管理局商用密码检测中心于 2024 年 1 月 12 日颁发，证明了 GoldenDB 在商用密码产品方面的合规性和安全性。
- 关系型数据库安全专项证书：由中国信通院于 2024 年 7 月 16 日颁发，证明了 GoldenDB 在关系型数据库安全能力方面均符合标准要求。
- TLCP 测试软件检测报告：由鼎铉商用密码测评技术有限公司于 2023 年 8 月 10 日出具，进一步验证了 GoldenDB 密码技术的先进性和可靠性。
- 网络安全等级保护核心系统等级测评报告（4 级）：GoldenDB 承接的项目由中国金融电子化公司测评中心于 2022 年 7 月 28 日出具，证明了 GoldenDB 核心系统在网络安全方面具有较高的防护能力。

通过一系列先进的安全技术和措施，如身份认证、权限管理、数据加密、审计日志、数据备份与恢复等，GoldenDB 全方位地保障了数据的安全性和完整性。同时，GoldenDB 还具备灵活扩展、高性能、高可用等特点，能够满足各种复杂业务场景的需求，为用户提供安全、可靠、高效的数据库服务。

4.5.1 安全审计

GoldenDB 支持界面化的安全审计功能，包括配置、查阅、备份、导出功能。该功能支持启停管理，并具备对数据库管理操作、用户数据库对象操作等的检测能力，同时能创建与用户关联的安全事件记录。还具备清理审计数据的能力，确保审计信息的时效性和系统性能，为数据库的安全监控提供全面保障。通过提供详细的审计日志和报告，帮助组织迅速响应风

险事件，追溯问题根源，并优化数据访问控制策略。

- 数据库管理操作

GoldenDB 的设计支持完整的安全管理功能，包括用户登陆、用户注销、用户权限控制，数据库启停、用户锁定解锁、数据库对象增删改和删除审计等。并且审计事件中包括主体身份，可以通过审计事件追溯到该事件对应的数据库用户。

- SQL 日志审计

审计是对数据库用户执行操作的记录和审核。GoldenDB 支持将所有的业务 SQL 写入审计日志文件，以便对 SQL 语句进行统计和审计，识别潜在的威胁。

- 审计数据可用性保证

GoldenDB 支持审计记录的查看、导出操作，但是必须具备对应权限的管理员或用户执行，无授权的用户无法查看或导出这些审计记录。审计数据具备加密存储的保护能力。

- 审计数据清理

GoldenDB 支持按策略周期清理最早的审计记录。

4.5.2 密码安全

GoldenDB 提供了密钥管理和密码运算功能的调用机制，以维护数据库管理系统中数据资产在存储和传输过程中的加密保护需求。GoldenDB 具备秘钥管理与标准密码运算能力。

- 密钥生成

GoldenDB 的密钥由安全管理员通过管理页面进行初始化和更新，使用符合 GM/T0028 标准的密钥生成算法，生成 CN 和 DN 工作密钥。

- 密钥变更

安全管理员通过管理界面进行密钥的更新，备份与销毁。

- 密码运算

GoldenDB 根据国家商用密码产品二级以上的特定的密码算法和密钥长度来执行密码运算列表，支持 SM2/SM3/SM4 加密算法。

- 密码验证

GoldenDB 支持通过国密算法对用户名密码进行解密校验。数据库设置了密码校验规则，包括长度、密码不能与用户名相同、英文字母大小写、特殊字符及数字等组合限制，如

果设置的密码不符合密码校验规则数据库会失败返回。

4.5.3 数据安全

GoldenDB 支持用户数据保护的能力，不仅支持精细的授权管理员角色，实现职责分离和角色约束，确保只有被授权的用户才能访问和操作数据库对象，而且支持数据库中的表列进行脱敏查询和数据传输加密功能，无论是与外部应用之间的交互还是数据库内部的数据传输，都能得到高级别的加密保护。此外，GoldenDB 提供事务处理和存储介质故障的数据恢复机制，以保障数据的完整性和可恢复性，而且支持数据副本间的准实时一致性，确保数据的实时同步，并在数据不一致时及时发出告警，为用户提供全面、可靠的数据保护服务。

4.5.3.1 访问控制

- **基于角色的访问控制**

GoldenDB 中有针对不同权限级别的各类系统表。GoldenDB 对不同的角色赋予或撤销不同的操作权限，有些角色赋予所有权限，有些角色只赋予读权限，有些角色只赋予写权限。角色赋予的权限会保存在对应的权限系统表中并持久化。创建用户后,将用户设置为一种角色，用户就拥有了该角色所对应的权限。

- **基于数据库对象的访问控制**

GoldenDB 可以对数据库对象配置不同的访问策略。如对于表的只读权限和写权限/敏感字段的访问权限/记录行的访问权限，不同的用户通过权限控制访问符合自己访问规则的数据。

- **身份鉴别标识**

用户登录数据库支持身份鉴别，身份鉴别标识包含用户名、密码等信息，在登录过程中需要对关键信息进行加密传输。

GoldenDB 通过’用户名’作为唯一标识。当用户登录后，根据唯一标识，从全部的权限系统表中加载该用户的全部权限。当执行某个 SQL 时，会根据执行的 SQL 语句类型、访问的库表等来汇总需要的全部权限,然后将该需要的权限集合与用户所拥有的权限做比较，如果当前用户所用有的权限满足需要的权限，则允许 SQL 执行，否则终止 SQL 执行并返回报错。通过设置不同的权限实现了精细化的权限访问控制。

- **会话状态控制**

GoldenDB通过客户端与CN端口建立会话，GoldenDB必须支持会话状态管理、SQL语句监控和活跃事务数监测，以及最大会话数控制，确保数据库操作的合规性。同时，GoldenDB必须支持多种会话终止方式、会话的锁定/解锁功能，并支持存储会话信息以追溯会话历史，提供全面的会话管理功能。这些措施共同保障GoldenDB会话管理的有效性，增强数据库访问的安全性和可管理性。支持设置最大连接数以及连接超时时间。

- **白名单客户端控制**

GoldenDB还支持白名单控制，通过IP白名单的方式过滤连接请求，只有在IP白名单中的客户端才可以访问，非白名单客户端连接报错。

- **数据库系统文件管理**

数据库系统的所有文件只有部署数据库的操作系统用户和操作系统的系统管理员才有权访问。

4.5.3.2 数据查询脱敏

GoldenDB支持对数据库中的表列进行脱敏查询，例如：模糊化信用卡号或身份证号中的某些数字。

4.5.3.3 数据传输保护

GoldenDB内部传输支持SSL加密。SSL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，有效保护数据的传输安全。GoldenDB内部传输使用SSL加密。原端将原始SQL语句进行SSL加密处理，然后将密文发送给目的端，目的端收到密文后进行解密，得到原始SQL语句进行正常解析和处理。此外GoldenDB也支持客户端使用TLCP加密的链路与CN交互。

4.5.3.4 数据加密存储

GoldenDB支持表级别和列级别的加密。通过列级别的加密，表中仅存储加密后的字符串，在不知道加密串的情况下，即使查询出来也会是乱码；表级别的加密可以加密整张表，在没有密钥文件的情况下，即使拷贝走表数据文件，也无法破解表内数据。

- 列级加密

对数据库中的某一列数据进行加密，这种加密方式具有较高的灵活性和选择性，可以根据实际需求选择需要加密的列。列加密支持加密算法包括 AES256 和 SM4。

- 表级加密

对整个数据表进行加密，这种加密方式适用于对整个表的数据进行保护。表加密支持的加密算法包括 AES256 和 SM4。

- 整个数据库级加密

对整个数据库进行加密，包括数据库中的所有表、视图、索引等对象。这种加密方式提供了最高级别的安全性，但也可能对性能产生较大影响。库加密实现上依赖于表加密。

4.5.3.5 数据运行安全

GoldenDB 支持数据运行安全，致力于为用户提供全方位的安全保障。GoldenDB 具备拦截大事务、拦截大结果集查询和预防误操作等多种机制来预防潜在的危险操作，确保数据在系统运行过程中的稳定性和可靠性。

- 拦截大事务操作

GoldenDB 能够智能地拦截大事务操作。在数据库环境中，大事务往往会对系统性能产生严重影响，甚至可能导致系统崩溃。通过拦截大事务，GoldenDB 能够避免这类操作对系统造成的冲击，确保系统能够持续稳定运行。

- 拦截大结果集查询

GoldenDB 还能够拦截大结果集查询。在某些情况下，用户可能会执行返回大量数据的查询操作，这不仅会消耗大量系统资源，还可能导致系统响应时间变长。通过拦截大结果集查询，GoldenDB 能够提醒用户注意查询操作的合理性，并引导用户采取适当的措施来优化查询性能。

- 预防误操作

GoldenDB 还具备预防误操作导致系统异常的能力。GoldenDB 通过预操作发现并避免可能导致系统异常的误操作。这一特性不仅提高了系统的安全性，还为用户提供了更加可靠的数据使用环境。

4.5.3.6 数据闪回

- **闪回**

闪回是一种用于恢复表数据到某个特定时间点或事务时刻的技术。闪回功能开启的情况下，清空数据后，可通过闪回功能恢复。GoldenDB 支持闪回功能，当表或是表分区出现误删除(drop)或是清理(truncate)时，可以通过闪回 FLASHBACK DATABASE/TABLE... TO BEFORE DROP/TRUNCATE 恢复删除的数据。

- **闪回查询**

闪回查询是指查询历史数据的能力。闪回查询的前提是需要在 Insight 页面打开闪回开关，数据库在执 DML（插入、更新、删除）操作时，DN 侧会将旧版本的数据副本保留下来。闪回查询通过指定时间戳，从副本中读取历史数据，而无需依赖备份还原。GoldenDB 支持闪回查询，通过分布式 MVCC 技术和 undo 日志，实现对历史版本数据的访问查询，使用 AS OF TIMESTAMP 关键字。

4.5.3.7 备份恢复

GoldenDB 拥有强大的备份和恢复功能，其备份类型分为两种：

- **全量备份**：全量物理备份。
- **增量备份**：最近一次全量备份后的增量部分备份（包含物理备份和逻辑备份）。

GoldenDB 的备份策略分为两种：

- **定时备份**：可以指定每周或每天指定时刻进行数据和日志备份
- **实时备份**：可以在需要时立即进行数据和日志备份任务

GoldenDB 备份的功能特点：

- **备份任务管理**

GoldenDB 支持在 Insight 上进行备份任务的管理，包括备份任务策略的配置、备份历史操作的查阅等。备份任务的发起可以有二种模式：一种为实时备份，即用户配置好后立刻会发起一次备份任务；另一种为定时备份，用户可以设置定时备份策略，典型的策略为每周日备份全量数据，其他时间分别在周日全量备份的基础上做一次增量备份，如

此，可通过全量数据和其他任一增量备份数据，快速恢复出想要的那天的数据。

- **备份文件存储**

GoldenDB 的备份文件可在分片节点本地挂载 NFS 共享目录，将备份结果文件统一存放；同时 GoldenDB 也支持与外部备份系统对接，如 IBM TSM、NBU、S3、COS 等，可以直接将备份结果存储到第三方存储介质。

- **备份文件存储策略**

GoldenDB 支持实例级存储配置、分片级存储配置、AZ 级存储配置、节点级存储配置等备份文件的存储策略，可以灵活应对局点的不同业务需求和网络隔离场景进行选择和配置。

GoldenDB 数据恢复功能，支持时间点和指定表的恢复功能：

- **可恢复到任意时刻**

由于 GoldenDB 有数据节点的全量及增量备份文件，同时有其运行过程中的日志，借助这些数据可以将系统恢复至任意需要的时刻，但需注意全量及增量备份文件恢复的速度要快于数据库日志的回放速度，因此如无特殊要求，建议选择恢复到某次备份的结束时刻，以便更快的完成数据恢复.

- **一致性的数据恢复**

由于 GoldenDB 备份了运行过程中的日志及每个时刻的全局事务列表快照，因此可以根据这些信息恢复出一个全局一致性的数据快照.

- **恢复任务管理**

GoldenDB 支持在运维平台上进行节点的一键恢复操作，用户直接在运维平台上选定要恢复的节点及要恢复到的时间点，即可进行自动化的数据恢复。

4.5.4 运维安全

GoldenDB 在运维安全领域提供了全面、高效且智能的解决方案，旨在为用户打造稳定、可靠且高效的数据库服务体验。

4.5.4.1 元数据管理能力

GoldenDB 支持元数据统一管理，并具备满足一致性要求的分布式数据库元数据管理能力。它不仅全面涵盖了元数据的创建、修改、删除、查询、可视化以及持久化等核心管理环节，还提供了对结构信息和分片信息等复杂元数据的管理功能。此外，GoldenDB 还具备对组件元数据的统一管理能力，能够修复各组件元数据不一致问题，确保各组件元数据的完整性和一致性。这一特性使得用户能够更加方便地进行元数据管理和维护。

4.5.4.2 异常问题快速诊断

GoldenDB 支持异常问题快速诊断功能。除了常规的集群服务器、数据库实例、慢 SQL、锁等待、死锁等问题诊断分析外，GoldenDB 还增加了对分布式事务、跨节点数据一致性等复杂问题的诊断能力。通过异常诊断的能力，GoldenDB 能够快速定位问题根源，提供准确的解决方案建议，有效缩短故障恢复时间。

4.5.4.3 监控管理告警能力

GoldenDB 支持一键向导式部署，使得部署过程更加简便快捷。图形化管理平台提供了丰富的监控信息，包括系统概况、运行情况、关键指标、集群情况、吞吐情况等，同时还支持自定义监控指标，满足用户个性化需求。当系统出现问题时，GoldenDB 能够实时发出告警，并支持快速跟踪和定位问题，确保系统稳定运行。

4.5.4.4 一键巡检及故障诊断

GoldenDB 支持一键巡检及故障诊断功能，能够按需收集日志及诊断信息，并进行智能分析和处理。支持日志打包及切片功能，有效的提升了问题诊断效率，同时 GoldenDB 还提供了丰富的诊断报告能力，包括按指定时间生成集群和数据库诊断报告等，为用户提供了全面的系统健康状况分析。

支持生成 AWR 报告，通过 AWR 报告评估数据库运行健康情况。支持生成 ASH 报告，通过 ASH 报告评估数据库运行健康状态。

- ASH 报告：

1）列出报告的用户事件（Top Events）；

2）列出执行的 SQL 类型、数量、占比、平均活跃 会话数（Top SQL Command Types）；

3）列出 TOP SQL 语句、及相应的响应时间占比（Top SQL with Top Events）；

4）列出会话 ID、等待事件、等待时间占比（Top Sessions）；

5）按照一个时间间隔对采集时间周期分组后,生成的每个时间间隔里的等待事件、占比（Activity Over Time）。

- AWR 报告：

1）列出集群、节点和租户的基本信息，包含开始时间、结束时间的 CPU、内存、存储资源分配情况；

2）列出集群和租户级别的等待事件；

3）列出执行时间、CPU 时间、物理读、逻辑读、影响行数、执行次数的 TOP SQL。

支持交易全程跟踪诊断，通过外部业务流水号或 Trace ID 可以输出指定事务或 SQL 请求在分布式数据库各节点的运行跟踪日志。

支持系统在运行中拦截大事务、拦截大结果集等危险操作。

支持不依赖第三方组件和产品的数据生命周期管理（数据自动归档和清理）。

4.5.4.5 主备节点数据一致性检查

GoldenDB 支持主备节点数据一致性检查功能，能够可视化实现数据库存储节点副本间主备数据的一致性检查。通过智能高效的数据处理能力，GoldenDB 能够实时监测主备节点数据的一致性状态，并在发现不一致情况时及时发出告警，确保数据的完整性和可靠性。同时，GoldenDB 还提供了丰富的数据修复工具和方法，帮助用户快速解决数据一致性问题。

