

GeminiDB Redis

用户指南

文档版本 01
发布日期 2025-11-13



版权所有 © 华为云计算技术有限公司 2025。保留一切权利。

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

商标声明



HUAWEI和其他华为商标均为华为技术有限公司的商标。

本文档提及的其他所有商标或注册商标，由各自的所有人拥有。

注意

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

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

目录

- 1 产品介绍..... 1
 - 1.1 GeminiDB Redis 产品优势..... 1
 - 1.2 GeminiDB Redis 产品架构..... 3
 - 1.3 业务场景推荐..... 6
 - 1.4 兼容接口和版本..... 7
 - 1.5 实例规格说明..... 7
 - 1.6 实例状态..... 12
 - 1.7 约束与限制..... 13
- 2 计费说明..... 17
 - 2.1 计费概述..... 17
 - 2.2 计费模式..... 18
 - 2.2.1 计费模式概述..... 18
 - 2.2.2 包年/包月..... 19
 - 2.2.3 按需计费..... 23
 - 2.3 计费项..... 27
 - 2.4 计费样例..... 29
 - 2.5 变更计费模式..... 31
 - 2.5.1 变更计费模式概述..... 31
 - 2.5.2 按需转包年/包月..... 32
 - 2.5.3 包年/包月转按需..... 34
 - 2.6 续费..... 36
 - 2.6.1 续费概述..... 36
 - 2.6.2 手动续费..... 37
 - 2.6.3 自动续费..... 40
 - 2.7 费用账单..... 42
 - 2.8 欠费说明..... 46
 - 2.9 停止计费..... 47
 - 2.10 成本管理..... 48
 - 2.10.1 成本构成..... 48
 - 2.10.2 成本分配..... 49
 - 2.10.3 成本分析..... 50
 - 2.10.4 成本优化..... 50
 - 2.11 计费 FAQ..... 51

2.11.1 包年/包月和按需计费有什么区别.....	51
2.11.2 GeminiDB Redis 支持包年/包月和按需计费模式相互转换吗.....	51
2.11.3 如何进行单个包年/包月或者批量实例续费.....	51
2.11.4 如何退订包年/包月实例.....	52
3 快速入门.....	55
3.1 快速了解 Redis.....	55
3.2 购买并连接集群版实例.....	56
3.3 购买并连接主备版实例.....	65
3.4 入门实践.....	75
4 用户指南.....	77
4.1 通过 IAM 授予使用 GeminiDB Redis 的权限.....	77
4.1.1 创建用户并授权使用 GeminiDB Redis.....	77
4.1.2 GeminiDB Redis 自定义策略.....	78
4.2 购买 GeminiDB Redis 实例.....	80
4.2.1 购买 GeminiDB Redis 集群版实例.....	80
4.2.2 购买 GeminiDB Redis 主备版实例.....	89
4.3 实例连接及管理.....	97
4.3.1 连接 GeminiDB Redis 实例方式介绍.....	97
4.3.2 通过 DAS 连接 GeminiDB Redis.....	98
4.3.3 通过内网连接 GeminiDB Redis.....	104
4.3.3.1 通过负载均衡地址连接实例（推荐）.....	105
4.3.3.2 通过内网域名连接实例.....	106
4.3.3.3 通过内网 IP 连接实例.....	112
4.3.4 通过公网连接 GeminiDB Redis.....	113
4.3.4.1 通过弹性负载均衡绑定弹性公网 IP 连接实例（推荐）.....	113
4.3.4.2 通过公网域名连接实例.....	116
4.3.4.3 通过弹性公网 IP 连接实例.....	122
4.3.5 通过程序代码连接实例示例.....	123
4.3.5.1 通过 Jedis 连接实例.....	123
4.3.5.2 通过 Redisson 连接实例.....	126
4.3.5.3 通过 Hiredis 连接实例.....	129
4.3.5.4 通过 NodeJs 连接实例.....	131
4.3.5.5 通过 PHP 连接实例.....	133
4.3.5.6 通过 Python 连接实例.....	135
4.3.5.7 通过 Go 语言连接实例.....	136
4.3.5.8 通过 C#语言连接实例.....	138
4.3.5.9 使用 Sentinel 兼容模式连接实例.....	140
4.3.6 连接信息管理.....	142
4.3.6.1 给 GeminiDB Redis 实例节点配置安全组规则.....	142
4.3.6.2 查看 GeminiDB Redis IP 地址和端口.....	145
4.3.6.3 修改 GeminiDB Redis 数据库端口.....	146
4.3.6.4 给 GeminiDB Redis 实例节点绑定弹性公网 IP	147

4.3.6.5 给 GeminiDB Redis 实例设置 SSL 数据加密.....	149
4.3.6.6 通过 SSL 方式连接 GeminiDB Redis 实例.....	150
4.3.6.7 修改 GeminiDB Redis 实例节点安全组.....	152
4.3.6.8 给 GeminiDB Redis 实例设置负载均衡内网访问控制.....	153
4.4 数据迁移.....	154
4.4.1 Redis 数据迁移方案概览.....	154
4.4.2 使用 DRS 服务将 GeminiDB Redis 迁移到 Redis（推荐）.....	156
4.4.3 阿里云数据库 Redis/Tair 到 GeminiDB Redis 的迁移.....	156
4.4.4 腾讯云 Redis 到 GeminiDB Redis 的迁移.....	160
4.4.5 使用 DRS 服务将自建 Redis 或者 Redis 集群迁移到 GeminiDB Redis（推荐）.....	164
4.4.6 通过 Redis-Shake 迁移工具将自建 Redis 迁移到 GeminiDB Redis.....	165
4.4.7 使用 Redis-Shake 工具将 RDB 文件/AOF 文件 导入到 GeminiDB Redis.....	169
4.4.8 使用数据导入功能将 RDB 文件恢复到 GeminiDB Redis(推荐).....	171
4.4.9 Kvrocks 到 GeminiDB Redis 的迁移.....	173
4.4.10 Pika 到 GeminiDB Redis 的迁移.....	176
4.4.11 SSDB 到 GeminiDB Redis 的迁移.....	184
4.4.12 LevelDB 到 GeminiDB Redis 的迁移.....	187
4.4.13 RocksDB 到 GeminiDB Redis 的迁移.....	188
4.4.14 AWS ElasticCache for Redis 数据库到 GeminiDB Redis 的迁移.....	188
4.4.15 迁移后 Redis 数据一致性校验.....	191
4.5 实例管理.....	192
4.5.1 GeminiDB Redis 实例会话管理.....	192
4.5.2 GeminiDB Redis 命令重命名.....	195
4.5.3 清除 GeminiDB Redis 数据.....	197
4.5.4 实例生命周期管理.....	199
4.5.4.1 重启 GeminiDB Redis 实例.....	199
4.5.4.2 导出实例.....	201
4.5.4.3 删除按需实例.....	201
4.5.4.4 GeminiDB Redis 实例回收站.....	202
4.6 变更实例.....	203
4.6.1 升级内核小版本.....	203
4.6.2 修改 GeminiDB Redis 实例名称.....	205
4.6.3 修改 GeminiDB Redis 数据库管理员密码.....	206
4.6.4 变更实例的 CPU 和内存规格.....	207
4.6.5 设置可维护时间段.....	210
4.6.6 扩容和缩容实例.....	211
4.6.6.1 扩容和缩容实例概述.....	211
4.6.6.2 扩容实例节点.....	213
4.6.6.3 扩容实例分片.....	218
4.6.6.4 缩容实例节点.....	219
4.6.7 扩容和缩容磁盘.....	221
4.6.7.1 扩容和缩容磁盘概述.....	221

4.6.7.2 手动扩容实例磁盘.....	223
4.6.7.3 自动扩容实例磁盘.....	226
4.6.7.4 手动缩容实例磁盘.....	230
4.6.8 GeminiDB Redis 主备切换.....	233
4.7 数据备份.....	234
4.7.1 备份概述.....	234
4.7.2 管理自动备份.....	235
4.7.3 管理手动备份.....	240
4.8 数据恢复.....	242
4.8.1 恢复方案.....	242
4.8.2 恢复备份到新实例.....	243
4.8.3 通过 PITR 回档恢复到当前实例.....	244
4.9 诊断分析.....	246
4.9.1 大 Key 诊断.....	247
4.9.2 热 key 诊断.....	248
4.10 账号与安全.....	249
4.10.1 开启免密访问.....	249
4.10.2 ACL 账号管理.....	250
4.10.3 开启 ACL 账号自动定向 DB 功能.....	255
4.10.4 防暴力破解.....	256
4.11 参数管理.....	256
4.11.1 修改 GeminiDB Redis 实例参数.....	256
4.11.2 创建参数模板.....	260
4.11.3 查看参数修改历史.....	262
4.11.4 导出参数模板.....	262
4.11.5 比较参数模板.....	264
4.11.6 复制参数模板.....	265
4.11.7 重置参数模板.....	266
4.11.8 应用参数模板.....	266
4.11.9 查看参数模板应用记录.....	266
4.11.10 修改参数模板描述.....	267
4.11.11 删除参数模板.....	267
4.12 日志与审计.....	267
4.12.1 日志配置管理.....	268
4.12.2 查看和导出慢日志.....	270
4.12.3 查看审计日志.....	272
4.12.4 CTS 审计.....	273
4.12.4.1 支持审计的 GeminiDB Redis 关键操作列表.....	273
4.12.4.2 查看 GeminiDB Redis 追踪事件.....	275
4.13 查看监控指标与配置告警.....	276
4.13.1 GeminiDB Redis 支持的监控指标.....	276
4.13.2 设置告警规则.....	338

4.13.3 GeminiDB Redis 指标告警策略推荐配置.....	342
4.13.4 查看 GeminiDB Redis 监控指标.....	345
4.13.5 配置 GeminiDB Redis 监控看板.....	347
4.13.6 事件监控.....	351
4.13.6.1 GeminiDB Redis 事件监控简介.....	351
4.13.6.2 GeminiDB Redis 事件监控支持的事件说明.....	352
4.13.6.3 创建 GeminiDB Redis 事件监控的告警通知.....	359
4.13.6.4 查看 GeminiDB Redis 事件监控数据.....	360
4.14 GeminiDB Redis 标签管理.....	361
4.15 GeminiDB Redis 用户资源配额.....	364
4.16 内存加速.....	365
4.16.1 通过 GeminiDB Redis 实现 RDS 内存加速.....	365
4.16.1.1 内存加速概述.....	365
4.16.1.2 开启和使用内存加速.....	366
4.16.1.3 修改和删除内存加速规则.....	373
4.16.1.4 查看和解除映射关系.....	374
4.16.2 通过 GeminiDB Redis 实现 TaurusDB 内存加速.....	375
4.16.2.1 内存加速概述.....	376
4.16.2.2 开启和使用内存加速.....	376
4.16.2.3 修改和删除内存加速规则.....	383
4.16.2.4 查看和解除映射关系.....	384
5 开发参考.....	386
5.1 开发运维规范.....	386
5.2 命令兼容列表.....	391
5.3 通过程序代码连接实例示例.....	395
5.3.1 通过 Jedis 连接实例.....	395
5.3.2 通过 Redisson 连接实例.....	398
5.3.3 通过 Hireis 连接实例.....	401
5.3.4 通过 NodeJs 连接实例.....	403
5.3.5 通过 PHP 连接实例.....	404
5.3.6 通过 Python 连接实例.....	407
5.3.7 通过 Go 语言连接实例.....	408
5.3.8 通过 C#语言连接实例.....	409
5.3.9 使用 Sentinel 兼容模式连接实例.....	411
5.4 Lua 脚本编写规范.....	414
5.5 使用紧凑型编码实现小对象存储.....	416
5.6 keyspace 通知功能.....	417
5.7 ExHash 命令列表.....	419
5.8 大 Bitmap 初始化.....	424
5.9 大 Bitmap 分页查询.....	428
5.10 客户端连接池参数配置.....	431
5.11 通过 openAPI 实现自定义周期备份.....	433

5.12 通过并行 SCAN 提升全库扫描速度.....	435
5.13 使用 Pipeline 访问 GeminiDB Redis.....	436
5.14 GeminiDB Redis 事务处理.....	438
5.15 GeminiDB Redis 客户端重试机制.....	439
5.16 GeminiDB Redis 消息发布与订阅.....	443
6 最佳实践.....	447
6.1 通过账号直达功能实现 GeminiDB Redis 实例的多租户管理.....	447
6.2 通过 FastLoad 极速导入实现广告投放业务 RTA.....	448
6.3 通过 PITR 实现游戏回档.....	450
6.4 通过 ExHash 实现广告频控业务.....	451
6.5 GeminiDB Redis 在 IM 场景中的应用.....	456
6.6 GeminiDB Redis 基于 lua 实现分布式锁.....	463
6.7 GeminiDB Redis 指标告警配置建议.....	465
6.8 GeminiDB Redis 在商品相关性分析的应用.....	467
6.9 GeminiDB Redis 在线课堂的应用.....	470
6.10 使用 GeminiDB Redis 在 Web 应用中实现会话管理.....	472
6.11 GeminiDB Redis 连接检测机制和客户端重连重试建议.....	475
7 性能白皮书.....	478
7.1 GeminiDB Redis 接口通用性能数据.....	478
7.1.1 性能测试方法.....	478
7.1.2 性能测试结果.....	481
7.2 GeminiDB Redis 接口广告 RTA 场景性能数据.....	483
7.2.1 性能测试方法.....	483
7.2.2 性能测试结果.....	486
8 常见问题.....	488
8.1 高频常见问题.....	488
8.2 产品咨询.....	489
8.2.1 GeminiDB Redis 和开源 Redis、其他开源 Redis 云服务有什么区别？	489
8.2.2 和开源 Redis 相比，GeminiDB Redis 性能如何？	490
8.2.3 GeminiDB Redis 兼容 Redis 哪些版本，兼容哪些命令，客户端连接是否需要修改.....	490
8.2.4 自建 Redis 是否可以搬迁至 GeminiDB Redis，需要注意什么.....	490
8.2.5 什么是 GeminiDB Redis 实例可用性.....	490
8.2.6 GeminiDB Redis 实例总容量是总内存吗，内存和容量之间是什么联系.....	491
8.2.7 购买 GeminiDB Redis 实例时，如何选择合适的节点规格和节点数量？	491
8.2.8 购买 x GB 的 GeminiDB Redis 的实例，优选主备还是集群？	491
8.2.9 GeminiDB Redis 持久化机制是怎样的，会丢数据吗.....	491
8.2.10 GeminiDB Redis 的内存淘汰策略是什么.....	491
8.2.11 GeminiDB Redis 是否支持布隆过滤器等 modules.....	492
8.3 计费相关.....	492
8.3.1 GeminiDB Redis 包年/包月和按需计费模式有什么区别.....	492
8.3.2 GeminiDB Redis 支持包年/包月和按需计费模式相互转换吗.....	492

8.4 数据库使用.....	492
8.4.1 scan 指定 match 参数，数据中确实存在匹配的 key，为什么返回的是空.....	492
8.4.2 业务侧原本做了数据分片，切换到 GeminiDB Redis 后如何处理这部分逻辑.....	495
8.4.3 GeminiDB Redis 接口是否支持 keys 命令的模糊查询.....	495
8.4.4 GeminiDB Redis 是否支持多 DB.....	495
8.4.5 对于 scan 类的操作，GeminiDB Redis 接口与开源 Redis 5.0 的返回值顺序为什么有差异.....	495
8.4.6 对于 scan 类的操作，返回的 cursor 为什么数字很大，与开源 Redis 不一样？	495
8.4.7 针对某些不合法命令，GeminiDB Redis 接口与开源 Redis 5.0 的报错信息为什么有差异.....	496
8.4.8 如何处理报错：CROSSSLOT Keys in request don't hash to the same slot.....	496
8.4.9 GeminiDB Redis 单次事务推荐包含的命令条数.....	496
8.4.10 GeminiDB Redis 集群版实例中，哪些命令需要使用 hashtag.....	496
8.4.11 如何处理报错“ERR unknown command sentinel”	498
8.4.12 对于阻塞命令，GeminiDB Redis 接口（主备实例）与开源 Redis 的返回值为什么可能有差异.....	499
8.4.13 GeminiDB Redis 存储扩容需要多久，对业务有影响吗？	499
8.4.14 GeminiDB Redis 多个节点同时扩容需要多长时间，对业务影响如何？	499
8.4.15 GeminiDB Redis 规格变更包含的在线变更和离线变更有什么区别，通常需要多长时间，对业务有哪些影响？	499
8.4.16 GeminiDB Redis 版本补丁升级包含的在线升级和离线升级有什么区别，通常需要多长时间，对业务有哪些影响？	500
8.4.17 GeminiDB Redis 备份文件是否可以下载到本地，是否支持线下恢复数据.....	500
8.4.18 GeminiDB Redis 数据备份工作机制是怎样的，对业务有哪些影响？	500
8.4.19 购买 GeminiDB Redis 1U*2 节点特惠型实例后，业务访问量比较少，但 CPU 占用率比较高，是什么原因？	500
8.4.20 GeminiDB Redis 监控面板上 key 数量下降又恢复至正常数量是什么原因？	500
8.4.21 GeminiDB Redis 节点 CPU 偶发冲高，可能是哪些原因.....	501
8.4.22 GeminiDB Redis 如何从 5.0 版本升级到 6.2 版本.....	501
8.4.23 GeminiDB Redis 什么时候进入只读.....	501
8.4.24 key 数量统计是什么机制？迁移完成后，GeminiDB 监控中 key 数量不准，还在持续增长，是什么原因？	501
8.5 数据库连接.....	502
8.5.1 如何接入 GeminiDB Redis.....	502
8.5.2 如何使用 GeminiDB Redis 提供的多个节点 IP 地址.....	502
8.5.3 GeminiDB Redis 提供的 ELB 的实现方式是怎样的.....	502
8.5.4 如何创建和连接弹性云服务器.....	502
8.5.5 GeminiDB Redis 实例购买成功后是否支持更换 VPC.....	503
8.5.6 绑定了弹性公网 IP 但是连接不上数据库.....	503
8.5.7 内网如何访问 GeminiDB Redis.....	503
8.5.8 GeminiDB Redis 自带的负载均衡地址是否能绑定公网 IP？ 如何通过公网连接 GeminiDB Redis 实例？	504
8.5.9 设置了安全组，还需要设置负载均衡内网访问控制吗？	504
8.5.10 如何处理客户端连接池报错“Could not get a resource from the pool”	504
8.5.11 常见客户端报错及解决方法.....	505
8.6 备份与恢复.....	506
8.6.1 GeminiDB Redis 实例能够保存多长时间的备份.....	506

8.7 区域和可用区.....	506
8.7.1 不同的可用区是否影响内网互通.....	506
8.7.2 GeminiDB Redis 购买成功后是否支持更换区域.....	506
8.8 数据迁移.....	506
8.8.1 如何处理报错 ERR the worker queue is full, and the request cannot be executed?	506
8.8.2 报错 ERR the request queue of io thread is full, and the request cannot be executed.....	507
8.8.3 报错 read error, please check source redis log or network.....	507
8.8.4 报错 slaveping_thread.cc-ThreadMain-90: error: Ping master error.....	507
8.8.5 同步状态正向迁移速度太慢.....	507
8.8.6 同步状态正向迁移速度太快, 报错: ERR server reply timeout, some responses may lose, but requests have been executed.....	507
8.8.7 4.0、5.0 以及 6.2 版本的自建 Redis 能迁移至 GeminiDB Redis 吗?	507
8.8.8 自建 Redis 主备、集群实例如何迁移到 GeminiDB Redis?.....	507
8.8.9 为什么阿里云 Redis、腾讯云 Redis 等云服务不能使用 DRS 进行数据迁移?	508
8.8.10 自建主备 Redis, 迁移到 GeminiDB Redis 集群, 需要考虑哪些因素?	508
8.8.11 迁移完成后数据量变少了, 100GB 的数据迁移到 GeminiDB Redis 只有 20-30GB, 数据是不是没迁移完?	508
8.8.12 GeminiDB Redis 是否支持 sync 和 psync 协议? 能否放开这些接口?	508
8.9 内存加速.....	509
8.9.1 开启内存加速, MySQL 数据更新后, 会把数据全部缓存到 GeminiDB Redis 吗?	509
8.9.2 开启内存加速, GeminiDB Redis 数据会不断增长, 需要扩容吗? 如何进行缓存数据管理?	509
8.9.3 客户已有业务实现“db”+“缓存”, 推荐使用内存加速吗? 哪种场景可以让客户使用内存加速方案?	509
8.9.4 RDS MySQL 到 GeminiDB Redis 数据延迟大概是多少, 会受哪些因素影响.....	509
8.9.5 开启内存加速后, 是否会对源端 MySQL 有影响.....	509
8.9.6 开启内存加速的 GeminiDB Redis 短时间要处理大量的 binlog, 会不会占用较多的资源, 导致在线业务受损.....	509
8.10 资源冻结/释放/删除/退订.....	510

1 产品介绍

1.1 GeminiDB Redis 产品优势

华为云GeminiDB是一款云原生Key-Value数据库服务，具有高稳定性、高性价比、高弹性、易运维等优势特性，100%兼容Redis协议，且支持PITR游戏回档、FastLoad特征极速入库、为HashKey设置Field过期、为高危Key设置黑名单等高级功能。

目前GeminiDB被广泛应用于游戏好友与排行榜、互联网广告投放、个性推荐、电商库存、车联网车机数据存储、企业ERP系统等应用场景，详见[业务场景推荐](#)。

相比用户自建开源KV数据库方案（例如自建Redis/自建Pika等），GeminiDB提供以下优势价值：

表 1-1 GeminiDB 与自建开源 KV 数据库方案对比项

分类	对比项	自建开源KV数据库方案	GeminiDB
稳定性	fork痛点问题引起的性能抖动	fork问题严重影响业务稳定性 生成RDB备份、AOF文件重写、全量数据同步，都会调用fork函数，引起时延增大、容易OOM等问题。	解决了fork痛点问题，提升了稳定性 GeminiDB备份及数据同步期间无性能抖动。
	大Key场景慢时延问题	单线程架构，后续请求都会变慢 采用单线程架构，大Key请求会导致后续所有请求变慢，还容易引起分片OOM和流控等问题。	多线程架构，能有效减少后续Key影响 GeminiDB采用多线程架构，提高并发的同时，也能减少大Key对后续其他key读写影响。

分类	对比项	自建开源KV数据库方案	GeminiDB
	业务高峰，带宽限流问题	容易触发流控，影响业务 自建开源方案通常采用混合部署方案，带宽严格限制，尤其小规格容易触发流控。	最高支持10Gbit/s，无惧业务高峰 采用独立容器部署方案，实例ELB最大带宽支持10Gbit/s。
	扩容对业务的影响	耗时从数分钟到数小时不等，对业务影响大 增加节点会涉及数据搬迁，对业务的影响是从数分钟到数小时不等。	平滑扩容，业务影响小 容量扩容：秒级完成，业务0感知。 节点扩容：不涉及数据搬迁，业务仅秒级抖动感知。
	节点宕机、主从倒换等HA场景	切换时间长，RTO > 30S+	秒级抖动，RTO < 10S
性能	QPS	单分片8-10w 自建开源方案单线程架构，单分片QPS不能随着CPU扩展而增加。	单分片支持1w-30w+的弹性范围选择 多线程架构，QPS可以随着CPU增加线性扩展。
	时延	低时延	低时延 通常业务场景下平均时延1ms，P99时延2ms左右。
运维能力	危险操作审计日志	不支持	支持高危命令可溯源
	紧急场景异常Key访问熔断能力	不支持	支持Key黑名单，一键熔断高危业务用法，避免实例整体受到影响
	慢日志	支持	支持，日志信息更详细
	大Key诊断	不支持	支持大Key在线分类诊断
	热key诊断	支持	支持热key在线诊断
成本	使用成本	数据全量存储在内存中，成本高	相同规格下，一般可节约30%左右成本 用户可根据业务需要，分别独立购买算力资源、容量资源，避免耦合浪费。

分类	对比项	自建开源KV数据库方案	GeminiDB
	数据压缩能力	不支持	压缩率通常可达到4:1，相同规格存储数据更多
	扩容	存算耦合，成本通常成倍增长	存算分离，支持资源单独扩容
可用性	/	任意一对主从节点故障，自建开源集群将不可用	即使N-1个节点同时故障，GeminiDB实例依然可用
数据可靠性	/	弱 节点重启，网络波动等场景通常存在数千条或者上万条数据丢失；数据弱一致，容易产生脏读问题。	高可靠 数据3副本可靠存储，可用作主数据库，替代传统的DB+Cache方案，简化业务架构；数据强一致，无脏读问题。
高阶特性	自动扩容	不支持	支持
	支持hash field过期	不支持	支持，减少业务设计复杂度&提高并发
	极速数据加载能力	不支持	支持FastLoad特征数据极速入库，减少对在线业务影响
	基于时间点恢复（PITR）能力	不支持	支持 支持PITR数据回档，原实例快速恢复数据。适用于游戏业务。
	灾备实例	不支持	支持同城，跨城灾备实例搭建

1.2 GeminiDB Redis 产品架构

GeminiDB Redis支持的架构类型有：Proxy集群版、Cluster集群版、主备版。

- Proxy集群和Cluster集群均支持水平和垂直扩展，可支撑百万QPS、数十TB级业务量。Cluster集群时延更低、并发更高、可扩展性更强，推荐使用Cluster架构。
- Proxy集群和Cluster集群所有节点可读可写，资源利用率更高，依赖共享存储实现高可用；主备架构仅有主节点可读写，推荐使用集群架构。

以下对架构类型和适用场景做出介绍：

架构类型	架构说明	架构图	适用场景
Proxy 集群版	采用分片集群架构，提供Proxy代理（访问），同时兼容单机、哨兵、Cluster接入方式。	参见图1-1。	- 架构优势：简单易用，用户无需关注数据分片，可以像使用单机使用集群。Proxy可以将用户请求分发到对应的Redis节点进行处理。 - 适用场景：简化客户使用逻辑，用户无需过多关注分片管理。例如，从单机迁移至集群，客户端不方便修改代码，可优先选择Proxy集群。对并发和时延要求更极致的，可使用Cluster架构。
Cluster 集群版 （推荐）	采用分片集群架构，无Proxy组件，兼容原生Redis Cluster的方式访问集群。	参见图1-2。	- 架构优势：与Proxy架构相比，实例无Proxy，单分片并发更高，客户端直连shard，时延更低。扩展性更强，最大可支持128节点。 - 适用场景：适用于时延更敏感、并发和扩展性要求更高的业务。
主备版	标准的master-replica架构，兼容单机、哨兵接入方式。	参见图1-3。	适用于原先采用 Redis 主备架构的业务场景，无需修改代码即可无缝切换使用。若新业务使用 GeminiDB，推荐使用集群架构，性能和可扩展性更优。

图 1-1 Proxy 集群版

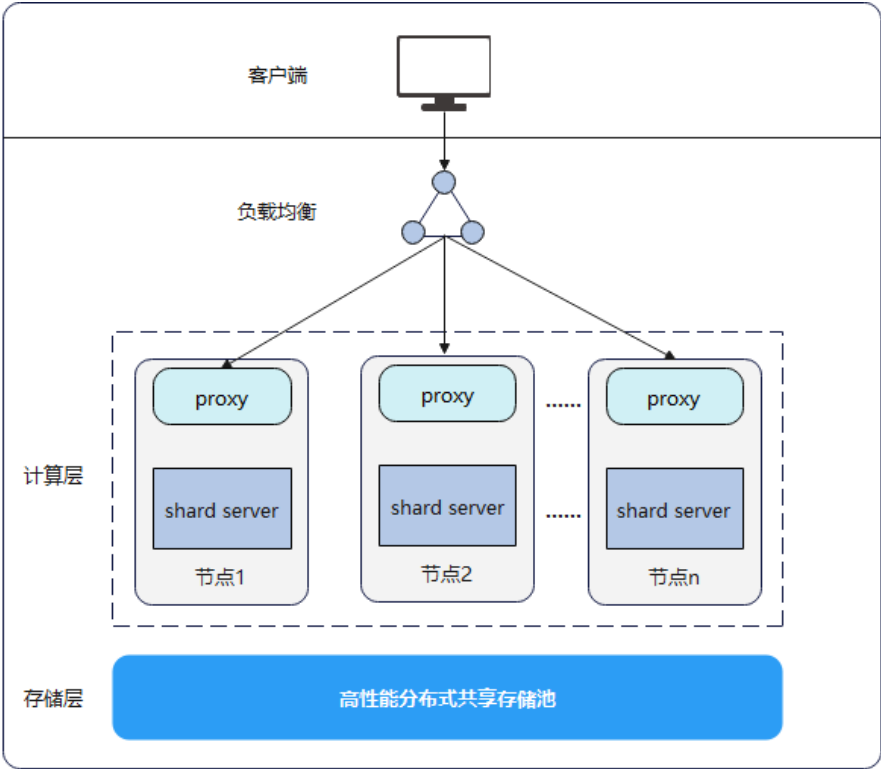


图 1-2 Cluster 集群版

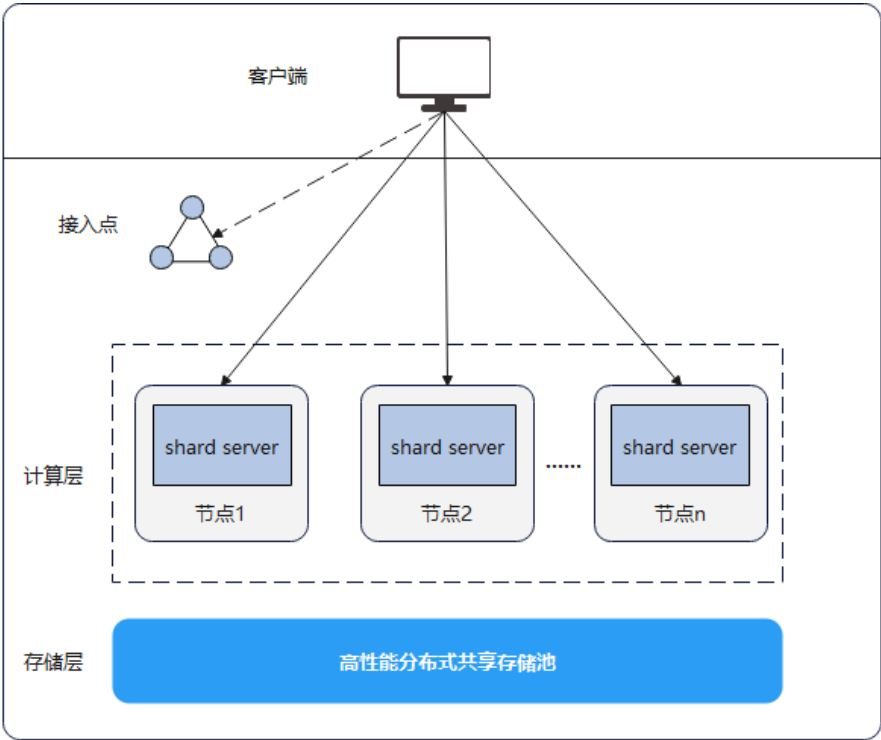
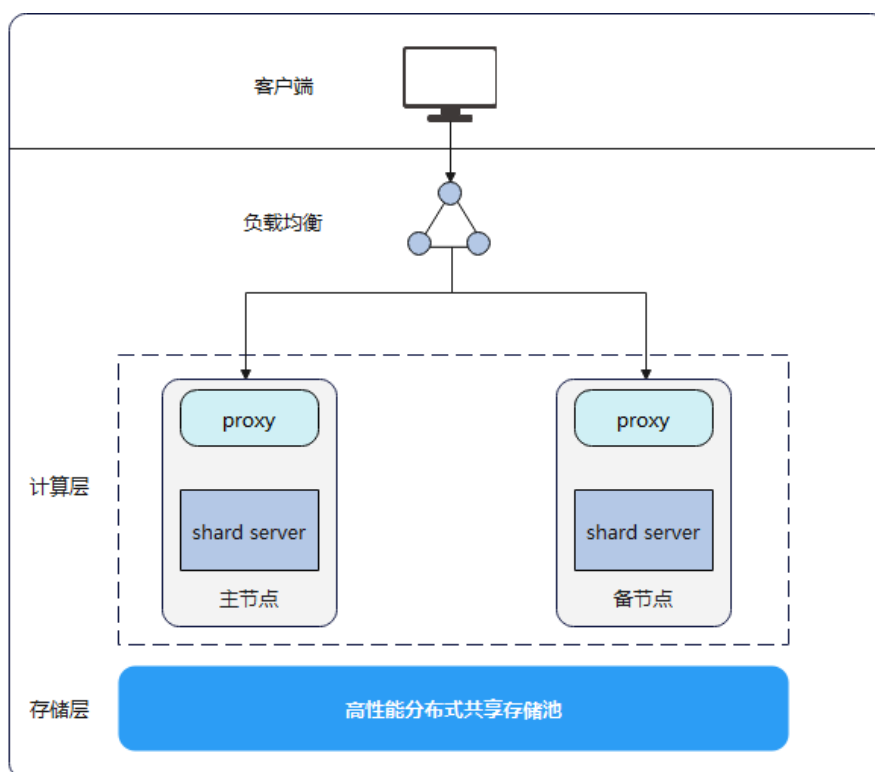


图 1-3 主备版



1.3 业务场景推荐

GeminiDB 100%兼容Redis接口，基于社区Redis提供诸多增强能力，如：秒级自动扩容、DB级多租户权限管理、HashKey设置Field粒度过期时间、PITR游戏数据原地回档、FastLoad广告特征极速入库，为DBA运维和业务开发带来诸多便利。

游戏全服

游戏业务场景丰富，对Redis丰富的数据类型和性能有强诉求，例如：好友关系、IM聊天、抢红包、全局排行榜等。

针对游戏业务，GeminiDB提供全服解决方案：

- 高弹性：为适应游戏运营生命周期，GeminiDB资源可随时灵活扩容、缩容，扩容仅需1秒，且业务无感知。值得一提的是，GeminiDB的CPU算力资源与数据容量资源解耦，提供高性价比。
- 零改造：GeminiDB 100%兼容社区Redis命令，无需修改代码即可平迁上云。
- 简化架构：GeminiDB兼顾低时延、高并发、数据可靠存储能力，因此很多新游采用GeminiDB做主数据库选型，可替换传统的DB + Cache，简化业务架构。
- PITR原地回档：支持在游戏发版失误时，将玩家数据回档到任意指定时间点，仅需数分钟即可完成回档，且支持原实例回档。