4.6 数据库管理功能

GoldenDB 具备开放便捷的运维支撑体系，设计以智能化、最简化为目标，提供了全面的运维管理功能，以满足不同场景下的数据库管理需求。

GoldenDB 提供 Insight 统一运维管理平台，提供 WEB 界面形式的统一管理入口，支撑各种场景的运维需求，支持对多个集群的集中管控，简化用户对集群的管理，满足不同的运维管理需求。Insight 同时提供丰富的 API 接口，供其他系统集成，满足客户构建自身统一运维平台的需求。支持对 GoldenDB 各个组件的日志进行一键采集，方便用户进行日志分析和故障排查。

4.6.1 安装与升级

GoldenDB 支持如下安装与升级功能：

- Insight 提供资源管理功能，将物理机、虚拟机纳入资源池统一管理和监控；物理机安装模式，从物理机或虚拟机资源池分配资源进行安装。对于云化，由云化管理平台纳管服务器资源分配容器进行安装，安装后由 Insight 纳管容器租户。
- 提供多样化的安装形式，包括：界面引导式白屏安装、一键式黑屏安装、API 接口安装，具备自动推荐安装参数快速安装，也支持客户自定义安装参数，支持多种场景的安装需求；支持从一个节点往多个节点部署的能力。
- 容灾安装模式，支持租户跨多地多可用区的灾备部署能力，如两地三中心、同城多中心。
- 支持 CPU 架构：X86 与 ARM。
- 支持多种主流操作系统，如 Redhat、麒麟。
- 支持命令行、界面化启停操作。
- 安装过程记录日志，可从界面查看安装日志。
- 支持一键式命令升级与界面引导式升级，支持任务编排，支持不停服务升级，保证升级期间的业务连续性。
- 提供扩缩容 API 接口，支持其他系统通过标准的 API 接口使用 Insight 的资源，完成扩缩容操作。

4.6.2 数据配置

GoldenDB 支持如下配置功能：

- 提供参数配置功能。支持查看、变更组件参数配置，具备对单个节点或同类组件多节点批量变更的能力；支持参数动态生效与静态生效。
- 提供参数比对功能。同类组件的多个节点间可以进行参数比对，发现差异。可以实时、定时进行比对，提供比对报告。

- 提供参数模板功能。参数模板即一组预先设定好的参数，在安装时进行应用，安装完成的参数就是预设值。
- 提供参数变更历史查询功能。
- 提供存储配置功能。可以配置数据存储目录、日志目录、备份目录。
- 提供资源规格配置功能。

4.6.3 运维

GoldenDB 支持如下运维功能：

- 支持统计监控功能。包括统计集群节点的 CPU、内存、磁盘、网络的使用情况。支持慢 SQL 统计，如语句类型、用户名、数据库名、次数、总耗时、平均耗时等。支持性能状态统计，如每秒事务数和查询数、SQL 平均响应时间，高频 SQL 等。Insight 平台内置丰富的监控报表，支持自定义监控报表。提供 API 拉取接口，支持主动上报 Kafka、Prometheus。
- 支持告警功能。包括支持事件告警和阈值告警。数据库运行中遇到重要事件、异常事件会产生事件告警；对于监控指标，设置告警基线，超过基线产生阈值告警。支持告警信息的实时展示与历史告警查询。告警上送支持多种形式，包括：邮件、REST 接口、Kafka、ActiveMQ、Syslog、SNMP。
- 支持实时监测 SQL 执行过程中资源使用情况；提供查询计划的缓存管理功能；提供 SQL 改写的优化建议。支持在线会话的分析诊断功能。支持 SQL 限流功能。支持历史 DML 操作在线查询功能。支持超长 SQL 与超大事务的自动查杀功能。
- 支持对各类事件如配置修改、表结构变更等进行日志记录的功能，支持多种维度的日志，如错误日志、慢日志、general 日志、DDL 日志、审计日志等，可通过日志查看操作内容、执行过程和结果。Insight 运维平台记录完整的操作日志，可用于审计。
- 支持基于角色、组、用户的权限管理模型，支持权限隔离，保障资源的使用安全。
- 提供丰富的接口供局点系统调用，包括 REST 接口、SQL 接口、命令行接口，可以查询状态、统计等多种信息，可以实现管理操作，也可以支持问题分析诊断。
- 系统具备高可用能力，自动检测故障，故障主动自愈修复。

- 支持巡检诊断功能，提供诊断报告。提供界面巡检以及工具巡检， InsightTool 作为独立的软件工具，可以单独安装在服务器上，内置丰富的诊断规则和算法，能够准确识别并定位 GoldenDB 中的潜在问题，如性能瓶颈、组件故障、可疑 SQL 等，支持脚本方式或者命令行方式对 InsightTool 进行卸载的能力。
- Insight 运维平台的监控以及运维操作均提供界面操作、指令操作以及 restful API，包括支持集群管理、租户管理、监控、日志、告警功能，均可通过 API 实现扩容，缩容和异地调度。
- 支持行锁死锁与分布式死锁及 MDL 死锁的检测与自动解锁功能。针对分布式死锁，可以配置 online_ddl_detect_switch=1 将功能打开。 支持该功能的相关参数，如重试次数、判定间隔、重试次数等的个性化配置。
- 支持定时任务的调用及管理功能，管理包括定时任务的修改及删除等。

4.7 GoldenDB 工具集

GoldenDB 的工具集极为丰富，专为支持其高效运行与管理而设计，全面覆盖了数据库迁移、测试、同步、诊断及优化等多个关键领域。其中，CACTool（Capture Analyze Convert Tool，简称：CAC）迁移评估工具，为业务迁移提供关键决策、改造工作量预估以及整体迁移策略的参考；Replay 回放工具则能够解析并回放生产流量，助力企业在不中断业务的前提下进行性能测试与故障排查；Sloth 迁移同步工具则实现了 GoldenDB 与其他数据库之间的全量及增量数据迁移与同步，确保了数据的准确性与完整性；而 InsightTool 巡检诊断工具，则以其快速、准确的特点，为数据库系统的故障诊断提供了强有力的支持。这些工具共同构成了 GoldenDB 强大的工具集，为企业提供了全方位、一站式的数据库管理解决方案。

4.7.1 CACTool 迁移评估工具

CACTool 迁移评估工具是 GoldenDB 数据库系统中一款重要的工具，支持元数据、数据库 Schema、数据库对象、表数据快速迁移的功能，旨在帮助用户在数据库迁移前评估业务迁移的可行性，并为迁移计划、工作量评估等提供关键支持。以下是对 CAC 的详细介绍：

- **CAC 功能概述**

CAC 主要用于对业务迁移进行深入的可行性评估，具备信息采集、对象转换、分片键推

荐、语法兼容性分析和生成迁移评估报告等功能。其目标是帮助用户全面了解从当前数据库（如 Oracle(11g-19c) 、DB2、MySQLL(5.7-8.0)）迁移到目标数据库 GoldenDB 的可行性和潜在的不兼容项。通过自动整改不兼容的语法，CAC 能够帮助定制迁移策略、简化迁移方案，提高迁移效率。

1. **语法兼容性评估：**CAC 能够识别并报告出当前数据库中与目标数据库不兼容的语法元素。CAC 支持对表结构、序列、函数、存储过程、同义词以及 SQL 语句采集后进行兼容性分析。通过对源库采集的 DDL、DML、DQL 等进行分析，CAC 工具可以自动进行 DDL 转换，并生成兼容性评估报告。支持结构、SQL 的自动转化，对无法直接兼容的语句提供修改建议，为兼容或修改后兼容的语句提供结构迁移功能；

2. **数据量分析：**CAC 能够统计和分析当前数据库系统的数据量，包括表和其他对象的数量等。兼容性分析报告支持导出；支持抓取应用发出的 SQL 语句；兼容评估工具可本地部署，无需将采集结果上传外网。

3. **迁移计划制定：**基于评估结果，CAC 能够协助用户制定详细的迁移计划。

4. **数据库对象转换：**支持异构数据库的元数据、数据库对象、PL/SQL 等快速迁移转换功能。

- **CAC 产品优势**

1. **提高迁移效率：**通过自动整改不兼容的语法并生成评估结果协助用户制定详细的迁移计划，CAC 能够显著缩短迁移周期，提高迁移效率。

2. **降低迁移风险：**通过全面的语法兼容性评估和数据量分析，CAC 能够帮助用户提前发现并解决潜在的问题，降低迁移过程中的风险和不确定性。

3. **优化资源分配：**基于迁移计划，CAC 能够指导用户合理分配资源，确保迁移工作的顺利进行。

4. **易用性：**提供图形化界面评估报告，降低理解难度并增强了用户体验。

- **CAC 使用场景**

在数据库迁移项目启动前，使用 CAC 分析源端数据库（如 Oracle、DB2、MySQL）的历史运行数据，并对目标数据库 GoldenDB 进行全面的评估，能够为迁移决策、改造工作量预估以及整体迁移策略的制定提供关键性的参考。

4.7.2 Sloth 数据迁移工具

数据库迁移是数据库管理中的一项重要任务，它涉及到将数据从一个数据库系统转移到另一个数据库系统的过程。这一过程可能由于多种原因发生，如技术升级、成本节约、性能提升或业务需求变化等。以下将详细介绍 GoldenDB 的数据库迁移工具 Sloth。

- **Sloth 功能概述**

Sloth 是一个集管控、运维、编排于一体的数据迁移工具，提供了多种数据处理能力，如存量数据的迁移能力、增量业务的实时同步能力、数据的快照比对能力、数据的实时同步比对能力、数据的修复能力等。用户可根据具体的诉求，按照操作指导，编排自己需要的任务，达成目的。还提供了任务监控、模块部署、表画像等能力。

1. **数据迁移：**支持将源端数据库的存量业务数据和实时增量业务数据无缝迁移至目标端数据库。

2. **数据库同步：**支持源端数据库指定位点之后产生的增量业务数据，实时同步到目标数据库。在同机房万兆网络带宽下，对行长 1200 字节的表进行插入压力测试，要求单向同步性能不低于 100000 记录/s。

3. **反向迁移：**支持源端数据库和目标数据库互换，支持在 oracle/mysql/pg/GoldenDB 等多个异构数据库之间进行双向数据迁移。

4. **同步比对：**支持在数据同步过程中对同步完成的数据进行源端和目标端数据库差异实时校验功能。

5. **快照比对：**支持基于快照点的全量数据校验功能。

6. **数据修复：**支持同步比对或者快照比对的差异数据进行数据修复，保证数据一致性。

7. **数据发布：**支持实时增量数据按时间产生的顺序同步到第三方存储设备 Kafka。

8. **运维管理：**Sloth 支持多种运维能力，包括但不限于权限管理、动态黑名单、全加增状态扭转、在线性能观测及调优等。

9. **监控功能：**Sloth 支持多种监控能力，如采集、回放、比对组件状态监控等。基于状态机变更，Sloth 配套实现了系统级高可用、单模块高可用。

10. **部署功能：**Sloth 支持图形化部署采集、回放、比对组件，可以在编排完任务时，

即拿即用。

11. **表画像：**Sloth 数据迁移及数据同步支持绘制表维度的数据画像，即表画像，在运维管理界面可以直观查看表维度的可视化描述。Sloth 表数据画像包含多个元素：表的基本信息、数据统计、采集、回放、比对组件的时间统计和时延统计、性能（流量、QPS）统计、类型/列/行过滤/HEX 转换等特殊配置、剔除某张表、关闭某张表的比对、一键关闭任务级三无表比对、暂停回放、暂停比对、绑定执行计划等运维操作。

12. **动态加减表：**Sloth 支持增量同步数据过程中动态修改同步表的黑白名单。

13. **断点续传：**Sloth 具备断点续传能力，数据迁移或数据同步过程中发生故障，导致数据传输中断，在故障恢复后，系统能够自动从故障点继续迁移或同步。

14. **对象迁移：**Sloth 支持迁移源库中的数据对象定义(表、函数、存储过程、触发器、视图、同义词等)，Sloth 的对象迁移发生在业务数据迁移之前，通过对象迁移保证源端数据库和目标端数据库对象结构的一致(异构数据库字段类型等通过映射来保证一致)，确保业务数据能够从源端数据库中完整、准确地迁移到目标数据库中。

15. **DDL 同步：**Sloth 支持在增量业务数据同步过程中同步 DDL。

- **Sloth 产品优势**

Sloth 是专注于异构（Oracle（11g-19c）/MySql（5.7-8.0）/Postgres）或同构数据库与 GoldenDB 进行数据交互的专业服务与解决方案，旨在助力客户实现同构或异构数据库向 GoldenDB 的实时数据迁移和数据同步。Sloth 迁移或同步解决方案，配置流程简单便捷、风险低、运维成本低、可靠性高、性能突出。

1. **可视化集中管理，操作便捷快速：**Sloth 无需单独安装驱动程序或应用程序，无需对源数据库做大幅改动，只需在 Sloth 提供的可视化管理界面进行简单配置即可开启数据传输。Sloth 管理界面中，只需花费几分钟就能设置一个迁移任务。您可以在迁移任务中定义用来执行迁移的各项参数，其中包括设置源数据库和目标数据库的连接，以及迁移类型和对象。Sloth 支持上千个节点的集群规模，可同时运行并集中管理大量数据迁移同步任务。

2. **不停服同步，业务无感知：**Sloth 的解决方案能够助力客户将数据库迁移至 GoldenDB，且几乎不需要停机拷贝。源数据库在迁移期间发生的所有数据更改都会复制到目标数据

库，因此迁移期间不影响源数据库对外提供服务。在数据复制完成后，源数据库和目标数据库将保持继续同步，您可自由选择业务切换时间。

3. **高性能数据迁移：**Sloth 使用高规格服务器来保证每条迁移同步链路都能拥有良好的传输性能。在数据迁移方面，数据传输服务底层采取了多种性能优化措施；相对于传统的数据迁移工具，极大提升了传输性能。

4. **支持多种数据源：**Sloth 目前已支持 Oracle、DB2、SQLServer、MySQL、PostgreSQL、Kafka 等类型的数据源向 GoldenDB 数据库进行迁移或同步，但具体的功能因源数据库类型不同而有所区别。

5. **内置多种过滤器：**Sloth 支持正则表达式过滤库表；支持配置库、表、字段、SQL 表达式等多种方式在数据同步过程中对源数据事件进行过滤。

6. **故障自动恢复：**Sloth 具备高可用功能，支持迁移任务在不同节点自由调度，少量工作节点宕机并不会影响进行中的任务。并且支持机房级故障自动修复的高可用功能。

7. **数据一致性校验：**Sloth 支持多种数据一致性校验方式，如快照比对、迁移/同步比对，高效识别数据差异并保证数据迁移的质量。

8. **差异数据修复：**Sloth 支持差异数据的修复功能，保证数据迁移的一致性。

- **Sloth 使用场景**

1. **数据迁移和同步：**在数据库迁移项目中，支持同构数据库（如 GoldenDB 到 GoldenDB，包括不同的分片数量场景）、异构数据库（如 Oracle、MySQL 等）之间的全量数据、增量数据的迁移和同步。

2. **数据比对：**数据库基于快照点的全量校验和数据同步过程中的数据实时校验，确保迁移数据的准确性和完整性。

3. **数据修复：**针对源端数据库和目标端数据库数据不一致的场景，进行差异数据修复。

4. **数据发布：**针对特殊业务场景，支持将数据同步发布给第三方 Kafka。

5. **对象迁移：**支持数据库对象迁移。

4.7.3 Replay 回放工具

Replay 回放工具是一款功能强大、易于使用的数据库测试工具，能够帮助开发人员和测试人员高效地回放生产流量，对 GoldenDB 进行全面的测试。通过使用该工具，可以提高 GoldenDB 的稳定性和可靠性，降低故障率和维护成本。以下是对 Replay 回放工具的详细介绍：

- Replay **功能概述**

Replay 回放工具是 GoldenDB 工具集中的一部分，主要用于将生产流量解析为可回放的格式，并在测试环境中，回放解析的生产流量，以模拟真实用户行为，从而对数据库系统进行全面的测试。