互联网广告、推荐

在常见的互联网广告RTA/DSP业务、精细化投放、频控等业务中，往往对时延性能敏感，同时也注重特征数据的存储成本。在此类业务中，GeminiDB是作为特征KV库的最佳选型：

- 性能稳定：平均时延<1ms，P99时延约2ms，性能表现可满足广告RTA等时延敏感类型业务。
- 数据降本：双重数据压缩，100G特征数据压缩后只占不到50G，解决长期成本痛点。
- FastLoad极速入库：FastLoad支持将Spark/Hadoop TB级特征数据极速导入到GeminiDB，比社区Redis逐条命令方案提速10倍。
- Hash Field过期：支持为Hash Key设置field自动过期时间，在频控、算法等业务中可发挥巨大价值。

车联网

在车联网中，数据的流动与车机数据的存储等业务特点对Redis数据库有高可靠的要求，GeminiDB具备以下优势：

- 可靠存储：数据3副本可靠保存，不担心社区Redis的丢数据问题。
- 稳定性高：3AZ同城高级别容灾，极端故障场景可秒级完成接管，数据无脏读。
- 分片不流控：GeminiDB每个节点都采用独享容器，不会对分片做限流。

1.4 兼容接口和版本

介绍GeminiDB Redis目前支持的兼容接口和版本。

表 1-2 兼容接口和版本

兼容接口	版本
Redis	7.0、6.2(包含6.2.x)、5.0及以下版本。

1.5 实例规格说明

本章节主要介绍GeminiDB Redis实例支持的实例规格信息。数据库实例规格与所选的CPU机型相关，请以实际环境为准。

GeminiDB Redis实现了数据冷热交换，支持远超内存的容量。热数据在内存中，全量数据存放在高性能存储池中。实例总容量指的是存储总容量，决定数据存储上限。节点内存请参考[表1-5](#)节点规格信息。

表 1-3 GeminiDB Redis 集群版(快速选择)实例规格

实例类型	实例存储总容量 (GB)	节点规格码	节点CPU (个)	节点内存 (GB)	节点数量	QPS	最大连接数	实例DB数	ACL子账号数量
集群版	4	geminidb.redis.medium.2	1	2	2	20,000	20,000	256	200
	8	geminidb.redis.medium.4	1	4	2	20,000	20,000	256	200
	16	geminidb.redis.large.4	2	8	2	40,000	20,000	256	200
	24	geminidb.redis.large.4	2	8	3	60,000	30,000	256	200
	32	geminidb.redis.large.4	2	8	4	80,000	40,000	256	200
	48	geminidb.redis.xlarge.4	4	16	3	120,000	30,000	1,000	200
	64	geminidb.redis.xlarge.4	4	16	4	160,000	40,000	1,000	200
	96	geminidb.redis.2xlarge.4	8	32	3	240,000	30,000	1,000	200
	128	geminidb.redis.2xlarge.4	8	32	4	320,000	40,000	1,000	200
	192	geminidb.redis.2xlarge.4	8	32	6	480,000	60,000	1,000	200
	256	geminidb.redis.2xlarge.4	8	32	8	640,000	80,000	1,000	200

实例类型	实例存储总容量 (GB)	节点规格码	节点CPU (个)	节点内存 (GB)	节点数量	QPS	最大连接数	实例DB数	ACL子账号数量
	384	geminidb.redis.2xlarge.4	8	32	10	800,000	100,000	1,000	200
	512	geminidb.redis.4xlarge.4	16	64	6	960,000	60,000	1,000	200
	768	geminidb.redis.4xlarge.4	16	64	9	1,440,000	90,000	1,000	200
	1024	geminidb.redis.4xlarge.4	16	64	12	1,920,000	120,000	1,000	200
	2048	geminidb.redis.4xlarge.4	16	64	22	3,520,000	220,000	1,000	200
	4096	geminidb.redis.8xlarge.4	32	128	24	7,680,000	240,000	1,000	200
	8192	geminidb.redis.8xlarge.4	32	128	36	11,520,000	360,000	1,000	200

表 1-4 GeminiDB Redis 主备版(快速选择)实例规格

实例类型	实例存储总容量 (GB)	节点规格码	节点CPU (个)	节点内存 (GB)	节点数量	分片数	QPS	最大连接数	实例DB数	ACL子账号数量
主备版	4	geminidb.redis.medium.2	1	2	2	1	8,000	10,000	1,000	200

实例类型	实例存储总容量 (GB)	节点规格码	节点CPU (个)	节点内存 (GB)	节点数量	分片数	QPS	最大连接数	实例DB数	ACL子账号数量
	8	geminidb.redis.medium.2	1	2	2	1	8,000	10,000	1,000	200
	16	geminidb.redis.medium.4	1	4	2	1	10,000	10,000	1,000	200
	24	geminidb.redis.large.4	2	8	2	1	20,000	10,000	1,000	200
	32	geminidb.redis.large.4	2	8	2	1	20,000	1,000	1,000	200
	48	geminidb.redis.xlarge.4	4	16	2	1	40,000	2,000	1,000	200
	64	geminidb.redis.xlarge.4	4	16	2	1	40,000	2,000	1,000	200
	96	geminidb.redis.2xlarge.4	8	32	2	1	80,000	2,000	1,000	200
	128	geminidb.redis.4xlarge.4	16	64	2	1	160,000	2,000	1,000	200

表 1-5 GeminiDB Redis 节点规格

节点规格码	vCPU (个)	内存 (GB)	单节点的最大持久化数据存储空间 (GB)	单节点的最大连接数	基准带宽 (Mbps)	实例DB数
geminidb.redis.medium.2	1	2	4	10,000	800	256

节点规格码	vCPU (个)	内存 (GB)	单节点的最大持久化数据存储空间 (GB)	单节点的最大连接数	基准带宽 (Mbps)	实例DB数
geminidb.redis.large.2	2	4	8	10,000	1200	256
geminidb.redis.xlarge.2	4	8	16	10,000	2500	1,000
geminidb.redis.2xlarge.2	8	16	32	10,000	5000	1,000
geminidb.redis.4xlarge.2	16	32	64	10,000	9000	1,000
geminidb.redis.8xlarge.2	32	64	128	10,000	18000	1,000
geminidb.redis.medium.4	1	4	8	10,000	800	256
geminidb.redis.large.4	2	8	16	10,000	1200	256
geminidb.redis.xlarge.4	4	16	32	10,000	2500	1,000
geminidb.redis.2xlarge.4	8	32	64	10,000	5000	1,000
geminidb.redis.4xlarge.4	16	64	128	10,000	9000	1,000
geminidb.redis.8xlarge.4	32	128	256	10,000	18000	1,000
geminidb.redis.medium.8	1	8	16	10,000	800	256
geminidb.redis.large.8	2	16	32	10,000	1200	256
geminidb.redis.xlarge.8	4	32	64	10,000	2500	1,000
geminidb.redis.2xlarge.8	8	64	128	10,000	5000	1,000
geminidb.redis.4xlarge.8	16	128	256	10,000	9000	1,000

节点规格码	vCPU (个)	内存 (GB)	单节点的最大持久化数据存储空间 (GB)	单节点的最大连接数	基准带宽 (Mbps)	实例DB数
geminidb.redis.8xlarge.8	32	256	512	10,000	18000	1,000

备注：存量部分老实例单个节点带宽为768Mbps，和其他Redis云服务产品保持一致。用户可关注节点带宽使用率，若老实例大流量场景，您可以在管理控制台右上角，选择“[工单](#) > [新建工单](#)”进行带宽咨询。

1.6 实例状态

数据库实例状态是数据库实例的运行情况。可以通过管理控制台查看数据库实例状态。

表 1-6 实例状态

状态	说明
正常	数据库实例正常和可用。
异常	数据库实例不可用。
创建中	正在创建数据库实例。
创建失败	数据库实例创建失败。
重启中	正在重启数据库实例。
重置密码中	正在重置管理员密码。
节点扩容中	正在扩容该实例下的节点个数。
节点缩容中	正在删除该实例下的节点。
存储扩容中	正在扩容实例的磁盘容量。
规格变更中	正在变更实例的CPU和内存规格。
转包周期中	实例的计费方式正在由“按需计费”转为“包年/包月”。
转按需中	实例的计费方式正在由“包年/包月”转为“按需计费”。
备份上传中	正在上传备份文件。
备份中	正在创建数据库备份。
恢复检查中	该实例下的备份正在恢复到新实例。
SSL切换中	正在开启或关闭SSL安全连接。

状态	说明
冻结	账户余额小于或等于0元，系统对该用户下的实例进行冻结。您需前往费用中心充值成功，欠款核销后，冻结的实例才会解冻。
解冻中	欠款核销后，正在解冻数据库实例。
包周期变更资源检查中	包周期实例在进行变更过程中所持续的状态。

1.7 约束与限制

云数据库 GeminiDB Redis在使用上有一些固定限制，用来提高实例的稳定性和安全性。

规格与限制

表 1-7 规格说明

资源类型	规格	说明
CPU和内存规格	GeminiDB Redis支持Proxy集群版、主备版。	- 不同版本规格具体参见实例规格说明。 - 当用户购买的实例的规格无法满足业务需求时，参见变更实例的CPU和内存规格进行规格变更。
存储空间	存储空间范围与所选 实例规格 有关。	存储容量变更目前既支持扩容，也支持缩容，具体请参见 概述 。
连接数	最大值为实例节点数 * 10000。	不同内存规格下的最大连接数不同，请参见 实例规格说明 。

配额限制

表 1-8 配额限制

资源类型	限制	说明
标签	每个实例最多支持20个标签配额。	更多信息，请参见 GeminiDB Redis 标签管理 。
免费备份空间	GeminiDB Redis将同比例赠送备份存储空间，用于存储备份数据。	更多信息，请参见 备份存储 。
自动备份保留天数	默认为7天，可设置范围为1 ~ 3660天。	更多信息，请参见 自动备份策略 。

命名限制

表 1-9 命名限制

限制项	说明
实例名称	- 长度在4~64个字符之间。 - 必须以字母开头（区分大小写），可以包含字母、数字、中划线或下划线，不能包含其他特殊字符。
备份名称	- 长度在4~64个字符之间。 - 必须以字母开头，区分大小写，可以包含字母、数字、中划线或者下划线，不能包含其他特殊字符。
参数模板名称	- 长度在1~64个字符之间。 - 区分大小写，可包含字母、数字、中划线、下划线或句点，不能包含其他特殊字符。

安全限制

表 1-10 安全限制

限制项	说明
数据库管理员rwuser账号的密码	- 长度为8~32个字符。 - 必须是大写字母、小写字母、数字、特殊字符中至少两种的组合，其中可输入~!@#%^*_-=+?特殊字符。更多信息，请参见修改GeminiDB Redis数据库管理员密码。 - 请妥善管理您的密码，因为系统将无法获取您的密码信息。
数据库端口	数据库的访问端口号。 GeminiDB Redis支持自定义端口号，取值范围为：1024~65535，其中，如下端口号为禁用端口：2180、2887、3887、6377、6378、6380、8018、8079、8091、8479、8484、8999、9864、9866、9867、12017、12333、50069。 当您不指定端口时，创建实例的访问端口默认为6379。
虚拟私有云	目前GeminiDB Redis实例创建完成后不支持切换虚拟私有云VPC，请谨慎选择所属虚拟私有云。
内网安全组	安全组限制安全访问规则，加强GeminiDB Redis与其他服务间的安全访问。请确保所选取的安全组允许客户端访问数据库实例。 如果没有可用的安全组，系统自动为您分配资源。

限制项	说明
访问控制	负载均衡地址不支持安全组，创建完成后请配置IP访问控制，如未配置白名单，所有与VPC互通的IP地址均可访问实例。具体操作请参考 给GeminiDB Redis实例设置负载均衡内网访问控制 。
ACL账户	GeminiDB Redis接口提供企业级的多租户能力，支持添加只读账号、读写账号，并且可约束每个账号可访问的数据库（DB）范围，避免误操作其他租户数据。每个实例最多能创建200个ACL账号。 更多信息，请参见 ACL账号管理 。

实例操作限制

表 1-11 实例操作限制

功能	使用限制
数据库访问	- 如果GeminiDB Redis实例未开通公网访问，则该实例必须与云主机弹性云服务器处在同一个虚拟私有云子网内才能相互访问。 - 弹性云服务器必须处于目标GeminiDB Redis实例所属安全组允许访问的范围内。 如果GeminiDB Redis实例与弹性云服务器处于不同的安全组，系统默认不能访问。需要在GeminiDB Redis的安全组添加一条“入”的访问规则。 - GeminiDB Redis实例的默认端口为6379。 - 数据库端口支持创建时设置，后期可修改。
实例部署	实例所部署的服务器，对用户都不可见，即只允许应用程序通过IP地址和端口访问数据库。
重启GeminiDB Redis实例	- 无法通过命令行重启，必须通过GeminiDB Redis的管理控制台操作重启实例。 - 重启实例会导致服务中断，请谨慎操作。请在业务低峰期重启，重启前请做好业务安排并确保应用有重连机制。
GeminiDB Redis备份查看	GeminiDB Redis实例在对象存储服务上的备份文件，对用户不可见。
变更GeminiDB Redis实例的CPU和内存规格	- 在线变更时，单个节点变更时会出现一次秒级闪断，因此，整个实例会出现数次秒级闪断，要求客户端有自动重连机制。建议在业务低峰期间变更。 - 正在进行变更的节点，其计算任务由其他节点分担，请在业务低峰变更，避免实例过载。

功能	使用限制
主备切换	只支持GeminiDB Redis主备版实例，主备切换时通常会发生小于10秒的连接闪断，会产生慢时延，或者命令执行失败，需要客户端有命令重试或者连接重试机制。为了避免影响业务，建议在业务低峰期操作。
数据恢复	为避免数据丢失，建议数据恢复前备份重要数据。
存储空间	若实例的磁盘空间已满，此时不可进行数据库写入操作，您需要扩容磁盘使实例恢复到正常状态。建议定期检查存储空间。 GeminiDB Redis支持自动扩容，可以有效避免数据量突增问题。建议参见自动扩容实例磁盘开启自动扩容。
回收站管理	• 支持将退订后的包年/包月实例和删除的按需实例，加入回收站管理。通过数据库回收站中重建实例功能，可以恢复1~7天内删除的实例。 • 回收站策略机制默认开启，且不可关闭，默认保留天数为7天，该功能免费。 • 目前回收站允许加入100个实例，超过该配额的实例将无法添加至回收站中。 • 当实例存储空间满时，删除之后的实例不会放入回收站。

其他有效评估以及改善业务系统稳定性的开发和运维规范，请参见[开发运维规范](#)。

2 计费说明

2.1 计费概述

通过阅读本文，您可以快速了解GeminiDB Redis的计费模式、计费项、续费、欠费等主要计费信息。

- **计费模式**

GeminiDB Redis提供包年/包月和按需计费两种计费模式，以满足不同场景下的用户需求。

- 包年/包月是一种预付费模式，即先付费再使用，按照订单的购买周期进行结算，因此在购买之前，您必须确保账户余额充足。
- 按需计费是一种后付费模式，即先使用再付费，按照数据库实例实际使用时长计费。

关于两种计费模式的详细介绍请参见[计费模式概述](#)。

在购买云数据库后，如果发现当前计费模式无法满足业务需求，您还可以变更计费模式。详细介绍请参见[变更计费模式概述](#)。

- **计费项**

GeminiDB Redis的计费项由实例规格费用、存储空间费用、备份空间费用和公网带宽费用组成。了解每种计费项的计费因子、计费公式等信息，请参考[计费项](#)。

如需了解实际场景下的计费样例以及各计费项在不同计费模式下的费用计算过程，请参见[计费样例](#)。

- **续费**

包年/包月数据库实例在到期后会影响到数据库实例的正常运行。如果您想继续使用数据库实例，需要在规定的时间内为数据库实例进行续费，否则计算、存储等资源将会自动释放，数据也可能会丢失。

续费包括手动续费和自动续费两种方式，您可以根据需求选择。了解更多关于续费的信息，请参见[续费概述](#)。

- **费用账单**

您可以在“费用中心 > 账单管理”查看与数据库实例相关的流水和明细账单，以便了解您的消费情况。如需了解具体操作步骤，请参见[费用账单](#)。

- **欠费**

在使用GeminiDB Redis时，账户的可用额度小于待结算的账单，即被判定为账户欠费。欠费后，可能会影响云服务资源的正常运行，需要及时充值。详细介绍请参见[欠费说明](#)。

- **停止计费**
当GeminiDB Redis资源不再使用时，可以将他们退订或删除，从而避免继续收费。详细介绍请参见[停止计费](#)。
- **成本管理**
使用GeminiDB Redis时，成本主要分为资源成本和运维成本。您可以从成本分配、成本分析和成本优化三方面来管理成本。更多详情，请参见[成本管理](#)。

2.2 计费模式

2.2.1 计费模式概述

GeminiDB Redis提供包年/包月和按需计费两种计费模式，以满足不同场景下的用户需求。

- **包年/包月**：一种预付费模式，即先付费再使用，按照订单的购买周期进行结算。购买周期越长，享受的折扣越大。一般适用于设备需求量长期稳定的成熟业务。
- **按需计费**：一种后付费模式，即先使用再付费，按照实例实际使用时长计费，秒级计费，按小时结算。按需计费模式允许您根据实际业务需求灵活地调整资源使用，无需提前预置资源，从而降低预置过多或不足的风险。一般适用于电商抢购等设备需求量瞬间大幅波动的场景。

[表2-1](#)列出了两种计费模式的区别。

表 2-1 计费模式

计费模式	包年/包月	按需计费
付费方式	预付费。 按照订单的购买周期结算。	后付费。 按照实例实际使用时长计费。
计费周期	按订单的购买周期计费。	秒级计费，按小时结算。
适用计费项	实例规格（vCPU和内存）、存储空间、备份空间、弹性公网IP	实例规格（vCPU和内存）、存储空间、备份空间、弹性公网IP
变更计费模式	支持变更为按需计费模式。当包年/包月资费模式到期后，按需的资费模式才会生效。详情请参考 包年/包月转按需 。	支持变更为包年/包月计费模式。详情请参考 按需转包年/包月 。
变更规格	支持变更实例规格。	支持变更实例规格。
适用场景	适用于可预估资源使用周期的场景，价格比按需计费模式更优惠。对于长期使用者，推荐该方式。	适用于计算资源需求波动的场景，可以随时开通，随时删除。

2.2.2 包年/包月

包年/包月是一种先付费再使用的计费模式，适用于对资源需求稳定且希望降低成本的用户。通过选择包年/包月的计费模式，您可以预先购买资源并获得一定程度的价格优惠。本文将介绍包年/包月GeminiDB Redis资源的计费规则。

适用场景

包年/包月计费模式需要用户预先支付一定时长的费用，适用于长期、稳定的业务需求。以下是一些适用于包年/包月计费模式的业务场景：

- 稳定业务需求：对于长期运行且资源需求相对稳定的业务，如企业官网、在线商城、博客等，包年/包月计费模式能提供较高的成本效益。
- 长期项目：对于周期较长的项目，如科研项目、大型活动策划等，包年/包月计费模式可以确保在整个项目周期内资源的稳定使用。
- 业务高峰预测：如果能预测到业务高峰期，如电商促销季、节假日等，可提前购买包年/包月资源以应对高峰期的需求，避免资源紧张。
- 数据安全要求高：对于对数据安全性要求较高的业务，包年/包月计费模式可确保资源的持续使用，降低因资源欠费而导致的数据安全风险。

适用计费项

包年包月包含以下计费项。

表 2-2 适用计费项

计费项	说明
实例规格	对所选的实例规格进行计费，包括vCPU和内存。
存储空间	对数据库存储空间进行计费，包年包月计费方式的存储空间如果超过当前容量，超出的部分将按需计费。
备份空间	GeminiDB Redis提供了部分免费存储空间，用于存放您的备份数据，其总容量为您购买存储容量的100%。 备份存储用量超过购买存储容量的100%，超出部分将按照备份计费标准收费，计费方式为按需计费（每小时扣费一次），不足一小时按照实际使用时长收费。
公网带宽（可选）	GeminiDB Redis实例支持公网访问，公网访问会产生带宽流量费；GeminiDB Redis数据库实例在云内部网络产生的流量不计费。

假设您计划购买一个规格为2vCPUs，3个节点，存储空间容量为12GB的GeminiDB Redis实例。在购买数据库实例页面底部，您将看到所需费用的明细（不包含备份空间费用），如图2-1所示。

图 2-1 配置费用

购买时长: 1, 2, 3

购买数量: 1

☐ 自动续费 [扣款规则](#) 和 [续费时长](#)

配置费用 **¥1,883.40**

配置费用将包括以下部分：

- 数据库实例：根据所选实例规格计算的费用。
- 数据库存储：对数据库存储空间进行计费。

说明

备份空间费用，使用后按照统一标准计费，购买时不包含在配置费用中，可通过[云数据库 GeminiDB 价格详情](#)查看。

备份空间计费信息

架构	按小时	价格单位
集群	0.000221	元/GB

计费周期

包年/包月 GeminiDB Redis 实例的计费周期是根据您购买的时长来确定的（以 UTC+8 时间为准）。一个计费周期的起点是您开通或续费资源的时间（精确到秒），终点则是到期日的 23:59:59。

例如，如果您在 2023/03/08 15:50:04 购买了一台时长为一个月的 GeminiDB Redis 实例，那么其计费周期为：2023/03/08 15:50:04 ~ 2023/04/08 23:59:59。

计费示例

假设您在 2023/03/08 15:50:04 购买了一个包年/包月 GeminiDB Redis 实例（规格：2 vCPUs 独享型，节点数量：3，存储空间：40GB，备份存储：50GB（赠送 40GB，后续收费空间 10GB）），计费资源包括实例规格（vCPU、内存、节点数量）、存储空间、备份空间、公网带宽。购买时长为一个月，并在到期前手动续费 1 个月，则：

- 第一个计费周期为：2023/03/08 15:50:04 ~ 2023/04/08 23:59:59
- 第二个计费周期为：2023/04/08 23:59:59 ~ 2023/05/08 23:59:59
 - 2023/04/08 23:59:59 ~ 2023/05/01 23:59:59 期间，使用免费备份空间 20GB。

- 2023/05/01 23:59:59~2023/05/08 23:59:59期间，使用计费备份空间10GB，计费时长168小时。

您需要为每个计费周期预先付费，各项GeminiDB Redis资源单独计费，计费公式如表2-3所示。

表 2-3 计费公式

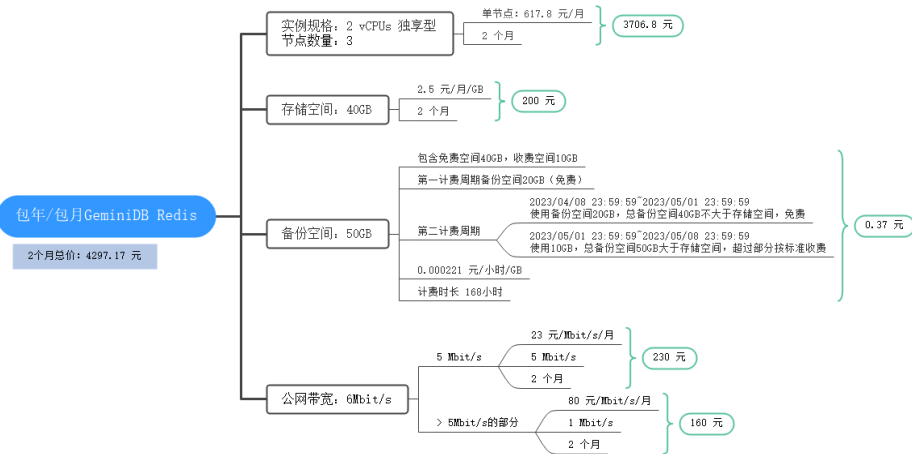
资源类型	计费公式	资源单价
实例规格（vCPU和内存）	实例规格单价 * 购买时长 * 节点数量	请参见 云数据库 GeminiDB价格详情 中的“集群-规格费用”。
存储空间	存储空间单价 * 购买时长 * 存储空间(GB)	请参见 云数据库 GeminiDB价格详情 中的“集群磁盘空间计费信息”。
备份空间	备份空间单价 * 计费时长 * (备份空间 - 存储空间)(GB) 说明 计费时长：备份超过免费空间大小的使用时长。	请参见 云数据库 GeminiDB价格详情 中的“备份空间计费信息”。
公网带宽	按固定带宽值计费	请参见 弹性公网IP价格详情 。

图2-2给出了上述示例配置的费用计算过程。

说明

图中价格仅供参考，实际计算请以[云数据库 GeminiDB价格详情](#)中的价格为准。

图 2-2 包年/包月 GeminiDB Redis 费用计算示例



变更配置后对计费的影响

当前包年/包月GeminiDB Redis实例的规格不满足您的业务需要时，您可以在控制台发起变更规格操作，变更时系统将按照如下规则为您计算变更费用：

- 实例升配：新配置价格高于老配置价格，此时您需要支付新老配置的差价。
- 实例降配：新配置价格低于老配置价格，此时华为云会将新老配置的差价退给您。

实例降配会影响云数据库性能，通常不建议您这样操作。这里以资源升配且无任何优惠的场景为例，假设您在2023/04/08购买了一个包年/包月GeminiDB Redis实例（1vCPU 6GB 3节点），购买时长为1个月，计划在2023/04/18变更规格为2vCPUs 12GB 3节点。旧配置价格为941.70 元/月，新配置价格为1883.40 元/月。计算公式如下：

升配费用=新配置价格*剩余周期-旧配置价格*剩余周期

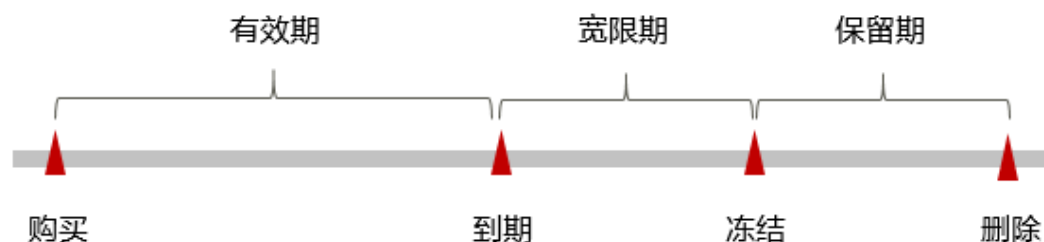
公式中的剩余周期为每个自然月的剩余天数/对应自然月的最大天数。本示例中，剩余周期=12（4月份剩余天数）/ 30（4月份最大天数）+ 8（5月份剩余天数）/ 31（5月份最大天数）=0.6581，代入公式可得升配费用=1883.4*0.6581-941.7*0.6581=619.73（元）

更多信息请参见[变更资源规格费用说明](#)。

到期后影响

图2-3描述了包年/包月GeminiDB Redis实例各个阶段的状态。购买后，在计费周期内实例正常运行，此阶段为有效期；实例到期而未续费时，将陆续进入宽限期和保留期。

图 2-3 包年/包月 GeminiDB Redis 实例生命周期



到期预警

包年/包月GeminiDB Redis实例在到期前，系统将向用户推送到期预警消息。预警消息将通过邮件、短信和站内信的方式通知到华为云账号的创建者。

- 包年资源到期前30、15、7、3、1天发送到期提醒。
- 包月资源到期前15、7、3、1天发送到期提醒。

到期后影响

当您的包年/包月GeminiDB Redis实例到期未续费，首先会进入宽限期，实例状态变为“已过期”。宽限期内您可以正常访问GeminiDB Redis实例，但以下操作将受到限制：

- 变更实例规格
- 包年/包月转按需
- 退订

如果您在宽限期内仍未续费包年/包月GeminiDB Redis实例，那么就会进入保留期，实例状态变为“已冻结”，您将无法对处于保留期的包年/包月资源执行任何操作。

保留期到期后，若包年/包月GeminiDB Redis实例仍未续费，那么实例将被释放，数据无法恢复。

 说明

- 华为云根据[客户等级](#)定义了不同客户的宽限期和保留期时长。
- 关于续费的详细介绍请参见[续费概述](#)。

2.2.3 按需计费

按需计费是一种先使用再付费的计费模式，适用于无需任何预付款或长期承诺的用户。本文将介绍按需计费GeminiDB Redis实例的计费规则。

适用场景

按需计费适用于具有不能中断的短期、突增或不可预测的应用或服务，例如电商抢购、临时测试、科学计算。

适用计费项

按需计费包含以下计费项。

表 2-4 适用计费项

计费项	说明
实例规格	对所选的实例规格进行计费，包括vCPU和内存。
存储空间	对数据库存储空间进行计费，按需计费的存储空间费用按照实际使用量每小时计费。
备份空间	GeminiDB Redis提供了部分免费存储空间，用于存放您的备份数据，其总容量为您购买存储容量的100%。 备份存储用量超过购买存储容量的100%，超出部分将按照备份计费标准收费，计费方式为按需计费（每小时扣费一次），不足一小时按照实际使用时长收费。
公网带宽（可选）	GeminiDB Redis实例支持公网访问，公网访问会产生带宽流量费；GeminiDB Redis数据库实例在云内部网络产生的流量不计费。

假设您计划购买一个规格为2vCPUs，3个节点，存储空间容量为12GB的GeminiDB Redis实例。在购买数据库实例页面底部，您将看到所需费用的明细（不包含备份空间费用），如[图2-4](#)所示。

图 2-4 配置费用

配置费用 **¥5.28**/小时 

配置费用将包括以下部分：

- 云数据库虚拟机：根据所选配置（包括vCPU和内存）计算的费用。
- 云数据库存储空间：根据所选存储空间计算的费用。

说明

备份空间费用，使用后按照统一标准计费，购买时不包含在配置费用中，可通过[云数据库 GeminiDB价格详情](#)查看。

备份空间计费信息

架构	按小时	价格单位
集群	0.000221	元/GB

计费周期

按需计费GeminiDB Redis实例按秒计费，每一个小时整点结算一次费用（以GMT+8时间为准），结算完毕后进入新的计费周期。计费的起点以GeminiDB Redis实例创建成功的时间点为准，终点以实例删除时间为准。

说明

实例从创建到启动需要一定时长，计费的起点是创建成功的时间点，而非开始创建的时间。您可以在实例详情页“基本信息”页签查看这个时间，创建成功的时间点对应界面上的“创建时间”。

例如，您在8:45:30购买了一个按需计费的GeminiDB Redis实例，相关资源包括计算资源（vCPU和节点数量）、存储容量和备份存储，然后在8:55:00将其删除，则计费周期为8:00:00 ~ 9:00:00，在8:45:30 ~ 8:55:30间产生费用，该计费周期内的计费时长为600秒。

计费示例

假设您在2023/04/18 9:59:30购买了一个按需计费实例（规格：0.5 vCPUs 独享型，节点数量：3，存储容量：40GB，备份存储：50GB（赠送40GB，后续收费空间10GB）），计费资源包括计算资源（vCPU和节点数量）和存储容量，然后在2023/04/18 10:45:46将其删除，则：

- 第一个计费周期为9:00:00 ~ 10:00:00，在9:59:30 ~ 10:00:00间产生费用，该计费周期内的计费时长为30秒。
- 第二个计费周期为10:00:00 ~ 11:00:00，在10:00:00 ~ 10:45:46间产生费用，该计费周期内的计费时长为2746秒。
 - 10:00:00 ~ 10:45:00期间，使用免费备份空间。
 - 10:45:00 ~ 10:45:46期间，使用计费备份空间10GB，计费时长46秒。

您需要为每个计费周期付费，各项GeminiDB Redis实例单独计费，计费公式如[表2-5](#)所示。产品价格详情中标出了实例的每小时价格，您需要将每小时价格除以3600，得到每秒价格。

表 2-5 计费公式

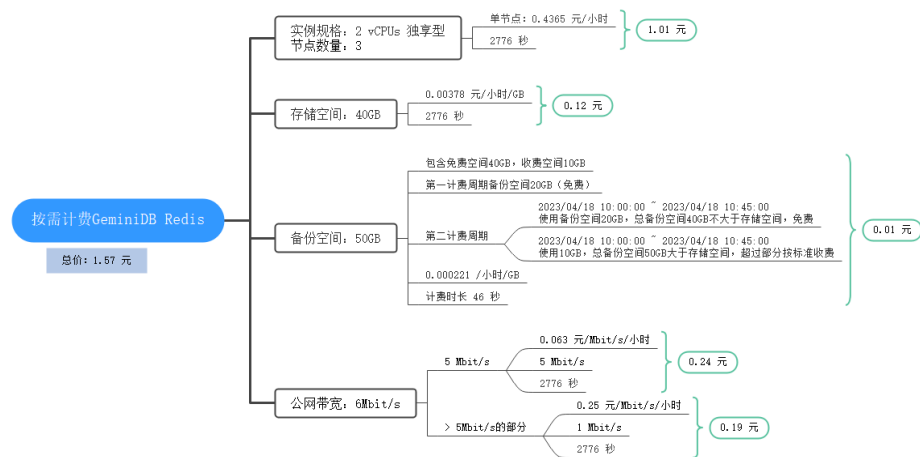
资源类型	计费公式	资源单价
计算资源 (vCPU和节点数量)	实例规格单价 * 计费时长	请参见 云数据库 GeminiDB价格详情 中的“集群-规格费用”。
存储容量	存储容量单价 * 购买时长	请参见 云数据库 GeminiDB价格详情 中的“集群磁盘空间计费信息”。
备份空间	备份空间单价 * 计费时长 * (备份容量 - 存储空间) (GB) 说明 计费时长：备份超过免费空间大小的使用时长。	请参见 云数据库 GeminiDB价格详情 中的“备份空间计费信息”。
公网流量	按固定带宽值采用阶梯计费 0Mbit/s~5Mbit/s (含)：均为一个统一的单价 - 大于5Mbit/s：按每Mbit/s计费	请参见 弹性云服务器价格详情 中的“带宽价格”，或者 弹性公网IP价格详情 。

图2-5给出了上述示例配置的费用计算过程。

说明

图中价格仅供参考，实际计算请以[云数据库 GeminiDB价格详情](#)中的价格为准。
在按需付费模式下，价格计算器上的金额如果遇小数点，则保留小数点后两位，第三位四舍五入。如遇四舍五入后不足¥0.01，则按¥0.01展示。

图 2-5 按需计费 GeminiDB Redis 费用计算示例



变更配置后对计费的影响

如果您在购买按需计费实例后变更了实例配置，会产生一个新订单并开始按新配置的价格计费，旧订单自动失效。

如果您在一个小时内变更了实例配置，将会产生多条计费信息。每条计费信息的开始时间和结束时间对应不同配置在该小时内的生效时间。

例如，您在9:00:00购买了一台按需计费实例，实例规格为 0.5vCPUs 3GB，并在9:30:00升配为 1vCPU 6GB，那么在9:00:00 ~ 10:00:00间会产生两条计费信息。