1. **流量录制：**Replay 回放工具支持将生产流量（包括各种 SQL 语句、事务操作等）解析为可回放的格式。

2. **流量回放：**在测试环境中，该工具可以回放之前录制的生产流量，以模拟真实的用户行为，对数据库系统进行压力测试、性能测试等，支持多线程并行回放，可在支持单机，多客户端和多压力端的分布式回放，支持负载倍增回放，加快负载验证效率。

3. **测试验证：**通过回放流量，可以验证数据库系统在特定场景下的表现，包括响应时间、错误率等关键指标，从而评估系统的稳定性和可靠性。

- Replay 产品优势

1. **高效性：**Replay 回放工具能够高效地回放数据库流量，减少测试时间和成本。

2. **准确性：**由于回放的是真实的生产流量，因此测试结果更加准确可靠。

3. **易用性：** 该工具提供丰富的集成选项，支持从 Oracle 捕获的 WCR 文件，从 MySQL 捕获的 General Log 文件直接解析和回放，方便用户进行配置和使用。

4. **可扩展性：**Replay 回放工具支持 Oracle、GoldenDB 等数据库，可以根据实际源端数据库类型进行定制回放给目标端 GoldenDB。

5. **可视化：**提供回放报告评估 SQL 执行效率以及 ORACLE、MySQL 等数据库的语法兼容性，以及每条 SQL 的性能对比。

- **Replay 使用场景**

1. **性能测试：**在版本更新或数据库迁移前，使用 Replay 回放工具对数据库系统进行性能测试，确保在同等数据量与并发压力下 GoldenDB 的稳定性和可靠性。

2. **故障排除：**当生产环境出现问题时，根据捕获的相关数据库流量并在隔离的环境中重放，以便开发人员安全地调试问题，定位并修复故障。

3. **回归测试：**在软件开发生命周期中，使用 Replay 回放工具进行回归测试，确保新版本的数据库系统不会引入旧版本中已经修复的错误。

4. **A/B 测试：**可以将流量发送到两个数据库系统（如新版本和旧版本、异构数据库迁移等），比较它们的表现，以便做出数据驱动的决策。

4.7.4 InsightTool 巡检诊断工具

InsightTool 巡检诊断工具是一款针对 GoldenDB 数据库系统设计的独立部署工具，它提供了全面的巡检、问题诊断以及体检报告式的诊断报告功能。以下是对 InsightTool 的详细介绍：

- **InsightTool 功能概述**

GoldenDB 提供独立部署的 InsightTool 巡检诊断工具，支持巡检、问题诊断，提供体检报告式的诊断报告。支持组件日志的一键采集。支持网络监控、磁盘性能下降监控。

1. **独立部署：**InsightTool 作为独立的软件工具，可以单独安装在服务器上。

2. **全面巡检：**自动对 GoldenDB 的各个组件进行巡检，包括管理节点、存储节点、计算节点等，确保 GoldenDB 的整体健康状态。

3. **精准诊断：**内置丰富的诊断规则和算法，能够准确识别并定位 GoldenDB 中的潜在问题，如性能瓶颈、组件故障、可疑 SQL 等。

4. **体检报告式诊断报告：**提供详细的体检报告式诊断报告，报告内容涵盖了巡检结果、问题诊断、建议措施等多个方面，帮助用户全面了解 GoldenDB 的健康状况。

5. **一键日志采集：**支持对 GoldenDB 各个组件的日志进行一键采集，方便用户进行日志分析和故障排查。

6. **网络监控：**实时监控 GoldenDB 的网络状态，包括网络延迟、带宽利用率等，确保系

统的网络性能稳定可靠。

7. **磁盘性能监控：**监控 GoldenDB 所在磁盘的性能指标，如 IOPS、吞吐量等，及时发现并处理磁盘性能下降的问题。

8. **一键卸载：**支持脚本方式或者命令行方式对 InsightTool 进行卸载的能力。

- **InsightTool 产品优势**

1. **高效巡检诊断：**InsightTool 能够自动化地完成巡检和诊断任务，大大提高了运维人员的工作效率。

2. **诊断报告精准：**内置的诊断规则和算法确保了问题的精准识别与定位，减少了误报和漏报的情况。

3. **报告内容全面：**提供了全面的巡检和诊断功能，涵盖了 GoldenDB 的各个组件和性能指标，确保系统的整体健康状态。

4. **支持定制与扩展：**InsightTool 支持与其他系统或工具的集成安装，方便用户根据实际需求进行定制和扩展。

- **InsightTool 使用场景**

1. **日常巡检：**运维人员可以使用 InsightTool 进行日常巡检，全面了解 GoldenDB 的健康状态，及时发现并处理潜在问题。

2. **故障排查：**当 GoldenDB 出现故障时，运维人员可以使用 InsightTool 进行问题诊断，快速定位故障原因，并采取相应的解决措施。

3. **性能优化：**开发人员和运维人员可以利用 InsightTool 提供的体检报告，分析数据库系统的性能瓶颈，进行相应的优化调整，提高系统的运行效率和响应速度。

4.7.5 GDC 可视化开发工具

GoldenDB 开发者中心（GoldenDB Developer Center，GDC）是数据库图形化开发工具，也是数据研发和生产变更管控协同平台。GDC 开发者工具支持创建和管理数据库对象，执行 SQL 语句/SQL 脚本，编辑和执行存储过程，函数语句，导入和导出表数据。具体功能：

- 支持查看表结构，视图语句相关信息，包含主键、分区、索引等。

- 支持管理和编辑现有数据库对象、表、视图、模式、数据库、权限和索引等。
- 具备简单便捷的 SQL 编辑器，支持导入、编辑、保存、执行 SQL 语句和 SQL 脚本功能；支持代码折叠，SQL 格式规范化，语法检查、颜色区分、以及多线程操作。
- 支持关键词显示标记、动态语法提示的 SQL 编辑器等。
- 支持查询返回结果的查看、编辑、分组、排序和过滤等。
- 支持执行计划图形化查看等。
- 支持编译和执行存储过程、函数语句
- 支持导入导出表数据。

5 应用场景

5.1 金融行业

金融尤其是银行核心 IT 应用，对性能、稳定性以及安全性有着极高的要求。随着移动互联网和电子商务的发展，金融行业的交易量激增，数据量呈现几何式增长。传统 IOE（IBM 小型机、Oracle 数据库、EMC 存储设备）集中式架构已无法满足持续增长的高并发和快速响应的需求。在这种背景下，GoldenDB 因其自主可控、高性能、高可靠等特点，在金融行业中的应用显得尤为重要。

1. 实现核心系统主机下移

优势：降低成本；灵活扩容；高可靠性

2. 构建新一代金融信息系统

优势：开放灵活；快速创新；平台化管理

应用场景：

- 新一代金融信息系统平台，支持金融科技与互联网金融业务的创新发展，推动业务数字化转型。

3. 建设数据库云服务平台

优势：云服务标准化；简化运维；资源优化

应用场景：

- 金融行业 IT 基础设施上云，采用全栈云服务方式，包括 IAAS、数据库、微服务框架、运维管理工具等，实现开发、部署、扩容、运维和升级的一系列优化。

4. 提升金融交易系统快速决策能力

优势：高效存储；混合处理；实时分析

应用场景：

- 实时风控系统：通过实时数据分析，提升风控能力，防范金融风险。
- 反欺诈系统：利用实时数据分析能力，快速识别欺诈行为。
- 实时经营分析：提供实时经营数据，支持快速决策。

场景案例

工商银行对公核心系统原先运行在 IBM 大机上，使用 DB2 数据库承载业务。随着金融核心系统业务量和数据量不断增长，原有的基于大机打造的集中式封闭架构已无法满足工商银行对该核心系统未来增长需求，工商银行核心业务系统亟需向分布式开放架构转型。工商银行于 2020 年开展对该核心系统下主机工作，进入分布式架构转型和国产化转型阶段，着眼于未来，借助分布式放平台实现对该核心系统安全可控、高并发、高容量、易扩展等特性。

工商银行选择 GoldenDB 开展对该核心系统主机下移项目，目前已成功将对该核心系统迁移至 GoldenDB，由 GoldenDB 独立支撑该核心业务运行，该系统已完全退出大机，GoldenDB 支撑工商银行上千万用户账户，日均处理 2 亿服务调用量，提供高效稳定的数据库服务，日均处理近 1.6 亿笔的交易量，承载交易金额日均超万亿。

GoldenDB 在金融行业尤其是银行核心 IT 应用中具有明显的优势，包括实现核心系统主机下移、构建新一代金融信息系统、建设数据库云服务平台以及提升金融交易系统快速决策能力。通过这些应用，GoldenDB 不仅帮助金融机构降低成本、提高效率，还能确保系统的高可用性和安全性，满足金融行业对高性能、高可靠性的严格要求。随着金融行业不断推进

数字化转型，GoldenDB 将继续发挥其在数据库领域的领先地位，助力金融机构实现业务创新和发展。

5.2 电信行业

电信行业是数据库产品主要市场之一，也是对数据库产品有很高要求的行业之一，尤其是在容量、性能和 Oracle 兼容性等方面有极致的要求。

电信业务运营支撑系统主要包括 BMO（BSS\MSS\OSS），其中计费/CRM/ERP 等数据库关系着运营商消费者的核心利益，是运营商的最重要数据，对性能、稳定性以及安全性都有着极高的要求，传统集中式架构一方面无法满足大容量、高并发、高可扩展、高可用等关键需求，另一方面，为降低业务改造成本，对国产数据库有很高的语法兼容和工具集要求，具体需求如下：

1. 高度兼容 Oracle 需求。CRM&BOSS 不仅使用大量的复杂 SQL，还使用大量的 Oracle 高级对象特性，如存储过程、包、自定义函数、触发器、匿名块、系统视图、Sequence、自治事务、JOB、XA、物化视图等。

2. 因无法进行全覆盖测试，必须通过 SQL 录制及回放工具进行覆盖性验证测试。

3. CRM&BOSS 不仅数据量大，而且业务逻辑复杂。涉及用户管理（开户/套餐变更/计费），计费计量（服务套餐/月度计费/充值等），产品配置（定价/阶梯服务）以及月初月末存在批量算费出账等跑批类业务，要求数据库具备处理是多复杂计算以及大规模批量并行计算能力。

4. 完整的数据库工具集，包括：迁移工具（全量+增量）、数据实时校验、回放工具、数据同步工具、数据库开发工具、巡检功能等等。

5. 根据系统安全等级要求，支持本地机房/本地同城/本地异地/本地同城异地的高可用架构，实现 RPO=0，RTO<8S。

6. 反向同步，支持故障回切，RPO=0。

目前，GoldenDB 已在运营商大面积商用，且在很多关键系统割接上线，如浙江移动账务库、山东移动营业及账务库、河北移动账务库/计费库/订单库、CMIOT 计费结算中心、中移动全网客户中心、安徽移动商品中心等落地案例。各系统割接上线后，系统运行稳定。在

整个国产化替换过程中，除了数据库驱动、导入导出工具等需要替换成 GoldenDB 的驱动或者导入导出工具外，业务核心代码几乎无改动。

5.3 政务行业

各级政企用户为了打造信息共享和业务协同的智能化政务服务，需要建立统一的基础数据标准，并形成相关的基础信息库。在人口大省和大市，容量和性能给传统数据库带来压力。而分布式数据库的大容量、高并发特性，成为应对解决各级政务数据瓶颈的利器。使用分布式数据库海量数据的存储和快速查询的解决方案能够给为业务带来单机数据库所无法提供的数据可扩展性及性能可扩展性；长期以来政府机关企事业单位使用的多是国外数据库 Oracle，对 Oracle 的语法、运维手段和性能都非常熟悉。政企客户对于数据库的应用场景和要求如下：

1. 强大的 Oracle 兼容性

GoldenDB 支持 Oracle 的大部分语法和功能，包括存储过程、包/包体、自定义函数、自定义数据类型以及游标等。这种兼容性使得现有基于 Oracle 的应用可以直接迁移到 GoldenDB 上，减少了重新开发的成本和复杂性。

优势：无缝迁移；业务无感

2. 强劲的性能表现

GoldenDB 在同等资源配置下，能够提供超越 Oracle 的性能表现，尤其是在高并发场景下。

优势：高并发处理；快速查询

3. 配套数据同步工具

GoldenDB 提供了配套的数据同步工具，支持核心系统与下游数据分析 OLAP 平台之间的数据同步，能够实现海量数据准实时同步。

优势：数据一致性；实时性

4. 合理规划资源投入

GoldenDB 具备多租户能力和动态缩扩容能力，可以根据不同租户的需求合理分配资源，

并在紧急情况下进行性能扩容。

优势：资源优化；灵活性。

5. 高可用能力

GoldenDB 具备健壮的高可用能力，支持多地多中心的灵活故障切换，确保核心系统的高可用性。

优势：故障切换；数据安全。

6. 完善的服务支撑体系

GoldenDB 提供多级保障体系，帮助政企用户建立健全数据库运维管理能力。

优势：技术支持；运维管理

场景案例

海关主要负责监管经各口岸进出境的运输工具、货物、行李物品、邮递物品和其他物品，征收关税和其他税费，查缉走私并编制海关统计和办理其他海关业务。IT 系统从使用场景上分为“对内”和“对外”。

海关业务系统数据体量没有金融行业庞大，但是业务场景更加复杂。其核心类系统有：通关、报关、物流、检验检疫、智能海关等。现阶段核心系统为第 2 代核心系统，主要运行在 Oracle 数据库上，小部分运行在 SQLServer 数据库上。

从 2023 年 2 月份开始，GoldenDB 参与海关新核心数据库选型 POC 测试，截止目前已经完成两个核心业务系统的兼容性对接和性能测试，其中，业务兼容性对接结果整体兼容率 95%以上。在充分完成业务兼容性对接测试之后，启动业务性能测试，目前无论是业务量还是耗时都完全超越了原有的 Oracle 数据库。

GoldenDB 作为一款数据库产品，在政务云环境中具有显著的优势，特别是在需要高并发、大容量数据处理和高性能的场景下。其强大的 Oracle 兼容性、强劲的性能表现、配套的数据同步工具、合理的资源规划、高可用能力和完善的服务支撑体系，使其成为政企客户理想的数据库解决方案。通过 GoldenDB 的应用，各级政府企业可以实现信息共享和业务协同的智能化政务服务，提高政务系统的效率和稳定性。

5.4 交通行业

交通行业是一个高度依赖数据管理和实时处理的领域，涵盖公共交通、道路运输、航空、铁路、港口等多个子领域。交通行业数据库的应用场景非常广泛，从高速过路费到旅客票务服务，从道路监控到港口装卸等，都需要高效、可靠的数据库支持。以下是交通行业中几个典型的数据库应用场景：

1. 交通收费管理系统

交通收费管理系统用于管理和处理各种交通收费，如高速公路收费、停车场收费等。

2. 交通规划与设计系统

交通规划与设计系统用于支持交通基础设施的规划和设计，需要处理大量的地理信息数据。

3. 交通数据分析系统

交通数据分析系统用于对交通数据进行深入分析，支持决策制定和优化。

4. 旅客信息服务系统

旅客信息服务系统为乘客提供实时的交通信息，如公交、地铁、火车、飞机的时刻表、延误信息等。

5. 港口管理系统

自动化集装箱码头操作系统（以下简称“TOS”）作为自动化集装箱码头的“大脑”，是集装箱码头实现自动化运行的关键所在。

场景案例

各省交通集团主要从事交通基础设施的投资建设运营、装备制造、城市综合开发等，为客户提供投资融资、咨询规划、设计建造、管理运营一揽子解决方案和综合一体化服务。信息化管理，主要包括数字施工、自动化设备管理以及数字化管理系统等，以及对客业务，包括计费功能，面对来往车辆提供快速、准确的服务。

在数据规模方面，核心收费业务相关的数据量大概在 2TB 左右，单表数据量大概在 1000 万行，整体规模不大。现网业务系统（如综合运营管理系统）主要是基于 Oracle。