- 第一条对应9:00:00 ~ 9:30:00，实例规格按照 0.5vCPUs 3GB计费。
- 第二条对应9:30:00 ~ 10:00:00，实例规格按照 1vCPU 6GB计费。

欠费影响

图2-6描述了按需计费GeminiDB Redis实例各个阶段的状态。购买后，在计费周期内实例正常运行，此阶段为有效期；当您的账号因按需GeminiDB Redis实例自动扣费导致欠费后，账号将变成欠费状态，实例将陆续进入宽限期和保留期。

图 2-6 按需计费 GeminiDB Redis 实例生命周期



欠费预警

系统会在每个计费周期后的一段时间对按需计费资源进行扣费。当您的账户被扣为负值时，我们将通过邮件、短信和站内信的方式通知到华为云账号的创建者。

欠费后影响

当您的账号因按需GeminiDB Redis实例自动扣费导致欠费后，账号将变成欠费状态。欠费后，按需实例不会立即停止服务，实例进入宽限期。您需支付按需实例在宽限期内产生的费用，相关费用可在“管理控制台 > 费用中心 > 总览”的“欠费金额”查看，华为云将会在您充值时自动扣取欠费金额。

如果您在宽限期内仍未支付欠款，那么就会进入保留期，实例状态变为“已冻结”，您将无法对处于保留期的按需计费实例执行任何操作。

保留期到期后，若您仍未支付账户欠款，那么实例将被释放，数据无法恢复。

说明

- 华为云根据[客户等级](#)定义了不同客户的宽限期和保留期时长。
- 关于充值的详细操作请参见[账户充值](#)。

2.3 计费项

计费说明

GeminiDB Redis的计费项由实例规格费用、存储空间费用、备份空间费用和公网流量费用组成。具体内容如[表2-6](#)所示。

 说明

标 * 的计费项为必选计费项。

表 2-6 GeminiDB Redis 实例计费项

计费项	计费项说明	适用的计费模式	计费公式
* 实例规格	计费因子：vCPU和内存，不同规格的实例类型提供不同的计算和存储能力。	包年/包月、按需计费	实例规格单价 * 购买时长 实例规格单价请参见 云数据库 GeminiDB价格详情 中的“集群-规格费用”。
* 存储空间	计费因子：存储空间，按统一标准进行计费。	包年/包月、按需计费	存储空间单价 * 存储容量 * 购买时长 请参见 云数据库 GeminiDB价格详情 中的“集群磁盘空间计费信息”。
备份空间	计费因子：备份空间，按统一标准进行计费。	按需计费	备份空间单价 * 备份收费容量 * 购买时长 请参见 云数据库 GeminiDB价格详情 中的“备份空间计费信息”。 说明 计费时长：备份超过免费空间大小的使用时长。

计费项	计费项说明	适用的计费模式	计费公式
公网流量	如有互联网访问需求，您需要购买弹性公网IP。 计费因子：带宽费、流量费和IP保有费。 - 包年/包月计费模式支持按带宽计费方式，收取带宽费。 - 按需计费模式支持按带宽计费、按流量计费和加入共享带宽三种计费方式，分别收取带宽费+IP保有费、流量费+IP保有费、带宽费+IP保有费。	包年/包月、按需计费 带宽费支持使用带宽加油包抵扣，流量费支持使用共享流量包抵扣。	按固定带宽值采用阶梯计费 0Mbit/s~5Mbit/s（含）：均为一个统一的单价 - 大于5Mbit/s：按每Mbit/s计费 公网带宽单价请参见 弹性云服务器价格详情 中的“带宽价格”，或者 弹性公网IP价格详情 。

计费示例

假设您在2023/03/08 15:50:04购买了一个包年/包月GeminiDB Redis实例（规格：2 vCPUs 独享型，节点数量：3，存储空间：40GB，备份存储：50GB（赠送40GB，后续收费空间10GB）），计费资源包括实例规格（vCPU、内存、节点数量）、存储空间、备份空间、公网带宽。购买时长为一个月，并在到期前手动续费1个月，则：

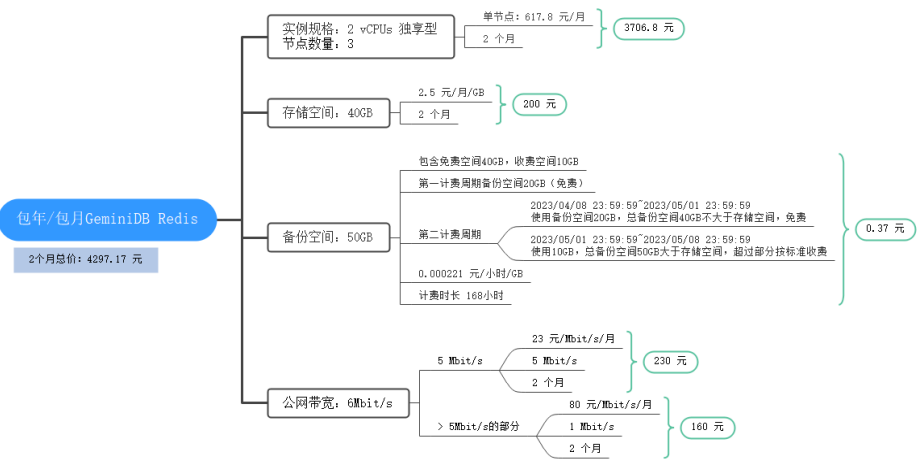
- 第一个计费周期为：2023/03/08 15:50:04 ~ 2023/04/08 23:59:59
- 第二个计费周期为：2023/04/08 23:59:59 ~ 2023/05/08 23:59:59
 - 2023/04/08 23:59:59~2023/05/01 23:59:59期间，使用免费备份空间20GB。
 - 2023/05/01 23:59:59~2023/05/08 23:59:59期间，使用计费备份空间10GB，计费时长168小时。

图2-7给出了上述示例配置的费用计算过程。

说明

图中价格仅为示例，实际计算请以[云数据库 GeminiDB价格详情](#)中的价格为准。

图 2-7 包年/包月 GeminiDB Redis 费用计算示例



按需计费模式下，各计费项的计费示例请参见[计费示例](#)。

2.4 计费样例

计费场景

某用户于2023/03/18 15:30:00购买了一个按需计费的GeminiDB Redis实例，规格配置如下：

- 规格：2vCPUs，12GB
- 节点数量：3
- 公网带宽：6Mbit/s

用了一段时间后，用户发现实例当前规格无法满足业务需要，于2023/03/20 9:00:00扩容规格为4vCPUs，24GB。因为该实例打算长期使用下去，于2023/03/20 10:30:00将实例转为包年/包月计费，购买时长为1个月。那么在3~4月份，该实例总共产生多少费用呢？

计费构成分析

可以将GeminiDB Redis的使用阶段按照计费模式分为两段：

- 2023/03/18 15:30:00 ~ 2023/03/20 10:30:00：按需计费
 - 2023/03/18 15:30:00 ~ 2023/03/20 9:00:00期间
 - 实例规格：2vCPUs，20GB
 - 节点数量：3
 - 使用存储空间：20GB
 - 使用备份空间：20GB
 - 公网带宽：6Mbit/s
 - 在2023/03/20 9:00:00 ~ 2023/03/20 10:30:00期间
 - 实例规格：4vCPUs，40GB
 - 节点数量：3
 - 使用存储空间：40GB
 - 使用备份空间：50GB（备份按需计费时间：2023/03/20 10:00:00~2023/03/20 10:30:00）
 - 公网带宽：6Mbit/s
- 2023/03/20 10:30:00 ~ 2023/04/20 23:59:59：包年/包月计费
 - 实例规格：4vCPUs，80GB
 - 节点数量：3
 - 使用存储空间：80GB
 - 使用备份空间：100GB（备份按需计费时间：2023/04/10 23:59:59 ~ 2023/04/20 23:59:59）

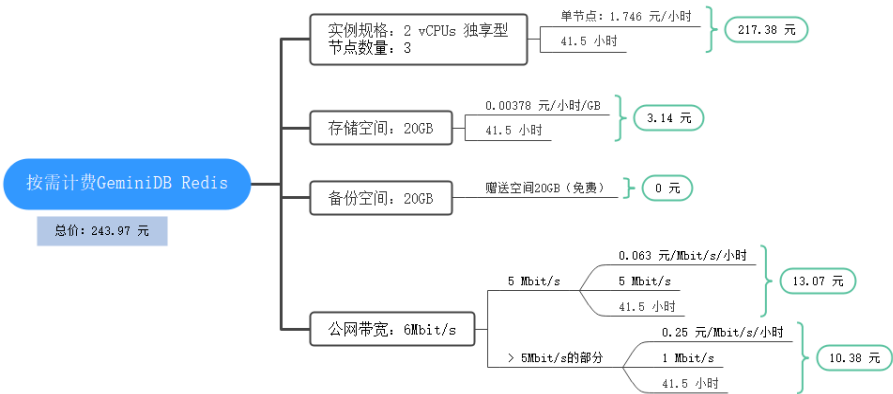
- 公网带宽：6Mbit/s
- 计费时长：1个月

说明

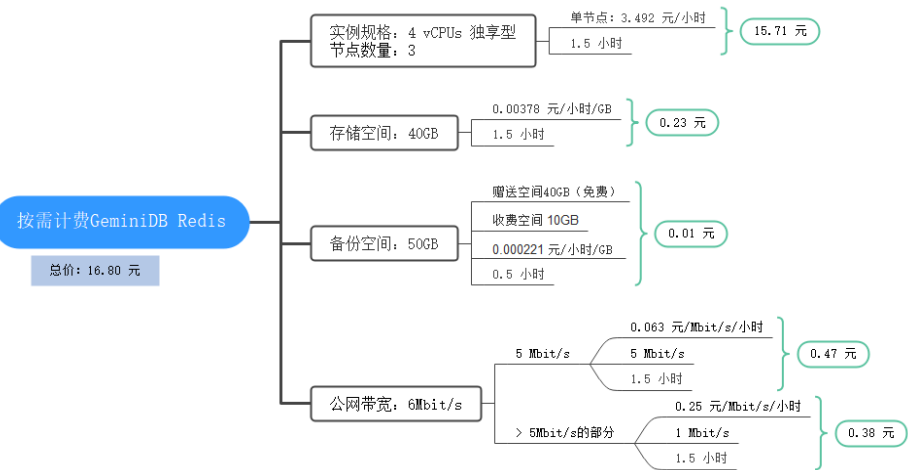
此案例中的单价仅为示例，且计算出的费用为估算值。单价的变动和实际场景中计算出来的费用可能会有偏差。请以华为云官网发布的数据为准。

按需计费

在2023/03/18 15:30:00 ~ 2023/03/20 9:00:00期间按照2vCPUs，20GB规格计费，计费时长为41.5小时，费用计算如下：

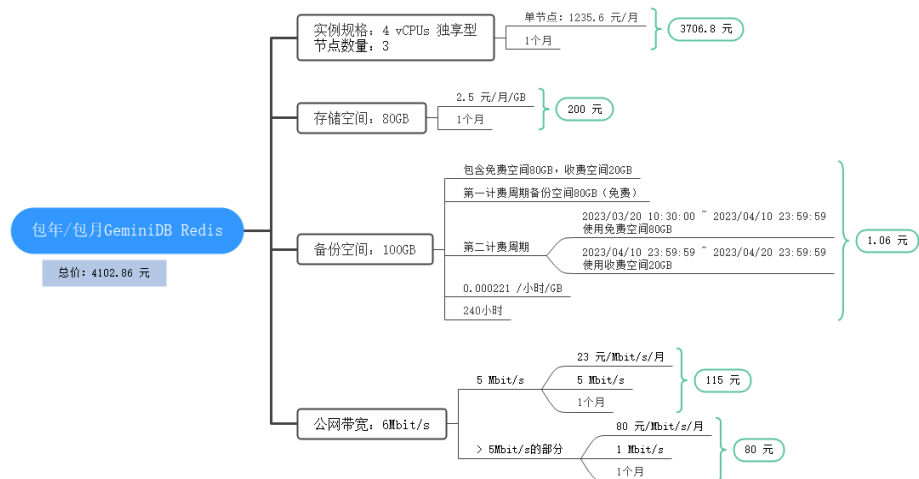


在2023/03/20 9:00:00 ~ 2023/03/20 10:30:00期间按照4vCPUs，40GB规格计费，计费时长为1.5小时，费用计算如下：



包年/包月计费

在2023/03/20 10:30:00 ~ 2023/04/20 23:59:59期间为包年/包月计费，计费时长为1个月，费用计算如下：



由此可见，在3~4月份，GeminiDB Redis总共产生的费用为：243.97 + 16.80 + 4102.86 = 4363.63 元。

2.5 变更计费模式

2.5.1 变更计费模式概述

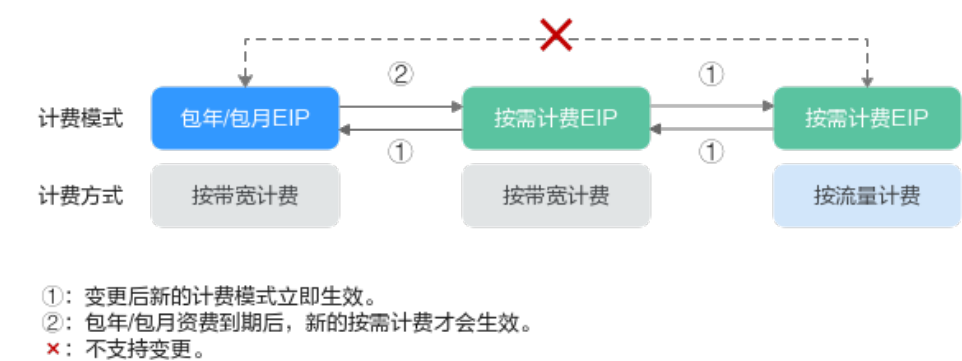
在购买GeminiDB Redis后，如果发现当前计费模式无法满足业务需求，您可以变更计费模式。支持变更计费模式的GeminiDB Redis计费项如表2-7所示。

表 2-7 支持变更计费模式的 GeminiDB Redis 计费项

计费项	变更说明	相关文档
实例规格 (vCPU和节点数量)	变更GeminiDB Redis实例的计费模式会同时变更计算资源（vCPU和节点数量）的计费模式。 - 将GeminiDB Redis实例的计费模式从按需计费转为包年/包月，可以让您享受一定程度的价格优惠。 - 将GeminiDB Redis实例的计费模式从包年/包月转为按需计费，可以帮助您回收部分成本，并更加灵活地使用GeminiDB Redis资源。 说明 包年/包月计费模式到期后，按需计费模式才会生效。	按需转包年/包月 包年/包月转按需

计费项	变更说明	相关文档
EIP	- 包年/包月EIP支持到期后转为按需、按带宽计费EIP。 - 按需、按带宽计费EIP支持转为包年/包月EIP。 - 按需、按带宽计费EIP支持和按需、按流量计费EIP互转。 具体变更方式可参考图2-8。	按需转包年/包月 包年/包月转按需

图 2-8 EIP 计费模式变更



2.5.2 按需转包年/包月

如果您需要长期使用当前按需购买的GeminiDB Redis实例，可以将该实例转为包年/包月计费模式，以节省开支。按需计费变更为包年/包月会生成新的订单，用户支付订单后，包年/包月资源将立即生效。

假设用户于2023/04/18 15:29:16购买了一个按需计费的GeminiDB Redis实例，由于业务需要，于2023/04/18 16:30:30执行按需转包年/包月操作，生成类型为“按需转包年/包月”的订单。用户支付订单后，按需转包年/包月的云数据库立即生效。在“费用中心 > 账单管理”页面将生成如下三条账单信息：

- 按需：2023/04/18 15:00:00 ~ 2023/04/18 16:00:00，在15:29:16 ~ 16:00:00间产生费用
- 按需：2023/04/18 16:00:00 ~ 2023/04/18 17:00:00，在16:00:00 ~ 16:30:30间产生费用
- 包年/包月：2023/04/18 16:30:30

约束与限制

按需计费云数据库绑定的资源（弹性公网IP）可能不支持随实例同步变更计费模式，请参考[表2-8](#)查看绑定资源的计费规则及处理措施。

表 2-8 弹性公网 IP 计费模式变更规则

资源	计费模式	计费方式	带宽类型	是否支持随 GeminiDB Redis按需转包年/包月	处理措施
弹性公网IP	按需计费	按带宽计费	独享带宽	是	在控制台的弹性公网IP页面进行按需转包年/包月操作。 详细内容，请参见 变更弹性公网IP计费方式 。
弹性公网IP	按需计费	按流量计费	独享带宽	否	按需、按流量计费EIP不支持直接转为包年/包月EIP。 变更方法如下： 1. 先转为按需、按带宽计费的EIP。 2. 再由按需、按带宽计费EIP转为包年/包月EIP。 详细内容，请参见 变更弹性公网IP计费方式 。

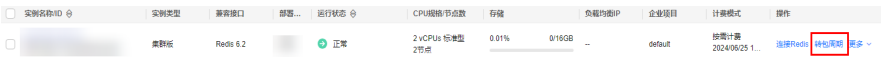
前提条件

- 实例计费模式为“按需计费”。
- 实例状态为“正常”。

操作步骤

- 步骤1 登录[云数据库GeminiDB控制台](#)。
- 步骤2 在“实例管理”页面，选择目标实例，选择操作列“转包周期”，进入“按需转包年/包月”页面。

图 2-9 按需转包年/包月



或者您也可以单击实例名称，进入基本信息页面，在“计费信息 > 计费模式”处，单击“转包周期”。

图 2-10 转包年/包月



说明

系统支持批量转包年/包月操作，即一次将多个按需计费的实例变更为包年/包月计费模式的。操作如下：

1. 勾选待转包年/包月的多个实例。
2. 单击实例列表上方“转包周期”。

步骤3 在“按需转包年/包月”页面，选择购买时长，以月为单位，最小包周期时长为一个月。

如果订单确认无误，单击“去支付”，进入“支付”页面。

步骤4 选择支付方式，单击“确认”。

步骤5 按需转包年/包月创建成功后，用户可以在“实例管理”页面对其进行查看和管理。

在实例列表的右上角，单击刷新列表，可查看到按需转包年/包月完成后，实例状态显示为“正常”。“计费模式”显示为“包年/包月”。

----结束

2.5.3 包年/包月转按需

创建一个包年/包月GeminiDB Redis实例后，您可以将该实例的计费模式转为按需计费，回收部分成本，同时更加灵活地按需使用实例。

假设用户于2023/04/18 15:29:16购买了一个包年包月的GeminiDB Redis实例，由于业务需要，于2023/05/18 16:30:00执行包年/包月转按需操作，生成类型为“包年包月转按需”。在“费用中心 > 账单管理”页面将生成如下账单信息：

- 包年/包月：2023/04/18 15:29:16 ~2023/05/18 23:59:59
- 按需：2023/05/18 23:59:59 ~ 按需使用结束时间，每小时生成一条账单

说明

包年/包月转按需，需包年/包月资费模式到期后，按需的资费模式才会生效，且自动续费功能会同步失效。

约束与限制

包年/包月的云数据库绑定的资源（弹性公网IP）可能不支持随云数据库同步变更计费模式，请参考[表2-9](#)查看绑定资源的计费规则及处理措施。

表 2-9 弹性公网 IP 计费模式变更规则

资源	计费模式	计费方式	带宽类型	是否支持随 GeminiDB Redis包年/包月转按需	处理措施
弹性公网IP	包年/包月	按带宽计费	独享带宽	否	在控制台的弹性公网IP页面进行按需转包年/包月操作。 详细内容，请参见 变更弹性公网IP计费方式 。
弹性公网IP	包年/包月	按流量计费	独享带宽	否	包年/包月EIP不支持直接转为按需、按流量计费EIP。变更方法如下： 1. 先转为按需、按带宽计费EIP。 2. 再由按需、按带宽计费EIP转为按需、按流量计费EIP。 详细内容，请参见 变更弹性公网IP计费方式 。

操作步骤

- 步骤1 登录[云数据库GeminiDB控制台](#)。
- 步骤2 在“实例管理”页面，选择目标实例，单击操作列“更多 > 转按需”，进入转按需页面。

图 2-11 包年/包月转按需



说明

系统支持批量转按需操作，即一次将多个包年/包月的实例变更为按需计费模式的。操作如下：

1. 勾选待转按需的多个实例。
2. 单击实例列表上方“转按需”。

- 步骤3 在转按需页面，核对实例信息无误后，单击“转按需”。包年/包月实例将在到期后转为按需计费实例。转按需成功后，自动续费将会被关闭，请谨慎操作。

- 步骤4 转按需申请提交后，在目标实例的“计费方式”列，会提示实例到期后转按需。
- 步骤5 如需取消转按需，您可以在费用中心的“续费管理”页签，在目标实例的“操作”列，选择“更多 > 取消转按需”。
- 步骤6 在弹出框中，单击“确定”，取消转按需申请。
- 结束

2.6 续费

2.6.1 续费概述

续费简介

包年/包月实例到期后会影响实例正常运行。如果您想继续使用，需要在指定的时间内为实例续费，否则vCPU、内存等资源会自动释放，数据丢失且不可恢复。

续费操作仅适用于包年/包月实例，按需计费实例不需要续费，只需要保证账户余额充足即可。

实例在到期前续费成功，所有资源得以保留，且实例的运行不受影响。实例到期后的状态说明，请参见[到期后影响](#)。

续费相关的功能

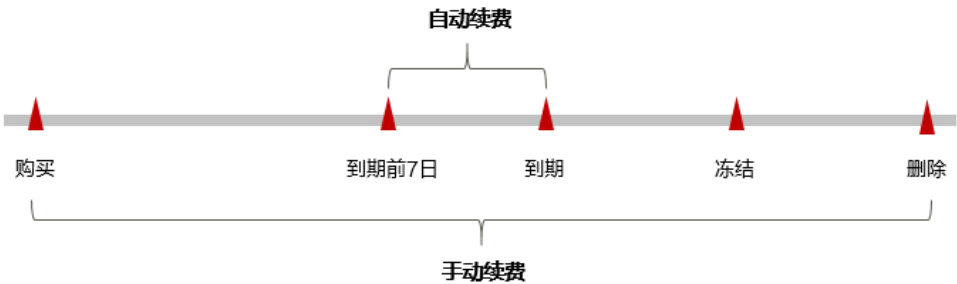
包年/包月实例续费相关的功能如[表2-10](#)所示。

表 2-10 续费相关的功能

功能	说明
手动续费	包年/包月实例从购买到被自动删除之前，您可以随时在控制台为实例续费，以延长实例的使用时间。
自动续费	开通自动续费后，实例会在每次到期前自动续费，避免因忘记手动续费而导致资源被自动删除。

在一台包年/包月实例生命周期的不同阶段，您可以根据需要选择一种方式进行续费，具体如[图2-12](#)所示。

图 2-12 实例生命周期



- 实例从购买到到期前，处于正常运行阶段，资源状态为“使用中”。
- 到期后，资源状态变为“已过期”。
- 到期未续费时，实例首先会进入宽限期，宽限期到期后仍未续费，资源状态变为“已冻结”。
- 超过宽限期仍未续费将进入保留期，如果保留期内仍未续费，资源将被自动删除。

 说明

- 华为云根据[客户等级](#)定义了不同客户的宽限期和保留期时长。

实例到期前均可开通自动续费，到期前7日凌晨3:00首次尝试自动续费，如果扣款失败，每天凌晨3:00尝试一次，直至实例到期或者续费成功。到期前7日自动续费扣款是系统默认配置，您也可以根据需要进行修改此扣款日。

2.6.2 手动续费

包年/包月实例从购买到被自动删除之前，您可以随时在控制台为实例续费，以延长实例的使用时间。

在云服务控制台续费

- 步骤1** 登录[云数据库GeminiDB控制台](#)。
- 步骤2** 在“实例管理”页面，选择需要续费的实例，单击操作列“续费”。

图 2-13 续费

实例名称ID	实例类型	兼容接口	部署...	运行状态	CPU/规格/节点数	存储	负载均衡IP	企业项目	计费模式	操作
	数据库	Redis 6.2		正常	2 vCPU/ 标准型 2节点	0.01% 0/16GB	..	default	包年/包月 30天宽限期	续费Redis 续费 更多

您也可以单击目标实例名称，进入实例的“基本信息”页面，在“计费信息”模块的“计费模式”处，单击“续费”。

图 2-14 续费

计费信息		订单
计费模式		CS2406261716BFYGA
包年/包月  开通自动续费		
创建时间		到期时间
2024/06/26 17:27:52 GMT+08:00		2024/07/26 23:59:59 GMT+08:00
到期策略		
到期进入宽限期 		

 说明

系统支持批量续费操作，即一次性为多个包年/包月的实例续费。操作如下：

1. 勾选待续费的多个实例。
2. 单击实例列表上方“续费”。

步骤3 进入续费页面，对实例进行续费。

----结束

在费用中心续费

步骤1 登录[云数据库GeminiDB控制台](#)。

步骤2 单击“控制台”页面上方“费用”，在下拉菜单中选择“续费管理”。
进入“续费管理”页面。

步骤3 自定义查询条件。

可在“手动续费项”、“自动续费项”、“到期转按需项”、“到期不续费项”页签查询全部待续费资源，对资源进行手动续费的操作。

图 2-15 续费管理



所有需手动续费的资源都可归置到“手动续费项”页签，具体操作请参见[如何恢复为手动续费](#)。

步骤4 手动续费资源。

- 单个续费：在资源页面找到需要续费的资源，单击操作列的“续费”。

图 2-16 单个续费



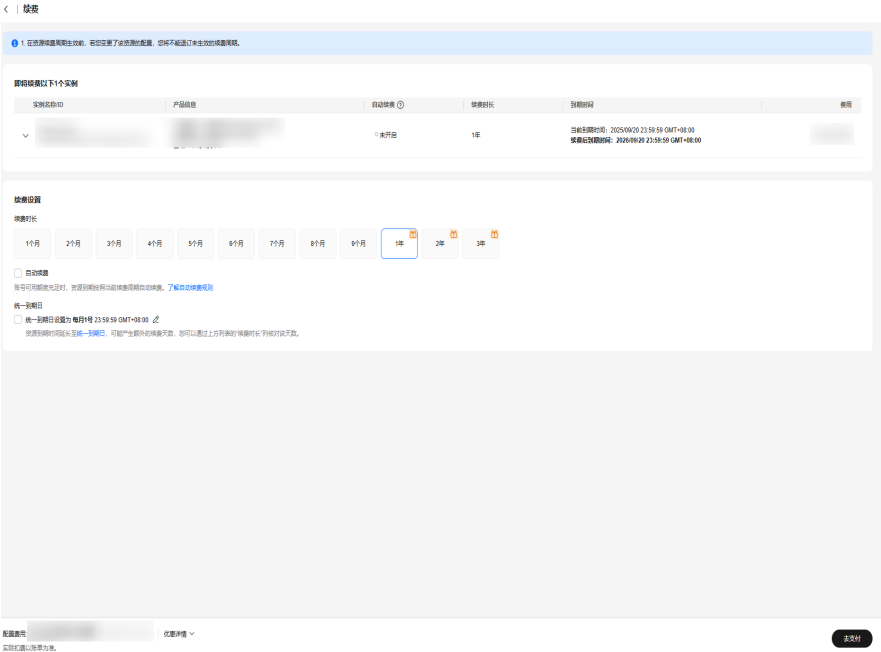
- 批量续费：在资源页面勾选需要续费的资源，单击列表左上角的“批量续费”。

图 2-17 批量续费



步骤5 选择续费时长，判断是否勾选“统一到期日”，将到期时间统一到各个月的某一天（详细介绍请参见[统一包年/包月资源的到期日](#)）。确认配置费用后单击“去支付”。

图 2-18 续费确认



步骤6 进入支付页面，选择支付方式，确认付款，支付订单后即可完成续费。
----结束

统一包年/包月资源的到期日

如果您持有多个到期日不同的实例，可以将到期日统一设置到一个日期，便于日常管理和续费。

图2-19展示了用户将两个不同时间到期的资源，同时续费一个月，并设置“统一到期日”后的效果对比。

图 2-19 统一到期日



更多关于统一到期日的规则请参见[如何设置统一到期日](#)。

2.6.3 自动续费

自动续费可以减少手动续费的管理成本，避免因忘记手动续费而导致云数据库被自动删除。自动续费的规则如下所述：

- 以实例的到期日计算第一次自动续费日期和计费周期。
- 实例自动续费周期以您选择的续费时长为准。
 - 按月购买：自动续费周期为1个月。
 - 按年购买：自动续费周期为1年。
- 在实例到期前均可开通自动续费，到期前7日凌晨3:00首次尝试自动续费，如果扣款失败，每天凌晨3:00尝试一次，直至实例到期或者续费成功。
- 开通自动续费后，还可以手动续费该实例。手动续费后，自动续费仍然有效，在新的到期时间前的第7天开始扣款。
- 自动续费的到期前7日自动扣款属于系统默认配置，您也可以根据需要修改此扣款日，如到期前6日、到期前5日等等。

更多关于自动续费的规则介绍请参见[自动续费规则说明](#)。

前提条件

请确认包年/包月实例还未到期。

在购买实例页面开通自动续费

您可以在购买数据库实例页面开通自动续费，如[图2-20](#)所示。更多购买实例的信息，请参见[购买实例](#)。

图 2-20 自动续费配置



在续费管理页面开通自动续费

- 步骤1** 登录[云数据库GeminiDB控制台](#)。
- 步骤2** 单击“控制台”页面上方“费用”，在下拉菜单中选择“续费管理”，进入“续费管理”页面。
- 步骤3** 自定义查询条件。
- 可在“自动续费项”页签查询已经开通自动续费的资源。
 - 可对“手动续费项”、“到期转按需项”、“到期不续费项”页签的资源开通自动续费。

图 2-21 续费管理



步骤4 为包年/包月资源开通自动续费。

- 单个资源开通自动续费：选择需要开通自动续费的实例，单击操作列“开通自动续费”。

图 2-22 单个资源开通自动续费



- 批量资源开通自动续费：选择需要开通自动续费的实例，单击列表左上角的“开通自动续费”。

图 2-23 多个资源开通自动续费



步骤5 选择续费时长，并根据需要设置自动续费次数，单击“开通”。

图 2-24 开通自动续费



----结束

2.7 费用账单

您可以在“费用中心 > 账单管理”查看资源的费用账单，以了解该资源在某个时间段的使用量和计费信息。

账单上报周期

包年/包月计费模式的资源完成支付后，会实时上报一条账单到计费系统进行结算。

按需计费模式的资源按照固定周期上报使用量到计费系统进行结算。按需计费模式产品根据使用量类型的不同，分为按小时、按天、按月三种周期进行结算，具体扣费规则可以参考[按需产品周期结算说明](#)。GeminiDB Redis的按需计费模式按小时进行结算。

按需计费资源的扣费时间可能会滞后于结算周期，例如：按小时结算的GeminiDB Redis实例在8:30删除资源，但是8:00~9:00期间产生的费用，通常会在10:00左右才进行扣费。在“费用中心 > 账单管理 > 流水和明细账单 > 流水账单”中，“消费时间”即按需产品的实际使用时间。

查看指定资源的账单

【方法一：使用资源ID查询账单】

- 步骤1 登录[云数据库GeminiDB控制台](#)。
- 步骤2 在“实例管理”页面，单击目标实例名称，进入基本信息页面。
- 步骤3 在如下图所示位置单击鼠标左键，复制实例ID。

图 2-25 获取实例 ID



- 步骤4 在控制台顶部菜单栏中选择“费用 > 费用账单”。

进入费用账单页面。

步骤5 选择“账单管理 > 流水和明细账单 > 明细账单”，在筛选条件中选择“资源ID”，并输入资源ID，单击图标即可搜索该资源的账单。

图 2-26 查询资源账单

账期2023/12

统计维度按使用量统计周期按账期按天按周按月如何查找资源?

资源名称: 添加筛选条件

xQ

账期	企业项目	账号	产品类型	产品	计费模式	账单类型	资源名称ID	资源标签	规格	区域	可用区	使用量类型	单价
2023/12	default		云数据库 Ge...	GeminiDB ...	按需	消费-使用	geminidb-ge... a9307387cb...	-			可用区...	时长	0.003
2023/12	default		云数据库 Ge...	GeminiDB ...	按需	消费-使用	geminidb-ge... e462183054...	-			可用区2	时长	1.7
2023/12	default		云数据库 Ge...	GeminiDB ...	按需	消费-使用	geminidb-ge... 3090a1909...	-			可用区1	时长	1.7
2023/12	default		云数据库 Ge...	GeminiDB ...	按需	消费-使用	geminidb-ge... 1e992180b9...	-			可用区3	时长	1.7
2023/12	default		云数据库 Ge...	GeminiDB ...	按需	消费-使用	geminidb-ge... a9307387cb...	-			可用区...	规格类型	0

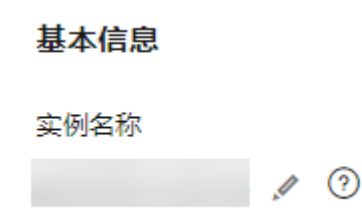
这里设置的统计维度为“按使用量”，统计周期为“按账期”，您也可以设置其他统计维度和周期，详细介绍请参见[流水与明细账单](#)。

----结束

【方法二：使用资源名称查询账单】

- 步骤1 登录[云数据库GeminiDB控制台](#)。
- 步骤2 在“实例管理”页面，单击目标实例名称，进入基本信息页面。
- 步骤3 在“基本信息”页面，获取实例名称。

图 2-27 获取实例名称



- 步骤4 在控制台顶部菜单栏中选择“费用 > 费用账单”。
- 进入费用账单页面。
- 步骤5 选择“账单管理 > 流水和明细账单 > 明细账单”，在筛选条件中选择“资源名称”，并输入资源名称，单击图标即可搜索该资源的账单。

图 2-28 查询资源账单

账期2023/12

统计维度按使用量

统计周期按账期

按天

明细

如何查找资源?