GoldenDB 的业务兼容性测试和性能测试满足交通集团国产数据库选型要求，后续将针对部署方案和高可用方案进一步深化讨论，进行投产落地。

5.5 医疗行业

医疗行业的核心系统包括医院信息系统(HIS)、电子病历系统(EMR)、实验室检验管理系统(LIS)、医学影像传输系统(PACS)、放射学信息系统(RIS)、临床信息系统(CIS)、合体检信息系统(PEIS)、医院资源管理系统（HRP）、医院客户关系管理系统（HCRM）等

医疗行业中使用的数据库种类较多，大多是随系统产品采购引入，具体使用哪种类型的数据库，还需要根据具体系统的需求来决定。

场景案例

HIS 系统是医院管理的核心系统，它集成了医院各个科室的医疗信息，包括病人信息、医嘱信息、药品信息、检验信息等，为医护人员提供全面的医疗信息支持。

HIS 系统应用数据库使用特点如下：

1. 用 Oracle 和 SQLServer 数据库的 HIS 占了绝大多数。

2. 数据类型基本覆盖关系数据库常见的各种类型，以及 clob，blob 等超大字段。

3. 严重依赖普通视图、嵌套视图、物化视图、函数、存储过程、索引、触发器、序列等数据库功能，大量业务逻辑在数据库中实现。

4. OLTP 与 OLAP 共存；而且数据量大。

目前 GoldenDB 已与一些行业内的医疗系统 ISV 进行了对接，与部分业务系统进行了认证与适配。

6 产品生态

GoldenDB 作为一款聚焦国产化生态适配的数据库产品，其兼容性工作构建了覆盖底层硬件到上层应用的全栈适配体系，不仅涵盖 CPU、服务器、操作系统、中间件、应用软件等核心层面，更通过深度技术打磨实现了与国产软硬件生态的全方位融合。GoldenDB 已经与400+合作伙伴、1000+个产品完成产品兼容性互认证，部分清单及证书如下：

航天壹进制容灾备份与恢复软件&GoldenDB 互认证

爱数备份与恢复系统&GoldenDB 互认证

宝兰德消息中间件软件&GoldenDB 互认证

宝兰德应用服务器软件&GoldenDB 互认证

长城数据备份与恢复系统&GoldenDB 互认证

鼎甲数据备份与恢复系统&GoldenDB 互认证

中创应用服务器软件&GoldenDB 互认证

东方通负载均衡软件&GoldenDB 互认证

金蝶 Apusic 应用服务器软件&GoldenDB 互认证

普元主数据管理平台软件&GoldenDB 互认证

PRIMETON 普元 ZTE中兴

产品兼容互认证证书

普元应用服务器软件[简称：PAS] V6.5是一款具备完整JavaEE功能，支持Web应用的开发、部署、监控、管理等能力，具备集群、高性能、高可用、易扩展的中间件产品。普元应用服务器软件基于微内核的轻量级开放架构，使其可以快速整合不同的配置和部署模型，定制符合用户需求的应用服务器，帮助用户快速整合已有业务，降低IT资源的投入。

GoldenDB是中兴通讯股份有限公司（以下简称“中兴通讯”）自主研发的关系型数据库，可为用户提供高性能、高可靠、高并发、可扩展的数据库解决方案。

针对上述产品的兼容性方面，普元信息技术股份有限公司和中兴通讯通过共同测试表明，普元应用服务器软件[简称：PAS] V6.5与GoldenDB V6、GoldenDB V7、GoldenDB Lite V7、GoldenDB EBASE V3能够在鲲鹏920、UOS操作系统服务器环境中相互兼容，系统功能运行稳定。

特此证明！

※声明：

(1)中兴通讯不对由普元应用服务器软件[简称：PAS] V6.5版本升级引起的兼容性负责，普元信息技术股份有限公司不对由GoldenDB V6、GoldenDB V7、GoldenDB Lite V7、GoldenDB EBASE V3升级引起的兼容性负责。此认证仅适用于GoldenDB V6、GoldenDB V7、GoldenDB Lite V7、GoldenDB EBASE V3及普元应用服务器软件[简称：PAS] V6.5（如上所列）。

(2)此认证仅在中国大陆地区适用（不包括港、澳、台地区）。

普元信息技术股份有限公司 2025年7月

中兴通讯股份有限公司 2025年7月

普元应用服务器软件&GoldenDB 互认证

PRIMETON 普元 ZTE中兴

产品兼容互认证证书

普元ESB软件[简称：Primeton ESB]V9是普元信息技术股份有限公司基于多年对大型企业的IT建设及分布式计算和集成能力的认识和技术积累而推出的服务集成产品，是部署和实现SOA的理想工具，支持协议转换、消息转换、消息路由、服务编排、服务注册、服务查找、服务监控等功能。

GoldenDB是中兴通讯股份有限公司（以下简称“中兴通讯”）自主研发的关系型数据库，可为用户提供高性能、高可靠、高并发、可扩展的数据库解决方案。

针对上述产品的兼容性方面，普元信息技术股份有限公司和中兴通讯通过共同测试表明，普元ESB软件[简称：Primeton ESB]V9与GoldenDB V6、GoldenDB V7、GoldenDB Lite V7、GoldenDB EBASE V3能够在鲲鹏920、UOS操作系统服务器环境中相互兼容，系统功能运行稳定。

特此证明！

※声明：

(1)中兴通讯不对由普元ESB软件[简称：Primeton ESB]V8版本升级引起的兼容性负责，普元信息技术股份有限公司不对由GoldenDB V6、GoldenDB V7、GoldenDB Lite V7、GoldenDB EBASE V3升级引起的兼容性负责。此认证仅适用于GoldenDB V6、GoldenDB V7、GoldenDB Lite V7、GoldenDB EBASE V3及普元ESB软件[简称：Primeton ESB]V9（如上所列）。

(2)此认证仅在中国大陆地区适用（不包括港、澳、台地区）。

普元信息技术股份有限公司 2025年7月

中兴通讯股份有限公司 2025年7月

普元 ESB 软件&GoldenDB 互认证

溪数科技 ZTE中兴

产品兼容性互认证明

溪数KetaOps智能运维平台软件[简称：KetaOps平台]是安徽溪数科技有限公司研发的运维产品，整体方向始终围绕运维领域展开，针对该领域所涉及的关键数据（Metrics、Logs、Events、Tracing、拓扑、运维资产配置）进行产品体系构建，提供智能运维、可观测性、应用性能监控等前沿的解决方案及场景落地能力。其模块包含：指标体系、故障管理、日志监控及管理、业务监控、基础监控及管理。

GoldenDB是中兴通讯股份有限公司（以下简称“中兴通讯”）自主研发的关系型数据库，可为用户提供高性能、高可靠、高并发、可扩展的数据库解决方案。

经安徽溪数科技有限公司与中兴通讯联合测试，GoldenDB V6、GoldenDB V7、GoldenDB Lite V7、GoldenDB EBASE V3与溪数KetaOps智能运维平台软件[简称：KetaOps平台]V3.0兼容良好，运行稳定，性能优越。

安徽溪数科技有限公司 2025 年 7 月

中兴通讯股份有限公司 2025年 7 月

证书编号：ZGA-20250701008

溪数 KetaOps 智能运维平台软件&GoldenDB 互认证

新炬网络 SNC Net ZTE中兴

产品兼容性互认证明

AIOPS智慧运维管理平台软件V4.0，是由新炬网络研发的智慧运维管理平台，产品提供全方位数字化运维管理能力，包括智能监控、统一告警、CMDB、ITSM、自动化运维、日志分析、应用性能分析（APM）、网络性能分析、业务拨测、全景观测、运维中台、低代码能力、大屏可视化以及应用发布等能力模块，助力企业实现IT运维的数字化转型，提升运维效率，降低成本，确保业务稳定安全。

GoldenDB是中兴通讯股份有限公司（以下简称“中兴通讯”）自主研发的关系型数据库，产品面向金融、电信、政府等行业，可为用户提供高性能、高可靠、高并发、可扩展的数据库解决方案。

经上海新炬网络技术有限公司与中兴通讯联合测试，GoldenDB V6、GoldenDB V7、GoldenDB Lite V7、GoldenDB EBASE V3与AIOPS智慧运维管理平台软件V4.0兼容良好，运行稳定，性能优越。

上海新炬网络技术有限公司 2025 年 07月

中兴通讯股份有限公司 2025 年 07月

证书编号：ZGA-20250701018

新炬 AIOPS 智慧运维管理平台软件&GoldenDB 互认证

新炬 ZnSQL 异构数据库智能管理平台软件&GoldenDB 互认证

新炬 ZnAiops 全栈智慧运维平台软件&GoldenDB 互认证

云信达企业数据备份与恢复系统软件&GoldenDB 互认证

中孚数据库内容保密检查系统&GoldenDB 互认证

移动云天元操作系统 BC-Linux&GoldenDB 互认证

英方数据备份与恢复管理软件&GoldenDB 互认证

SuperSync 大型数据库高性能复制平台&GoldenDB 互认证

DataXone 大数据综合实时采集与共享交换平台&GoldenDB 互认证

数据网关（DG）软件&GoldenDB 互认证

数据雷达（DR）软件&GoldenDB 互认证

InforCube 智能运维安全管理平台&GoldenDB 互认证

敏捷数据库管理平台（ADM）软件&GoldenDB 互认证

InforCube 移动安全管理平台软件&GoldenDB 互认证

鸿数隐私数据保护管理软件&GoldenDB 互认证

中安数联备份与恢复系统&GoldenDB 互认证

DataPipeline 实时数据融合平台&GoldenDB 互认证

NewStart ZTE中兴

兼容性认证

广东中兴新支点技术有限公司与中兴通讯股份有限公司对新支点服务器操作系统 V6和GoldenDB V6、GoldenDB V7、GoldenDB Lite V7适配性进行严格的联合测试验证，得出以下结论：

新支点服务器操作系统 V6与GoldenDB V6、GoldenDB V7、GoldenDB Lite V7完全兼容，功能、性能和兼容性等各方面表现良好，运行稳定、安全，可以满足用户的需求。

广东中兴新支点技术有限公司（盖章）
2025年7月

中兴通讯股份有限公司（盖章）
2025年7月

证书编号：F20250274

新支点服务器操作系统 V6 和 GoldenDB 互认证

附录 1：GoldenDB 部分关键指标测试结果

1. 功能性指标

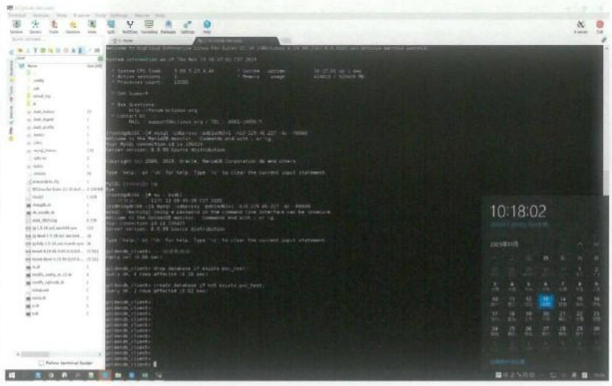
TTL泰尔实验室 CAICT 中国信通院

检 验 报 告 附 件 一

报告编号：25V01Y001802-001 第 1页 共 287 页

1. 功能性

1.1 字段类型

序号：UseCase101
测试项目：字段类型
测试目的：验证是否支持各类数据类型
预置条件： 1) 数据库正常运行
测试步骤： 1) 创建含有数值型（精确到小数点后 5 位）、浮点型（精确和非精确）、字符串型、时间日期型（精确到毫秒）、大字段（支持大字段值的查询）、二进制类型、伪列类型、布尔类型、JSON 类型、XML 类型、GIS 用户自定义类型的数据模型； 2) 用语句或工具向数据库中导入数据； 3) 查询数据。
预期结果： 1)创建成功； 2)数据写入成功； 3)成功查出数据，验证无误。
测试结果： 

```
goldendb_client> -- 字段类型
Empty set (0.00 sec)

goldendb_client> use poc_test;
Database changed
goldendb_client> DROP TABLE IF EXISTS TestAllTypes;
Query OK, 0 rows affected, 2 warnings (0.01 sec)

goldendb_client> CREATE TABLE TestAllTypes (
    ->     exact_number DECIMAL(10, 5),
    ->     precise_float DOUBLE,
    ->     imprecise_float FLOAT,
    ->     fixed_string CHAR(10),
    ->     variable_string VARCHAR(255),
    ->     event_time TIMESTAMP(3),
    large_text TEXT,
    binary_data BLOB,
    row_identifier SERIAL,
    ->     large_text TEXT,
    ->     binary_data BLOB,
    custom_type ENUM('small', 'medium', 'large')
    ->     row_identifier SERIAL,
    ->     is_active BOOLEAN,
    ->     json_data JSON,
    ->     custom_type ENUM('small', 'medium', 'large')
    -> );
Query OK, 4 rows affected (0.02 sec)

goldendb_client> INSERT INTO TestAllTypes (
    ->     exact_number, precise_float, imprecise_float,
    ->     fixed_string, variable_string, event_time,
    ->     large_text, binary_data, is_active, json_data, custom_type
    -> )
    123.45678, 123.456789012, 123.456789012,
    -> VALUES (
    ->     123.45678, 123.456789012, 123.456789012,
    ->     'fixedStr', 'This is a test string', CURRENT_TIMESTAMP(3),
    -> );    'This is a large text field.', 0xDEADBEEF, TRUE,
    ->     '{"key": "value", "array": [1, 2, 3]}', 'medium'
    -> );
Query OK, 1 row affected (0.01 sec)
```

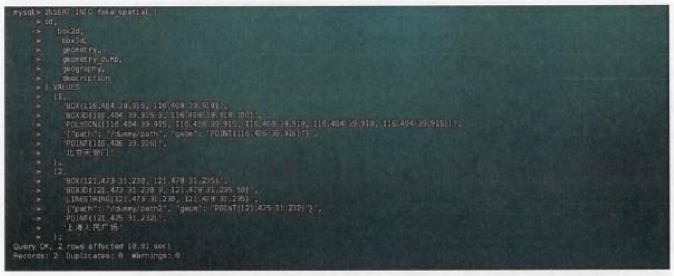
-- XML 类型

```
goldendb_client> -- 创建含 XML 列的表
Empty set (0.00 sec)

goldendb_client> use poc_test;
Database changed
goldendb_client>
goldendb_client> CREATE TABLE test_xml (
    ->     id INT PRIMARY KEY,
    ->     xml_data XMLTYPE COMMENT '存储XML格式数据'
    -> ) distributed by hash(id)(g1,g2);
Query OK, 4 rows affected (0.02 sec)

goldendb_client> -- 插入数据
Empty set (0.00 sec)

goldendb_client> INSERT INTO test_xml VALUES
    -> (1, '<employee><name>Alice</name><age>25</age></employee>'),
    -> (2, '<employee><name>Bob</name><age>30</age></employee>');
Query OK, 2 rows affected (0.00 sec)
Records: 2  Duplicates: 0  Warnings: 0

goldendb_client> -- 查询
Empty set (0.00 sec)

goldendb_client> SELECT id,
    ->        ExtractValue(xml_data, '/employee/name') AS name,
    ->        ExtractValue(xml_data, '/employee/age') AS age
    -> FROM test_xml;
+----+-------+------+
| ID | NAME  | AGE  |
+----+-------+------+
|  1 | Alice | 25   |
|  2 | Bob   | 30   |
+----+-------+------+
2 rows in set (0.00 sec)
```

-- 伪列支持

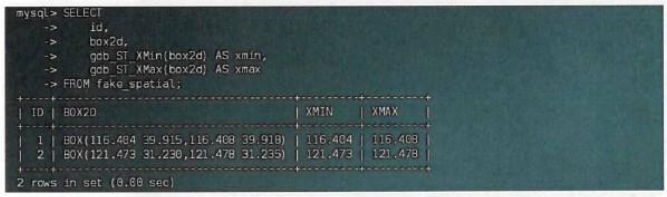
-- GIS 用户自定义类型

```
mysql> CREATE TABLE test_spatial (
    ->     id          NUMBER PRIMARY KEY,
    ->     box2d       box2d,
    ->     box3d       box3d,
    ->     geometry    geometry,
    ->     geometry_dump geometry_dump,
    ->     geography   geography,
    ->     description VARCHAR2(200)
    -> );
    -> //
Query OK, 4 rows affected (0.07 sec)
```



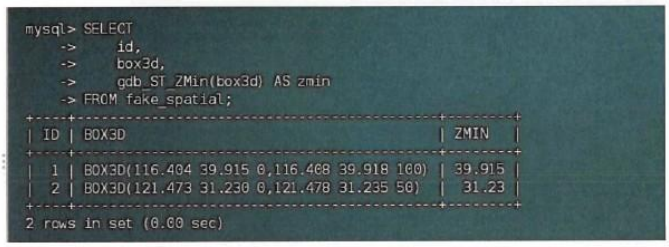
查询数据：

-- 测试 box2d 查询



```
mysql> SELECT
    ->     id,
    ->     box2d,
    ->     gdb_ST_XMin(box2d) AS xmin,
    ->     gdb_ST_XMax(box2d) AS xmax
    -> FROM fake_spatial;
+----+-----------------------------------+---------+---------+
| ID | BOX2D                             | XMIN    | XMAX    |
+----+-----------------------------------+---------+---------+
|  1 | BOX(116.404 39.915,116.408 39.918) | 116.404 | 116.408 |
|  2 | BOX(121.473 31.230,121.478 31.235) | 121.473 | 121.478 |
+----+-----------------------------------+---------+---------+
2 rows in set (0.00 sec)
```

-- 测试 box3d 查询



```
mysql> SELECT
    ->     id,
    ->     box3d,
    ->     gdb_ST_ZMin(box3d) AS zmin
    -> FROM fake_spatial;
+----+-----------------------------------------------+--------+
| ID | BOX3D                                         | ZMIN   |
+----+-----------------------------------------------+--------+
|  1 | BOX3D(116.404 39.915 0,116.408 39.918 160)    | 39.915 |
|  2 | BOX3D(121.473 31.230 0,121.478 31.235 50)     |  31.23 |
+----+-----------------------------------------------+--------+
2 rows in set (0.00 sec)
```

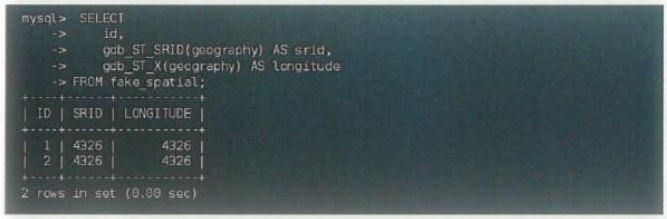
-- 测试 geometry 查询

```
mysql> SELECT
    ->     id,
    ->     gdb_ST_AsText(geometry) AS geom_text
    -> FROM fake_spatial;
+----+-----------------------------------------------------------------------+
| ID | GEOM_TEXT                                                             |
+----+-----------------------------------------------------------------------+
|  1 | POLYGON((116.404 39.915,116.408 39.915,116.408 39.918,116.404 39.918)) |
|  2 | LINESTRING(121.473 31.230,121.478 31.235)                             |
+----+-----------------------------------------------------------------------+
2 rows in set (0.01 sec)
```

-- 测试 geometry_dump 查询

```
mysql> select geometry_dump from fake_spatial;
+-----------------------------------------------------------------------+
| GEOMETRY_DUMP                                                         |
+-----------------------------------------------------------------------+
| {"path": "/dummy/path", "geom": "POINT(116.406 39.916)"}              |
| {"path": "/dummy/path2", "geom": "POINT(121.475 31.232)"}             |
+-----------------------------------------------------------------------+
2 rows in set (0.01 sec)
```

-- 测试 geography 查询



```
mysql>  SELECT
    ->      id,
    ->      gcb_ST_SRID(geography) AS srid,
    ->      gcb_ST_X(geography) AS longitude
    -> FROM fake_spatial;
+----+------+-----------+
| ID | SRID | LONGITUDE |
+----+------+-----------+
|  1 | 4326 |      4326 |
|  2 | 4326 |      4326 |
+----+------+-----------+
2 rows in set (0.00 sec)
```

备注：

1.2 函数

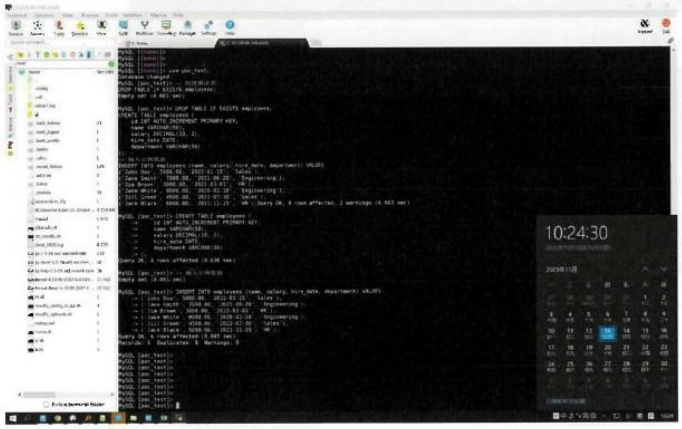
序号：UseCase102
测试项目：函数
测试目的：验证是否支持各类函数
预置条件： 1) 数据库正常运行
测试步骤： 1) 验证是否支持聚合函数包括：COUNT/SUM/AVG/DISTINCT/MAX/MIN/GROUP_CONCAT/STDDEV（计算标准差）/VARIANCE（计算方差）； 2) 验证是否支持字符函数包括：LEFT/RIGHT/UPPER/LENGTH/SUBSTRING/TRIM/LIKE/CONCAT（字符串连接）/REPLACE（字符串替换）/POSITION（查找子字符串位置）/LPAD/RPAD（左/右填充字符串）； 3) 验证是否支持数值函数包括：DIV/MOD/ABS/ROUND（四舍五入）/FLOOR（向下取整）/CEIL/CEILING（向上取整）/SIGN（返回数的符号）/POWER/POW（幂运算）/SQRT（开平方）； 4) 验证是否支持类型转换函数：TO_CHAR/TO_DATE/TO_NUMBER/CONVERT； 5) 验证是否支持的日期和时间函数包括：CURTIME/TIME_FORMAT/DATE_FORMAT/STR_TO_DATE/DATEADD（日期加法）/DATE_SUB/DATEDIFF（日期减法）/DAYNAME/MONTHNAME（返回星期名/月份名）/YEAR/MONTH/DAY（提取年/月/日）/ 是否支持 NOW()函数，NOW()值显示形式； 6) 验证是否支持加密解密函数包括：AES_ENCRYPT/AES_DECRYPT/非对称加密； 7) 验证是否支持行列转行函数 PIVOT(支持使用多个聚合函数)/UNPIVOT（支持多个列）； 8) 其他：系统函数（用于获取系统信息，如当前用户、数据库版本等）、条件逻辑函数（CASE 语句、IF 函数）、窗口函数（如 RANK、DENSE_RANK、ROW_NUMBER 等）。

预期结果：

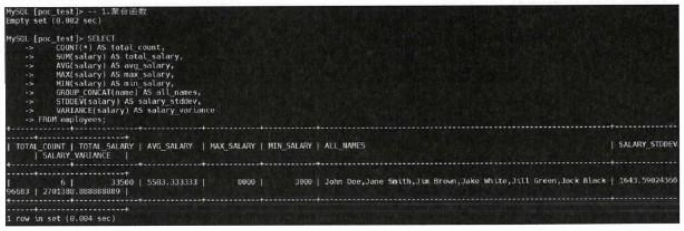
1) 数据库支持各类函数运算。

测试结果：

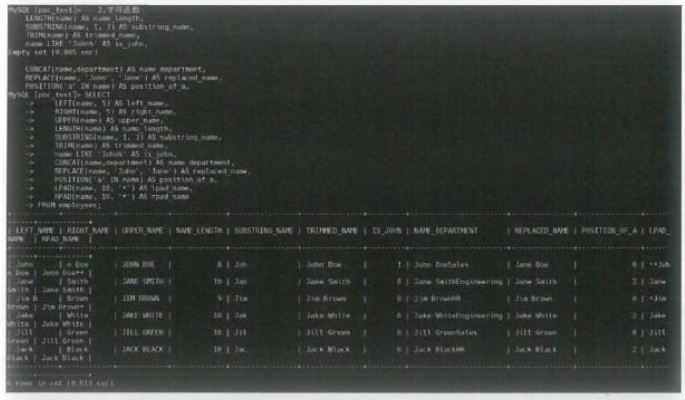
创建数据库，插入数据



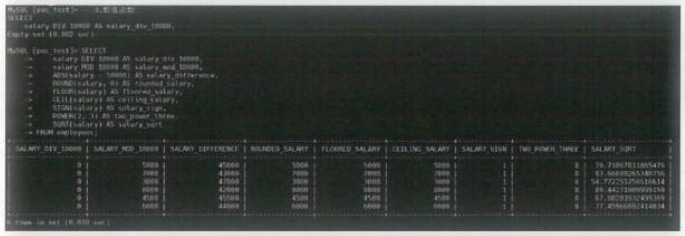
聚合函数



字符函数



数值函数



类型转换函数

```
MySQL [poc_test]> -- 4.类型转换函数
Empty set (0.002 sec)

MySQL [poc_test]> SELECT
    ->     TO_CHAR(salary) AS salary_char,
    ->     TO_DATE('2024-01-01', 'YYYY-MM-DD') AS date_value,
    ->     TO_NUMBER('1234.56') AS number_value,
    ->     CONVERT(name USING utf8) AS utf8_name
    -> FROM employees;
+-------------+---------------------+--------------+------------+
| SALARY_CHAR | DATE_VALUE          | NUMBER_VALUE | UTF8_NAME  |
+-------------+---------------------+--------------+------------+
| 5000        | 2024-01-01 00:00:00 |      1234.56 | John Doe   |
| 7000        | 2024-01-01 00:00:00 |      1234.56 | Jane Smith |
| 3000        | 2024-01-01 00:00:00 |      1234.56 | Jim Brown  |
| 8000        | 2024-01-01 00:00:00 |      1234.56 | Jake White |
| 4500        | 2024-01-01 00:00:00 |      1234.56 | Jill Green |
| 6000        | 2024-01-01 00:00:00 |      1234.56 | Jack Black |
+-------------+---------------------+--------------+------------+
6 rows in set (0.008 sec)
```

日期和时间函数

加密解密函数

```
MySQL [poc_test]> -- 6.加密解密函数
Empty set (0.005 sec)

MySQL [poc_test]> SELECT
    ->     AES_ENCRYPT('mysecret', 'encryption_key') AS encrypted_value,
    ->     AES_DECRYPT(AES_ENCRYPT('mysecret', 'encryption_key'), 'encryption_key') AS decrypted_value
    -> FROM employees;
+--------------------+-----------------+
| ENCRYPTED_VALUE    | DECRYPTED_VALUE |
+--------------------+-----------------+
| □Y  6a N SE        | mysecret        |
| □Y  6a N SE        | mysecret        |
| □Y  6a N SE        | mysecret        |
| □Y  6a N SE        | mysecret        |
| □Y  6a N SE        | mysecret        |
| □Y  6a N SE        | mysecret        |
+--------------------+-----------------+
6 rows in set (0.004 sec)
```

行列转行函数

验证行转列函数 PIVOT

```
goldendb_client> SELECT SELECT DEPT_ID,CHUJI_SUM_SALARY,CHUJI_AVG_SALARY,GAOJI_SUM_SALARY,GAOJI_AVG_SALARY
    -> FROM (
    ->     SELECT DEPT_ID, JOB_LEVEL, SALARY
    ->     FROM EMPLOYEE_SALARY
    -> )
    -> PIVOT (
    ->     SUM(SALARY) AS SUM_SALARY,
    ->     AVG(SALARY) AS AVG_SALARY
    ->     FOR JOB_LEVEL IN ('初级' AS chuji, '高级' AS gaoji)
    -> )
    -> ORDER BY DEPT_ID;
+---------+-----------------+-----------------+-----------------+-----------------+
| DEPT_ID | CHUJI_SUM_SALARY | CHUJI_AVG_SALARY | GAOJI_SUM_SALARY | GAOJI_AVG_SALARY |
+---------+-----------------+-----------------+-----------------+-----------------+
| 1       | 102000          | 51000           | 165000          | 82500           |
| 2       | 113000          | 56500           | 175000          | 87500           |
+---------+-----------------+-----------------+-----------------+-----------------+
2 rows in set (0.00 sec)
```

验证列转行函数 UNPIVOT

```
goldendb_client> SELECT STUDENT_ID,STUDENT_NAME,SUBJECT,SCORE
    -> FROM STUDENT_GRADES_WIDE
    -> UNPIVOT (
    ->     SCORE
    ->     FOR SUBJECT IN (
    ->         CHINESE_SCORE AS 'chinese',
    ->         MATH_SCORE AS 'math',
    ->         ENGLISH_SCORE AS 'english',
    ->         SCIENCE_SCORE AS 'science'
    ->     )
    -> )
    -> ORDER BY STUDENT_ID, SUBJECT;
+------------+--------------+---------+-------+
| STUDENT_ID | STUDENT_NAME | SUBJECT | SCORE |
+------------+--------------+---------+-------+
|          1 | 张三         | chinese |    85 |
|          1 | 张三         | math    |    92 |
|          1 | 张三         | english |    78 |
|          1 | 张三         | science |    88 |
|          2 | 李四         | chinese |    90 |
|          2 | 李四         | math    |    85 |
|          2 | 李四         | english |    92 |
|          2 | 李四         | science |    80 |
|          3 | 王五         | chinese |    78 |
|          3 | 王五         | math    |    88 |
|          3 | 王五         | english |    85 |
|          3 | 王五         | science |    90 |
|          4 | 赵六         | chinese |    92 |
|          4 | 赵六         | math    |    76 |
|          4 | 赵六         | english |    89 |
|          4 | 赵六         | science |    84 |
+------------+--------------+---------+-------+
16 rows in set (0.00 sec)
```

其他

系统函数

```
MySQL [poc_test]> -- 系统函数
Empty set (0.012 sec)

MySQL [poc_test]> SELECT  CURRENT_USER() from dual;
+----------------+
| CURRENT_USER() |
+----------------+
| dbproxy@%      |
+----------------+
1 row in set (0.011 sec)

MySQL [poc_test]> select     VERSION() from dual;
+-----------+
| VERSION() |
+-----------+
| 8.9.99    |
+-----------+
1 row in set (0.002 sec)
```

```
MySQL [poc_test]> -- CASE 语句
Empty set (0.009 sec)

MySQL [poc_test]> SELECT
    ->     name,
    ->     CASE
    ->         WHEN salary < 3000 THEN 'Low'
    ->         WHEN salary BETWEEN 3000 AND 7000 THEN 'Medium'
    ->         ELSE 'High'
    ->     END AS salary_range
    -> FROM employees;
+------------+--------------+
| NAME       | SALARY_RANGE |
+------------+--------------+
| John Doe   | Medium       |
| Jane Smith | Medium       |
| Jim Brown  | Medium       |
| Jake White | High         |
| Jill Green | Medium       |
| Jack Black | Medium       |
+------------+--------------+
6 rows in set (0.006 sec)
```

```
mysql> SELECT
    ->     name,
    ->     salary,
    ->     RANK() OVER (ORDER BY salary DESC) AS salary_rank,
    ->     DENSE_RANK() OVER (ORDER BY salary DESC) AS dense_salary_rank,
    ->     ROW_NUMBER() OVER (ORDER BY salary DESC) AS row_number
    -> FROM employees;
+------------+--------+-------------+-------------------+------------+
| NAME       | SALARY | SALARY_RANK | DENSE_SALARY_RANK | ROW_NUMBER |
+------------+--------+-------------+-------------------+------------+
| Jake White |   8000 |           1 |                 1 |          1 |
| Jane Smith |   7000 |           2 |                 2 |          2 |
| Jack Black |   6000 |           3 |                 3 |          3 |
| John Doe   |   5000 |           4 |                 4 |          4 |
| Jill Green |   4500 |           5 |                 5 |          5 |
| Jim Brown  |   3000 |           6 |                 6 |          6 |
+------------+--------+-------------+-------------------+------------+
6 rows in set (0.03 sec)
```