资源名称:

添加筛选条件

x

q

+

-

账期	企业项目	账号	产品类型	产品	计费模式	账单类型	资源名称ID	资源标签	规格	区域	可用区	使用量类型	单价
2023/12	default		云数据库 Ge...	GeminiDB ...	按需	消费-使用	geminidb-6e-a9307387cb...	-			可用区...	时长	0.003
2023/12	default		云数据库 Ge...	GeminiDB ...	按需	消费-使用	geminidb-6e-e4e2103b54...	-			可用区2	时长	1.7
2023/12	default		云数据库 Ge...	GeminiDB ...	按需	消费-使用	geminidb-6e-308da1900...	-			可用区1	时长	1.7
2023/12	default		云数据库 Ge...	GeminiDB ...	按需	消费-使用	geminidb-6e-1e9921809...	-			可用区3	时长	1.7
2023/12	default		云数据库 Ge...	GeminiDB ...	按需	消费-使用	geminidb-6e-a9307387cb...	-			可用区...	规格类型	0

这里设置的统计维度为“按使用量”，统计周期为“按账期”，您也可以设置其他统计维度和周期，详细介绍请参见[流水与明细账单](#)。

----结束

场景示例：核对资源用量是否与实际相符

假设用户在2023/04/08 10:09:06购买了一个按需计费GeminiDB Redis实例，并在2023/04/08 12:09:06时刻将其删除。

- GeminiDB Redis实例流水账单
按需计费GeminiDB Redis实例按秒计费，每一个小时整点结算一次费用，您可以在流水账单中核对每一个计费周期的信息是否和实际相符，流水账单中计费资源是分开计费的，以“GeminiDB Redis存储”为例，具体如[表2-11](#)所示。

表 2-11 GeminiDB Redis 流水账单

产品类型	GeminiDB Redis
产品	GeminiDB Redis存储
计费模式	按需
消费时间	2023/04/08 10:09:06 ~ 2023/04/08 12:09:06时段计费系统将生成6笔流水账单，对应每一个计费周期，分别如下： 2023/04/08 10:09:06 ~ 2023/04/08 11:00:00 2023/04/08 11:00:00 ~ 2023/04/08 12:00:00 2023/04/08 12:00:00 ~ 2023/04/08 12:09:06
官网价	官网价=使用量*单价*容量 本例中，在第一个计费周期内GeminiDB Redis的使用量为3054秒，单价可在 云数据库 GeminiDB价格详情 中查询，以0.00378元/GB/小时为例，容量为40GB，那么官网价=(3054 ÷ 3600) * 0.00378 * 40 = 0.128268 元。同理，您可以计算剩余计费周期内资源的官网价。
优惠金额	用户使用云服务享受折扣优惠如商务折扣、伙伴授予折扣以及促销优惠等减免的金额。基于官网价的优惠金额。

抹零金额	华为云产品定价精度为小数点后8位（单位：元），因此在计费过程中会产生小数点后8位的资源使用费用。而在实际扣费时，仅扣除到小数点后2位，小数点后第3位到第8位部分金额会被舍弃，这种舍弃部分的金额称作抹零金额。 以第一个计费周期为例，抹零金额为：0.008268 元
应付金额	应付金额=官网价-优惠金额-抹零金额 以第一个计费周期为例，假设优惠金额为0，那么应付金额=0.128268 - 0 - 0.008268 = 0.12 元。

- GeminiDB Redis实例明细账单
明细账单可以通过多维度展示客户账单的详细信息。一般通过设置统计维度为“按使用量”，统计周期为“按账期”来统计资源在某个月份的总开销，以“GeminiDB Redis存储”为例，建议您核对[表2-12](#)所示的信息是否和实际相符。

表 2-12 GeminiDB Redis 明细账单

产品类型	GeminiDB Redis
产品	GeminiDB Redis存储
计费模式	按需
资源名称/ID	GeminiDB Redis的名称和ID 例如：nosql-b388，21e8811a64bf4de88bc2e2556da17983in12
规格	GeminiDB Redis存储
使用量类型	按需计费GeminiDB Redis的使用量类型为“时长”
单价	按需计费模式为简单定价（使用量*单价）时提供单价信息，其他的定价（如EIP公网带宽的阶梯定价）等不提供单价。 按需计费GeminiDB Redis属于简单定价，您可以在 云数据库 GeminiDB价格详情 中查中查询单价。
单价单位	在 云数据库 GeminiDB价格详情 中查中查询到的单价单位：元/GB/小时
使用量	按产品单价单位显示使用量，GeminiDB Redis的单价单位为元/GB/小时，因此使用量以小时为单位。本例中，2023/04/08 10:09:06 ~ 2023/04/08 12:09:06时段总计使用量为2小时。
使用量单位	小时
官网价	官网价=使用量*单价*容量 本例中，使用量为2小时，单价可在 云数据库 GeminiDB价格详情 中查中查询，以0.00378元/GB/小时为例，容量为40GB，那么官网价=2 * 0.00378 * 40 = 0.3024 元。

优惠金额	用户使用云服务享受折扣优惠如商务折扣、伙伴授予折扣以及促销优惠等减免的金额。基于官网价的优惠金额。
应付金额	用户使用云服务享受折扣优惠后需要支付的费用金额。

2.8 欠费说明

用户在使用实例时，账户的可用额度小于待结算的账单，即被判定为账户欠费。欠费后，可能会影响实例资源的正常运行，请及时充值。

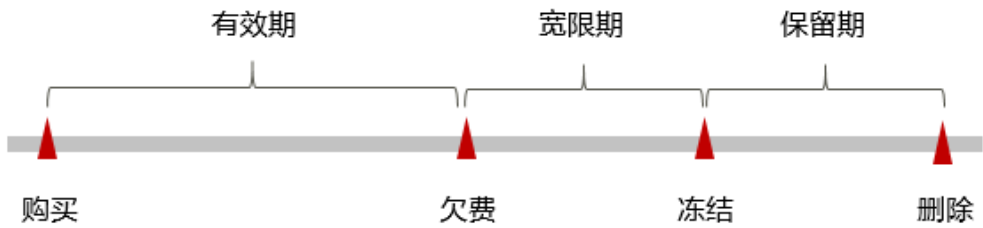
欠费原因

未购买包年包月实例，在按需计费模式下账户的余额不足。

欠费影响

- 包年/包月
对于包年/包月GeminiDB Redis资源，用户已经预先支付了资源费用，因此在账户出现欠费的情况下，已有的包年/包月GeminiDB Redis资源仍可正常使用。然而，对于涉及费用的操作，如新购GeminiDB Redis、升级规格、续费订单等，用户将无法正常进行。
- 按需计费
当您的账号因按需GeminiDB Redis资源自动扣费导致欠费后，账号将变成欠费状态。欠费后，按需资源不会立即停止服务，资源进入宽限期。您需支付按需资源在宽限期内产生的费用，相关费用可在管理控制台 > 费用中心 > 总览“欠费金额”查看，华为云将在您充值时自动扣取欠费金额。
如果您在宽限期内仍未支付欠款，那么就会进入保留期，资源状态变为“已冻结”，您将无法对处于保留期的按需计费资源执行任何操作。
保留期到期后，若您仍未支付账户欠款，那么计算资源（vCPU和内存）和弹性公网IP都将被释放，数据无法恢复。

图 2-29 按需计费资源生命周期



说明

华为云根据[客户等级](#)定义了不同客户的宽限期和保留期时长。

避免和处理欠费

欠费后需要及时充值，详细操作请参见[账户充值](#)。

若GeminiDB Redis实例不再使用，您可以将他们删除，以免继续扣费。

您可以在“费用中心 > 总览”页面设置“可用额度预警”功能，当可用额度的总额度低于预警阈值时，系统自动发送短信和邮件提醒。

2.9 停止计费

包年/包月资源

对于包年/包月计费模式的资源，例如包年/包月的GeminiDB Redis实例，用户在购买时会一次性付费，服务将在到期后自动停止使用。

- 如果在计费周期内不再使用包年/包月资源，您可以执行退订操作，系统将根据资源是否属于五天无理由退订、是否使用代金券和折扣券等条件返还一定金额到您的账户。详细的退订规则请参见[云服务退订](#)。
- 如果您已开启“自动续费”功能，为避免继续产生费用，请在自动续费扣款日（默认为到期前7日）之前关闭自动续费。

按需计费资源

对于按需计费模式的资源，例如按需计费的GeminiDB Redis实例，若不再使用这些资源且需停止计费，请删除相应资源。

通过账单查找云服务资源并停止计费

为了确保所有计费资源都能够被找到并删除，您可以在费用账单中获取计费资源的ID，通过这些ID信息找到具体的资源，然后将其一一删除。具体操作如下：

【方法一：使用资源ID查找资源】

- 步骤1** 登录[云数据库GeminiDB控制台](#)。
- 步骤2** 在顶部菜单栏中选择“费用 > 费用账单”。进入费用账单页面。
- 步骤3** 选择“账单管理 > 流水和明细账单 > 明细账单”，在如下图所示位置单击鼠标左键，复制资源ID。

图 2-30 复制资源 ID



- 步骤4** 登录[云数据库GeminiDB控制台](#)。
- 步骤5** 选择资源所在的区域，在筛选条件中选择“实例ID”，并输入**步骤3**中复制的资源ID，单击图标即可查找到该资源。

图 2-31 查找资源



步骤6 单击操作列的“更多 > 删除实例”，对实例执行删除操作，确保列表中已无该资源。

说明

按需计费资源删除后，可能还会存在账单信息，因为系统通常会在使用后1个小时内对上一个结算周期的费用进行扣款。例如在8:30删除按小时结算的实例资源，但是8:00~9:00期间产生的费用，通常会在10:00左右才进行扣费。

----结束

【方法二：使用资源名称查找资源】

- 步骤1 登录[云数据库GeminiDB控制台](#)。
- 步骤2 在顶部菜单栏中选择“费用 > 费用账单”。进入费用账单页面。
- 步骤3 选择“账单管理 > 流水和明细账单 > 明细账单”，在如下图所示位置单击鼠标左键，复制资源名称。

图 2-32 复制资源名称



- 步骤4 登录[云数据库GeminiDB控制台](#)。
- 步骤5 在搜索框中输入**步骤3**中复制的资源名称，单击图标即可查找到该资源。

图 2-33 查找资源



步骤6 单击操作列的“更多 > 删除实例”，对实例执行删除操作，确保列表中已无该资源。

说明

按需计费资源删除后，可能还会存在账单信息，因为系统通常会在使用后1个小时内对上一个结算周期的费用进行扣款。例如在8:30删除按小时结算的实例资源，但是8:00~9:00期间产生的费用，通常会在10:00左右才进行扣费。

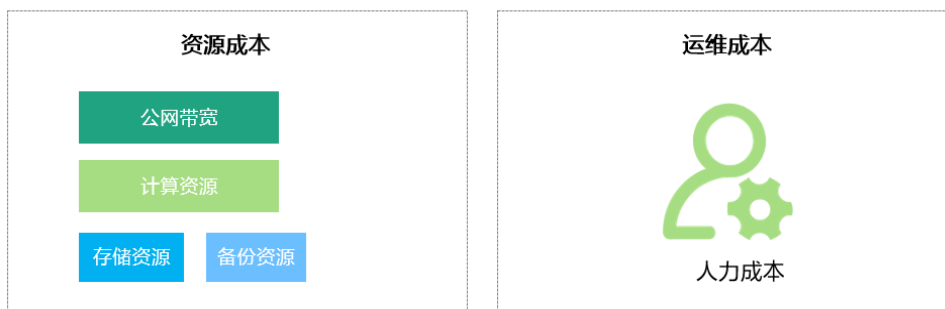
----结束

2.10 成本管理

2.10.1 成本构成

GeminiDB Redis的成本主要包括两方面：

- 资源成本：计算、存储等在内的资源成本，详细介绍请参见[计费模式](#)。
- 运维成本：使用GeminiDB Redis过程中产生的人力成本。



2.10.2 成本分配

成本管理的基础是树立成本责任制，让各部门、各业务团队、各责任人参与进来，为各自消耗云服务产生的成本负责。企业可以通过成本分配的方式，将云上成本分组，归集到特定的团队或项目业务中，让各责任组织及时了解各自的成本情况。

华为云[成本中心](#)支持通过多种不同的方式对成本进行归集和重新分配，您可以根据需要选择合适的分配工具。

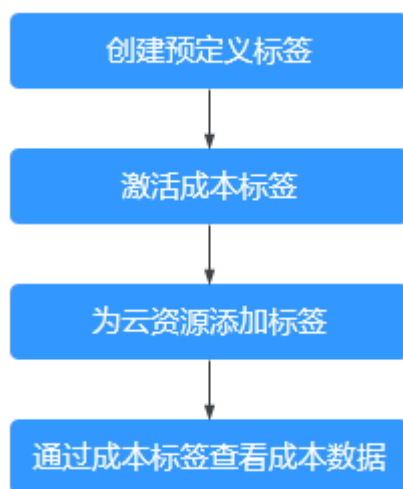
- **通过关联账号进行成本分配**
企业主客户可以使用关联账号对子客户的成本进行归集，从而对子账号进行财务管理。详细介绍请参见[通过关联账号维度查看成本分配](#)。
- **通过企业项目进行成本分配**
在进行成本分配之前，建议开通企业项目并做好企业项目的规划，可以根据企业的组织架构或者业务项目来划分。在购买云资源时选择指定企业项目，新购云资源将按此企业项目进行成本分配。详细介绍请参见[通过企业项目维度查看成本分配](#)。

图 2-34 选择企业项目



- **通过成本标签进行成本分配**
标签是华为云为了标识云资源，按各种维度（例如用途、所有者或环境）对云资源进行分类的标记。推荐企业使用预定义标签进行成本归集，具体流程如下：

图 2-35 添加标签



详细介绍请参见[通过成本标签维度查看成本分配](#)。

- **使用成本分组进行成本分配**

企业可以使用[成本中心](#)的“成本分组”来拆分公共成本。公共成本是指多个部门共享的计算、网络、存储或资源包产生的云成本，或无法直接通过企业项目、成本标签分配的云成本。这些成本不能直接归属于单一所有者，因此不能直接归属到某一类别。使用拆分规则，可以在各团队或业务部门之间公平地分配这些成本。详细介绍请参见[使用成本分组查看分配成本](#)。

2.10.3 成本分析

企业只有了解组织中哪些方面产生了成本，才能正确地控制和优化成本。[成本中心](#)支持使用“成本分析”的汇总和过滤机制可视化企业的原始成本和摊销成本，从而通过各种角度、范围分析成本和用量的趋势及驱动因素。

企业还可以通过[成本中心](#)的“成本监控”，及时发现计划外费用，做到成本的可监控、可分析和可追溯。

详细介绍请参见[通过成本分析探索成本和使用量](#)、[通过成本监控识别异常成本](#)。

2.10.4 成本优化

在您通过成本中心了解和分析您的成本情况后，您可以确定成本偏高的原因，然后采取针对性的优化措施。

资源优化

- 通过CES查看GeminiDB Redis监控指标，例如CPU、内存、磁盘的使用率，如果当前配置过高，可以通过规格变更降低配置。
- 监控GeminiDB Redis资源闲置情况，及时删除闲置的实例。

- 如果您的业务对性能稳定性要求较低，可以考虑购买通用型规格实例，以此来降低您的成本。例如通用型4U24GB的规格相比较于独享型的4U24GB的规格，每月可降低约30%的成本。

计费模式优化

不同类型的业务对资源使用周期有不同的要求，为每一类业务确定合适的计费模式，灵活组合以达到最优效果。

- 针对长期稳定的成熟业务，使用包年/包月计费模式。
- 针对不能中断的短期、突增或不可预测的业务，使用按需计费模式。
- 监控实例生命周期，对即将到期的包周期资源进行及时续费。

2.11 计费 FAQ

2.11.1 包年/包月和按需计费有什么区别

包年/包月的计费模式也称为包周期计费模式，是一种预付费方式，按订单的购买周期计费，适用于可预估资源使用周期的场景，价格比按需计费模式更优惠。对于长期使用用户，推荐该方式。

按需计费是后付费模式，按实际使用时长计费，这种购买方式比较灵活，可以即开即停。以自然小时为单位整点计费，不足一小时按使用时长计费。

2.11.2 GeminiDB Redis 支持包年/包月和按需计费模式相互转换吗

包年/包月和按需计费模式支持相互转换。

- GeminiDB Redis包年/包月实例转按需实例请参见[包年/包月转按需](#)。
- GeminiDB Redis按需实例转包年/包月实例请参见[按需转包年/包月](#)。

2.11.3 如何进行单个包年/包月或者批量实例续费

您可根据业务需要，对GeminiDB Redis“包年/包月”实例进行续费。

使用须知

- GeminiDB Redis的计费详情请参见[计费说明](#)。
- 按需计费的实例不支持续费。

单个包年/包月实例续费

步骤1 登录[云数据库GeminiDB控制台](#)。

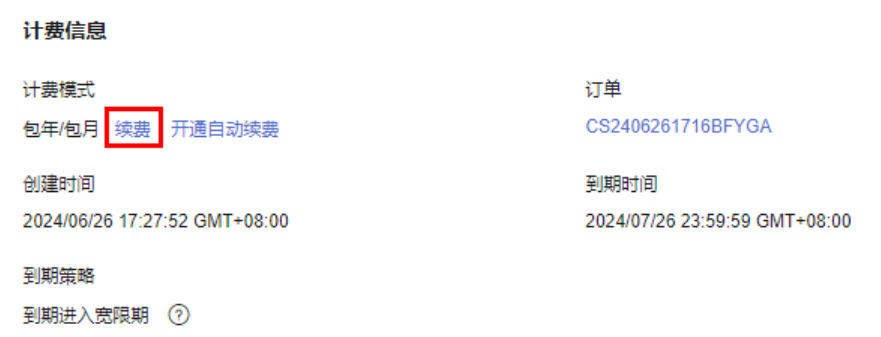
步骤2 在“实例管理”页面，选择需要续费的实例，单击操作列“续费”。

图 2-36 续费

实例名称ID	实例类型	实例规格	部署模式	运行状态	CPU规格/节点数	存储	负载均衡IP	企业项目	计费模式	操作
	通用型	Redis 6.2	正常		2 vCPU/ 标准型 2节点	0.01%	0/15GB	default	包年包月 365天到期	续费 更多

您也可以单击目标实例名称，进入实例的“基本信息”页面，在“计费信息”模块的“计费模式”处，单击“续费”。

图 2-37 续费



步骤3 进入续费页面，对实例进行续费。

----结束

批量续费

- 步骤1 登录[云数据库GeminiDB控制台](#)。
- 步骤2 在“实例管理”页面，勾选目标实例，单击实例列表上方的“续费”。

图 2-38 批量续费



步骤3 在弹出框中确认需要续费的实例，单击“是”，进入续费页面，对实例进行续费。

----结束

2.11.4 如何退订包年/包月实例

对于“包年/包月”模式的数据库实例，您需要退订订单，从而删除数据库实例资源。

使用须知

- GeminiDB Redis的计费详情请参见[计费说明](#)。
- 退订操作无法恢复，请谨慎操作。如需保留数据，请您在退订之前先创建一个[手动备份](#)。
- 提交退订后，资源和数据将会被删除并无法找回。如需保留数据，请务必确认完成数据备份后再提交退订。

退订单个包年/包月实例

步骤1 登录[云数据库GeminiDB控制台](#)。

步骤2 在“实例管理”页面，选择需要退订的目标实例，在“操作”列选择“更多 > 退订”。

图 2-39 退订包年/包月



- 步骤3** 在弹出框中，单击“是”，进入“退订资源”页面。
- 步骤4** 在“退订资源”页面，确认待退订实例信息，并选择退订原因，单击“退订”。
- 资源退订相关信息，请参考[退订规则说明](#)。
- 步骤5** 在弹出框中确认是否退订并删除该资源，单击“是”，提交退订申请。

说明

- 1. 提交退订后，资源和数据将会被删除并无法找回。
- 2. 如需保留数据，请务必确认完成数据备份后再提交退订。

步骤6 查看退订结果。数据库实例订单退订成功后，实例将会被删除，即“实例管理”页面，将不再显示该订单对应的数据库实例。

----结束

批量退订包年/包月实例

- 步骤1** 登录[云数据库GeminiDB控制台](#)。
- 步骤2** 在“实例管理”页面，选择需要退订的目标实例，单击实例列表上方的“退订”。

图 2-40 批量退订包年/包月



- 步骤3** 在弹出框中，单击“是”，进入“退订资源”页面。
- 步骤4** 在“退订资源”页面，确认待退订实例信息，并选择退订原因，单击“退订”。
- 资源退订相关信息，请参考[退订规则说明](#)。
- 步骤5** 在弹出框中确认是否退订并删除该资源，单击“是”，提交退订申请。

说明

1. 提交退订后，资源和数据将会被删除并无法找回。
2. 如需保留数据，请务必确认完成数据备份后再提交退订。

步骤6 查看退订结果。数据库实例订单退订成功后，实例将会被删除，即“实例管理”页面，将不再显示该订单对应的数据库实例。

----结束

3 快速入门

对于初次使用GeminiDB Redis的用户，从实例创建到管理运维的过程中可能会遇到多种挑战，比如配置复杂、连接不稳定等。本章节将帮助您快速掌握GeminiDB Redis实例从创建、连接、开发到管理运维的整体流程，提供详细的步骤指导和最佳实践建议。

3.1 快速了解 Redis

本章节主要介绍GeminiDB Redis产品类型和实例类型，并帮助您快速掌握GeminiDB Redis实例创建、连接的整体流程。

表 3-1 产品类型

产品类型	使用场景	支持的实例类型
标准型	提供稳定的低延迟性能，适用于广告&推荐、游戏、电商、车联网等各类常见业务场景。	- Proxy集群版 - Cluster集群版 - 主备版
容量型	提供大容量Key-Value存储，适用于对性能要求不高且有降低成本诉求的业务。	Proxy集群版

表 3-2 实例类型

实例类型	使用场景	购买并连接实例
集群版	- Proxy集群版实例采用分片集群架构，提供Proxy代理连接，同时兼容单机、哨兵、Cluster接入方式。性能水平扩展能力强，可支撑百万QPS、数十TB级业务量。 - Cluster集群版实例采用原生Cluster分片集群架构，完全兼容Redis Cluster直连访问接入，时延更低，性能更优。	购买并连接集群版实例

实例类型	使用场景	购买并连接实例
主备版	标准的master-replica架构，兼容单机、哨兵接入方式。适用于无法使用hashtag的业务场景。	购买并连接主备版实例

连接方式介绍

数据管理服务（Data Admin Service，简称DAS）是一款专业的简化数据库管理工具，提供优质的可视化操作界面，大幅提高工作效率，让数据管理变得既安全又简单。您可以通过数据管理服务连接并管理实例。云数据库GeminiDB Redis服务默认为您开通了远程主机登录权限，推荐您使用更安全便捷的数据管理服务连接实例。

表 3-3 DAS 连接方式

连接方式	使用场景	说明
DAS连接	无需使用IP地址，通过控制台即可登录实例。	- 易用、安全、高级、智能。 - 默认为您开通了远程主机登录权限，推荐您使用更安全便捷的数据管理服务连接实例。

更多连接操作

- [连接方式介绍](#)

3.2 购买并连接集群版实例

本章节主要介绍了如何在云数据库 GeminiDB管理控制台购买Redis数据库的Proxy集群版或者Cluster集群版实例并通过控制台连接实例。

- Proxy集群版实例采用分片集群架构，提供Proxy代理连接，同时兼容单机、哨兵、Cluster接入方式。性能水平扩展能力强，可支撑百万QPS、数十TB级业务量。
- Cluster集群版实例采用原生Cluster分片集群架构，完全兼容Redis Cluster直连访问接入，时延更低，性能更优。

每个租户下GeminiDB Redis实例的默认总配额为50。如需申请扩大配额，您可以在管理控制台右上角，选择“[工单 > 新建工单](#)”，联系客服申请开通。

- [步骤1：购买实例](#)
- [步骤2：通过DAS连接实例](#)

如果需要其他方式连接实例请参考[其他连接实例方式](#)。

步骤 1：购买实例

- 登录[云数据库GeminiDB控制台](#)。
- 在“实例管理”页面，单击“购买数据库实例”，进入“服务选型”页面。

3. 在“服务选型”页面，选择计费模式，填写并选择实例相关信息后，单击“立即购买”。

下面的配置项仅作为示例，实际购买时根据用户需求选择合适的规格参数，具体参数详见[参数详解](#)。

图 3-1 计费模式和基本信息（Proxy 集群版）

计费模式

包年包月

按需计费

区域

华东-上海

不同区域的价格之间可能不一致，请按照您业务所在的区域，可以降低网络时延，提高访问速度。

实例名称

gemini-1

购买多个数据库实例时，名称必须按顺序递增。例如：输入Instance_从Instance-0101-120101-01开始命名，每日按Instance-0101-120101-01，到Instance-0101-120101-62开始命名。

兼容接口

Redis

Cassandra

DynamoDB

HBase

InfluxDB

部署模式

集群

云原生

传统架构，独立部署。

产品规格

4核8G

8核16G

按规格选择规格名称，适用于广域网、游戏、电商、车联网等场景可灵活使用。

实例架构

Proxy集群版

Cluster集群版

主备版

按架构选择规格名称，按架构选择规格名称，按架构选择规格名称，按架构选择规格名称。

实例版本

7.0

按版本选择规格名称，按版本选择规格名称，按版本选择规格名称，按版本选择规格名称。

CPU规格

4核

按CPU规格选择规格名称，按CPU规格选择规格名称，按CPU规格选择规格名称，按CPU规格选择规格名称。

可用区

可用区1（可用区2、可用区3）

按可用区选择规格名称，按可用区选择规格名称，按可用区选择规格名称，按可用区选择规格名称。

图 3-2 计费模式和基本信息（Cluster 集群版）

计费模式

包年包月

按需计费

区域

华东-上海

不同区域的价格之间可能不一致，请按照您业务所在的区域，可以降低网络时延，提高访问速度。

实例名称

gemini-1

购买多个数据库实例时，名称必须按顺序递增。例如：输入Instance_从Instance-0101-120101-01开始命名，每日按Instance-0101-120101-01，到Instance-0101-120101-62开始命名。

兼容接口

Redis

Cassandra

DynamoDB

HBase

InfluxDB

部署模式

集群

云原生

传统架构，独立部署。

产品规格

4核8G

8核16G

按规格选择规格名称，适用于广域网、游戏、电商、车联网等场景可灵活使用。

实例架构

Proxy集群版

Cluster集群版

主备版

按架构选择规格名称，按架构选择规格名称，按架构选择规格名称，按架构选择规格名称。

实例版本

7.0

按版本选择规格名称，按版本选择规格名称，按版本选择规格名称，按版本选择规格名称。

CPU规格

4核

按CPU规格选择规格名称，按CPU规格选择规格名称，按CPU规格选择规格名称，按CPU规格选择规格名称。

可用区

可用区1（可用区2、可用区3）

按可用区选择规格名称，按可用区选择规格名称，按可用区选择规格名称，按可用区选择规格名称。

参数	示例	参数说明
计费模式	按需计费	实例的计费模式。 - 包年/包月：一种预付费模式，即先付费再使用，按照订单的购买周期进行结算。购买周期越长，享受的折扣越大。一般适用于需求量长期稳定的成熟业务。 - 按需计费：一种后付费模式，即先使用再付费，按照云数据库GeminiDB实际使用时长计费，秒级计费，按小时结算。按需计费模式允许您根据实际业务需求灵活地调整资源使用，无需提前预置资源，从而降低预置过多或不足的风险。

参数	示例	参数说明
区域	华北-北京四	租户所在的区域。 说明 请就近选择靠近您业务的区域，可减少网络时延，提高访问速度。不同区域内的产品内网不互通，且购买后不能更换，请谨慎选择。
实例名称	自定义	设置实例名称时，需要满足如下规则。 - 实例名称允许和已有名称重复。 - 实例名称长度在4个到64个字符之间，必须以字母或中文字开头，区分大小写，可以包含字母、数字、中划线、下划线或中文（一个中文字符占用3个字节），不能包含其他特殊字符。如果名称包含中文，则不超过64字节。
兼容接口	Redis	云数据库 GeminiDB目前兼容Redis、DynamoDB、Cassandra、HBase、InfluxDB和MongoDB主流NoSQL接口，您当前选择的是Redis接口，其他接口的选择具体参见 如何选择接口 。
部署模式	经典	- 经典：GeminiDB经典的存算分离架构。 - 云原生：新一代存算分离架构，更灵活，支持更多的可用区。
产品类型	标准型	- 标准型：提供稳定的低延迟性能，适用于广告&推荐、游戏、电商、车联网等各类常见业务场景。 - 容量型：提供大容量Key-Value存储，适用于对性能要求不高且有降低成本诉求的业务。 说明 云原生部署模式下产品类型暂不支持标准型。
实例架构	Proxy集群版	- Proxy集群版： 采用分片集群架构，提供Proxy代理连接，同时兼容单机、哨兵、Cluster接入方式。性能水平扩展能力强，可支撑百万QPS、数十TB级业务量。 - Cluster集群版： 采用原生Cluster分片集群架构，完全兼容Redis Cluster直连访问接入，时延更低，性能更优。
兼容版本	7.0	7.0、6.2(包含6.2.x)，5.0及以下版本。

参数	示例	参数说明
CPU类型	x86	x86类型的CPU架构采用复杂指令集（CISC），CISC指令集的每个小指令可以执行一些较低阶的硬件操作，指令数目多而且复杂，每条指令的长度并不相同。由于指令执行较为复杂所以每条指令花费的时间较长。
可用区	可用区1，可用区2，可用区3	指在同一区域下，电力、网络隔离的物理区域，可用区之间内网互通，不同可用区之间物理隔离。

图 3-3 规格与存储

规格选择模式

快速选择

自定义资源

实例规格	实例规格	实例数量	CPU规格	最大内存	实例规格	实例规格	实例规格
☐ 4 GB	规格型 1 vCPU	2	16,000	20,000	1,000	200	
☒ 8 GB	规格型 1 vCPU	2	20,000	20,000	1,000	200	
☐ 16 GB	规格型 2 vCPU	2	40,000	20,000	1,000	200	
☐ 32 GB	规格型 2 vCPU	3	60,000	30,000	1,000	200	
☐ 64 GB	规格型 2 vCPU	4	80,000	40,000	1,000	200	
☐ 128 GB	规格型 4 vCPU	3	120,000	50,000	1,000	200	
☐ 256 GB	规格型 4 vCPU	4	160,000	60,000	1,000	200	

当前规格 4 GB | 实例规格 规格型 1 vCPU | 实例数量 2 | CPU规格 16,000 | 最大内存 20,000 | 实例规格 1,000 | 实例规格 200

参数	示例	参数说明
规格选择模式	快速选择	集群版实例支持快速选择和自定义资源两种规格选择模式。 快速选择 快速选择模式为您提供了常用的实例推荐配置，您可以根据业务需要选择合适的配置，无需自定义节点规格、节点数量和存储容量。 自定义资源 您可以根据业务需要自定义实例的节点规格、节点数量和存储容量。 目前节点数上限为36个，如需更多，您可以在管理控制台右上角，选择“工单 > 新建工单”联系客服进行咨询。
实例规格	2U8GB	CPU规格越高，性能越好。请根据实际业务场景，选取符合业务类型的性能规格。 实例的CPU和内存，请参见实例规格说明。

图 3-4 网络和数据库配置

虚拟私有云

vpc-19-0

查看虚拟私有云

目前GeminiDB Redis实例创建完成后不支持切换虚拟私有云VPC，请谨慎选择所属虚拟私有云。如需与ECS进行内网通信，建议将GeminiDB Redis实例创建在同一个虚拟私有云下，或者不同虚拟私有云之间配置对等连接。如需创建新的虚拟私有云，可前往VPC控制台创建。

子网

subnet-

查看子网

创建子网时最多支持17个VPC，当前选择的子网下可创建实例数量17个。

内网安全组

default

查看安全组

您可为实例指定安全组，创建实例后请配置实例安全组。

数据库端口

6379

查看端口规则

如该实例计划用于搭建高可用数据库，请配置为6379端口。

访问控制

前往设置

创建访问控制

设置密码

前往设置

创建访问控制

密码

确认密码

高级配置

前往设置

创建访问控制

企业项目

选择企业项目

查看企业项目

购买数量

-

1

+

参数	示例	参数说明
虚拟私有云	default_vpc	实例所在的虚拟专用网络，可以对不同业务进行网络隔离。您可以根据需要创建或选择所需的虚拟私有云。 说明 - 目前GeminiDB Redis实例创建完成后不支持切换虚拟私有云VPC，请谨慎选择所属虚拟私有云。 - 如需与ECS进行内网通信，GeminiDB Redis实例与需要通信的ECS需要处于同一个虚拟私有云下，或者不同虚拟私有云之间配置对等连接。
子网	default_subnet	通过子网提供与其他网络隔离的、可以独享的网络资源，以提高网络安全性。
数据库端口	6379	数据库的访问端口号，当您不指定端口时，创建实例的访问端口默认为6379。GeminiDB Redis支持自定义端口号，取值范围为：1024~65535，其中，如下端口号为禁用端口：2180、2887、3887、6377、6378、6380、8018、8079、8091、8479、8484、8999、12017、12333、50069。

参数	示例	参数说明
访问控制	创建后设置	• 创建后设置：若不设置则默认根据VPC的访问策略进行限制。 • 现在设置： 可以选择允许所有IP访问、白名单和黑名单。 允许所有IP访问：不进行访问控制，允许所有IP访问此实例。 白名单：仅允许IP地址组中的IP访问此实例。 黑名单：不允许IP地址组中的IP访问此实例。 说明 Cluster集群版暂不支持访问控制。
设置密码	创建后设置	• 创建后设置：支持创建实例后设置数据库密码。 • 现在设置：创建实例时设置数据库密码。 说明 Cluster集群版暂不支持创建后设置密码。
密码	根据密码策略配置	如果“设置密码”选择“现在设置”，用户需要设置数据库密码。 • 长度为8~32个字符。 • 必须是大写字母、小写字母、数字、特殊字符中至少两种的组合，其中可输入~!@#%^*_-=+?特殊字符。 • 系统会进行弱密码校验，安全起见，请输入高强度密码。 请妥善保管您的密码，因为系统将无法获取您的密码信息。
免密访问配置	创建后设置	配置免密的网段，连接实例时，可以免输入实例密码。 具体参见 表4-5 。 说明 Cluster集群版暂不支持免密访问配置。

其他参数选择默认配置。

6. 单击“返回实例列表”按钮，返回实例列表页。

7. 在“实例管理”页面，您可以查看并管理实例。

- 实例创建过程中，运行状态显示为“创建中”，此过程约5~9分钟。
- 实例创建完成后，运行状态显示为“正常”。

图 3-5 购买成功

步骤 2: 通过 DAS 连接实例

数据管理服务（Data Admin Service，简称DAS）是一款专业的简化数据库管理工具，提供优质的可视化操作界面，大幅提高工作效率，让数据管理变得既安全又简单。您可以通过数据管理服务连接并管理实例。GeminiDB Redis默认为您开通了远程主机登录权限，推荐您使用更安全便捷的数据管理服务连接实例。