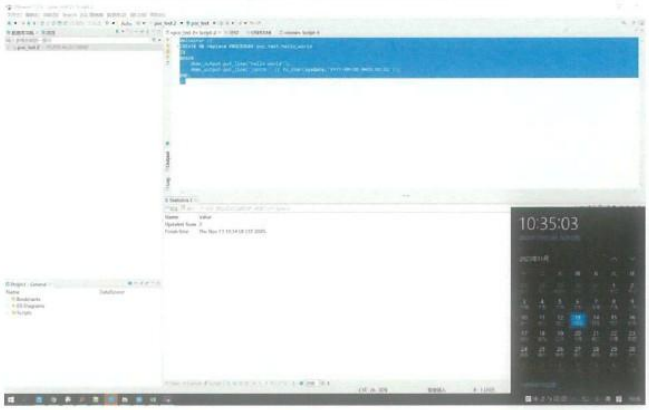
备注：

1.3 存储过程调试

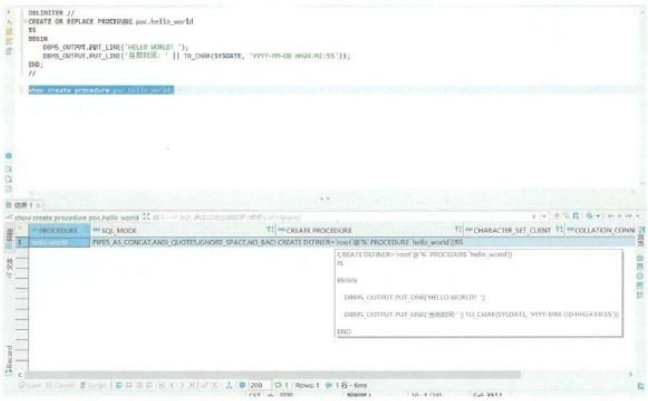
序号：UseCase103
测试项目：存储过程调试
测试目的：验证是否支持图形化存储过程调试工具
预置条件： 1) 数据库正常运行
测试步骤： 1) 使用图形化存储过程调试工具进行调试。
预期结果： 1) 调试成功。

测试结果：

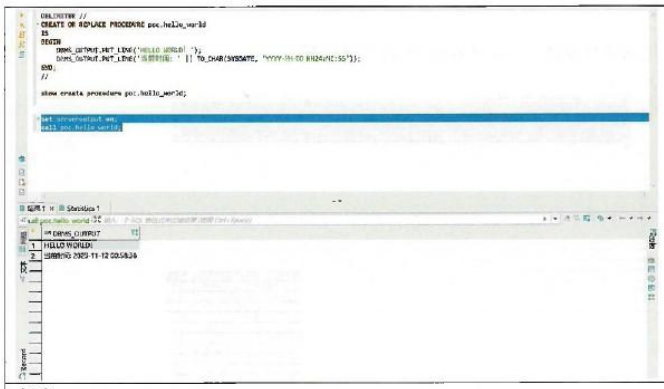
1、使用主流图形化存储过程调试工具 dbeaver 创建存储过程



2、展示该存储过程 ddl



3、执行该存储过程

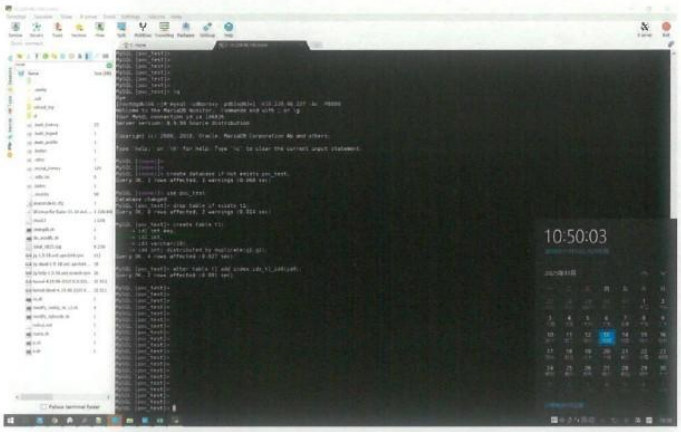


备注：

1.4 执行计划

序号：UseCase104
测试项目：执行计划
测试目的：验证是否支持查看执行计划
预置条件： 1) 数据库正常运行
测试步骤： 1) 使用 explain 查看执行计划； 2) 使用 hint 强制干预查询执行路径； 3) 执行计划自动绑定。
预期结果： 1) 查看成功； 2) 执行成功； 3) 结果符合预期。
测试结果： 1、创建测试库表 create database if not exists poc_test; use poc_test drop table if exists t1; create table t1(id1 int key, id2 int, id3 varchar(10), id4 int) distributed by duplicate(g1,g2);

alter table t1 add index idx_t1_id4(id4);



2、查看语句的原始执行计划

explain select id2 from t1 where id4 = 10 and id1 < 1000\G

```
MySQL [poc_test]> explain select id2 from t1 where id4 = 10 and id1 < 1000\G
*************************** 1. row ***************************
           ID: 10001
  SELECT_TYPE: SQLNode
        TABLE: 1
   PARTITIONS:
         TYPE:
POSSIBLE_KEYS:  select id2 from `poc_test`.t1 where id4 = 10 and id1 < 1000
          KEY: Cluster1,g2
      KEY_LEN:
          REF: Parent=NULL,Child=NULL,Next=NULL
         ROWS:
     FILTERED:
        EXTRA: 4, poc_test.t1=rep
*************************** 2. row ***************************
           ID: 1
  SELECT_TYPE: SIMPLE
        TABLE: t1
   PARTITIONS: NULL
         TYPE: range
POSSIBLE_KEYS: PRIMARY,idx_t1_id4
          KEY: PRIMARY
      KEY_LEN: 4
          REF: NULL
         ROWS: 1
     FILTERED: 100.00
        EXTRA: Using where, g2
2 rows in set (0.002 sec)
```

3、支持 hint

explain /*!gdb_isolation_grade=ur*/select id2 from t1 where id4 = 10 and id1 < 1000\G

```
MySQL [poc_test]> explain /*!gdb_isolation_grade=ur*/select id2 from t1 where id4 = 10 and id1 < 1000\G
*************************** 1. row ***************************
           ID: 10001
  SELECT_TYPE: SQLNode
        TABLE: 1
   PARTITIONS:
         TYPE:
POSSIBLE_KEYS:  select id2 from `poc_test`.t1 where id4 = 10 and id1 < 1000
          KEY: Cluster1.g1
      KEY_LEN:
          REF: Parent=NULL,Child=NULL,Next=NULL
         ROWS:
     FILTERED:
        EXTRA: 1, poc_test.t1=rep
*************************** 2. row ***************************
           ID: 1
  SELECT_TYPE: SIMPLE
        TABLE: t1
   PARTITIONS: NULL
         TYPE: range
POSSIBLE_KEYS: PRIMARY,idx_t1_id4
          KEY: PRIMARY
      KEY_LEN: 4
          REF: NULL
         ROWS: 1
     FILTERED: 100.00
        EXTRA: Using where, g1
2 rows in set (0.002 sec)
```

4、使用 OUTLINE 绑定执行计划

CREATE OUTLINE ot101 ON SELECT/*+ INDEX(t1 idx_t1_id4) */ id2 FROM t1 WHERE id4 = 10 and id1 < 1000;

```
MySQL [poc_test]> CREATE OUTLINE ot101 ON SELECT/*+ INDEX(t1 idx_t1_id4) */ id2 FROM t1 WHERE id4 = 10 and id1 < 1000;
Query OK, 0 rows affected (0.010 sec)
```

5、查看执行计划缓存

select * from mysql.outline\G

```
MySQL [poc_test]> select * from mysql.outline\G
*************************** 1. row ***************************
      DIGEST: 962a0d1e190a5911fe7347db5b1d6ee51eb2f9ee5a238c0ebb9a381ae558984
DATABASE_NAME: NULL
 OUTLINE_NAME: ot101
      PATTERN: /* select#1 */ select `id2` from `poc_test`.`t1` where ((`id4` = ?) and (`id1` < ?))
    STATEMENT: SELECT /*+ SET_VAR(active_gtm_gtid_list='A0AAAgBWcwAAAAAAFkLAAAAAAAA') SET_VAR(next_gtm_gtid=2906)*//*+ INDEX(t1 idx_t1_id4) */
id2 FROM `poc_test`.`t1` WHERE id4 = 10 and id1 < 1000
      LITERAL: [1]
         HINT: {"1": [[10, true, "t1", null, "idx_t1_id4"]]}
      ENABLED: YES
  UPDATE_TIME: 2024-09-24 16:39:47
1 row in set (0.003 sec)
```

6、进行 explain 判断执行计划是否绑定成功

explain select id2 from t1 where id4 = 10 and id1 < 1000\G

```
MySQL [poc_test]> explain select id2 from t1 where id4 = 10 and id1 < 1000\G
*************************** 1. row ***************************
           ID: 10001
  SELECT_TYPE: SQLNode
        TABLE: 1
   PARTITIONS:
         TYPE:
POSSIBLE_KEYS:  select id2 from `poc_test`.t1 where id4 = 10 and id1 < 1000
          KEY: Cluster1,g1
      KEY_LEN:
          REF: Parent=NULL,Child=NULL,Next=NULL
         ROWS:
     FILTERED:
        EXTRA: 4, poc_test.t1=rep
*************************** 2. row ***************************
           ID: 1
  SELECT_TYPE: SIMPLE
        TABLE: t1
   PARTITIONS: NULL
         TYPE: range
POSSIBLE_KEYS: idx_t1_id4
          KEY: idx_t1_id4
      KEY_LEN: 9
          REF: NULL
         ROWS: 1
     FILTERED: 100.00
        EXTRA: Using index condition; Using outline, g1
2 rows in set (0.007 sec)
```

7、删除 OUTLINE

DROP OUTLINE otl01;

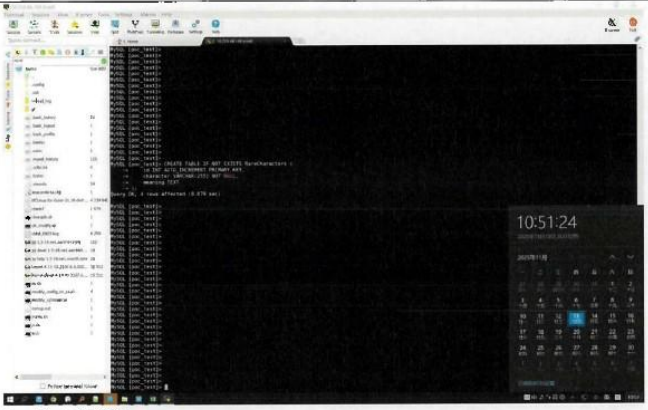
select * from mysql.outline\G

```
MySQL [poc_test]> DROP OUTLINE otl01;
Query OK, 0 rows affected (0.004 sec)

MySQL [poc_test]> select * from mysql.outline\G
Empty set (0.005 sec)
```

备注：

1.5 生僻字

序号：UseCase105
测试项目：生僻字
测试目的：验证是否支持生僻字存储
预置条件： 1）数据库正常运行
测试步骤： 1）创建表，插入生僻字，查询生僻字，验证是否支持生僻字存储。
预期结果： 1）结果符合预期，支持生僻字存储。
测试结果： 1、创建表



2、插入生僻字

```
MySQL [poc]> INSERT INTO RareCharacters (character, meaning) VALUES
    -> ('龘', '龙腾飞的样子'),
    -> ('靐', '雷声'),
    -> ('齉', '鼻子不通气，发音不清'),
    -> ('爨', '用力的样子'),
    -> ('灪', '烧火做饭');
Query OK, 5 rows affected (0.010 sec)
Records: 5  Duplicates: 0  Warnings: 0
```

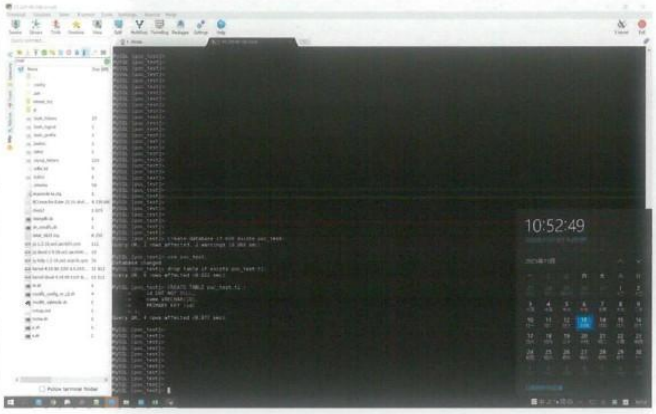
3、查询生僻字

```
MySQL [poc]> select * from  RareCharacters;
+----+-----------+--------------------------------+
| ID | CHARACTER | MEANING                        |
+----+-----------+--------------------------------+
|  1 | 龘        | 龙腾飞的样子                   |
|  2 | 靐        | 雷声                           |
|  3 | 齉        | 鼻子不通气，发音不清           |
|  4 | 爨        | 用力的样子                     |
|  5 | 灪        | 烧火做饭                       |
+----+-----------+--------------------------------+
5 rows in set (0.002 sec)

MySQL [poc]>
```

备注：

1.6 闪回

序号：UseCase106
测试项目：闪回
测试目的：验证是否支持闪回功能
预置条件：

1) 数据库正常运行
测试步骤： 1) 使用数据库闪回 flashback 功能。
预期结果： 1) 闪回操作成功.
测试结果： 分布式数据库支持分区表的 drop 表、truncate 表闪回和闪回查询 1、验证删除表、truncate 表后的闪回功能 1)创建表并预置数据    2)下图显示，GoldenDB 支持删除表后闪回

```
mysql> -- 3、删除表
Empty set (0.00 sec)

mysql> drop table poc_test.case_flash ;
Query OK, 0 rows affected (0.07 sec)

mysql> -- 4、查询表数据（报错表不存在）
Empty set (0.00 sec)

mysql> select * from  poc_test.case_flash order by id;
ERROR 942 (HY000): ORA-00942:Table or View 'poc_test.case_flash' doesn't exist
mysql>
mysql> -- 5、闪回表结构和表数据
Empty set (0.00 sec)

mysql> flashback table poc_test.case_flash to before drop;
Query OK, 0 rows affected (0.05 sec)

mysql>
mysql> -- 6、查询表数据（正常查询）
Empty set (0.00 sec)

mysql> select * from  poc_test.case_flash order by id;
+-------+-----------+------+-------+------+------+---------+---------+-------+
| ID    | HOTELNAME | YEAR | MONTH | DAY  | RENT | REMARK1 | REMARK2 | DATE1 |
+-------+-----------+------+-------+------+------+---------+---------+-------+
|     1 | rujia     | 1988 |     1 |    1 |  198 | NULL    | NULL    | NULL  |
|     2 | rujia     | 1988 |    10 |    2 |  199 | NULL    | NULL    | NULL  |
|     3 | NULL      | 1988 |     2 | NULL |  199 | NULL    | NULL    | NULL  |
|  1001 | shiji2    | 1920 |     6 |    1 | 3192 | NULL    | NULL    | NULL  |
|  2001 | shiji3    | 2020 |     7 |    1 | 3193 | NULL    | NULL    | NULL  |
|  3001 | shiji4    | 2020 |     8 |    1 | 3194 | NULL    | NULL    | NULL  |
| 19001 | shiji20   | 2013 |     1 |    1 | 3120 | NULL    | NULL    | NULL  |
+-------+-----------+------+-------+------+------+---------+---------+-------+
7 rows in set (0.00 sec)
```

3)下图显示，GoldenDB 支持 TRUNCATE 表后闪回

```
mysql> -- 7、清空表
Empty set (0.00 sec)

mysql> truncate table poc_test.case_flash ;
Query OK, 0 rows affected (0.05 sec)

mysql>
mysql> -- 8、查询表数据（查询结果为空）
Empty set (0.00 sec)

mysql> select * from  poc_test.case_flash order by id;
Empty set (0.01 sec)

mysql> -- 9、闪回表结构和表数据
Empty set (0.00 sec)

mysql> flashback table poc_test.case_flash to before truncate;
Query OK, 0 rows affected (0.08 sec)

mysql> -- 10、查询表数据 （正常查询）
Empty set (0.00 sec)

mysql> select * from poc_test.case_flash order by id;
+-------+-----------+------+-------+------+-------+---------+---------+---------+
| ID    | HOTELNAME | YEAR | MONTH | DAY  | RENT  | REMARK1 | REMARK2 | DATE1   |
+-------+-----------+------+-------+------+-------+---------+---------+---------+
|     1 | rujia     | 1988 |     1 |    1 |   198 | NULL    | NULL    | NULL    |
|     2 | rujia     | 1988 |    10 |    2 |   199 | NULL    | NULL    | NULL    |
|     3 | NULL      | 1988 |     2 | NULL |   199 | NULL    | NULL    | NULL    |
|  1001 | shiji2    | 1920 |     6 |    1 |  3192 | NULL    | NULL    | NULL    |
|  2001 | shiji3    | 2020 |     7 |    1 |  3193 | NULL    | NULL    | NULL    |
|  3001 | shiji4    | 2020 |     8 |    1 |  3194 | NULL    | NULL    | NULL    |
| 19001 | shiji20   | 2013 |     1 |    1 |  3120 | NULL    | NULL    | NULL    |
+-------+-----------+------+-------+------+-------+---------+---------+---------+
7 rows in set (0.00 sec)
```

2、验证 as of timestamp 闪回查询功能

1)创建测试库表并预置数据

```
Empty set (0.00 sec)

mysql> create database if not exists poc_test;
Query OK, 1 row affected, 1 warning (0.02 sec)

mysql> use poc_test;
Database changed
mysql> drop table if exists poc_test.t1;
Query OK, 0 rows affected, 1 warning (0.03 sec)

mysql> create table poc_test.t1 (id int primary key,name varchar(10)) distributed by hash(id)(g1);
Query OK, 0 rows affected (0.06 sec)

mysql>
mysql> insert into poc_test.t1 values (1,'zte1'),(2,'zte2');
Query OK, 2 rows affected (0.01 sec)
Records: 2  Duplicates: 0  Warnings: 0

mysql> select * from poc_test.t1;
+----+------+
| ID | NAME |
+----+------+
|  1 | zte1 |
|  2 | zte2 |
+----+------+
2 rows in set (0.01 sec)
```

2)等待 1 分钟后插入新数据并执行闪回查询,下图显示支持闪回查询功能

```
mysql> -- 创建库表并预置数据
Empty set (0.00 sec)

mysql> create database if not exists poc_test;
Query OK, 1 row affected, 1 warning (0.03 sec)

mysql> use poc_test;
Database changed
mysql> drop table if exists poc_test.t1;
Query OK, 0 rows affected (0.11 sec)

mysql> CREATE TABLE poc_test.t1 (
    ->     id INT NOT NULL,
    ->     name VARCHAR(10),
    ->     PRIMARY KEY (id)
    -> )
    -> PARTITION BY HASH(id) PARTITIONS 4
    -> distributed by hash(id)(g1);
Query OK, 0 rows affected (0.15 sec)

mysql>
mysql> insert into poc_test.t1 values (1,'zte1'),(2,'zte2');
Query OK, 2 rows affected (0.01 sec)
Records: 2  Duplicates: 0  Warnings: 0

mysql> select * from poc_test.t1;
+----+------+
| ID | NAME |
+----+------+
|  1 | zte1 |
|  2 | zte2 |
+----+------+
2 rows in set (0.00 sec)
```

3、验证 as of scn 闪回查询功能

1)创建测试库表

```
mysql> create database if not exists poc_test;
Query OK, 1 row affected, 1 warning (0.02 sec)

mysql> use poc_test;
Database changed
mysql> drop table if exists poc_test.t1;
Query OK, 0 rows affected (0.05 sec)

mysql> CREATE TABLE poc_test.t1 (
    ->     id INT NOT NULL,
    ->     name VARCHAR(10),
    ->     PRIMARY KEY (id)
    -> )
    -> PARTITION BY HASH(id) PARTITIONS 4
    -> distributed by hash(id)(g1);
Query OK, 0 rows affected (0.09 sec)

mysql>
mysql>
mysql> insert into poc_test.t1 values (1,'zte1'),(2,'zte2');
Query OK, 2 rows affected (0.01 sec)
Records: 2  Duplicates: 0  Warnings: 0

mysql> select * from poc_test.t1;
+----+------+
| ID | NAME |
+----+------+
|  1 | zte1 |
|  2 | zte2 |
+----+------+
2 rows in set (0.01 sec)

mysql>
mysql> -- 查询当前scn
Empty set (0.00 sec)

mysql> select current_scn from VSdatabase;
+-------------+
| CURRENT_SCN |
+-------------+
| 863743      |
+-------------+
1 row in set (0.03 sec)
```

2)插入新数据，通过 scn 闪回查询，下图显示支持 scn 闪回查询

```
mysql> -- 插入新数据并查询表数据
Empty set (0.00 sec)

mysql> insert into poc_test.t1 values (3,'zte3'),(4,'zte4');
Query OK, 2 rows affected (0.03 sec)
Records: 2  Duplicates: 0  Warnings: 0

mysql> select * from poc_test.t1;
+----+------+
| ID | NAME |
+----+------+
|  4 | zte4 |
|  1 | zte1 |
|  2 | zte2 |
|  3 | zte3 |
+----+------+
4 rows in set (0.00 sec)

mysql> select * from poc_test.t1 as of scn 863743;
+----+------+
| ID | NAME |
+----+------+
|  1 | zte1 |
|  2 | zte2 |
+----+------+
2 rows in set (0.00 sec)
```

备注：

1.7 字符集

序号：UseCase107
测试项目：字符集
测试目的：验证是否支持各类字符集
预置条件： 1) 数据库正常运行
测试步骤： 1) 验证是否支持 GBK、GB18030、UTF-8、we8iso8859p1、Latin1、SQL_ASCII、GB18030-2022； 2) Oracle 中存在 we8iso8859p1 字符集数据，导入信创库转换为 UTF-8 字符集，查看数据是否显示乱码； 3) DB2 中存在 GBK 字符集数据，导入信创库转换为 UTF-8 字符集，查看数据是否显示乱码。
预期结果： 1) 数据库支持各类标准字符集；

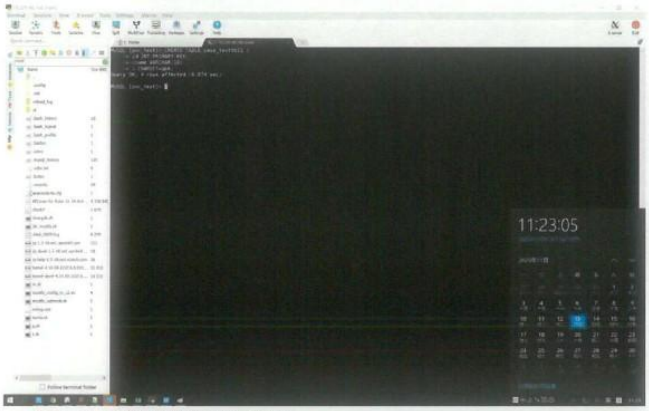
2) 数据显示无乱码；

3) 数据显示无乱码。

测试结果：

1.验证是否支持 GBK、GB18030、UTF-8、we8iso8859p1、Latin1、SQL_ASCII、GB18030-2022；

-- 创建表并指定字符集 gbk



-- 修改表并指定字符集 GB18030

```
MySQL [poc]> ALTER TABLE cmss_testtbl1 CONVERT TO CHARACTER SET  GB18030 ;
Query OK, 0 rows affected (0.034 sec)
```

-- 修改表并指定字符集 UTF-8

```
MySQL [poc]> ALTER TABLE cmss_testtbl1 CONVERT TO CHARACTER SET  UTF8 ;
Query OK, 0 rows affected, 4 warnings (0.032 sec)
```

-- 修改表并指定字符集 we8iso8859p1

```
MySQL [poc]> ALTER TABLE cmss_testtbl1 CONVERT TO CHARACTER SET  we8iso8859p1 ;
Query OK, 0 rows affected (0.033 sec)
```

-- 修改表并指定字符集 Latin1

```
MySQL [poc]> ALTER TABLE cmss_testtbl1 CONVERT TO CHARACTER SET  Latin1 ;
Query OK, 0 rows affected (0.031 sec)
```

-- 修改表并指定字符集 SQL_ASCII

```
MySQL [poc]> ALTER TABLE cmss_testtbl1 CONVERT TO CHARACTER SET  ASCII ;
Query OK, 0 rows affected (0.032 sec)
```

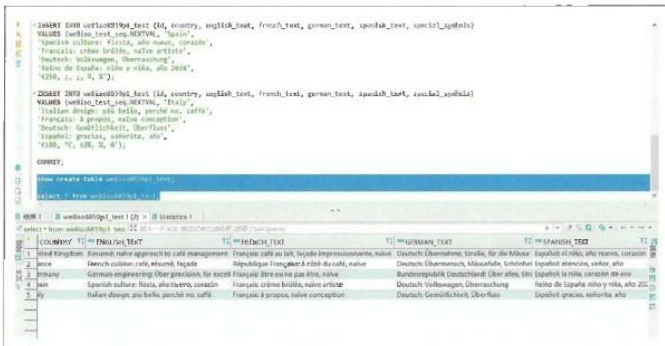
-- 修改表并指定字符集 GB18030

```
MySQL [poc]> ALTER TABLE cmss_testtbl1 CONVERT TO CHARACTER SET  GB18030-2022;
Query OK, 0 rows affected (0.021 sec)
```

2.we8iso8859p1 字符集数据，导入信创库转换为 UTF-8 字符集，查看数据是否显示乱码；

--创建表并插入数据

---查看表字符集以及数据是否乱码



3.GBK 字符集数据，导入信创库转换为 UTF-8 字符集，查看数据是否显示乱码

--创建表并插入数据

--查看表字符集以及数据是否乱码

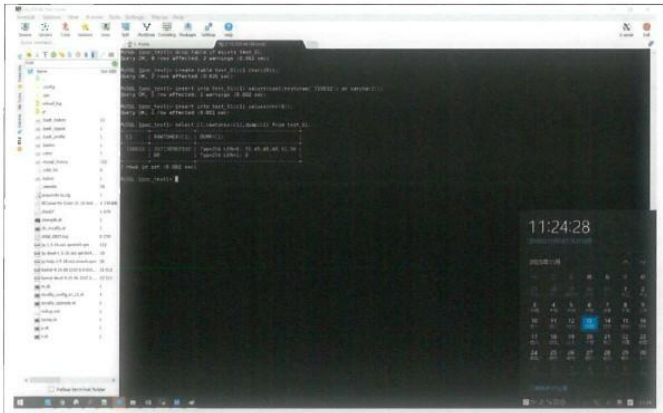
备注：

1.8 低值支持

序号：UseCase108
测试项目：低值支持
测试目的：验证是否支持低值
预置条件： 1) 数据库正常运行

测试步骤：
1) 针对 char 字段，执行以下代码：
drop table if exists test_01;
create table test_01(c1 char(20));
insert into test_01(c1) values(cast_to_varchar2(hextoraw('310032')));
insert into test_01(c1) values(chr(0));
select c1,rawtohex(c1),dump(c1) from test_01;
2) 针对 varchar 字段，执行以下代码：
drop table if exists test_01;
create table test_01(c1 varchar(20));
insert into test_01(c1) values(dbe_raw.cast_to_varchar2(hextoraw('310032')));
insert into test_01(c1) values(chr(0));
select c1,rawtohex(c1),dump(c1) from test_01;

预期结果： 1) 查询成功； 2) 查询成功。
测试结果： char 字段：



varchar 字段：

```
MySQL [poc]> drop table if exists test_01;
Query OK, 2 rows affected (0.021 sec)

MySQL [poc]> create table test_01(c1 varchar(20));
Query OK, 2 rows affected (0.019 sec)

MySQL [poc]> insert into test_01(c1) values(cast(hextoraw('310032') as varchar2));
Query OK, 1 row affected, 2 warnings (0.004 sec)

MySQL [poc]> insert into test_01(c1) values(chr(0));
Query OK, 1 row affected (0.002 sec)

MySQL [poc]> select c1,rawtohex(c1),dump(c1) from test_01;
+--------+-------------+--------------------------------+
| C1     | RAWTOHEX(C1) | DUMP(C1)                       |
+--------+-------------+--------------------------------+
| 310032 | 333130303332 | Typ=15 Len=6: 51,49,48,48,51,50 |
|        | 00           | Typ=15 Len=1: 0                |
+--------+-------------+--------------------------------+
2 rows in set (0.002 sec)

MySQL [poc]>
```

备注：

1.9 索引

序号：UseCase109
测试项目：索引
测试目的：验证是否支持各类索引
预置条件： 1) 数据库正常运行
测试步骤： 1) 在线创建复合索引、全局唯一索引、全局索引、不包含分区键和主键的全局索引、函数索引，使用索引进行查询； 2) 在线创建全文索引，使用 catsearch 全文检索函数（测试完需要将索引删除还原环境）。

预期结果：

1) 索引创建成功，查询成功；

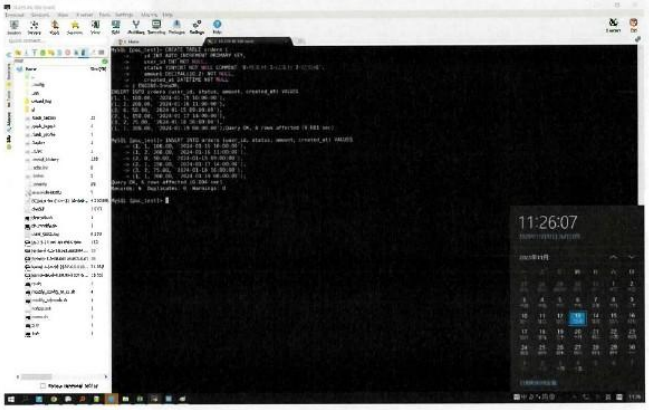
2) 索引创建成功，catsearch 全文索引函数可以正常应用。

测试结果：

1、支持复合索引，全局唯一索引，全局索引，不包含分区键和主键的全局索引、函数索引

--复合索引

1) 创建表并插入数据



2) 创建复合索引

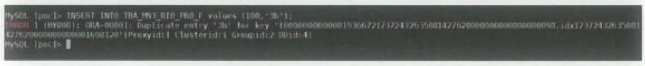
```
MySQL [poc]> ALTER TABLE orders ADD INDEX idx_user_status_time (user_id, status, created_at);
Query OK, 2 rows affected (0.039 sec)

MySQL [poc]>
```

3) 查看执行计划

--全局唯一索引

1) 建表并插入数据

2) 插入重复值检验全局唯一索引



--全局索引

1) 预置表和数据

```
MySQL [poc]> CREATE TABLE t3(
    ->       name VARCHAR (30),
    ->          id int key
    ->     ) distributed by hash(id)(g1,g2);
Query OK, 4 rows affected (0.030 sec)

MySQL [poc]> insert into t3 values('aaa', 1),('bbb',2),('cccc',3);
Query OK, 3 rows affected (0.006 sec)
Records: 3  Duplicates: 0  Warnings: 0

MySQL [poc]>
```

2) 创建不包含分区键和主键的全局索引

```
MySQL [poc]> create global index idx2 on t3(name);
Query OK, 0 rows affected (0.054 sec)

MySQL [poc]>
```