1. 登录[云数据库GeminiDB控制台](#)。
2. 在实例列表中选择目标实例，单击操作列“连接Redis”。

图 3-6 连接 Redis

您也可以单击实例名称，进入“基本信息”页面，在页面右上角单击“连接Redis”。

图 3-7 连接 Redis

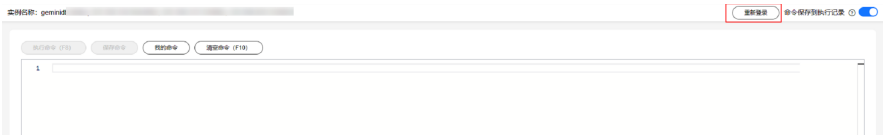
3. 输入实例的密码登录实例。

图 3-8 登录 Redis 实例



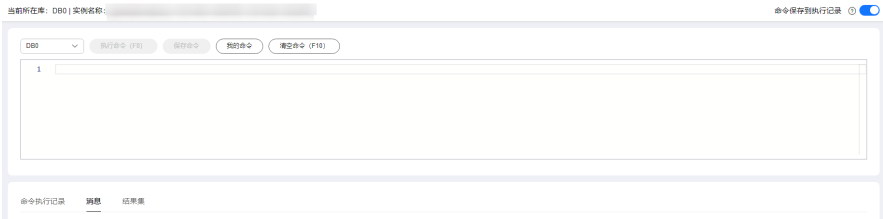
如果重置密码后，需要重新登录，请单击右上角的“重新登录”按钮，使用新密码进行登录。

图 3-9 重新登录



4. 根据业务需要进行相关数据库管理操作。

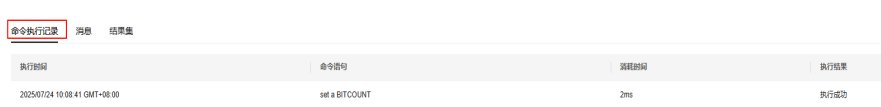
图 3-10 实例首页



- 命令保存到执行记录

默认开启命令保存到执行记录的功能，主要用于保存最近执行的命令到服务器中，方便您查看最近执行过的操作记录。
该功能开启后，您可以在页面下方的命令执行记录中查看操作记录。

图 3-11 查看命令执行记录



关闭该功能后，页面将不再展示后续的命令执行记录。您可以单击右上角“命令保存到执行记录”后的  关闭此功能。

- 执行命令
- 您可以在命令执行框中输入需要执行的命令，单击“执行命令”或者通过快捷键“F8”执行。

 说明

- 不建议使用事务，Lua脚本，不允许使用Pub/Sub和其他具有阻塞语义的命令。
- 支持多DB的版本可以在前端切换当前库，但不允许使用SELECT切换。

图 3-12 执行命令



命令执行成功后，您可以在页面下方结果集中查看执行结果。

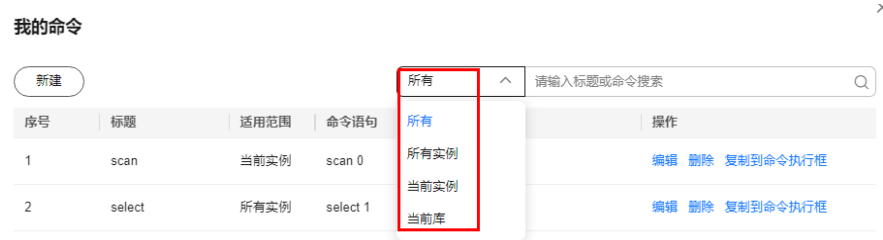
- 保存命令
- 您可以将命令保存于所有实例、当前实例、当前库。保存成功后可以在“我的命令”中查看详情。

图 3-13 保存命令



- 我的命令
- 我的命令主要展示的是一些常见操作命令。
- 您可以根据命令的适用范围筛选命令，当选择“所有”时可以查询到当前账号下所保存的所有命令，其他类型可根据相应的适用范围进行查询。

图 3-14 筛选命令



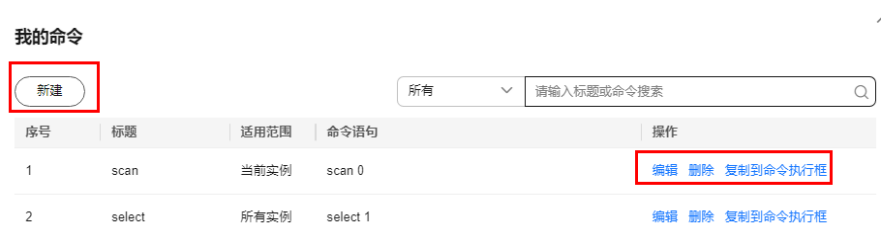
您也可以通过搜索框按照命令标题和命令语句进行查询。

图 3-15 搜索命令



另外，您还可以在“我的命令”页签中新建命令、编辑命令、删除命令、复制到命令执行框。

图 3-16 命令管理



- 清空命令
清除当前命令执行框中的命令，您也可以通过快捷键“F10”操作。

图 3-17 清空命令



常见问题

问题：在实例列表的操作列单击“连接Redis”时或者在“基本信息”页面单击“连接Redis”时，无法跳转至DAS页面怎么办？
解决方法：请将您的浏览器设置为允许弹出窗口，然后重试。

3.3 购买并连接主备版实例

本章节主要介绍了如何在云数据库 GeminiDB管理控制台购买Redis数据库的主备版实例并通过控制台连接实例。
主备版实例是标准的master-replica架构，兼容单机、哨兵接入方式。适用于无法使用hashtag的业务场景。

每个租户下GeminiDB Redis实例的默认总配额为50。如需申请扩大配额，您可以在管理控制台右上角，选择“[工单 > 新建工单](#)”，联系客服申请开通。

- [步骤1：购买实例](#)
- [步骤2：通过DAS连接实例](#)

如果需要其他方式连接实例请参考[其他连接实例方式](#)。

步骤 1：购买实例

1. 登录[云数据库GeminiDB控制台](#)。
2. 在“实例管理”页面，单击“购买数据库实例”，进入“服务选型”页面。
3. 在“服务选型”页面，选择计费模式，填写并选择实例相关信息后，单击“立即购买”。

下面的配置项仅作为示例，实际购买时根据用户需求选择合适的规格参数，具体参数详见[参数详解](#)。

图 3-18 计费模式和基本信息

计费模式: 包年包月 (selected) | 按量计费

区域: [Region Selection]

实例名称: geminidb [icon]

兼容接口: Redis (selected) | Cassandra | DynamoDB | HBase | InfluxDB

部署模式: 经典 (selected) | 云原生

产品类型: 标准型 (selected) | 容量型

实例架构: Proxy集群版 | Cluster集群版 | 主备版 (selected)

兼容版本: 7.0 (selected)

CPU类型: x86 (selected)

可用区: 可用区1 | 可用区2 | 可用区3 (selected)

主可用区: 可用区1 (selected) | 可用区2 | 可用区3

备可用区: 可用区1 | 可用区2 (selected) | 可用区3

参数	示例	参数说明
计费模式	按需计费	实例的计费模式。 - 包年/包月：一种预付费模式，即先付费再使用，按照订单的购买周期进行结算。购买周期越长，享受的折扣越大。一般适用于需求量长期稳定的成熟业务。 - 按需计费：一种后付费模式，即先使用再付费，按照云数据库GeminiDB实际使用时长计费，秒级计费，按小时结算。按需计费模式允许您根据实际业务需求灵活地调整资源使用，无需提前预置资源，从而降低预置过多或不足的风险。
区域	华北-北京四	租户所在的区域。 说明 请就近选择靠近您业务的区域，可减少网络时延，提高访问速度。不同区域内的产品内网不互通，且购买后不能更换，请谨慎选择。
实例名称	自定义	设置实例名称时，需要满足如下规则。 - 实例名称允许和已有名称重复。 - 实例名称长度在4个到64个字符之间，必须以字母或中文字开头，区分大小写，可以包含字母、数字、中划线、下划线或中文（一个中文字符占用3个字节），不能包含其他特殊字符。如果名称包含中文，则不超过64字节
兼容接口	Redis	云数据库 GeminiDB目前兼容Redis、DynamoDB、Cassandra、HBase、InfluxDB和MongoDB主流NoSQL接口，您当前选择的是Redis接口，其他接口的选择具体参见 如何选择接口 。
部署模式	经典	- 经典：GeminiDB经典的存算分离架构。 - 云原生：新一代存算分离架构，更灵活，支持更多的可用区。
产品类型	标准型	- 标准型：提供稳定的低延迟性能，适用于广告&推荐、游戏、电商、车联网等各类常见业务场景。 - 容量型：提供大容量Key-Value存储，适用于对性能要求不高且有降低成本诉求的业务。
实例架构	主备版	主备版 标准的master-replica架构，兼容单机、哨兵接入方式。适用于无法使用hashtag的业务场景。
兼容版本	7.0	7.0、6.2(包含6.2.x)，5.0及以下版本。

参数	示例	参数说明
CPU类型	x86	x86类型的CPU架构采用复杂指令集（CISC），CISC指令集的每个小指令可以执行一些较低阶的硬件操作，指令数目多而且复杂，每条指令的长度并不相同。由于指令执行较为复杂所以每条指令花费的时间较长。
可用区	可用区1，可用区2，可用区3	指在同一区域下，电力、网络隔离的物理区域，可用区之间内网互通，不同可用区之间物理隔离。多可用区时需要选择主可用区和备可用区。 目前支持将实例部署在单可用区或3可用区。 ● 如果业务要求实例之间的网络延时较低，则建议您选择单可用区，将实例的组件部署在同一个可用区内。 ● 如果业务需要较高的容灾能力，请选择3可用区，此时实例下的节点根据您的需要选择主可用区和备可用区。 - 主可用区：主节点所在的可用区。 - 备可用区：备节点所在的可用区。

图 3-19 规格与存储

规格名称	节点数	CPU规格	最大内存	存储容量	AQL引擎规格
4 GB	1	8,000	10,000	1,000	200
8 GB	1	8,000	10,000	1,000	200
16 GB	1	16,000	10,000	1,000	200
24 GB	1	20,000	10,000	1,000	200
32 GB	1	20,000	10,000	1,000	200
48 GB	1	40,000	20,000	1,000	200
64 GB	1	40,000	20,000	1,000	200

参数	示例	参数说明
规格选择模式	快速选择	实例支持快速选择和自定义资源两种规格选择模式。 ● 快速选择 快速选择模式为您提供了常用的实例推荐配置，您可以根据业务需要选择合适的配置，无需自定义节点规格、节点数量和存储容量。 ● 自定义资源 您可以根据业务需要自定义实例的节点规格、节点数量和存储容量。

参数	示例	参数说明
实例规格	2U8GB	CPU规格越高，性能越好。请根据实际业务场景，选取符合业务类型的性能规格。 实例的CPU和内存，请参见 实例规格说明 。

图 3-20 网络和数据库配置

虚拟私有云

vpv-19-9

查看虚拟私有云

目前GeminiDB Redis实例创建完成后不支持切换虚拟私有云VPC，请谨慎选择所属虚拟私有云。如需与ECS进行内网通信，建议将GeminiDB Redis实例与需要通信的ECS放在同一个虚拟私有云下，或者不同虚拟私有云之间配置对等连接。如需创建新的虚拟私有云，可前往VPC控制台创建。

子网

subnet-

查看子网

创建实例需要选择VPC，当前所选的子网下可用私有IP数量17个。

内网安全组

default

查看安全组

实例所属安全组不支持安全组。创建实例后应配置其安全组。

数据库端口

6379

查看端口规则

创建实例时指定用于数据库的端口，请配置为6379端口。

访问控制

现在设置

创建访问控制

设置密码

现在设置

创建时设置

密码

确认密码

高级选项配置

现在设置

创建时设置

企业项目

--请选择--

查看项目列表

购买数量

-

1

+

参数	示例	参数说明
虚拟私有云	default_vpc	实例所在的虚拟专用网络，可以对不同业务进行网络隔离。您可以根据需要创建或选择所需的虚拟私有云。 说明 - 目前GeminiDB Redis实例创建完成后不支持切换虚拟私有云VPC，请谨慎选择所属虚拟私有云。 - 如需与ECS进行内网通信，GeminiDB Redis实例与需要通信的ECS需要处于同一个虚拟私有云下，或者不同虚拟私有云之间配置对等连接。
子网	default_subnet	通过子网提供与其他网络隔离的、可以独享的网络资源，以提高网络安全性。

参数	示例	参数说明
数据库端口	6379	数据库的访问端口号，当您不指定端口时，创建实例的访问端口默认为6379。GeminiDB Redis支持自定义端口号，取值范围为：1024~65535，其中，如下端口号为禁用端口：2180、2887、3887、6377、6378、6380、8018、8079、8091、8479、8484、8999、12017、12333、50069。
访问控制	创建后设置	- 创建后设置：若不设置则默认根据VPC的访问策略进行限制。 - 现在设置： 可以选择允许所有IP访问、白名单和黑名单。 允许所有IP访问：不进行访问控制，允许所有IP访问此实例。 白名单：仅允许IP地址组中的IP访问此实例。 黑名单：不允许IP地址组中的IP访问此实例。
设置密码	创建后设置	- 创建后设置：支持创建实例后设置数据库密码。 - 现在设置：创建实例时设置数据库密码。
密码	根据密码策略配置	如果“设置密码”选择“现在设置”，用户需要设置数据库密码。 - 长度为8~32个字符。 - 必须是大写字母、小写字母、数字、特殊字符中至少两种的组合，其中可输入~!@#%^*_-=+?特殊字符。 - 系统会进行弱密码校验，安全起见，请输入高强度密码。 请妥善保管您的密码，因为系统将无法获取您的密码信息。
免密访问配置	创建后设置	配置免密的网段，连接实例时，可以免输入实例密码。 具体参见表4-5。

参数	示例	参数说明
企业项目	default	该参数针对企业用户使用。 企业项目是一种云资源管理方式，企业项目管理服务提供统一的云资源按项目管理，以及项目内的资源管理、成员管理，默认项目为default。 请在下拉框中选择所在的企业项目。更多关于企业项目的信息，请参见《企业管理用户指南》。

其他参数选择默认配置。

5. 在“订单详情确认”页面，核对实例信息，如果需要修改，单击“上一步”，修改实例信息。核对无误后，勾选协议，单击“提交”，开始创建实例。
6. 单击“返回实例列表”按钮，返回实例列表页。
7. 在“实例管理”页面，您可以查看并管理实例。
 - 实例创建过程中，运行状态显示为“创建中”，此过程约5~9分钟。
 - 实例创建完成后，运行状态显示为“正常”。

图 3-21 购买成功

实例名称ID	实例类型	部署模式	部署模式	运行状态	所属租户ID	企业项目	计费模式	操作
gemini-7089	主实例	Redis 7.0	独享	正常	IPV4	default	按量计费	2025/07/04 10:23:55 GMT+08:00 连接Redis 修改配置 更多

步骤 2：通过 DAS 连接实例

数据管理服务（Data Admin Service，简称DAS）是一款专业的简化数据库管理工具，提供优质的可视化操作界面，大幅提高工作效率，让数据管理变得既安全又简单。您可以通过数据管理服务连接并管理实例。GeminiDB Redis默认为您开通了远程主机登录权限，推荐您使用更安全便捷的数据管理服务连接实例。

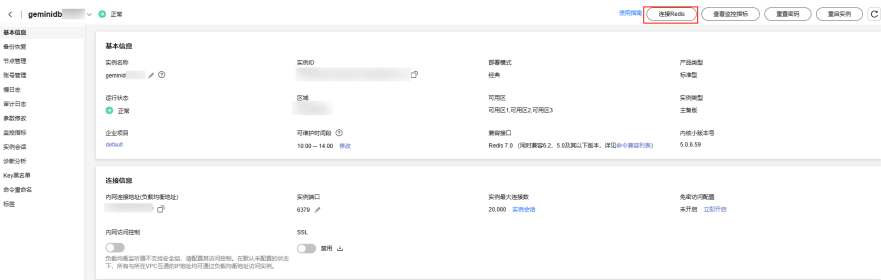
1. 登录[云数据库GeminiDB控制台](#)。
2. 在实例列表中选择目标实例，单击操作列“连接Redis”。

图 3-22 连接 Redis

实例名称ID	实例类型	部署模式	部署模式	运行状态	所属租户ID	企业项目	计费模式	操作
gemini-7089	主实例	Redis 7.0	独享	正常	IPV4 192.168.8.42	default	按量计费	2025/07/04 10:23:55 GMT+08:00 连接Redis 修改配置 更多

您也可以单击实例名称，进入“基本信息”页面，在页面右上角单击“连接Redis”。

图 3-23 连接 Redis



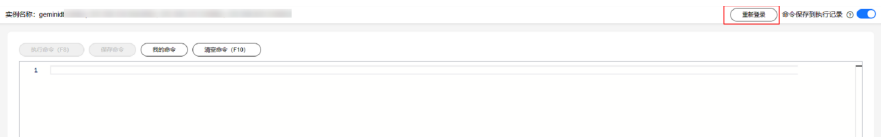
3. 输入实例的密码登录实例。

图 3-24 登录 Redis 实例



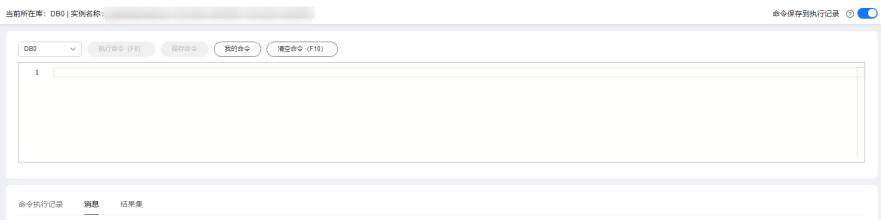
如果重置密码后，需要重新登录，请单击右上角的“重新登录”按钮，使用新密码进行登录。

图 3-25 重新登录



4. 根据业务需要进行相关数据库管理操作。

图 3-26 实例首页



命令保存到执行记录

默认开启命令保存到执行记录的功能，主要用于保存最近执行的命令到服务器中，方便您查看最近执行过的操作记录。
该功能开启后，您可以在页面下方的命令执行记录中查看操作记录。

图 3-27 查看命令执行记录

命令执行记录

消息

结果集

执行时间	命令语句	消耗时间	执行结果
2025/07/24 10:38:18 GMT+08:00	set a BITCOUNT	3ms	执行成功

关闭该功能后，页面将不再展示后续的命令执行记录。您可以单击右上角“命令保存到执行记录”后的  关闭此功能。

- 执行命令

您可以在命令执行框中输入需要执行的命令，单击“执行命令”或者通过快捷键“F8”执行。

 说明

- 不建议使用事务，Lua脚本，不允许使用Pub/Sub和其他具有阻塞语义的命令。
- 支持多DB的版本可以在前端切换当前库，但不允许使用SELECT切换。

图 3-28 执行命令

DB0

执行命令 (F8)

保存命令

我的命令

清空命令 (F10)

1

SCAN 0

命令执行记录

消息

结果集

命令语句	游标	执行结果
SCAN 0	0	(empty list or set)

命令执行成功后，您可以在页面下方结果集中查看执行结果。

- 保存命令

您可以将命令保存于所有实例、当前实例、当前库。保存成功后可以在“我的命令”中查看详情。

图 3-29 保存命令

DB0

执行命令 (F8)

保存命令

我的命令

1

SCAN 0

保存命令

标题

适用范围

所有实例

当前实例

当前库

确定

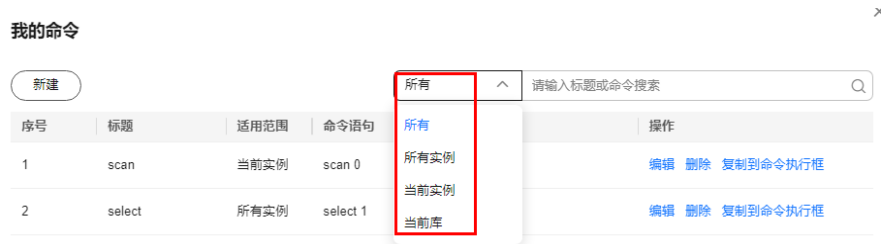
取消

- 我的命令

我的命令主要展示的是一些常见操作命令。

您可以根据命令的适用范围筛选命令，当选择“所有”时可以查询到当前账号下所保存的所有命令，其他类型可根据相应的适用范围进行查询。

图 3-30 筛选命令



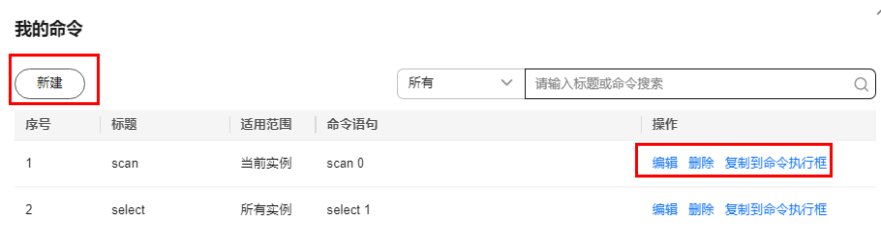
您也可以通过搜索框按照命令标题和命令语句进行查询。

图 3-31 搜索命令



另外，您还可以在“我的命令”页签中新建命令、编辑命令、删除命令、复制到命令执行框。

图 3-32 命令管理



- 清空命令
清除当前命令执行框中的命令，您也可以通过快捷键“F10”操作。

图 3-33 清空命令



常见问题

问题：在实例列表的操作列单击“连接Redis”时或者在“基本信息”页面单击“连接Redis”时，无法跳转至DAS页面怎么办？

解决方法：请将您的浏览器设置为允许弹出窗口，然后重试。

3.4 入门实践

当您购买并连接了GeminiDB Redis数据库实例后，可以根据自身的业务需求使用GeminiDB Redis提供的一系列常用实践。

表 3-4 常用最佳实践

实践		描述
数据迁移	阿里云数据库Redis/Tair到GeminiDB Redis的迁移	本实践主要介绍阿里云数据库Redis/Tair到GeminiDB Redis的迁移方案。使用阿里云的数据迁移工具DTS中的数据迁移功能。
	腾讯云Redis到GeminiDB Redis的迁移	本实践主要介绍腾讯云Redis到GeminiDB Redis的迁移方案。使用腾讯云的数据迁移工具DTS中的数据同步功能。
	通过Redis-Shake迁移工具将自建Redis迁移到GeminiDB Redis	本实践主要介绍自建Redis到GeminiDB Redis的迁移方案。使用Redis-Shake迁移工具进行源端Redis到目标端GeminiDB Redis的迁移。
	使用数据导入功能将RDB文件恢复到GeminiDB Redis(推荐)	本实践主要介绍RDB到GeminiDB Redis的迁移方案。使用Redis-Shake迁移工具进行源端RDB到目标端GeminiDB Redis的迁移。迁移过程支持全量+增量迁移。
	Kvrocks到GeminiDB Redis的迁移	本实践主要介绍Kvrocks到GeminiDB Redis的迁移方案。使用开源工具kvrocks2redis进行Kvrocks到GeminiDB Redis的迁移。
	Pika到GeminiDB Redis的迁移	本实践主要介绍利用pika-port工具实现Pika到GeminiDB Redis的迁移方案。
	SSDB到GeminiDB Redis的迁移	本实践主要介绍利用ssdb-port工具实现SSDB到GeminiDB Redis的迁移方案。
	LevelDB到GeminiDB Redis的迁移	本实践主要介绍利用自研迁移工具leveldb-port实现LevelDB到GeminiDB Redis的迁移方案。

实践		描述
	RocksDB到GeminiDB Redis的迁移	本实践主要介绍利用自研迁移工具 rocksdb-port实现RocksDB到 GeminiDB Redis的迁移方案。
	AWS ElasticCache for Redis数据库到GeminiDB Redis的迁移	本实践主要介绍AWS ElasticCache for Redis到GeminiDB Redis的迁移方案。AWS ElasticCache for Redis通过备份导出RDB文件后，通过Redis-Shake迁移工具的数据恢复（restore）将数据迁移到GeminiDB Redis。
数据备份	管理自动备份	介绍如何在数据库实例的备份时段中创建数据库实例的自动备份，系统根据您的指定的备份保留期保存数据库实例的自动备份。
	管理手动备份	介绍如何为数据库实例手动备份数据，用户可以通过手动备份恢复数据，从而保证数据可靠性。
数据恢复	恢复备份到新实例	介绍如何使用已有的自动备份或手动备份恢复实例数据到新建实例，恢复后的数据与该备份生成时的实例数据一致。
日志管理	查看和导出慢日志	GeminiDB Redis支持查看数据库级别的慢日志，执行时间的单位为ms。通过该日志，可查找出执行效率低的语句，以便优化。

4 用户指南

4.1 通过 IAM 授予使用 GeminiDB Redis 的权限

4.1.1 创建用户并授权使用 GeminiDB Redis

如果您需要对您所拥有的云数据库 GeminiDB进行精细的权限管理，您可以使用[统一身份认证服务](#)（Identity and Access Management，简称IAM），通过IAM，您可以：

- 根据企业的业务组织，在您的华为账号中，给企业中不同职能部门的员工创建IAM用户，让员工拥有唯一安全凭证，并使用云数据库 GeminiDB资源。
- 根据企业用户的职能，设置不同的访问权限，以达到用户之间的权限隔离。
- 将云数据库 GeminiDB资源委托给更专业、高效的其他华为账号或者云服务，这些账号或者云服务可以根据权限进行代运维。

如果华为账号已经能满足您的要求，不需要创建独立的IAM用户，您可以跳过本章节，不影响您使用云数据库 GeminiDB服务的其它功能。

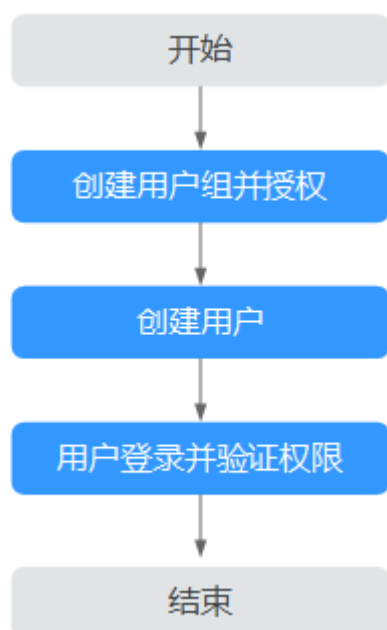
本章节为您介绍对用户授权的方法，操作流程如[图4-1](#)所示。

前提条件

给用户组授权之前，请您了解用户组可以添加的云数据库 GeminiDB权限，并结合实际需求进行选择，云数据库 GeminiDB支持的系统权限，请参见：[云数据库 GeminiDB系统权限](#)。若您需要对除云数据库 GeminiDB之外的其它服务授权，IAM支持服务的所有权限请参见[权限策略](#)。

示例流程

图 4-1 给用户授权云数据库 GeminiDB 权限流程



1. 创建用户组并授权

在IAM控制台创建用户组，并授予云数据库 GeminiDB服务“GeminiDB FullAccess”权限。

📖 说明

如果需要使用到对接其他服务的一些功能时，除了需要配置“GeminiDB FullAccess”权限外，还需要配置对应服务的权限。

例如：使用控制台连接实例时，除了需要配置“GeminiDB FullAccess”权限外，您还需要配置数据管理服务“DAS FullAccess”权限后，才可正常使用控制台登录数据库实例。

2. 创建用户并加入用户组

在IAM控制台创建用户，并将其加入1中创建的用户组。

3. 用户登录并验证权限

新创建的用户参考[IAM用户登录华为云](#)登录，切换至授权区域，验证权限：

在“服务列表”中选择云数据库 GeminiDB服务，进入云数据库 GeminiDB主界面，单击右上角“购买数据库实例”，尝试购买云数据库 GeminiDB实例，若可以正常购买数据库实例，则表示所需权限策略均已生效。

4.1.2 GeminiDB Redis 自定义策略

如果系统预置的云数据库 GeminiDB权限，不满足您的授权要求，可以创建自定义策略。自定义策略中可以添加的授权项（Action）请参见[云数据库GeminiDB服务授权项说明](#)。

目前华为云支持以下两种方式创建自定义策略：

- 可视化视图创建自定义策略：无需了解策略语法，按可视化视图导航栏选择云服务、操作、资源、条件等策略内容，可自动生成策略。

- JSON视图创建自定义策略：可以在选择策略模板后，根据具体需求编辑策略内容；也可以直接在编辑框内编写JSON格式的策略内容。

具体创建步骤请参见：[创建自定义策略](#)。本章为您介绍常用的云数据库 GeminiDB自定义策略样例。

自定义策略样例

- 示例1：授权用户创建云数据库 GeminiDB实例

```
{
  "Version": "1.1",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "nosql:instance:create"
      ]
    }
  ]
}
```

- 示例2：拒绝用户删除云数据库 GeminiDB数据库实例

拒绝策略需要同时配合其他策略使用，否则没有实际作用。用户被授予的策略中，一个授权项的作用如果同时存在Allow和Deny，则遵循**Deny优先原则**。

如果您给用户授予GeminiDB FullAccess的系统策略，但不希望用户拥有GeminiDB FullAccess中定义的删除云数据库 GeminiDB实例权限，您可以创建一条拒绝删除云数据库 GeminiDB实例的自定义策略，然后同时将GeminiDB FullAccess和拒绝策略授予用户，根据Deny优先原则，则用户可以对云数据库 GeminiDB执行除了删除云数据库 GeminiDB实例外的所有操作。拒绝策略示例如下：

```
{
  "Version": "1.1",
  "Statement": [
    {
      "Effect": "Deny",
      "Action": [
        "nosql:instance:delete"
      ],
    }
  ]
}
```

- 示例3：多个授权项策略

一个自定义策略中可以包含多个授权项，且除了可以包含本服务的授权项外，还可以包含其他服务的授权项，可以包含的其他服务必须跟本服务同属性，即都是项目级服务或都是全局级服务。多个授权语句策略描述如下：

```
{
  "Version": "1.1",
  "Statement": [
    {
      "Action": [
        "nosql:instance:create",
        "nosql:instance:rename",
        "nosql:instance:delete",
        "vpc:publicips:list",
        "vpc:publicips:update"
      ],
      "Effect": "Allow"
    }
  ]
}
```

```
}  
}
```

4.2 购买 GeminiDB Redis 实例

4.2.1 购买 GeminiDB Redis 集群版实例

本章节主要介绍了如何在云数据库 GeminiDB 管理控制台购买集群版的 Redis 数据库实例。

- Proxy 集群版实例采用分片集群架构，提供 Proxy 代理连接，同时兼容单机、哨兵、Cluster 接入方式。性能水平扩展能力强，可支撑百万 QPS、数十 TB 级业务量。
- Cluster 集群版采用原生 Cluster 分片集群架构，完全兼容 Redis Cluster 直连访问接入，时延更低，性能更优。

每个租户下 GeminiDB Redis 实例的默认总配额为 50。如需申请扩大配额，您可以在管理控制台右上角，选择“[工单 > 新建工单](#)”，联系客服申请开通。

前提条件

- [注册华为账号并开通华为云](#)。
- 账户余额大于或等于 0 元。

操作步骤

步骤1 登录[云数据库 GeminiDB 控制台](#)。

步骤2 在“实例管理”页面，单击“购买数据库实例”，进入“服务选型”页面。

步骤3 在“服务选型”页面，选择计费模式，填写并选择实例相关信息后，单击“立即购买”。

图 4-2 计费模式和基本信息（Proxy 集群版）

The screenshot displays the 'Purchase Instance' (购买实例) page for GeminiDB Redis Proxy Cluster Edition. The page is organized into several sections with configuration options:

- 计费模式 (Billing Mode):** Includes '包年包月' (Subscription) and '按量付费' (Pay-as-you-go) buttons.
- 区域 (Region):** A dropdown menu for selecting the region, with a note: '不同区域的数据之间互不互通，请谨慎选择业务发生的区域，可以降低网络时延，提高访问速度。' (Data across different regions is not互通, please carefully select the business area to reduce network latency and improve access speed.)
- 实例名称 (Instance Name):** A text input field with a placeholder 'gemini-...' and a note: '购买多个数据库实例时，名称需按命名规范命名，即名称和两位数字组成。例如：gemini-0101-1234567890101-01 开始命名。' (When purchasing multiple database instances, the name must follow the naming convention, consisting of the name and two digits. For example: gemini-0101-1234567890101-01.)
- 实例端口 (Instance Port):** A dropdown menu with options: 'Proxy', 'Cassandra', 'DynamoDB', 'HBase', and 'InfluxDB'. 'Proxy' is selected.
- 部署模式 (Deployment Mode):** Includes '主备' (Master-Slave) and '云原生' (Cloud Native) buttons.
- 产品规格 (Product Specification):** Includes '标准版' (Standard Edition) and '企业版' (Enterprise Edition) buttons.
- 实例架构 (Instance Architecture):** Includes 'Proxy 集群版' (Proxy Cluster Edition), 'Cluster 集群版' (Cluster Cluster Edition), and '主备版' (Master-Slave Edition) buttons.
- 实例版本 (Instance Version):** Includes '6.0' and '7.0' buttons.
- CPU 规格 (CPU Specification):** Includes 'x86' and 'arm' buttons.
- 可用区 (Availability Zone):** Includes '可用区1 (高可用)' (Availability Zone 1 (High Availability)) and '可用区2 (高可用)' (Availability Zone 2 (High Availability)) buttons.