3) 查看执行计划

--不包含分区键和主键的函数索引

1）预置表和数据

```
MySQL [poc]> CREATE TABLE orders (
    ->     id INT PRIMARY KEY AUTO_INCREMENT,
    ->     order_number VARCHAR(20) NOT NULL,
    ->     order_date DATETIME NOT NULL,
    ->     customer_id INT NOT NULL,
    ->     total_amount DECIMAL(10,2)
    -> );
Query OK, 4 rows affected (0.032 sec)

MySQL [poc]> INSERT INTO orders (order_number, order_date, customer_id, total_amount)
    -> VALUES
    ->     ('ORD20230306001', '2025-03-01 10:00:00', 101, 1500.00),
    ->     ('ORD20230306002', '2025-03-05 14:30:00', 102, 2300.50),
    -> ('ORD20230306003', '2025-02-28 09:15:00', 103, 899.99);
Query OK, 3 rows affected (0.006 sec)
Records: 3  Duplicates: 0  Warnings: 0

MySQL [poc]>
```

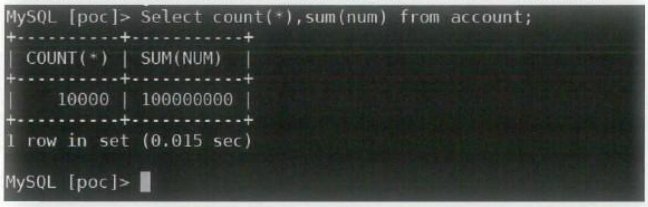
2）针对按月查询场景创建函数索引

```
MySQL [poc]> CREATE INDEX idx_month ON orders ((MONTH(order_date)));
Query OK, 0 rows affected (0.035 sec)

MySQL [poc]>
```

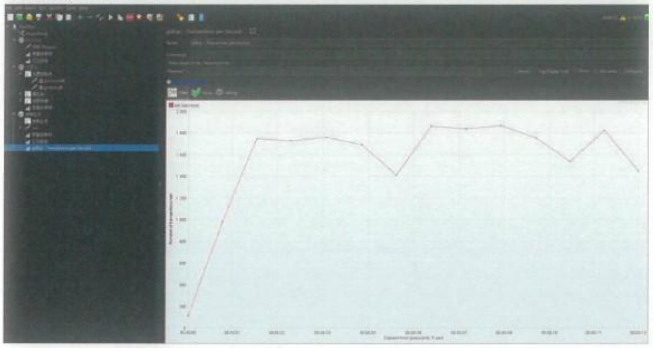
3）查询 3 月份的所有订单

2.验证是否支持在线创建索引（测试完需要将索引删除还原环境）

--使用 jmeter 预置库表和数据，数据为 10000 账户，每账户 10000 金额



--开始压测转账业务

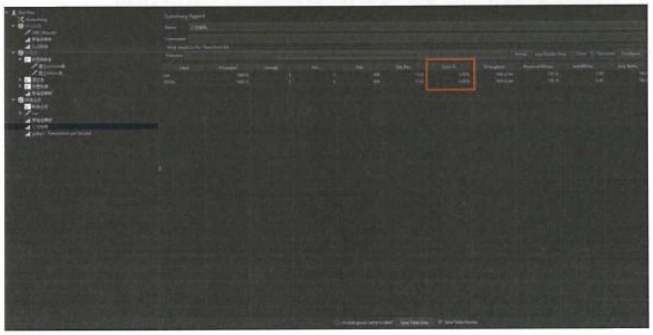


--在线创建索引

```
MySQL [poc]> create index idx1 on history(id1);
Query OK, 0 rows affected (0.008 sec)

MySQL [poc]> show create table history\G
*************************** 1. row ***************************
       TABLE: history
CREATE TABLE: CREATE TABLE `history` (
  `n` int NOT NULL AUTO_INCREMENT,
  `id1` decimal(20,0) DEFAULT NULL,
  `id2` decimal(20,0) DEFAULT NULL,
  `a` varchar(40) DEFAULT NULL,
  PRIMARY KEY (`n`),
  INDEX `idx1` (`id1`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_bin
 DISTRIBUTED BY HASH(`n`)(g1,g2)
1 row in set (0.002 sec)

MySQL [poc>
```

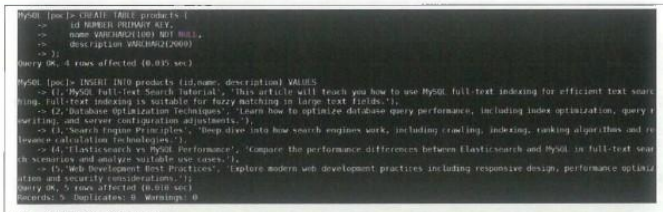
--查看数据一致性以及压测是否有报错

```
MySQL [poc]> Select count(*),sum(num) from account;
+----------+-----------+
| COUNT(*) | SUM(NUM)  |
+----------+-----------+
|    10000 | 100000000 |
+----------+-----------+
1 row in set (0.005 sec)

MySQL [poc]>
```



3.验证是否支持全文索引，支持 catsearch 全文检索函数

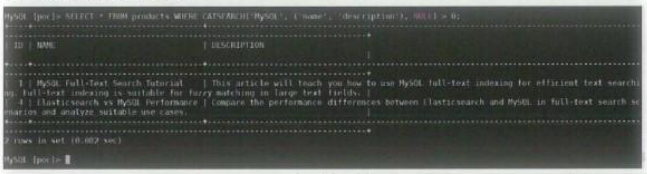
1）建表并插入数据



2) 创建全文索引

```
MySQL [poc]> CREATE INDEX products_fti ON products(name, description)
    ->     INDEXTYPE IS CTXSYS.CONTEXT;
Query OK, 2 rows affected (0.048 sec)
```

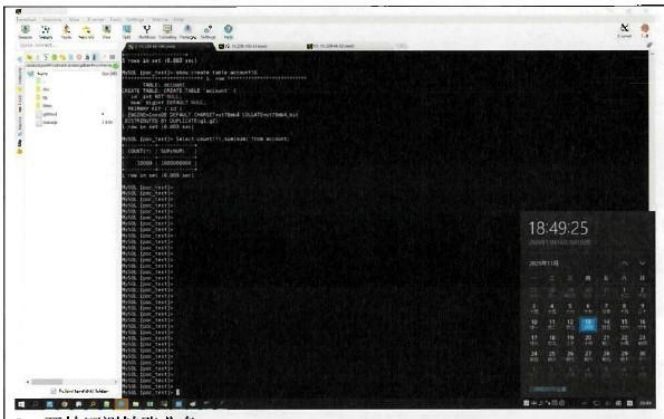
3) 使用 catsearch 查询



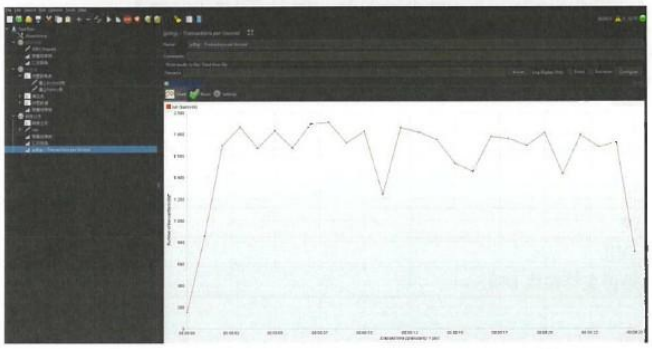
备注：

1.10 在线 DDL

序号：UseCase110
测试项目：在线 DDL
测试目的：验证是否支持在线 DDL
预置条件： 1) 数据库正常运行
测试步骤： 1) 在背景压力下，完成在线 DDL 的测试，主要包括：新增字段、修改字段类型、修改字段长度。
预期结果： 1) 执行成功。
执行结果： 1、使用 jmeter 预置库表和数据，数据为 10000 账户，每账户 10000 金额



2、开始压测转账业务



3、在线 ddl 测试

--新增字段

```
MySQL [poc]> alter table history add column a int;
Query OK, 0 rows affected (0.002 sec)

MySQL [poc]> show create table history\G
*************************** 1. row ***************************
       TABLE: history
CREATE TABLE: CREATE TABLE `history` (
  `n` int NOT NULL AUTO_INCREMENT,
  `id1` decimal(20,0) DEFAULT NULL,
  `id2` decimal(20,0) DEFAULT NULL,
  `a` int DEFAULT NULL,
  PRIMARY KEY (`n`),
  INDEX `idx1` (`id1`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_bin
 DISTRIBUTED BY HASH(`n`)(g1,g2)
1 row in set (0.002 sec)
```

--修改字段类型

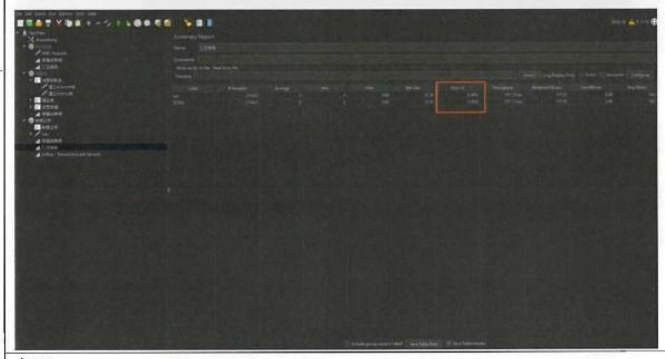
```
MySQL [poc]> ALTER TABLE history MODIFY COLUMN a varchar(10);
Query OK, 0 rows affected (0.013 sec)

MySQL [poc]> show create table history\G
*************************** 1. row ***************************
       TABLE: history
CREATE TABLE: CREATE TABLE `history` (
  `n` int NOT NULL AUTO_INCREMENT,
  `id1` decimal(20,0) DEFAULT NULL,
  `id2` decimal(20,0) DEFAULT NULL,
  `a` varchar(10) DEFAULT NULL,
  PRIMARY KEY (`n`),
  INDEX `idx1` (`id1`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_bin
 DISTRIBUTED BY HASH(`n`)(g1,g2)
1 row in set (0.002 sec)
```

--修改字段长度

```
MySQL [poc]> ALTER TABLE history MODIFY COLUMN a varchar(40);
Query OK, 0 rows affected (0.007 sec)

MySQL [poc]> show create table history\G
*************************** 1. row ***************************
       TABLE: history
CREATE TABLE: CREATE TABLE `history` (
  `n` int NOT NULL AUTO_INCREMENT,
  `id1` decimal(20,0) DEFAULT NULL,
  `id2` decimal(20,0) DEFAULT NULL,
  `a` varchar(40) DEFAULT NULL,
  PRIMARY KEY (`n`),
  INDEX `idx1` (`id1`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_bin
 DISTRIBUTED BY HASH(`n`)(g1,g2)
1 row in set (0.004 sec)
```

4、查看数据一致性以及压测是否有报错

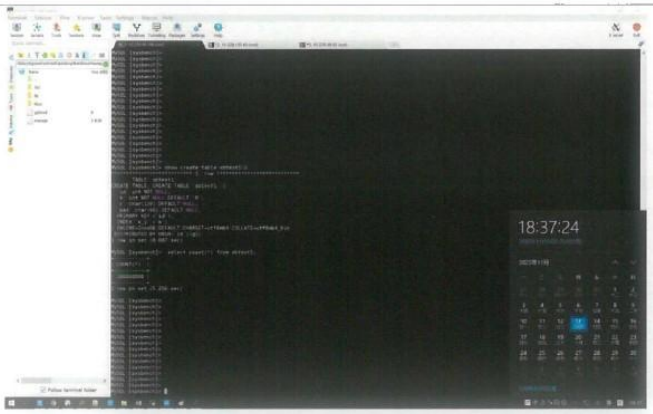
```
MySQL [poc]> Select count(*),sum(num) from account;
+----------+-----------+
| COUNT(*) | SUM(NUM)  |
+----------+-----------+
|    10000 | 100000000 |
+----------+-----------+
1 row in set (0.005 sec)

MySQL [poc]>
```



备注：

1.11 毫秒级加列

序号：UseCase111
测试项目：毫秒级加列
测试目的：验证是否支持毫秒级加列
预置条件： 1) 数据库正常运行
测试步骤： 1) 执行在线表结构变更。
预期结果： 1) 支持不影响生产业务的在线无阻塞 DDL 运维能力，在性能测试背景下，集中式下亿级大表在线增加带缺省值的列可以在毫秒级完成。
测试结果： 1、sysbench 构造 1 亿数据



2、sysbenh 进行压测

3、在线增加带缺省值的列并查看表结构

```
MySQL [sysbench]> alter table sbtest1 add column a int default 0;
Query OK, 0 rows affected (0.187 sec)

MySQL [sysbench]> show create table sbtest1\G
*************************** 1. row ***************************
       TABLE: sbtest1
CREATE TABLE: CREATE TABLE `sbtest1` (
  `id` int NOT NULL,
  `k` int NOT NULL DEFAULT '0',
  `c` char(120) DEFAULT NULL,
  `pad` char(60) DEFAULT NULL,
  `a` int DEFAULT '0',
  PRIMARY KEY (`id`),
  INDEX `k_1` (`k`),
  INDEX `idx1` (`k`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_bin
 DISTRIBUTED BY HASH(`id`)(g1)
1 row in set (0.009 sec)

MySQL [sysbench]>
```

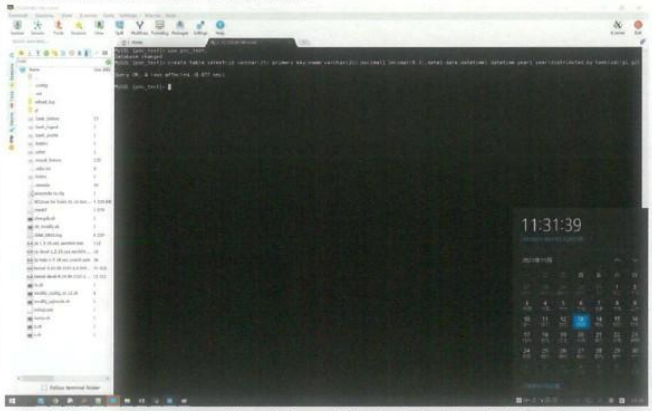
4、查看 sysbench 是否报错

备注：

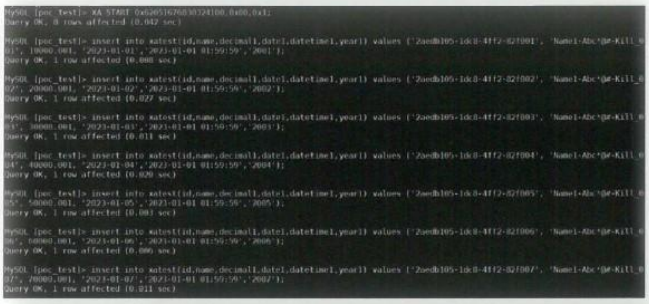
1.12 分布式 XA 事务控制能力

序号：UseCase112
测试项目：分布式 XA 事务控制能力
测试目的：验证是否支持分布式 XA 事务控制
预置条件： 1) 数据库正常运行
测试步骤： 1) 进行分布式 XA 事务控制操作。
预期结果： 1) 执行成功，且支持 XID 为 64 位的 XA 事务和 XA 分支事务。
测试结果：

1.创建测试表 xatest 表，并预置数据：



2.客户端 1，启动一个 XA 分支事务，关联全局事务标识和分支事务 1 标识，事务中执行 dml 语句，在分支事务结束前和结束后分别进行一次 xatest 表的数据查询；



3.客户端 2，对此 XA 事务进行加入操作，关联分支事务 1，在事务中执行 dml 语句，在 join 分支事务结束前和结束后分别进行一次 xatest 表的数据查询；

泰尔实验室 CAICT 中国信通院

检 验 报 告 附 件 一

报告编号：25V01Y001802-001 第 40页 共 287 页

4.客户端 3，对此 XA 事务启动分支事务，分支事务 2，事务中执行 dml 语句，在第 2 个分支事务结束前和结束后分别进行一次 xatest 表的数据查询；

5.任一客户端，对此 XA 事务进行准备提交和提交操作，操作前后分别进行一次 xatest 表的数据查询；

备注：