图 4-3 计费模式和基本信息（Cluster 集群版）

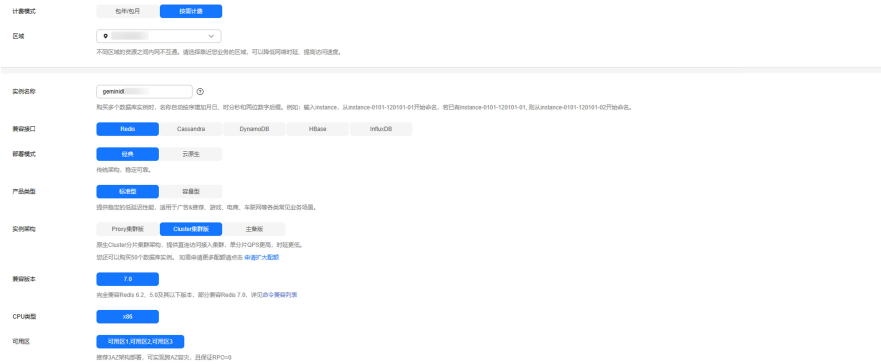


表 4-1 计费模式

参数	描述
计费模式	选择“包年包月”或“按需计费”。 - 包年包月 - 用户选购完服务配置后，可以根据需要设置购买时长，系统会一次性按照购买价格对账户余额进行扣费。 - 创建成功后，如果包年/包月实例到期后不再长期使用资源，可将“包年/包月”实例转为“按需计费”，到期后将转为按需计费实例。具体请参见包年/包月转按需。 - 按需计费 - 用户选购完服务配置后，无需设置购买时长，系统会根据消费时长对账户余额进行扣费。 - 创建成功后，如果需要长期使用资源，可将“按需计费”实例转为“包年/包月”，继续使用这些资源的同时，享受包年/包月的低资费。具体请参见按需转包年/包月。 说明 “包年/包月”方式购买的实例就是包周期实例，“包年/包月”方式购买的实例不能直接删除，仅支持资源退订操作，如何退订资源请参见如何退订包年/包月实例。

表 4-2 基本信息

参数	描述
区域	租户所在的区域。 说明 请就近选择靠近您业务的区域，可减少网络时延，提高访问速度。不同区域内的产品内网不互通，且购买后不能更换，请谨慎选择。

参数	描述
实例名称	设置实例名称时，需要满足如下规则。 - 实例名称允许和已有名称重复。 - 实例名称长度在4个到64个字符之间，必须以字母或中文字开头，区分大小写，可以包含字母、数字、中划线、下划线或中文（一个中文字符占用3个字节），不能包含其他特殊字符。如果名称包含中文，则不超过64字节。 创建成功后，可修改实例名称，请参见修改GeminiDB Redis实例名称。
兼容接口	Redis 云数据库 GeminiDB目前兼容Redis、DynamoDB、Cassandra、HBase、InfluxDB和MongoDB主流NoSQL接口，您当前选择的是Redis接口，其他接口的选择具体参见如何选择接口。
部署模式	- 经典：GeminiDB经典的存算分离架构。 - 云原生：新一代存算分离架构，更灵活，支持更多的可用区。 说明 - 云原生部署模式只支持Proxy集群版实例类型。 - 经典和云原生只是部署模式有差异，对用户使用无差异；云原生支持的可用区更多，对于同时支持经典和云原生的，用户选择任意部署模式即可。
产品类型	- 标准型：提供稳定的低延迟性能，适用于广告&推荐、游戏、电商、车联网等各类常见业务场景。 - 容量型：提供大容量Key-Value存储，适用于对性能要求不高且有降低成本诉求的业务。 说明 云原生部署模式下暂不支持“标准型”的产品类型。
实例架构	- Proxy集群版： 采用分片集群架构，提供Proxy代理连接，同时兼容单机、哨兵、Cluster接入方式。性能水平扩展能力强，可支撑百万QPS、数十TB级业务量。 - Cluster集群版： 采用原生Cluster分片集群架构，完全兼容Redis Cluster直连访问接入，时延更低，性能更优。
兼容版本	7.0, 6.2(包含6.2.x), 5.0及以下版本。
CPU类型	x86。 x86类型的CPU架构采用复杂指令集（CISC），CISC指令集的每个小指令可以执行一些较低阶的硬件操作，指令数目多而且复杂，每条指令的长度并不相同。由于指令执行较为复杂所以每条指令花费的时间较长。

参数	描述
可用区	指在同一区域下，电力、网络隔离的物理区域，可用区之间内网互通，不同可用区之间物理隔离。 目前支持将实例部署在单可用区或3可用区。 - 如果业务要求实例之间的网络延时较低，则建议您选择单可用区，将实例的组件部署在同一个可用区内。 - 如果业务需要较高的容灾能力，请选择3可用区，此时实例下的节点Hash均衡部署在3个可用区内。

图 4-4 存储与规格（经典标准型）



图 4-5 存储与规格（云原生容量型）



表 4-3 规格与存储

参数	描述
规格选择模式	集群版实例支持快速选择和自定义资源两种规格选择模式。 快速选择 快速选择模式为您提供了常用的实例推荐配置，您可以根据业务需要选择合适的配置，无需自定义节点规格、节点数量和存储容量。 说明 - 仅单可用区支持8GB和16GB规格的实例，具体支持的实例规格以实际界面为准。 - QPS基准值仅为参考值，具体需要以业务实测为准。 自定义资源 您可以根据业务需要自定义实例的规格类型、节点规格、节点数量和实例存储总容量。

参数	描述
实例规格	当规格选择模式选择“快速选择”时，需要选择实例规格。 CPU规格越高，性能越好。请根据实际业务场景，选取符合业务类型的性能规格。 具体请参见 实例规格说明 。
规格类型	当规格选择模式选择“自定义资源”时，需要选择规格类型。 ● 标准型：CPU和内存为1:4，兼顾时延和并发，推荐默认选择1:4。 ● 增强型：CPU和内存为1:8，内存更大，访问命中内存概率更大，时延更低。
节点规格	当规格选择模式选择“自定义资源”时，“经典”部署模式，需要选择节点规格。 您可以根据业务需要选择节点规格，具体请参见 实例规格说明 。
节点数量	当规格选择模式选择“自定义资源”时，“经典”部署模式，需要选择节点数量。 节点数量请根据实际需要进行选取。创建成功后可以添加节点。 目前节点数上限为36个，如需更多，您可以在管理控制台右上角，选择“ 工单 > 新建工单 ”联系客服进行咨询。
实例存储总容量	当规格选择模式选择“自定义资源”时，需要选择实例存储总容量。 CPU规格越高，性能越好。请根据实际业务场景，选取符合业务类型的性能规格。 实例的CPU和内存，请参见 实例规格说明 。
实例规格预览	选择实例规格后，自动生成实例总容量、节点规格、节点数量、QPS基准、总连接数、数据副本数信息，方便用户知晓已选择的实例规格。
自动扩容	可根据业务需要选择是否自动扩容。 ● 存储容量使用率：存储容量使用率超出该值时，存储容量会自动扩容。可选值：60%，65%，70%，75%，80%，85%，90%。 ● 扩容步长：存储容量每次自动扩容的大小占存储总量的比率，可选值：10%，15%，20%。

参数	描述
静态数据加密	您可以根据业务需要选择是否进行静态数据加密。 - 不加密：未开启加密功能。 - 加密：当启用加密功能，用户创建数据库实例成功后，磁盘数据会在服务端加密成密文后存储。用户下载加密对象时，存储的密文会先在服务端解密为明文，再提供给用户，用于提高数据安全，但对数据库读写性能有少量影响。 密钥名称：选择或创建密钥。 说明 - 该功能目前为公测阶段，如需使用，您可以在管理控制台右上角，选择“工单 > 新建工单”，联系客服申请开通。 - 开启静态数据加密，将会创建委托。 - 实例创建成功后，不可修改静态数据加密状态，且无法更改密钥。 - 密钥在使用过程中不能被禁用、删除或冻结，否则会导致数据库不可用。 - 创建密钥请参见《密码安全中心用户指南》的创建密钥章节内容。

表 4-4 网络

参数	描述
虚拟私有云	实例所在的虚拟专用网络，可以对不同业务进行网络隔离。您可以根据需要创建或选择所需的虚拟私有云。 如何创建虚拟私有云，请参见《虚拟私有云用户指南》中的“创建虚拟私有云基本信息及默认子网”。 您还可以通过共享VPC功能，使用其他账号共享的VPC和子网。 共享VPC是基于资源访问管理（Resource Access Manager，简称RAM）服务的机制，VPC的所有者可以将VPC内的子网共享给其他账号使用，以实现网络资源的共享和统一管理，提升资源管控效率、降低运维成本。 有关VPC子网共享的更多信息，请参见《虚拟私有云用户指南》的“共享VPC”相关内容。 如果没有可用的VPC，系统自动为您分配资源。 说明 - 目前GeminiDB Redis实例创建完成后不支持切换虚拟私有云VPC，请谨慎选择所属虚拟私有云。 - 如需与ECS进行内网通信，GeminiDB Redis实例与需要通信的ECS需要处于同一个虚拟私有云下，或者不同虚拟私有云之间配置对等连接。
子网	通过子网提供与其他网络隔离的、可以独享的网络资源，以提高网络安全。 说明 目前不支持选择IPV6网段的子网，建议您在使用时创建并选择IPV4网段的子网。

参数	描述
内网安全组	安全组限制安全访问规则，加强GeminiDB Redis与其他服务间的安全访问。请确保所选取的安全组允许客户端访问数据库实例。 如果没有可用的安全组，系统自动为您分配资源。
数据库端口	数据库的访问端口号。 GeminiDB Redis支持自定义端口号，取值范围为：1024~65535，其中，如下端口号为禁用端口：2180、2887、3887、6377、6378、6380、8018、8079、8091、8479、8484、8999、12017、12333、50069。 当您不指定端口时，创建实例的访问端口默认为6379。
访问控制	创建后设置 - 创建后设置：若不设置则默认根据VPC的访问策略进行限制。 - 现在设置： 可以选择允许所有IP访问、白名单和黑名单。 允许所有IP访问：不进行访问控制，允许所有IP访问此实例。 白名单：仅允许IP地址组中的IP访问此实例。 黑名单：不允许IP地址组中的IP访问此实例。 说明 Cluster集群版暂不支持访问控制。

表 4-5 数据库配置

参数	描述
设置密码	- 创建后设置：支持创建实例后设置数据库密码。 - 现在设置：创建实例时设置数据库密码。 说明 Cluster集群版暂不支持创建后设置密码。
密码	数据库管理员rwuser账号设置的密码。 - 长度为8~32个字符。 - 必须是大写字母、小写字母、数字、特殊字符中至少两种的组合，其中可输入~!@#%^*_-=+?特殊字符。 - 系统会进行弱密码校验，安全起见，请输入高强度密码。 请妥善保管您的密码，因为系统将无法获取您的密码信息。
确认密码	必须和上述数据库密码一致。

参数	描述
免密访问配置	配置免密的网段，连接实例时，可以免输入实例密码。 - 创建后设置 选择创建后设置，可以在GeminiDB Redis实例创建成功后，设置免密访问，操作详情请参见开启免密访问。 - 现在设置 填写免密网段，最多可配置30个免密网段。 说明 Cluster集群版实例暂不支持免密访问配置。

表 4-6 企业项目

参数	描述
企业项目	该参数针对企业用户使用。 企业项目是一种云资源管理方式，企业项目管理服务提供统一的云资源按项目管理，以及项目内的资源管理、成员管理，默认项目为default。 请在下拉框中选择所在的企业项目。更多关于企业项目的信息，请参见《企业管理用户指南》。

表 4-7 更多配置

参数	描述
SSL安全连接	SSL证书是一种遵守SSL协议的服务器数字证书，能在客户端和服务端之间建立加密通道，保证数据在传输过程中不被窃取或篡改。 您可以开启SSL连接，提高数据安全性，实例创建成功后，通过SSL的方式连接实例。 说明 如果您在创建实例时未开启SSL安全连接，可以待实例创建成功后开启，操作详情请参见给GeminiDB Redis实例设置SSL数据加密。
参数模板	用户可以使用数据库参数模板中的参数来管理数据库接口配置。数据库参数模板就像是接口配置值的容器，这些值可应用于一个或多个数据库实例。 每个用户可以创建100个参数模板，同一项目下所有类型的实例可以共享该配额。具体参照参见创建参数模板。

参数	描述
标签	可选配置，对GeminiDB Redis的标识。使用标签可以方便识别和管理您拥有的GeminiDB Redis资源。每个实例默认最多支持20个标签配额。 如您的组织已经设定GeminiDB Redis的相关标签策略，则需按照标签策略规则为实例添加标签。标签如果不符合标签策略的规则，则可能会导致实例创建失败，请联系组织管理员了解标签策略详情。 标签由标签“键”和标签“值”组成。 - 键：如果要为数据库实例添加标签，该项为必选参数。对于每个实例，每个标签的键唯一。长度范围1到128个字符，不能以“_sys_”开头和以空格开头、结尾，且只能包含数字、英文字母、下划线、点、中划线、中文、空格、冒号、加号、等号、符号@和/。 - 值：如果要为数据库实例添加标签，该项为可选参数。可以为空。长度不超过255个字符，只能包含数字、英文字母、下划线、点、中划线、中文、空格、冒号、加号、等号、符号@和/。 实例创建成功后，您可以单击实例名称，在“标签”页签下查看对应标签。同时，支持为已有实例添加、修改、删除标签，具体操作请参见GeminiDB Redis标签管理。

表 4-8 购买时长

参数	描述
购买时长	选择“包年/包月”方式的用户需要设置购买时长，最短为1个月，最长为3年。
自动续费	- 默认不勾选，不进行自动续费。 - 勾选后实例自动续费，自动续费周期与原订单周期一致。

步骤4 在“订单详情确认”页面，核对实例信息。

- 包年/包月
 - 如果需要修改，单击“上一步”，修改实例信息。
 - 核对无误后，勾选协议，单击“去支付”，进入“付款”页面，选择支付方式，完成支付。
- 按需计费
 - 如果需要修改，单击“上一步”，修改实例信息。
 - 核对无误后，勾选协议，单击“提交”，开始创建实例。

步骤5 在“实例管理”页面，您可以查看并管理实例。

实例创建过程大约持续5-15分钟，实例创建完成后，运行状态显示为“正常”。

如果页面长时间未刷新，您可以单击页面右上角的  刷新页面查看实例运行状态。

----结束

4.2.2 购买 GeminiDB Redis 主备版实例

本章节主要介绍了如何在云数据库 GeminiDB管理控制台购买主备版Redis数据库实例。

每个租户下GeminiDB Redis实例的默认总配额为50。如需申请扩大配额，您可以在管理控制台右上角，选择“[工单 > 新建工单](#)”，联系客服申请开通。

前提条件

- [注册华为账号并开通华为云](#)。
- 账户余额大于或等于0元。

操作步骤

- 步骤1** 登录[云数据库GeminiDB控制台](#)。
- 步骤2** 在“实例管理”页面，单击“购买数据库实例”，进入“服务选型”页面。
- 步骤3** 在“服务选型”页面，选择计费模式，填写并选择实例相关信息后，单击“立即购买”。

图 4-6 计费模式和基本信息

计费模式

包年包月

按需计费

区域

不同区域的资源之间内网不互通。请选择靠近您业务的区域，可以降低网络时延，提高访问速度。

实例名称

gemini01

购买多个数据库实例时，名称自动按序增加月、日、时分秒和两位数字后缀。例如：输入instance，从instance-0101-120101-01开始命名。若已有instance-0101-120101-01，则从instance-0101-120101-02开始命名。

兼容接口

Redis

Cassandra

DynamoDB

HBase

InfluxDB

部署模式

经典

云原生

传统架构，稳定可靠。

产品类型

标准型

容量型

提供稳定的低延迟性能，适用于广告&推荐、游戏、电商、车联网等各类常见业务场景。

实例架构

Proxy集群版

Cluster集群版

主备版

标准master-replica架构，兼容单机/两节点接入方式。
您还可以购买49个数据库实例。如需申请更多配额请点击 [申请扩大配额](#)

兼容版本

7.0

完全兼容Redis 6.2、5.0及其以下版本。部分兼容Redis 7.0，详见[命令兼容列表](#)

CPU类型

x86

可用区

可用区1可用区2可用区3

推荐3AZ架构部署，可实现跨AZ容灾，且保证RPO=0

主可用区

可用区1

可用区2

可用区3

备可用区

可用区1

可用区2

可用区3

表 4-9 计费模式

参数	描述
计费模式	选择“包年包月”或“按需计费”。 - 包年包月 - 用户选购完服务配置后，可以根据需要设置购买时长，系统会一次性按照购买价格对账户余额进行扣费。 - 创建成功后，如果包年/包月实例到期后不再长期使用资源，可将“包年/包月”实例转为“按需计费”，到期后将转为按需计费实例。具体请参见包年/包月转按需。 - 说明 “包年/包月”方式购买的实例就是包周期实例，“包年/包月”方式购买的实例不能直接删除，仅支持资源退订操作，如何退订资源请参见如何退订包年/包月实例。 - 按需计费 - 用户选购完服务配置后，无需设置购买时长，系统会根据消费时长对账户余额进行扣费。 - 创建成功后，如果需要长期使用资源，可将“按需计费”实例转为“包年/包月”，继续使用这些资源的同时，享受包年/包月的低资费。具体请参见按需转包年/包月。

表 4-10 基本信息

参数	描述
区域	租户所在的区域。 说明 请就近选择靠近您业务的区域，可减少网络时延，提高访问速度。不同区域内的产品内网不互通，且购买后不能更换，请谨慎选择。
实例名称	设置实例名称时，需要满足如下规则。 - 实例名称允许和已有名称重复。 - 实例名称长度在4个到64个字符之间，必须以字母或中文字开头，区分大小写，可以包含字母、数字、中划线、下划线或中文（一个中文字符占用3个字节），不能包含其他特殊字符。如果名称包含中文，则不超过64字节。 创建成功后，可修改实例名称，请参见修改GeminiDB Redis实例名称。
兼容接口	Redis。 说明 云数据库 GeminiDB目前兼容Redis、DynamoDB、Cassandra、HBase、InfluxDB和MongoDB主流NoSQL接口，您当前选择的是Redis接口，其他接口的选择具体参见如何选择接口。
部署模式	经典：GeminiDB经典的存算分离架构。
产品类型	标准型：提供稳定的低延迟性能，适用于广告&推荐、游戏、电商、车联网等各类常见业务场景。

参数	描述
实例架构	主备版。 说明 标准的master-replica架构，兼容单机、哨兵接入方式。适用于无法使用hashtag的业务场景。
兼容版本	7.0，6.2(包含6.2.x)，5.0及以下版本。
CPU类型	x86。 说明 x86类型的CPU架构采用复杂指令集（CISC），CISC指令集的每个小指令可以执行一些较低阶的硬件操作，指令数目多而且复杂，每条指令的长度并不相同。由于指令执行较为复杂所以每条指令花费的时间较长。
可用区	指在同一区域下，电力、网络隔离的物理区域，可用区之间内网互通，不同可用区之间物理隔离。多可用区时需要选择主可用区和备可用区。目前支持将实例部署在单可用区或3可用区。 - 如果业务要求实例之间的网络延时较低，则建议您选择单可用区，将实例的组件部署在同一个可用区内。 - 如果业务需要较高的容灾能力，请选择3可用区，此时实例下的节点根据您的需要选择主可用区和备可用区。 - 主可用区：主节点所在的可用区。 - 备可用区：备节点所在的可用区。

图 4-7 规格与存储

规格选择模式

快速选择自定义资源

实例规格 ①	总容量 (存储)	分片数	QPS规格 ②	最大连接数	实例内存数 ③	ACL子程序数量
☐ 4 GB	1	8,000	10,000	1,000	200	
☐ 8 GB	1	8,000	10,000	1,000	200	
☒ 16 GB	1	10,000	10,000	1,000	200	
☐ 24 GB	1	20,000	10,000	1,000	200	
☐ 32 GB	1	20,000	10,000	1,000	200	
☐ 48 GB	1	40,000	20,000	1,000	200	
☐ 64 GB	1	40,000	20,000	1,000	200	

实例规格选择

节点规格 标准型 1 vCPU 4 GB (节点数量 2个) 总内存 4 GB (存储总容量 16 GB) QPS规格 10,000 (总连接数 10,000) 数据副本数 3

数据实例容量。存储总容量大于内存时，可提供超内存的存储容量。实例最大数据容量由存储总容量决定。若需要灵活配置内存和存储，可通过自定义资源。

表 4-11 规格与存储

参数	描述
规格选择模式	主备版实例支持快速选择和自定义资源两种规格选择模式。 - 快速选择 快速选择模式为您提供了常用的实例推荐配置，您可以根据业务需要选择合适的配置，无需自定义节点规格、节点数量和存储容量。 说明 - QPS基准值仅为参考值，具体需要以业务实测为准。 - 自定义资源 您可以根据业务需要自定义实例的节点规格、节点数量和存储容量。

参数	描述
实例规格	当规格选择模式选择“快速选择”时，需选择实例规格。 CPU规格越高，性能越好。请根据实际业务场景，选取符合业务类型的性能规格。 实例的CPU和内存，请参见实例规格说明。
规格类型	当规格选择模式选择“自定义资源”时，需选择规格类型。 标准型：CPU和内存为1:4，兼顾时延和并发，推荐默认选择1:4。
节点规格	当规格选择模式选择“自定义资源”时，需要选择节点规格。 您可以根据业务需要选择节点规格，具体请参见实例规格说明。
节点数量	2。一主一备2节点，故障时主备可以自动进行切换。
实例存储总容量	当规格选择模式选择“自定义资源”时，需要选择实例存储总容量。 CPU规格越高，性能越好。请根据实际业务场景，选取符合业务类型的性能规格。 实例的CPU和内存，请参见实例规格说明。
实例规格预览	选择实例规格后，自动生成实例总容量、节点规格、节点数量、QPS基准、总连接数、数据副本数信息，方便用户知晓已选择的实例规格。
自动扩容	可根据业务需要选择是否自动扩容。 - 存储容量使用率：存储容量使用率超出该值时，存储容量会自动扩容。可选值：60%，65%，70%，75%，80%，85%，90%。 - 扩容步长：存储容量每次自动扩容的大小占存储总量的比率，可选值：10%，15%，20%。
静态数据加密	您可以根据业务需要选择是否进行静态数据加密。 - 不加密：未开启加密功能。 - 加密：当启用加密功能，用户创建数据库实例成功后，磁盘数据会在服务端加密成密文后存储。用户下载加密对象时，存储的密文会先在服务端解密为明文，再提供给用户，用于提高数据安全，但对数据库读写性能有少量影响。 密钥名称：选择或创建密钥。 说明 - 该功能目前为公测阶段，如需使用，您可以在管理控制台右上角，选择“工单 > 新建工单”，联系客服申请开通。 - 开启静态数据加密，将会创建委托。 - 实例创建成功后，不可修改静态数据加密状态，且无法更改密钥。 - 密钥在使用过程中不能被禁用、删除或冻结，否则会导致数据库不可用。 - 创建密钥请参见《密码安全中心用户指南》的创建密钥章节内容。

图 4-8 网络和数据库配置

网络配置

vmc-15-0

查看或私有化云

目前Docker私有化镜像不支持对私网私有云VPC，请谨慎选择所属虚拟机私有云。如需与ECI进行内网通信，建议将OpenSSH与需要通信的ECI放置到同一个虚拟机私有云下，避免不同虚拟机私有云之间需跨网段通信。如需新建的虚拟机私有云，可前往VPC控制台创建。

子网

subnet-15-0-0-0-024

查看子网

设置当前实例需要6个IP，当前所选的子网下可用私有IP数量171个。

网络安全组

default-09963024-1984-4734-8148-0247472

查看安全组

实例均要求必须不支持安全组，创建实例时需提前配置其访问控制。

数据源端口

6379

查看端口规则

如需实例计划用于搭建双节点redis集群，请配置为私网端口。

访问控制

现在设置

创建访问策略

设置密码

现在设置

创建新设置

密码

确认密码

免密访问配置

现在设置

创建新设置

云盘挂载

挂载策略

创建新策略

云盘挂载

挂载策略

查看策略管理

网络配置

—请选择—

查看策略管理

网络配置

— 1 — +

表 4-12 网络

参数	描述
虚拟私有云	实例所在的虚拟专用网络，可以对不同业务进行网络隔离。您可以根据需要创建或选择所需的虚拟私有云。 如何创建虚拟私有云，请参见《虚拟私有云用户指南》中的“创建虚拟私有云基本信息及默认子网”。 您还可以通过共享VPC功能，使用其他账号共享的VPC和子网。 共享VPC是基于资源访问管理（Resource Access Manager，简称RAM）服务的机制，VPC的所有者可以将VPC内的子网共享给其他账号使用，以实现网络资源的共享和统一管理，提升资源管控效率、降低运维成本。 有关VPC子网共享的更多信息，请参见《虚拟私有云用户指南》的“共享VPC”相关内容。 如果没有可用的VPC，系统自动为您分配资源。 说明 - 目前实例创建完成后不支持切换虚拟私有云VPC，请谨慎选择所属虚拟私有云。 - 如需与ECS进行内网通信，实例与需要通信的ECS需要处于同一个虚拟私有云下，或者不同虚拟私有云之间配置对等连接。
子网	通过子网提供与其他网络隔离的、可以独享的网络资源，以提高网络安全。 说明 目前不支持选择IPv6网段的子网，建议您在创建并选择IPv4网段的子网。
内网安全组	安全组限制安全访问规则，加强与其他服务间的安全访问。请确保所选取的安全组允许客户端访问数据库实例。 如果没有可用的安全组，系统自动为您分配资源。

参数	描述
数据库端口	数据库的访问端口号。 GeminiDB Redis支持自定义端口号，取值范围为：1024~65535，其中，如下端口号为禁用端口：2180、2887、3887、6377、6378、6380、8018、8079、8091、8479、8484、8999、12017、12333、50069。 当您不指定端口时，创建实例的访问端口默认为6379。
访问控制	创建后设置 - 创建后设置：若不设置则默认根据VPC的访问策略进行限制。 - 现在设置： 可以选择允许所有IP访问、白名单和黑名单。 允许所有IP访问：不进行访问控制，允许所有IP访问此实例。 白名单：仅允许IP地址组中的IP访问此实例。 黑名单：不允许IP地址组中的IP访问此实例。

表 4-13 数据库配置

参数	描述
设置密码	- 创建后设置：支持创建实例后设置数据库密码。 - 现在设置：创建实例时设置数据库密码。
密码	数据库管理员rwuser账号设置的密码。 - 长度为8~32个字符。 - 必须是大写字母、小写字母、数字、特殊字符中至少两种的组合，其中可输入~!@#%&*()_-=+?\${}&特殊字符。 - 系统会进行弱密码校验，安全起见，请输入高强度密码。请妥善保管您的密码，因为系统将无法获取您的密码信息。
确认密码	必须和管理员密码一致。
免密访问配置	配置免密的网段，连接实例时，可以免输入实例密码。 - 创建后设置 选择创建后设置，可以在GeminiDB Redis实例创建成功后，设置免密访问，操作详情请参见开启免密访问。 - 现在设置 填写免密网段，最多可配置30个免密网段。
企业项目	该参数针对企业用户使用。 企业项目是一种云资源管理方式，企业项目管理服务提供统一的云资源按项目管理，以及项目内的资源管理、成员管理，默认项目为default。 请在下拉框中选择所在的企业项目。更多关于企业项目的信息，请参见 《企业管理用户指南》 。

表 4-14 更多配置

参数	描述
静态数据加密	静态加密 (encryption at rest) 即在存储数据时进行数据加密。 - 不加密 - 加密 开启静态数据加密，可以提高数据安全性，但对数据库读写性能有少量影响。开启磁盘加密，将会创建委托。
方式选择	“静态数据加密”选择“加密”时，需要选择“方式选择”： - 下拉选择：选择“密钥名称”。 - 手动输入：手动输入密钥ID。注意：ID不能为空，且您的KMS密钥必须位于当前Region。
密钥名称	可以选择已有密钥名称，或者创建密钥。密钥在使用过程中不能被禁用、删除或冻结，否则会导致数据库不可用。
SSL安全连接	SSL证书是一种遵守SSL协议的服务器数字证书，能在客户端和服务端之间建立加密通道，保证数据在传输过程中不被窃取或篡改。 您可以开启SSL连接，提高数据安全性，实例创建成功后，通过 SSL的方式 连接实例。 说明 如果您在创建实例时未开启SSL安全连接，可以待实例创建成功后开启，操作详情请参见 给GeminiDB Redis实例设置SSL数据加密 。
命令重命名	GeminiDB Redis支持命令重命名，为了避免误操作导致数据丢失，实例重启，性能抖动等问题，您可以对高危命令进行重命名。具体参见 GeminiDB Redis命令重命名 。
参数模板	用户可以使用数据库参数模板中的参数来管理数据库接口配置。数据库参数模板就像是接口配置值的容器，这些值可应用于一个或多个数据库实例。 每个用户可以创建100个参数模板，同一项目下所有类型的实例可以共享该配额。具体参照参见 创建参数模板 。
可维护时间段	用户指定对数据库实例有影响的变更任务的时间范围，例如规格变更、补丁升级等操作的时间窗。 - 暂不设置 - 选择时段

参数	描述
标签	可选配置，对的标识。使用标签可以方便识别和管理您拥有的资源。每个实例默认最多支持20个标签配额。 如您的组织已经设定GeminiDB Redis的相关标签策略，则需按照标签策略规则为实例添加标签。标签如果不符合标签策略的规则，则可能会导致实例创建失败，请联系组织管理员了解标签策略详情。 标签由标签“键”和标签“值”组成。 - 键：如果要为数据库实例添加标签，该项为必选参数。对于每个实例，每个标签的键唯一。长度范围1到128个字符，不能以“_sys_”开头和以空格开头、结尾，且只能包含数字、英文字母、下划线、点、中划线、中文、空格、冒号、加号、等号、符号@和/。 - 值：如果要为数据库实例添加标签，该项为可选参数。可以为空。长度不超过255个字符，只能包含数字、英文字母、下划线、点、中划线、中文、空格、冒号、加号、等号、符号@和/。 实例创建成功后，您可以单击实例名称，在“标签”页签下查看对应标签。同时，支持为已有实例添加、修改、删除标签，具体操作请参见GeminiDB Redis标签管理。

表 4-15 购买时长

参数	描述
购买时长	选择“包年/包月”方式的用户需要设置购买时长，最短为1个月，最长为3年。
自动续费	- 默认不勾选，不进行自动续费。 - 勾选后实例自动续费，自动续费周期与原订单周期一致。

步骤4 在“订单详情确认”页面，核对实例信息。

- 包年/包月
 - 如果需要修改，单击“上一步”，修改实例信息。
 - 核对无误后，勾选协议，单击“去支付”，进入“付款”页面，选择支付方式，完成支付。
- 按需计费
 - 如果需要修改，单击“上一步”，修改实例信息。
 - 核对无误后，勾选协议，单击“提交”，开始创建实例。

步骤5 在“实例管理”页面，您可以查看并管理实例。

实例创建过程大约持续5-15分钟，实例创建完成后，运行状态显示为“正常”。

如果页面长时间未刷新，您可以单击页面右上角的  刷新页面查看实例运行状态。

----结束

4.3 实例连接及管理

4.3.1 连接 GeminiDB Redis 实例方式介绍

GeminiDB Redis兼容开源Redis接口，支持应用程序使用多种SDK访问。同时还提供数据管理服务（Data Admin Service，简称DAS）、内网、公网的访问途径。

连接GeminiDB Redis实例的使用流程介绍如图4-9所示。

图 4-9 连接方式介绍

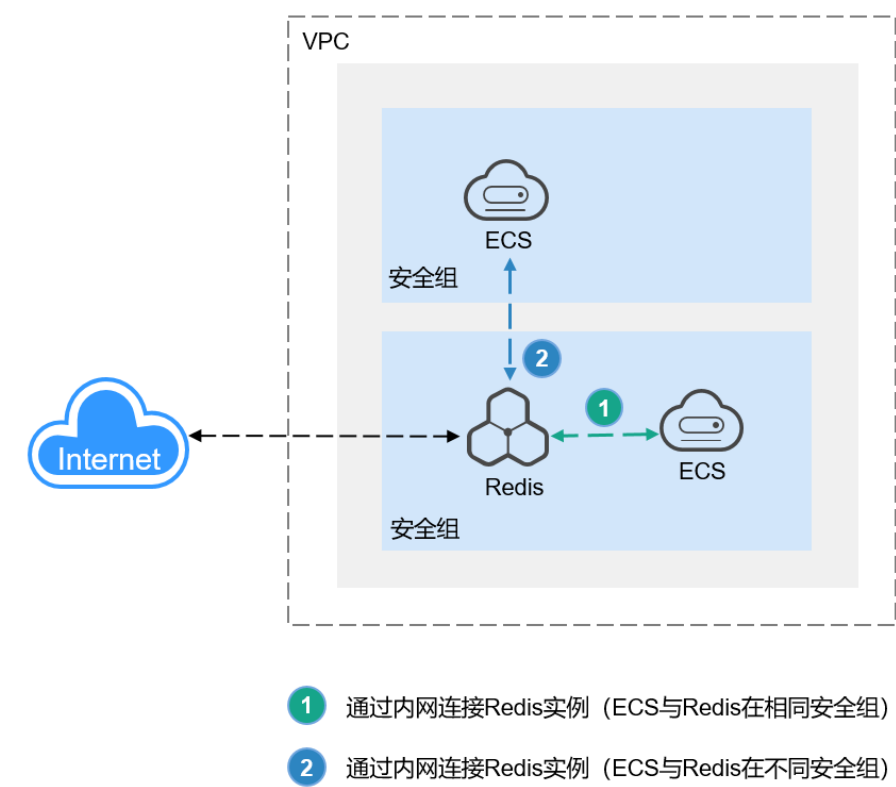


表 4-16 连接方式

连接方式	使用场景	默认端口	说明
DAS连接	GeminiDB Redis支持通过管理控制台的Web客户端连接Redis实例	-	-

连接方式	使用场景	默认端口	说明
内网连接	介绍通过内网IP、内网域名或负载均衡地址连接 GeminiDB Redis实例的方法。 该方式适用于当应用部署在弹性云服务器上，且该弹性云服务器与数据库实例处于同一区域、同一VPC内时连接数据库实例。	6379	- 推荐使用负载均衡地址连接实例，可靠性高，可消除单点故障带来的影响。 - 安全性高，可实现数据库实例的较好性能。 - ECS与Redis实例在相同安全组默认ECS与Redis实例内网互通，无需设置安全组规则。 - ECS与Redis实例在不同安全组时，需要为Redis和ECS分别设置安全组规则。 - 设置Redis安全组规则：为Redis所在安全组配置相应的入方向规则，详见设置安全组规则。 - 设置ECS安全组规则：安全组默认规则为出方向上数据报文全部放行，此时，无需对ECS配置安全组规则。当在ECS所在安全组为非默认安全组且出方向规则非全放通时，需要为ECS所在安全组配置相应的出方向规则。
公网连接	介绍通过公网域名、弹性公网IP或负载均衡器绑定弹性公网IP连接 GeminiDB Redis实例的方法。 该方式适用于不能通过内网方式访问数据库实例时，可以单独使用公网域名或弹性公网IP连接弹性云服务器（或公网主机）与数据库实例。	6379	- 为了获得更快的传输速率和更高的安全性，建议您将应用迁移到与您的数据库实例在同一VPC子网内，使用内网连接。 - 推荐负载均衡器绑定弹性公网IP的方式，可靠性高，可消除单点故障影响。 - 用户需要购买弹性公网IP，请参见弹性公网IP计费说明。 说明 Cluster集群版实例暂不支持公网访问。
程序代码连接	介绍通过不同代码的方式访问GeminiDB Redis实例的方法，详情请参见通过程序代码连接实例示例。	6379	-

4.3.2 通过 DAS 连接 GeminiDB Redis

数据管理服务（Data Admin Service，简称DAS）是一款专业的简化数据库管理工具，提供优质的可视化操作界面，大幅提高工作效率，让数据管理变得既安全又简

单。您可以通过数据管理服务连接并管理实例。GeminiDB Redis默认为您开通了远程主机登录权限，推荐您使用更安全便捷的数据管理服务连接实例。

权限配置

使用IAM（统一身份认证）的用户，需要给对应的子用户授予DAS FullAccess权限，否则子用户无法正常使用该功能。具体授权的操作请参见[创建用户并授权](#)。

同时您也可以通过创建自定义策略的方式来限制有权限的数据库类型，具体操作如下：

- 1. 登录统一身份认证服务页面，单击“创建自定义策略”。

图 4-10 创建自定义策略



- 2. 设置策略名称、策略配置方式、策略内容等信息。

图 4-11 配置自定义策略

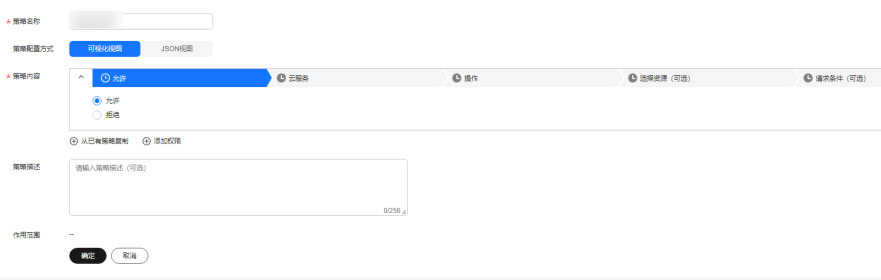


表 4-17 自定义策略信息

参数名称	描述
策略名称	输入策略名称。
策略配置方式	此处选择“JSON视图”。

参数名称	描述
策略内容	在策略内容中配置如下自定义策略： <pre>{ "Version": "1.1", "Statement": [{ "Action": ["das:*:*", "nosql:instance:list"], "Effect": "Allow" }] }</pre> 或者您也可以在“策略内容”区域，单击“从已有策略复制”，选择“DAS FullAccess”作为模板，然后只保留对应的DB类型就可以指定权限的范围，此处只需要保留“nosql:instance:list”即可。
策略描述	输入策略描述。
作用范围	默认项目级服务

3. 单击“确定”，返回权限页面，可以查看到创建成功的自定义策略。

图 4-12 查看自定义策略



4. 创建用户组。

图 4-13 创建用户组

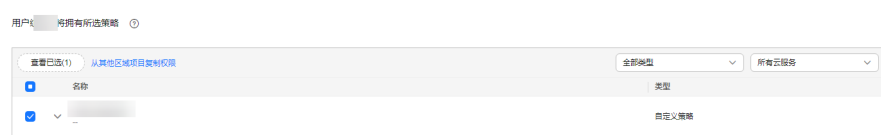


5. 为4创建的用户组授权3已创建好的自定义策略。

图 4-14 授权自定义策略



图 4-15 选择自定义策略



6. 单击用户组名称，进入用户组管理，添加对应的子用户即可。

图 4-16 添加子用户



前提条件

已创建GeminiDB Redis实例且实例状态处于“正常”。

操作步骤

- 步骤1** 登录[云数据库GeminiDB控制台](#)。
- 步骤2** 在实例列表中选择目标实例，单击操作列“连接Redis”。

图 4-17 连接 Redis

实例名称	实例ID	实例类型	网络类型	部署模式	运行状态	实例规格	计费模式	操作
gemini-01	62a35c33866481a48f0d0c0902084612	Redis 7.0	经典	正常	IPv4	默认	按量计费	连接Redis 帮助文档 更多

您也可以单击实例名称，进入“基本信息”页面，在页面右上角单击“连接Redis”。

图 4-18 连接 Redis



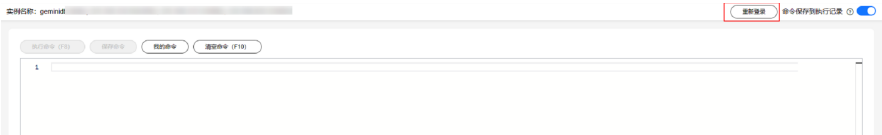
- 步骤3** 输入实例的密码登录实例。

图 4-19 登录 Redis 实例



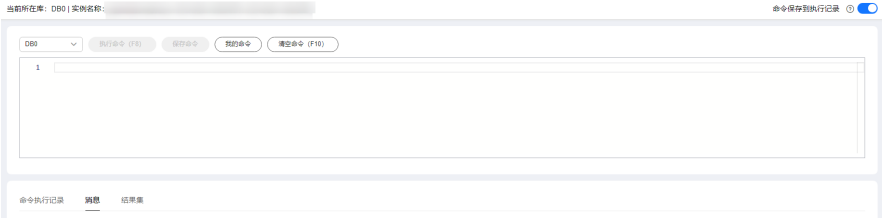
如果重置密码后，需要重新登录，请单击右上角的“重新登录”按钮，使用新密码进行登录。

图 4-20 重新登录



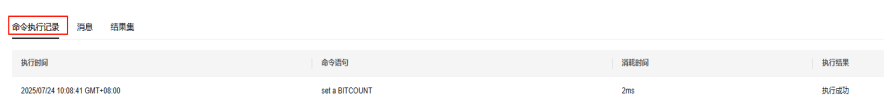
步骤4 根据业务需要进行相关数据库管理操作。

图 4-21 实例首页



- 命令保存到执行记录
默认开启命令保存到执行记录的功能，主要用于保存最近执行的命令到服务器中，方便您查看最近执行过的操作记录。
该功能开启后，您可以在页面下方的命令执行记录中查看操作记录。

图 4-22 查看命令执行记录



关闭该功能后，页面将不再展示后续的命令执行记录。您可以单击右上角“命令保存到执行记录”后的  关闭此功能。

- 执行命令

您可以在命令执行框中输入需要执行的命令，单击“执行命令”或者通过快捷键“F8”执行。

 说明

- 不建议使用事务，Lua脚本，不允许使用Pub/Sub和其他具有阻塞语义的命令。
- 支持多DB的版本可以在前端切换当前库，但不允许使用SELECT切换。

图 4-23 执行命令



命令执行成功后，您可以在页面下方结果集中查看执行结果。

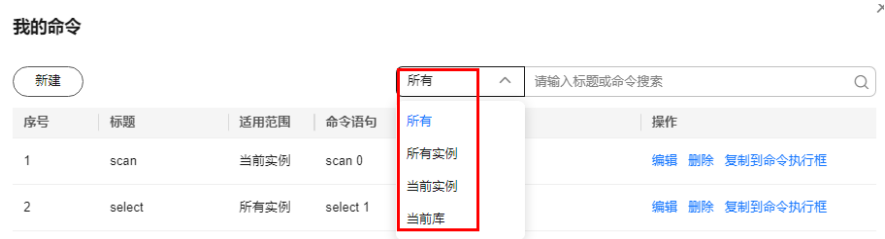
- 保存命令
您可以将命令保存于所有实例、当前实例、当前库。保存成功后可以在“我的命令”中查看详情。

图 4-24 保存命令



- 我的命令
我的命令主要展示的是一些常见操作命令。
您可以根据命令的适用范围筛选命令，当选择“所有”时可以查询到当前账号下所保存的所有命令，其他类型可根据相应的适用范围进行查询。

图 4-25 筛选命令



您也可以通过搜索框按照命令标题和命令语句进行查询。

图 4-26 搜索命令



另外，您还可以在“我的命令”页签中新建命令、编辑命令、删除命令、复制到命令执行框。

图 4-27 命令管理



- 清空命令
清除当前命令执行框中的命令，您也可以通过快捷键“F10”操作。

图 4-28 清空命令



----结束

常见问题

问题：在实例列表的操作列单击“连接Redis”时或者在“基本信息”页面单击“连接Redis”时，无法跳转至DAS页面怎么办？
解决方法：请将您的浏览器设置为允许弹出窗口，然后重试。

4.3.3 通过内网连接 GeminiDB Redis

当应用部署在弹性云服务器上，并且该弹性云服务器与GeminiDB Redis数据库实例位于同一区域和VPC内时，可以通过内网IP、内网域名或负载均衡地址连接GeminiDB Redis实例。

4.3.3.1 通过负载均衡地址连接实例（推荐）

本章节介绍如何在Linux系统的弹性云服务器上，使用负载均衡地址连接GeminiDB Redis实例，可以提升数据可靠性，消除单点故障影响。

使用须知

- 目标实例必须与弹性云服务器在同一个虚拟私有云和子网内才能访问。
- 弹性云服务器必须处于目标实例所属安全组允许访问的范围内。
场景一：如果目标实例所属安全组为**默认安全组**，则无需设置安全组规则。
场景二：如果目标实例所属安全组**非默认安全组**，请查看安全组规则是否允许该弹性云服务器访问。
 - 如果安全组规则允许弹性云服务器访问，即可连接实例。
 - 如果安全组规则不允许弹性云服务器访问，需要在实例安全组添加一条“入”的访问规则。设置安全组的具体操作请参考[给GeminiDB Redis实例节点配置安全组规则](#)。

前提条件

- 创建弹性云服务器，以Linux操作系统为例。详情请参见《弹性云服务器快速入门》中[购买弹性云服务器](#)的内容。
- 下载[Redis客户端安装包](#)。

操作步骤

步骤1 登录弹性云服务器，详情请参见《弹性云服务器快速入门》中[登录弹性云服务器](#)的内容。

步骤2 获取Redis客户端。

方法一：

通过如下命令，下载Redis客户端。

```
wget http://download.redis.io/releases/redis-6.2.0.tar.gz
```

方法二：

通过[前提条件](#)中的地址下载Redis客户端，并将Redis客户端安装包上传到弹性云服务器。

步骤3 解压客户端工具包。

```
tar -xzf redis-6.2.0.tar.gz
```

步骤4 在“src”目录下，连接数据库实例。

```
cd redis-6.2.0
```

```
make
```

```
cd src
```

```
./redis-cli -h <DB_HOST> -p <DB_PORT> -a <DB_PWD>
```

示例：

```
./redis-cli -h 192.xx.xx.xx -p 6379 -a <DB_PWD>
```

表 4-18 参数说明

参数	说明
<DB_HOST>	待连接实例的负载均衡地址。 您可以单击实例名称，进入“基本信息”页面，在连接信息区域获取到“负载均衡地址”。
<DB_PORT>	待连接实例的负载均衡地址对应的访问端口。 具体查看方法如下： 在“实例管理”页面，单击实例名称，进入“基本信息”页面，在“连接信息 > 实例端口”处获取当前端口信息。
<DB_PWD>	购买GeminiDB Redis实例时设置的管理员密码。

步骤5 出现如下信息，说明连接成功。

```
IP:port>
```

----结束

4.3.3.2 通过内网域名连接实例

本章节介绍如何在Linux系统的弹性云服务器上，使用内网域名连接GeminiDB Redis实例。

使用须知

- 目标实例必须与弹性云服务器在同一个虚拟私有云和子网内才能访问。
- 弹性云服务器必须处于目标实例所属安全组允许访问的范围内。
场景一：如果目标实例所属安全组为**默认安全组**，则无需设置安全组规则。
场景二：如果目标实例所属安全组**非默认安全组**，请查看安全组规则是否允许该弹性云服务器访问。
 - 如果安全组规则允许弹性云服务器访问，即可连接实例。
 - 如果安全组规则不允许弹性云服务器访问，需要在实例安全组添加一条“入”的访问规则。设置安全组的具体操作请参考[给GeminiDB Redis实例节点配置安全组规则](#)。

前提条件

- 创建弹性云服务器，以Linux操作系统为例。详情请参见《弹性云服务器快速入门》中[购买弹性云服务器](#)的内容。
- 下载[Redis客户端安装包](#)。

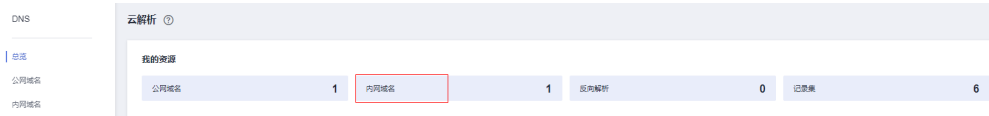
步骤 1 配置 GeminiDB Redis 实例的内网域名

创建内网域名：

步骤1 登录[云解析服务DNS控制台](#)。

步骤2 在云解析服务页面，选择“内网域名”，进入“内网域名”页面。

图 4-29 内网域名



步骤3 单击“创建内网域名”，进行域名创建。

图 4-30 创建域名



步骤4 填选内网域名信息。

图 4-31 配置域名信息

创建内网域名

*

域名

例如: example.com

*

区域

*

VPC

查看VPC

*

企业项目

—请选择—

新建企业项目

标签

如果您需要使用同一标签标识多种云资源，即所有服务均可在标签输入框下拉选择同一标签，建议在TMS中创建预定义标签。

查看预定义标签

在下方键/值输入框输入内容后单击'添加'，即可将标签加入此处

请输入标签键

请输入标签值

添加

您还可以添加10个标签。

描述

0/255

确定

取消

表 4-19 参数说明

参数名称	说明	示例
域名	创建的内网域名名称。 支持创建顶级域，但需符合域名命名规范。 域名的格式请参见 域名格式与级别 。	example.com
区域	租户所在的区域。	华东-上海一
VPC	内网域名关联的VPC需要和 GeminiDB Redis实例所在的 VPC一致，否则内网域名解析不成功。	-
企业项目	内网域名关联的企业项目，用于将内网域名按照企业项目进行管理。 说明 仅当用户使用的“账号类型”为“企业账号”时，显示该参数，且参数必选。 配置原则： • 如果不通过企业项目管理域名资源，则采用默认值“default”。 • 如果通过企业项目管理域名资源，则在下拉列表中选择已经创建的企业项目。	default

参数名称	说明	示例
标签	可选参数。域名的标识，包括键和值，每个域名可以创建20个标签。 键和值的命名规则如下： 键： - 不能为空。 - 对于同一资源键值唯一。 - 长度不超过128个字符。 - 标签的键可以包含任意语种字母、数字、空格，以及_ . : = + - @特殊字符，但首尾不能含有空格，不能以_sys_开头。 值： - 长度不超过255个字符。 - 标签的值可以包含任意语种字母、数字、空格，以及_ . : / = + - @特殊字符。	example_key1 example_value1
描述	可选参数。域名的描述信息，长度不超过255个字符。	This is a zone example.

步骤5 单击“确定”，在“内网域名”页面，查看域名的创建状态。

当域名状态为正常时，说明域名已创建成功。

图 4-32 查看域名状态

您还可以创建38个内网域名。

删除	所有项目	所有状态	域名	Q	标签搜索	C
域名	状态	记录集个数	已关联的VPC	企业项目	描述	操作
test2	正常	2	vpc-C	default	--	关联VPC 修改 删除
reditest	正常	2	vpc-C	default	--	关联VPC 修改 删除

----结束

为域名添加解析记录集:

内网域名创建成功后，需要为内网域名设置解析记录集，以便使用域名来访问实例。以下操作以添加A类型记录集为例，其他更多类型记录集请参见[添加内网域名解析记录](#)。

步骤1 单击新建的内网域名名称，进入“解析记录”页面，单击右上方“添加记录集”。

图 4-33 添加记录集



步骤2 在“添加记录集”弹出框中，根据界面提示填写记录集参数。

值：填写实例的负载均衡地址。

图 4-34 添加解析记录集

添加记录集

主机记录

reditest. ?

* 类型

A - 将域名指向IPv4地址

* TTL (秒)

3005分钟1小时12小时1天 ?

* 值

例:
192.168.10.10

?

标签

如果您需要使用同一标签标识多种云资源，即所有服务均可在标签输入框下拉选择同一标签，建议在TMS中创建预定义标签。[查看预定义标签](#) C

在下方键/值输入框输入内容后单击'添加'，即可将标签加入此处

请输入标签键

请输入标签值

添加

您还可以添加10个标签。

描述

0/255

确定

取消

参数配置说明请参见[添加内网域名解析记录](#)。

- 步骤3** 单击“确定”。
- 步骤4** 返回“解析记录”页面。
- 步骤5** 添加完成后，您可以在域名对应的记录集列表中查看已添加的记录集。当记录集的状态显示为“正常”时，表示记录集添加成功。
- 结束

步骤 2 登录弹性云服务器并通过 Redis 客户端连接实例

- 步骤1** 登录弹性云服务器，详情请参见《弹性云服务器快速入门》中[登录弹性云服务器](#)的内容。
- 步骤2** 获取Redis客户端。
- 方法一：**
- 通过如下命令，下载Redis客户端。
- ```
wget http://download.redis.io/releases/redis-6.2.0.tar.gz
```
- 方法二：**
- 通过[前提条件](#)中的地址下载Redis客户端，并将Redis客户端安装包上传到弹性云服务器。
- 步骤3** 解压客户端工具包。
- ```
tar -xzf redis-6.2.0.tar.gz
```
- 步骤4** 在“src”目录下，连接数据库实例。
- ```
cd redis-6.2.0
make
cd src
./redis-cli -h <DB_Domain_Name> -p <DB_PORT> -a <DB_PWD>
```
- 示例：
- ```
./redis-cli -h redis.com -p 6379 -a <DB_PWD>
```

表 4-20 参数说明

参数	说明
<DB_Domain_Name>	待连接实例的内网域名，内网域名为 步骤1 配置GeminiDB Redis实例的内网域名 中创建的域名。
<DB_PORT>	待连接实例的端口，请以实际端口为准。 获取实例端口的方法如下： 在“实例管理”页面，单击实例名称，进入“基本信息”页面，在“连接信息 > 实例端口”处获取当前实例的端口信息。
<DB_PWD>	购买GeminiDB Redis实例时设置的管理员密码。

步骤5 出现如下信息，说明连接成功。

```
Domain_Name:port>
```

----结束

4.3.3.3 通过内网 IP 连接实例

GeminiDB Redis支持使用弹性云服务器通过内网IP连接GeminiDB Redis实例。

本章节以Linux系统的弹性云服务器为例，指导您通过Redis-cli客户端连接GeminiDB Redis实例。您可以根据业务需求选用非SSL方式或SSL方式连接实例，其中，SSL连接实现了数据加密功能，具有更高的安全性，详情请参见[通过SSL方式连接GeminiDB Redis实例](#)。本章节中主要介绍非SSL方式连接GeminiDB Redis实例的步骤。

为了保障数据可靠性，推荐您使用[负载均衡地址](#)或者[域名](#)的方式访问实例。

使用须知

- 目标实例必须与弹性云服务器在同一个虚拟私有云和子网内才能访问。
- 弹性云服务器必须处于目标实例所属安全组允许访问的范围内，详情请参见[给GeminiDB Redis实例节点配置安全组规则](#)。
- 使用非SSL方式连接GeminiDB Redis实例，必须关闭SSL安全连接。如何关闭SSL，请参见[给GeminiDB Redis实例设置SSL数据加密](#)。

前提条件

创建弹性云服务器，以Linux操作系统为例。详情请参见《弹性云服务器快速入门》中[购买弹性云服务器](#)的内容。

操作步骤

步骤1 登录弹性云服务器，详情请参见《弹性云服务器快速入门》中[登录弹性云服务器](#)的内容。

步骤2 获取Redis客户端。

方法一：

通过如下命令，下载Redis客户端。

```
wget --no-check-certificate https://download.redis.io/releases/redis-6.2.0.tar.gz
```

方法二：

下载[Redis客户端安装包](#)，并将Redis客户端安装包上传到弹性云服务器。

步骤3 解压客户端工具包。

```
tar -xzf redis-6.2.0.tar.gz
```

步骤4 在“src”目录下，连接数据库实例。

```
cd redis-6.2.0
make
cd src
./redis-cli -h <DB_HOST> -p <DB_PORT> -a <DB_PWD>
```

示例：

```
./redis-cli -h 192.xx.xx.xx -p 6379 -a <DB_PWD>
```

表 4-21 参数说明

参数	说明
<DB_HOST>	待连接实例的内网IP。 您可以在“实例管理”页面，单击实例名称，进入“基本信息”页面，在左侧导航栏单击“节点管理”，在节点信息列表中获得“内网IP”。 如果您购买的实例有多个节点，选择其中任意一个节点的内网IP即可连接GeminiDB Redis实例。
<DB_PORT>	待连接实例的端口，具体请以实际端口为准。 获取实例端口的方法如下： 在“实例管理”页面，单击实例名称，进入“基本信息”页面，在“连接信息 > 实例端口”处获取当前实例的端口信息。
<DB_PWD>	购买GeminiDB Redis实例时设置的管理员密码。

步骤5 出现如下信息，说明连接成功。

```
IP:port>
```

----结束

4.3.4 通过公网连接 GeminiDB Redis

当不满足通过内网地址访问GeminiDB Redis实例的条件时，可以使用公网访问，建议单独绑定弹性公网IP连接弹性云服务器（或公网主机）与GeminiDB Redis实例。

说明

为了获得更快的传输速率和更高的安全性，建议您将应用迁移到与您的数据库实例在同一VPC子网内，使用[内网连接](#)。

4.3.4.1 通过弹性负载均衡绑定弹性公网 IP 连接实例（推荐）

本章介绍通过创建弹性负载均衡（ELB）并绑定弹性公网IP的方式，使用公网访问GeminiDB Redis实例。生产环境中为避免单点故障，实现流量负载均衡的效果，推荐使用弹性负载均衡绑定弹性公网IP的方式连接GeminiDB Redis实例。

如果您有公网连接GeminiDB Redis实例的需求，为了保障实例可靠性，推荐您使用[通过公网域名连接实例](#)。

使用须知

Cluster集群版实例暂不支持该功能。

新建独享型负载均衡器并进行配置

步骤1 购买独享型负载均衡器，详情参考[创建独享型负载均衡器](#)。需要注意以下事项：

- 选择“规格”时，需要选择网络型（TCP/UDP）。
- 在网络配置中，必须开启‘跨VPC后端’，以支持负载均衡器添加后端IP。
- 需要使用新创建或者已有的“弹性公网IP”，以支持公网访问。

步骤2 在目标负载均衡器的操作列，单击“添加监听器”，详情参考[添加TCP监听器](#)。需要注意以下事项：

图 4-35 添加监控器

名称ID	监控	状态	规格	服务地址与所属网络	监听器 (前端协议/端口)	公网信息	操作
		运行中	网络型(TCP/UDP) 1...	(IPv4私网地址) ddc-st-test-vpc (虚拟私有云)	未添加监听器 去添加	IPv4 5 Mbit/s	添加监听器 更多

- 在“配置监听器”时，前端协议选择TCP，前端端口建议设置为Redis常用端口6379。
- 在配置“添加后端服务器”时，选择“跨VPC后端”，单击“添加跨VPC后端”。在跨VPC后端IP，配置GeminiDB Redis实例的[负载均衡地址](#)、[端口信息](#)。
- “配置健康检查”保持开启即可。

步骤3 如果ELB和GeminiDB Redis在同VPC下，执行步骤[步骤7](#)。否则，执行步骤[步骤4](#)创建对等连接。

步骤4 左侧导航栏单击“虚拟私有云 > 对等连接”进入“对等连接”页面，单击“创建对等连接”，选择“本端VPC”和“对端VPC”。

“本端VPC”选择ELB所在的VPC。

- 若选择的“所属VPC”和GeminiDB Redis在同一个虚拟私有云（VPC）中，“对端VPC”选择任一个非ELB所在的VPC。
- 若选择的“所属VPC”和GeminiDB Redis不在同一个虚拟私有云（VPC）中，“对端VPC”选择GeminiDB Redis实例所在的VPC。

详情参考[创建相同账户下的对等连接](#)。

步骤5 进入对等连接基本信息页面。

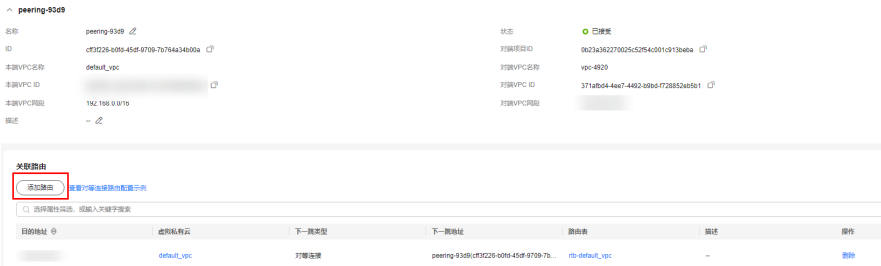
- 在“对等连接创建完成”弹框中，单击“立即添加”，进入对等连接基本信息页面。
- 在“对等连接创建完成”弹框中，单击“暂不添加”，回到“对等连接”页面，单击创建好的对等连接名称，进入对等连接基本信息页面。

图 4-36 对等连接

名称ID	状态	本端VPC	本端VPC网络	对端VPC	对端VPC网络	描述	操作
gemini-3337 7975443-8008a	已接受	ddc-st-vpc	192.168.0.0/16	0023a363277005c525 4c091c913860a	vpc-4523 192.168.0.0/16	--	修改 删除

步骤6 单击“添加路由”，配置对等连接的本端路由、对端路由。

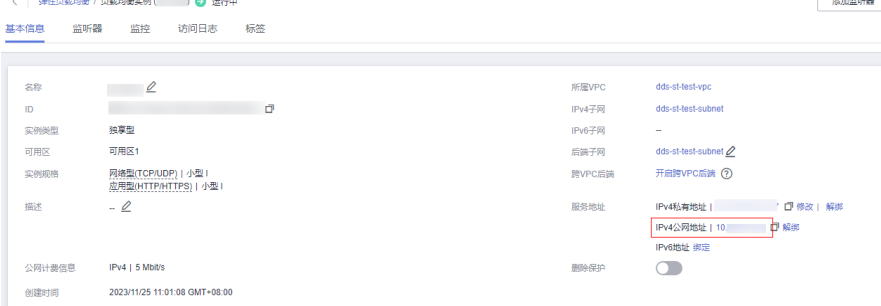
图 4-37 添加路由



- 添加本端路由：在“添加路由”弹框中，“目的地址”输入对等连接中“对端VPC网段”值，“下一跳类型”选择“对等连接”，“下一跳”选择步骤4中创建的对等连接，单击“确定”。
- 添加对端路由：在“添加路由”弹框中，“目的地址”输入对等连接中“本端VPC网段”值，“下一跳类型”选择“对等连接”，“下一跳”选择步骤4中创建的对等连接，单击“确定”。

步骤7 对添加的GeminiDB Redis的负载均衡地址进行健康检查。在左侧导航栏，单击“弹性负载均衡 > 我的ELB”，单击目标负载均衡器名称，进入“监听器”页签，当健康检查结果为“正常”时，表示正常使用。用户可通过“基本信息”页签的“服务地址”访问GeminiDB Redis实例。

图 4-38 服务地址



----结束

连接步骤

- 步骤1** 登录弹性云服务器，详情请参见《弹性云服务器快速入门》中[登录弹性云服务器](#)的内容。
- 步骤2** 获取Redis客户端。
- 方法一：**
- 通过如下命令，下载Redis客户端。
- ```
wget --no-check-certificate http://download.redis.io/releases/redis-6.2.0.tar.gz
```
- 方法二：**
- 下载[Redis客户端安装包](#)，并将Redis客户端安装包上传到弹性云服务器。
- 步骤3** 解压客户端工具包。

```
tar -xzf redis-6.2.0.tar.gz
```

**步骤4** 在“src”目录下，连接数据库实例。

```
cd redis-6.2.0
make
cd src
./redis-cli -h <DB_HOST> -p <DB_PORT> -a <DB_PWD>
```

示例：

```
./redis-cli -h 192.168.0.208 -p 6379 -a <DB_PWD>
```

表 4-22 参数说明

| 参数        | 说明                                                                                                                                                                                                                                                  |
|-----------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <DB_HOST> | 待连接实例的弹性公网IP。<br>您可以在“实例管理”页面，单击实例名称，进入“基本信息”页面，在左侧导航栏单击“节点管理”，在节点信息列表中获得“弹性IP”。<br>如果您购买的实例有多个节点，选择其中任意一个节点的弹性公网IP即可连接GeminiDB Redis实例。<br>如果“节点信息 > 弹性IP”处提示“未绑定”，请您参考 <a href="#">给 GeminiDB Redis实例节点绑定弹性公网IP</a> 为当前实例绑定弹性公网IP后，再根据本章节的操作连接实例。 |
| <DB_PORT> | 待连接实例的端口，具体请以实际端口为准。<br>获取实例端口的方法如下：<br>在“实例管理”页面，单击实例名称，进入“基本信息”页面，在“连接信息 > 实例端口”处获取当前实例的端口信息。                                                                                                                                                     |
| <DB_PWD>  | 购买GeminiDB Redis实例时设置的管理员密码。                                                                                                                                                                                                                        |

**步骤5** 出现如下信息，说明连接成功。

```
IP:port>
```

----结束

#### 4.3.4.2 通过公网域名连接实例

公网域名是指在Internet中，使用公网解析的域名，用于访问网站或Web应用程序。

您可以使用华为云的云解析服务（DNS）提供的公网域名解析功能，将常用的域名（如www.example.com）转换成用于计算机连接的IP地址（如1.2.3.4），这样您就可以通过解析后的IP地址访问GeminiDB Redis实例。

本章节以Linux操作系统为例，介绍使用DNS服务配置的公网域名连接GeminiDB Redis实例的过程。

#### 使用须知

Cluster集群版实例暂不支持该功能。

## 前提条件

- 创建弹性云服务器，以Linux操作系统为例。详情请参见《弹性云服务器快速入门》中[购买弹性云服务器](#)的内容。
- 已注册域名和弹性公网IP。
- GeminiDB Redis实例节点需要绑定弹性公网IP并设置安全组规则，具体操作请参见[给GeminiDB Redis实例节点绑定弹性公网IP](#)和[给GeminiDB Redis实例节点配置安全组规则](#)。

### 说明

GeminiDB Redis实例下存在多个节点，选择任意一个节点绑定弹性公网IP即可。

- 下载[Redis客户端安装包](#)。

## 步骤 1 配置 GeminiDB Redis 实例的公网域名

华为云注册的域名：

通过华为云的[域名注册服务](#)注册的域名无需添加，您可以直接在云解析服务的“域名列表”中查看注册的域名。

**步骤1** 登录[云解析服务DNS控制台](#)。

**步骤2** 在左侧树状导航栏，选择“公网域名”。

图 4-39 公网域名页面



**步骤3** 在“公网域名”页面，您可以查看到注册的域名已经被添加到域名列表中。

**步骤4** 在域名所在行，单击域名的名称，进入“解析记录”页面。

**步骤5** 在“解析记录”页面，可以通过“添加记录集”为域名配置解析记录，详细内容请参见[解析记录简介](#)。

----结束

非华为云注册的域名：

通过第三方域名注册商注册的域名，需要通过“创建公网域名”的操作将域名添加至云解析服务。

**步骤1** 登录[云解析服务DNS控制台](#)。

**步骤2** 在左侧树状导航栏，选择“公网域名”，进入“公网域名”页面。

图 4-40 公网域名页面



- 步骤3 在页面右上角，单击“创建公网域名”。
- 步骤4 在“创建公网域名”页面中，输入域名及相关参数。

图 4-41 创建公网域名

创建公网域名

★ 域名

例如：example.com

★ 企业项目

-请选择-

C ? 新建企业项目

标签

如果您需要使用同一标签标识多种云资源，即所有服务均可在标签输入框下拉选择同一标签，建议在TMS中创建预定义标签。查看预定义标签 C

在下方键/值输入框输入内容后单击添加，即可将标签加入此处

请输入标签键

请输入标签值

添加

您还可以添加10个标签。

描述

0/255

确定

取消

表 4-23 公网域名参数说明

| 参数名称 | 说明                                                                                                                                                                                                                             | 示例          |
|------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------|
| 域名   | 从域名注册商处获得的授权域名。<br>支持添加主域名及主域名的子域名，即最多支持添加二级域名，例如： <ul style="list-style-type: none"><li>example.com的子域名<br/>abc.example.com ↵</li><li>example.com.cn的子域名<br/>abc.example.com.cn。</li></ul> 域名的格式请参见 <a href="#">域名格式与级别</a> 。 | example.com |

文档版本 01 (2025-11-13)

版权所有 © 华为云计算技术有限公司

118

| 参数名称 | 说明                                                                                                                                                                                                                                                                                                                                                                  | 示例                             |
|------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------|
| 企业项目 | 公网域名关联的企业项目，用于将公网域名资源按照企业项目进行管理。<br><b>说明</b><br>仅当用户使用的“账号类型”为“企业账号”时，显示该参数，且参数必选。<br><b>配置原则：</b> <ul style="list-style-type: none"><li>• 如果不通过企业项目管理域名资源，则采用默认值“default”。</li><li>• 如果通过企业项目管理域名资源，则在下拉列表中选择已经创建的企业项目。</li></ul>                                                                                                                                 | default                        |
| 标签   | 可选参数。域名的标识，包括键和值，每个域名可以创建20个标签。<br>键和值的命名规则如下：<br><b>键：</b> <ul style="list-style-type: none"><li>• 不能为空。</li><li>• 对于同一资源键值唯一。</li><li>• 长度不超过128个字符。</li><li>• 标签的键可以包含任意语种的字母、数字和空格，以及_!:=+~@字符，但首尾不能包含空格，且不能以_sys_开头。</li></ul> <b>值：</b> <ul style="list-style-type: none"><li>• 长度不超过255个字符。</li><li>• 标签的值可以包含任意语种的字母、数字和空格，以及_!:=+~@字符，但首尾不能包含空格。</li></ul> | example_key1<br>example_value1 |
| 描述   | 可选参数。域名的描述信息，长度不超过255个字符。                                                                                                                                                                                                                                                                                                                                           | This is a zone example.        |

**步骤5** 单击“确定”。

创建完成后，您可以在“公网域名”页面的域名列表中查看新创建的域名信息。

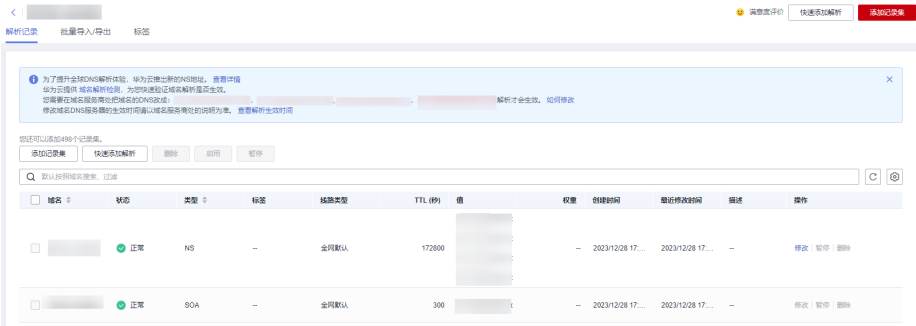
----结束

**为域名添加解析记录集：**

公网域名创建成功后，需要为公网域名设置解析记录集，以便使用域名来访问实例。以下操作以添加A类型记录集为例，其他更多类型记录集请参见[添加公网域名解析记录](#)。

**步骤1** 单击新建的公网域名名称，进入“解析记录”页面，单击右上方“添加记录集”。

图 4-42 添加记录集



步骤2 在“添加记录集”弹出框中，根据界面提示填写记录集参数。

图 4-43 添加解析记录集

添加记录集

主机记录

1234456.com. ?

\* 类型

A – 将域名指向IPv4地址

\* 别名

☐ 是 ☒ 否 ?

\* 线路类型

全网默认 ?

\* TTL (秒)

300

5分钟

1小时

12小时

1天

 ?

\* 值

例:  
192.168.10.10

 ?

权重

1

 ?

标签

如果您需要使用同一标签标识多种云资源，即所有服务均可在标签输入框下拉选择同一标签，建议在TMS中创建预定义标签。 [查看预定义标签](#) C

在下方键/值输入框输入内容后单击“添加”，即可将标签加入此处

请输入标签键

请输入标签值

添加

您还可以添加10个标签。

描述

 0/255

确定

取消

文档版本 01 (2025-11-13)

版权所有 © 华为云计算技术有限公司

120

参数配置说明请参见[添加公网域名解析记录](#)。

- 步骤3 单击“确定”。
- 步骤4 返回“解析记录”页面。
- 步骤5 添加完成后，您可以在域名对应的记录集列表中查看已添加的记录集。当记录集的状态显示为“正常”时，表示记录集添加成功。

----结束

步骤 2 登录弹性云服务器并通过 Redis 客户端连接实例

- 步骤1 登录弹性云服务器，详情请参见《弹性云服务器快速入门》中[登录弹性云服务器](#)的内容。
- 步骤2 获取Redis客户端。

方法一：

通过如下命令，下载Redis客户端。

```
wget http://download.redis.io/releases/redis-6.2.0.tar.gz
```

方法二：

通过[前提条件](#)中的地址下载Redis客户端，并将Redis客户端安装包上传到弹性云服务器。

- 步骤3 解压客户端工具包。

```
tar -xzf redis-6.2.0.tar.gz
```

- 步骤4 在“src”目录下，连接数据库实例。

```
cd redis-6.2.0
make
cd src
./redis-cli -h <DB_Domain_Name> -p <DB_PORT> -a <DB_PWD>
```

示例：

```
./redis-cli -h redis.com -p 6379 -a <DB_PWD>
```

表 4-24 参数说明

| 参数               | 说明                                                                                              |
|------------------|-------------------------------------------------------------------------------------------------|
| <DB_Domain_Name> | 待连接实例的公网域名，公网域名为 <a href="#">步骤1 配置GeminiDB Redis实例的公网域名</a> 中创建的域名。                            |
| <DB_PORT>        | 待连接实例的端口，具体请以实际端口为准。<br>获取实例端口的方法如下：<br>在“实例管理”页面，单击实例名称，进入“基本信息”页面，在“连接信息 > 实例端口”处获取当前实例的端口信息。 |
| <DB_PWD>         | 购买GeminiDB Redis实例时设置的管理员密码。                                                                    |

**步骤5** 出现如下信息，说明连接成功。

```
Domain_Name:port>
```

----结束

#### 4.3.4.3 通过弹性公网 IP 连接实例

GeminiDB Redis支持使用弹性云服务器或本地设备，通过公网连接GeminiDB Redis实例。

本章节以Linux系统的弹性云服务器为例，指导您通过Redis-cli客户端连接GeminiDB Redis实例。生产环境中为避免单点故障，实现流量负载均衡的效果，推荐使用[通过弹性负载均衡绑定弹性公网IP连接实例](#)的方式连接GeminiDB Redis实例。

您可以根据业务需求选用非SSL方式或SSL方式连接实例，其中，SSL连接实现了数据加密功能，具有更高的安全性，详情请参见[通过SSL方式连接GeminiDB Redis实例](#)。本章节中主要介绍非SSL方式连接GeminiDB Redis实例的步骤。

#### 使用须知

- 使用非SSL方式连接GeminiDB Redis实例，必须关闭SSL安全连接。如何关闭SSL，请参见[给GeminiDB Redis实例设置SSL数据加密](#)。
- 用户需要评估业务所需带宽，购买带宽资源足够的弹性公网IP，**因公网网络性能造成的客户端访问异常不计入SLA**。
- Cluster集群版实例暂不支持该功能。

#### 前提条件

1. 创建弹性云服务器，以Linux操作系统为例。详情请参见《弹性云服务器快速入门》中[购买弹性云服务器](#)的内容。
2. GeminiDB Redis实例节点需要绑定弹性公网IP并设置安全组规则，具体操作请参见[给GeminiDB Redis实例节点绑定弹性公网IP](#)和[给GeminiDB Redis实例节点配置安全组规则](#)。

#### 说明

GeminiDB Redis实例下存在多个节点，选择任意一个节点绑定弹性公网IP即可。

#### 操作步骤

**步骤1** 登录弹性云服务器，详情请参见《弹性云服务器快速入门》中[登录弹性云服务器](#)的内容。

**步骤2** 获取Redis客户端。

**方法一：**

通过如下命令，下载Redis客户端。

```
wget http://download.redis.io/releases/redis-6.2.0.tar.gz
```

**方法二：**

下载[Redis客户端安装包](#)，并将Redis客户端安装包上传到弹性云服务器。



**步骤3** 解压客户端工具包。

```
tar -xzf redis-6.2.0.tar.gz
```

**步骤4** 在“src”目录下，连接数据库实例。

```
cd redis-6.2.0
make
cd src
./redis-cli -h <DB_HOST> -p <DB_PORT> -a <DB_PWD>
```

示例：

```
./redis-cli -h 192.168.0.208 -p 6379 -a <DB_PWD>
```

表 4-25 参数说明

| 参数        | 说明                                                                                                                                                                                                                                                 |
|-----------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <DB_HOST> | 待连接实例的弹性公网IP。<br>您可以在“实例管理”页面，单击实例名称，进入“基本信息”页面，在左侧导航栏单击“节点管理”，在节点信息列表中获取“弹性IP”。<br>如果您购买的实例有多个节点，选择其中任意一个节点的弹性公网IP即可连接GeminiDB Redis实例。<br>如果“节点信息 > 弹性IP”处提示“未绑定”，请您参考 <a href="#">给GeminiDB Redis实例节点绑定弹性公网IP</a> 为当前实例绑定弹性公网IP后，再根据本章节的操作连接实例。 |
| <DB_PORT> | 待连接实例的端口，具体请以实际端口为准。<br>获取实例端口的方法如下：<br>在“实例管理”页面，单击实例名称，进入“基本信息”页面，在“连接信息 > 实例端口”处获取当前实例的端口信息。                                                                                                                                                    |
| <DB_PWD>  | 购买GeminiDB Redis实例时设置的管理员密码。                                                                                                                                                                                                                       |

**步骤5** 出现如下信息，说明连接成功。

```
IP:port>
```

----结束

## 4.3.5 通过程序代码连接实例示例

### 4.3.5.1 通过 Jedis 连接实例

本章节介绍使用Jedis（JAVA客户端）访问GeminiDB Redis实例的方法。

由于GeminiDB Redis的“Proxy集群”架构提供了统一的负载均衡地址，并提供高可用能力，因此推荐使用JedisPool单机模式轻松接入。

此外，GeminiDB Redis对JedisSentinelPool、JedisCluster两种模式也提供支持。

### 前提条件

- 已成功创建GeminiDB Redis实例，且实例状态正常。创建GeminiDB Redis实例的方法请参见[购买GeminiDB Redis集群版实例](#)。

- 已创建弹性云服务器，创建弹性云服务器的方法，请参见《弹性云服务器快速入门》中“[购买弹性云服务器](#)”章节。
- 弹性云服务器上已经安装GCC等编译工具。
- 创建的弹性云服务器与GeminiDB Redis实例要保证区域、可用区、VPC和安全组一致。

## pom 文件依赖

```
<dependency>
 <groupId>redis.clients</groupId>
 <artifactId>jedis</artifactId>
 <version>4.3.2</version>
</dependency>
<dependency>
 <groupId>org.springframework.boot</groupId>
 <artifactId>spring-boot-starter-data-redis</artifactId>
 <version>2.3.6.RELEASE</version>
</dependency>
```

## 使用 JedisPool 访问（推荐）

### 样例代码

```
import org.apache.commons.pool2.impl.GenericObjectPoolConfig;
import redis.clients.jedis.Jedis;
import redis.clients.jedis.JedisPool;

public class JedisPoolTests {

 private static void testPool() {
 // 认证用的用户名和密码直接写到代码中有很大的安全风险，建议在配置文件或者环境变量中存放(密码应密文存放、使用时解密)，确保安全；
 // 本示例以用户名和密码保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量(环境变量名称请根据自身情况进行设置)EXAMPLE_USERNAME_ENV和EXAMPLE_PASSWORD_ENV。
 String pwd = System.getenv("EXAMPLE_PASSWORD_ENV");
 JedisPool pool = new JedisPool(new GenericObjectPoolConfig(), "172.xx.xx.xx", 6379,
 2000, pwd);
 Jedis jedis = pool.getResource();
 try {
 System.out.println(jedis.hgetAll("676296"));
 System.out.println(jedis.set("key1", "value1"));
 } finally {
 jedis.close();
 }
 pool.destroy();
 }

 public static void main(String[] args) {
 testPool();
 }
}
```

- 上述代码中“172.xx.xx.xx”为待连接GeminiDB Redis实例的负载均衡地址。您可以单击实例名称，进入“基本信息”页面，在网络信息区域获取“负载均衡地址”。

图 4-44 查看负载均衡地址



- 上述代码中的“6379”为待连接GeminiDB Redis实例的端口，具体请以实际端口为准。获取端口的方法请参见[查看GeminiDB Redis IP地址和端口](#)。
- GeminiDB Redis支持与限制的命令，详情请参见[开发运维规范](#)。
- 由于GeminiDB Redis和开源Redis集群计算hash算法不同，因此GeminiDB Redis的部分命令需要给对应的key打上hashtag，否则会出现非预期的行为。hashtag使用建议参考[开发运维规范](#)。

## 使用 JedisCluster 访问

### 样例代码

```
import org.apache.commons.pool2.impl.GenericObjectPoolConfig;
import redis.clients.jedis.HostAndPort;
import redis.clients.jedis.JedisCluster;

public class ClusterTests {

 private static void testCluster() {
 String pwd = "a";
 JedisCluster cluster = new JedisCluster(new HostAndPort("172.xx.xx.xx", 6379),
 200, 2000, 5, pwd, new GenericObjectPoolConfig());
 System.out.println(cluster.hgetAll("676296"));
 System.out.println(cluster.set("key1", "value1"));
 }

 public static void main(String[] args) {
 testCluster();
 }
}
```

- 上述代码中“172.xx.xx.xx”为待连接GeminiDB Redis实例的负载均衡地址。您可以单击实例名称，进入“基本信息”页面，在网络信息区域获取“负载均衡地址”。

图 4-45 查看负载均衡地址



- 上述代码中的“6379”为待连接GeminiDB Redis实例的端口，具体请以实际端口为准。获取端口的方法请参见[查看GeminiDB Redis IP地址和端口](#)。
- GeminiDB Redis支持与限制的命令，详情请参见[开发运维规范](#)。
- 由于GeminiDB Redis和开源Redis集群计算hash算法不同，因此GeminiDB Redis的部分命令需要给对应的key打上hashtag，否则会出现非预期的行为。hashtag使用建议参考[开发运维规范](#)。

## 使用 JedisSentinelPool 访问

### 样例代码

```
import org.apache.commons.pool2.impl.GenericObjectPoolConfig;
import redis.clients.jedis.Jedis;
import redis.clients.jedis.JedisSentinelPool;
import java.util.HashSet;
import java.util.Set;
```

```
public void SentinelTest {
 public static void main(String[] args) {
 GenericObjectPoolConfig<Jedis> config = new GenericObjectPoolConfig<Jedis>();
 Set<String> mySentinels = new HashSet<String>();
 mySentinels.add("172.xx.xx.xx:6379");
 JedisSentinelPool pool = new JedisSentinelPool(master-name, mySentinels, config, 1000, password, 0);
 Jedis jedis = pool.getResource();
 jedis.auth(password);
 jedis.set("foo", "bar");
 String s = jedis.get("foo");
 System.out.println(s);
 jedis.close();
 pool.close();
 }
}
```

- 上述代码中“172.xx.xx.xx”为待连接GeminiDB Redis实例的负载均衡地址。您可以单击实例名称，进入“基本信息”页面，在网络信息区域获取“负载均衡地址”。

图 4-46 查看负载均衡地址



- 上述代码中的“6379”为待连接GeminiDB Redis实例的端口，具体请以实际端口为准。获取端口的方法请参见[查看GeminiDB Redis IP地址和端口](#)。
- 上述代码中的master-name只能是“mymaster”。
- GeminiDB Redis支持与限制的命令，详情请参见[开发运维规范](#)。
- 由于GeminiDB Redis和开源Redis集群计算hash算法不同，因此GeminiDB Redis的部分命令需要给对应的key打上hashtag，否则会出现非预期的行为。hashtag使用建议参考[开发运维规范](#)。

### 4.3.5.2 通过 Redisson 连接实例

本章节介绍使用Java的开源库Redisson访问GeminiDB Redis实例的方法，主要涉及单机模式和sentinel模式连接。

#### 前提条件

- 已成功创建GeminiDB Redis实例，且实例状态正常。创建GeminiDB Redis实例的方法请参见[购买GeminiDB Redis集群版实例](#)。
- 已创建弹性云服务器，创建弹性云服务器的方法，请参见《弹性云服务器快速入门》中“[购买弹性云服务器](#)”章节。
- 弹性云服务器上已经安装GCC等编译工具。
- 创建的弹性云服务器与GeminiDB Redis实例要保证区域、可用区、VPC和安全组一致。

#### 使用 SingleServer 模式连接实例

样例代码：

```
import org.redisson.Redisson;
import org.redisson.api.RedissonClient;
import org.redisson.config.Config;

public class SingleServerTests {

 private static void testSingleServer() {
 Config config = new Config();
 // 认证用的用户名和密码直接写到代码中有很大的安全风险，建议在配置文件或者环境变量中存放(密码应密文存放、使用时解密)，确保安全；
 // 本示例以用户名和密码保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量(环境变量名称请根据自身情况进行设置)EXAMPLE_USERNAME_ENV和EXAMPLE_PASSWORD_ENV。
 String password = System.getenv("EXAMPLE_PASSWORD_ENV");
 config.useSingleServer().setAddress("redis://172.xx.xx.xx:6379")
 .setPassword(password);
 RedissonClient redisson = Redisson.create(config);
 execute(redisson); // send requests to database
 redisson.shutdown();
 }

 public static void main(String[] args) {
 testSingleServer();
 }
}
```

- 上述代码中“172.xx.xx.xx”为待连接GeminiDB Redis实例的负载均衡地址。您可以单击实例名称，进入“基本信息”页面，在网络信息区域获取“负载均衡地址”。

图 4-47 查看负载均衡地址



- 上述代码中的“6379”为待连接GeminiDB Redis实例的端口，具体请以实际端口为准。获取端口的方法请参见[查看GeminiDB Redis IP地址和端口](#)。
- GeminiDB Redis支持与限制的命令，详情请参见[开发运维规范](#)。
- 由于GeminiDB Redis和开源Redis集群计算hash算法不同，因此GeminiDB Redis的部分命令需要给对应的key打上hashtag，否则会出现非预期的行为。hashtag使用建议参考[开发运维规范](#)。

## 使用 Sentinel 模式连接实例

### 样例代码：

```
import org.redisson.Redisson;
import org.redisson.api.RedissonClient;
import org.redisson.config.Config;
import static org.redisson.config.ReadMode.MASTER;

public class SingleServerTests {

 public static void testSentinel() {
 Config config = new Config();
 // 认证用的用户名和密码直接写到代码中有很大的安全风险，建议在配置文件或者环境变量中存放(密码应密文存放、使用时解密)，确保安全；
 // 本示例以用户名和密码保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量(环境变量名称请根据自身情况进行设置)EXAMPLE_USERNAME_ENV和EXAMPLE_PASSWORD_ENV。
 String password = System.getenv("EXAMPLE_PASSWORD_ENV");
```

```
config.useSentinelServers()
 .setMasterName(master_name)
 .setCheckSentinelsList(false)
 .setReadMode(MASTER)
 .setPassword(password)
 .addSentinelAddress("redis://172.xx.xx.xx:6379");
RedissonClient redisson = Redisson.create(config);
execute(redisson); // send requests to database
redisson.shutdown();
}

public static void main(String[] args) {
 testSentinel();
}
```

- 上述代码中“172.xx.xx.xx”为待连接GeminiDB Redis实例的负载均衡地址。您可以单击实例名称，进入“基本信息”页面，在网络信息区域获取“负载均衡地址”。

图 4-48 查看负载均衡地址



- 上述代码中的“6379”为待连接GeminiDB Redis实例的端口，具体请以实际端口为准。获取端口的方法请参见[查看GeminiDB Redis IP地址和端口](#)。
- GeminiDB Redis只是实现了Sentinel模式的接入方式，不采用原生sentinel的可用性功能。所以样例代码中，master\_name固定为“mymaster”，必须设置“CheckSentinelsList”为“false”，必须设置ReadMode为MASTER。
- GeminiDB Redis支持与限制的命令，详情请参见[开发运维规范](#)。
- 由于GeminiDB Redis和开源Redis集群计算hash算法不同，因此GeminiDB Redis的部分命令需要给对应的key打上hashtag，否则会出现非预期的行为。hashtag使用建议参考[开发运维规范](#)。

## 使用 ClusterServer 模式连接实例

### 样例代码：

```
import org.redisson.Redisson;
import org.redisson.api.RedissonClient;
import org.redisson.config.Config;

public class ClusterServerTests {

 private static void testClusterServer() {
 Config config = new Config();
 config.useClusterServers()
 .addNodeAddress("redis://172.xx.xx.xx:6379")
 .setPassword(password);

 RedissonClient redisson = Redisson.create(config);
 execute(redisson); // send requests to database
 redisson.shutdown();
 }

 public static void main(String[] args) {
 testClusterServer();
 }
}
```

```
}
}
```

- 上述代码中“172.xx.xx.xx”为待连接GeminiDB Redis实例的负载均衡地址。您可以单击实例名称，进入“基本信息”页面，在网络信息区域获取“负载均衡地址”。

图 4-49 查看负载均衡地址



- 上述代码中的“6379”为待连接GeminiDB Redis实例的端口，具体请以实际端口为准。获取端口的方法请参见[查看GeminiDB Redis IP地址和端口](#)。
- GeminiDB Redis支持与限制的命令，详情请参见[开发运维规范](#)。
- 由于GeminiDB Redis和开源Redis集群计算hash算法不同，因此GeminiDB Redis的部分命令需要给对应的key打上hashtag，否则会出现非预期的行为。hashtag使用建议参考[开发运维规范](#)。

### 4.3.5.3 通过 Hiredis 连接实例

本章节主要介绍使用Hiredis访问GeminiDB Redis实例的方法。

#### 前提条件

- 已成功创建GeminiDB Redis实例，且实例状态正常。创建GeminiDB Redis实例的方法请参见[购买GeminiDB Redis集群版实例](#)。
- 已创建弹性云服务器，创建弹性云服务器的方法，请参见《弹性云服务器快速入门》中“[购买弹性云服务器](#)”章节。
- 弹性云服务器上已经安装GCC等编译工具。
- 创建的弹性云服务器与GeminiDB Redis实例要保证区域、可用区、VPC和安全组一致。

#### 操作步骤

**步骤1** 获取GeminiDB Redis实例的负载均衡地址和端口。

- 负载均衡地址的查看方法请参见[查看负载均衡地址及端口](#)。
- 端口的获取方法请参见[查看实例节点端口](#)。

**步骤2** 登录弹性云服务器，具体操作请参见《弹性云服务器快速入门》中“[登录弹性云服务器](#)”。

**步骤3** 使用如下命令，下载并解压Hiredis。

```
wget https://github.com/redis/hiredis/archive/master.zip
```

**步骤4** 进入到解压目录后编译安装Hiredis。

```
make
make install
```

**步骤5** 编写测试代码connRedisTst.cc。 **说明**

关于Hiredis的详细用法，请参考[redis官网](#)的使用介绍。

**代码如下：**

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <hiredis.h>
int main(int argc, char **argv) {
 unsigned int j;
 redisContext *conn;
 redisReply *reply;
 if (argc < 3) {
 printf("Usage: example {instance_ip_address} 6379 {password}\n");
 exit(0);
 }
 const char *hostname = argv[1];
 const int port = atoi(argv[2]);
 const char *password = argv[3];
 struct timeval timeout = { 1, 500000 }; // 1.5 seconds
 conn = redisConnectWithTimeout(hostname, port, timeout);
 if (conn == NULL || conn->err) {
 if (conn) {
 printf("Connection error: %s\n", conn->errstr);
 redisFree(conn);
 } else {
 printf("Connection error: can't allocate redis context\n");
 }
 exit(1);
 }
 /* AUTH */
 reply = redisCommand(conn, "AUTH %s", password);
 printf("AUTH: %s\n", reply->str);
 freeReplyObject(reply);

 /* Set */
 reply = redisCommand(conn, "SET %s %s", "key", "hiredis test ok!");
 printf("SET: %s\n", reply->str);
 freeReplyObject(reply);

 /* Get */
 reply = redisCommand(conn, "GET key");
 printf("GET key: %s\n", reply->str);
 freeReplyObject(reply);

 /* Disconnects and frees the context */
 redisFree(conn);
 return 0;
}
```

**步骤6** 执行如下命令进行编译。

```
gcc connRedis.c -o connRedis -I /usr/local/include/hiredis -lhiredis
```

如果有报错，可查找hiredis.h文件路径，并修改编译命令。

编译完后得到一个可执行文件connRedis。

**步骤7** 执行如下命令，连接GeminiDB Redis实例。

```
export LD_LIBRARY_PATH=/usr/local/lib/:$LD_LIBRARY_PATH
```

```
./connRedis <redis_ip_address> 6379 <password>
```

其中，以下信息需按照实际情况进行替换：



- <redis\_ip\_address>为[步骤1](#)中获取到的GeminiDB Redis实例的负载均衡地址。
- “6379”为GeminiDB Redis实例的端口。
- <password>为创建GeminiDB Redis实例时自定义的密码。

**步骤8** 返回如下回显信息，表示成功连接GeminiDB Redis实例。

```
AUTH: OK
SET: OK
GET key: Hello, hiredis test ok!
```

----结束

#### 4.3.5.4 通过 NodeJs 连接实例

本章节主要介绍使用NodeJs访问GeminiDB Redis实例的方法。

##### 前提条件

- 已成功创建GeminiDB Redis实例，且实例状态为“正常”。
- 已创建弹性云服务器，创建弹性云服务器的方法，请参见《弹性云服务器快速入门》中“[购买弹性云服务器](#)”章节。
- 如果弹性云服务器为Linux系统，该弹性云服务器上必须已经安装GCC等编译工具。
- 创建的弹性云服务器与GeminiDB Redis实例要保证区域、可用区、VPC和安全组一致。

##### 操作步骤

**步骤1** 获取GeminiDB Redis实例的负载均衡地址和端口。

- 负载均衡地址的查看方法请参见[查看负载均衡地址及端口](#)。
- 端口信息的获取方法请参见[查看实例节点端口](#)。

**步骤2** 登录弹性云服务器，具体操作请参见《弹性云服务器快速入门》中“[登录弹性云服务器](#)”。

**步骤3** 安装NodeJs。

- **方法一：**使用如下命令安装NodeJs。

```
yum install nodejs
```

##### 说明

以CentOS([redhat系列](#))为例，如果是Ubuntu([debian系列](#))，则需要使用其对应的安装命令。

- **方法二：**如果方法一安装不了，也可以通过如下方式进行安装。

```
wget https://nodejs.org/dist/v0.12.4/node-v0.12.4.tar.gz --no-check-certificate ;
```

```
tar -xvf node-v0.12.4.tar.gz;
```

```
cd node-v0.12.4;
```

```
./configure;
```

```
make;
```

```
make install;
```

 说明

以CentOS(**redhat系列**)为例, 如果是Ubuntu(**debian系列**), 则需要使用其对应的安装命令。

**步骤4** NodeJs安装完成后, 可执行如下命令, 查看其版本号, 确认NodeJs已安装成功。

**node -v**

**步骤5** 安装JS包管理工具npm。

**yum install npm**

**步骤6** 安装NodeJs redis客户端ioredis。

**npm install ioredis**

**步骤7** 编辑连接GeminiDB Redis实例的示例脚本。

- 使用NodeJs单机SDK连接GeminiDB Redis集群

```
var Redis = require('ioredis');
// 认证用的用户名和密码直接写到代码中有很大的安全风险, 建议在配置文件或者环境变量中存放(密码应
// 密文存放、使用时解密), 确保安全;
// 本示例以用户名和密码保存在环境变量中为例, 运行本示例前请先在本地环境中设置环境变量(环境变量
// 名称请根据自身情况进行设置)EXAMPLE_USERNAME_ENV和EXAMPLE_PASSWORD_ENV。
var pwd = process.env.EXAMPLE_PASSWORD_ENV;
var redis = new Redis({
 port: 6379, // 此处为步骤1中获取到的GeminiDB Redis实例的端口。
 host: '192.xx.xx.xx', //此处为步骤1中获取到的GeminiDB Redis实例的负载均衡地址。
 family: 4, // 此处填写4表示IPv4, 6表示IPv6。
 password: pwd,
 db: 0
});
redis.set('key', 'Nodejs tst ok!');
redis.get('key', function (err, result) {
 console.log(result);
});
```

- 使用NodeJs集群SDK连接GeminiDB Redis集群

```
const Redis = require("ioredis");
// 认证用的用户名和密码直接写到代码中有很大的安全风险, 建议在配置文件或者环境变量中存放(密码应
// 密文存放、使用时解密), 确保安全;
// 本示例以用户名和密码保存在环境变量中为例, 运行本示例前请先在本地环境中设置环境变量(环境变量
// 名称请根据自身情况进行设置)EXAMPLE_USERNAME_ENV和EXAMPLE_PASSWORD_ENV。
var pwd = process.env.EXAMPLE_PASSWORD_ENV;
const cluster = new Redis.Cluster([
 {
 port: 6379, // 此处为步骤1中获取到的GeminiDB Redis实例的端口。
 host: '192.xx.xx.xx', //此处为步骤1中获取到的GeminiDB Redis实例的负载均衡地址。
 family: 4, // 此处填写4表示IPv4, 6表示IPv6。
 password: pwd,
 db: 0
 },
]);
cluster.set("foo", "nodejs is ok!");
cluster.get("foo", (err, res) => {
 console.log(res);
});
```

**步骤8** 运行示例脚本, 确认结果正常。

**node ioredisdemo.js**

**----结束**

### 4.3.5.5 通过 PHP 连接实例

本章节主要介绍使用PHP访问GeminiDB Redis实例的方法。

#### 前提条件

- 已成功创建GeminiDB Redis实例，且实例状态为“正常”。
- 已创建弹性云服务器，创建弹性云服务器的方法，请参见《弹性云服务器快速入门》中“[购买弹性云服务器](#)”章节。
- 弹性云服务器上已经安装GCC等编译工具。
- 创建的弹性云服务器与GeminiDB Redis实例要保证区域、可用区、VPC和安全组一致。

#### 操作步骤

**步骤1** 获取GeminiDB Redis实例的负载均衡地址和端口。

- 负载均衡地址的查看方法请参见[查看负载均衡地址及端口](#)。
- 端口信息的获取方法请参见[查看实例节点端口](#)。

**步骤2** 登录弹性云服务器，具体操作请参见《弹性云服务器快速入门》中“[登录弹性云服务器](#)”。

**步骤3** 安装PHP开发包与命令行工具。

执行如下命令，使用yum方式直接安装。

```
yum install php-devel php-common php-cli
```

#### 说明

以CentOS([redhat系列](#))为例，如果是Ubuntu([debian系列](#))，则需要使用其对应的安装命令。

**步骤4** 安装完后可查看版本号，确认成功安装。

```
php --version
```

**步骤5** 安装Redis的PHP客户端。

1. 下载phpredis源文件。

```
wget http://pecl.php.net/get/redis-4.1.0RC3.tgz
```

#### 说明

以上是当前最新版本，也可以通过[PHP官网](#)下载其他版本的phpredis客户端。

2. 解压phpredis源文件包。

```
tar -zxvf redis-4.1.0RC3.tgz
```

```
cd redis-4.1.0RC3
```

3. 编译前先执行扩展命令。

```
phpize
```

4. 配置php-config文件。

```
./configure --with-php-config=/usr/bin/php-config
```

 说明

不同的操作系统，安装PHP的方式不同，该配置文件位置可能不同。建议在配置前，先确认该文件的目录，命令如下：**find / -name php.ini**。

5. 编译和安装phpredis客户端。

**make && make install**

6. 安装完后在php.ini文件中增加extension配置项，用于增加redis模块的引用配置。使用如下命令，查找到php.ini文件：

**vim /usr/local/php/etc/php.ini**

在php.ini文件中，增加如下配置项：

```
extension = "/usr/lib64/php/modules/redis.so"
```

 说明

php.ini和redis.so两个文件的目录可能不同，可通过如下命令先查找确认。

**find / -name php.ini**

**find / -name redis.so**

7. 保存退出后确认扩展生效。

**php -m |grep redis**

如果以上命令返回了redis，表示php redis客户端环境搭建好了。

## 步骤6 使用phpredis客户端连接GeminiDB Redis实例。

1. 编写测试代码redis.php。

- 使用PHP单机SDK连接GeminiDB Redis集群

```
<?php
// 认证用的用户名和密码直接写到代码中有很大的安全风险，建议在配置文件或者环境变量中存放(密码应密文存放、使用时解密)，确保安全；
// 本示例以用户名和密码保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量(环境变量名称请根据自身情况进行设置)EXAMPLE_USERNAME_ENV和EXAMPLE_PASSWORD_ENV。
$pwd = getenv('EXAMPLE_PASSWORD_ENV');
$redis_host = "192.xx.xx.xx"; //此处为步骤1获取到的负载均衡地址
$redis_port = 6379;
$user_pwd = $pwd;
$redis = new Redis();
if ($redis->connect($redis_host, $redis_port) == false) {
 die($redis->getLastError());
}
if ($redis->auth($user_pwd) == false) {
 die($redis->getLastError());
}
if ($redis->set("key", "php test ok!") == false) {
 die($redis->getLastError());
}
$value = $redis->get("key");
echo $value;
$redis->close();
?>
```

- 使用PHP集群SDK连接GeminiDB Redis集群

```
<?php
$redis_host = "192.xx.xx.xx"; //此处为步骤1获取到的负载均衡地址
$redis_port = 6379;
$user_pwd = "pwd";
// Connect with read/write timeout as well as specify that phpredis should use
// persistent connections to each node.
$redis = new RedisCluster(NULL, Array("$redis_host:$redis_port"), 1.5, 1.5, true,
$user_pwd);
if ($redis->set("key", "php test ok!") == false) {
```

```
die($redis->getLastError());
}
$value = $redis->get("key");
echo $value;
$redis->close();
?>
```

2. 执行redis.php，确认结果正常。

----结束

### 4.3.5.6 通过 Python 连接实例

本章节主要介绍使用Python访问GeminiDB Redis实例的方法。

#### 前提条件

- 已成功创建GeminiDB Redis实例，且实例状态为“正常”。
- 已创建弹性云服务器，创建弹性云服务器的方法，请参见《弹性云服务器快速入门》中“[购买弹性云服务器](#)”章节。
- 弹性云服务器上已经安装GCC等编译工具。
- 创建的弹性云服务器与GeminiDB Redis实例要保证区域、可用区、VPC和安全组一致。

#### 操作步骤

**步骤1** 获取GeminiDB Redis实例的负载均衡地址和端口。

- 负载均衡地址的查看方法请参见[查看负载均衡地址及端口](#)。
- 端口信息的获取方法请参见[查看实例节点端口](#)。

**步骤2** 登录弹性云服务器，具体操作请参见《弹性云服务器快速入门》中“[登录弹性云服务器](#)”。

**步骤3** 安装Python和Redis的Python客户端Redis-py。

1. 如果系统没有自带Python，可以使用yum方式安装。

```
yum install python
```

2. 下载并解压redis-py。

```
wget https://github.com/andymccurdy/redis-py/archive/master.zip
```

3. 进入到解压目录后安装Redis的Python客户端Redis-py。

```
python setup.py install
```

4. 安装后执行python命令，返回如下信息说明成功安装Redis-py：

```
Python 2.6.6 (r266:84292, Aug 18 2016, 15:13:37)
[GCC 4.4.7 20120313 (Red Hat 4.4.7-17)] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> import redis
>>>
```

**步骤4** 使用Redis-py客户端连接GeminiDB Redis实例。

以下步骤以命令行模式进行示例（也可以将命令写入python脚本中再执行）：

- **使用单机方式连接GeminiDB Redis集群**

- a. 执行python命令，进入命令行模式。

返回如下信息说明已进入命令行模式：

```
Python 2.6.6 (r266:84292, Aug 18 2016, 15:13:37)
[GCC 4.4.7 20120313 (Red Hat 4.4.7-17)] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> import redis
>>>
```

- b. 在命令行中执行以下命令，确认结果正常。

```
>>> r = redis.StrictRedis(host='192.xx.xx.xx', port=6379, password='pwd');
>>> r.set('key', 'Python tst ok!')
True
>>> r.get('key')
'Python tst ok!'
```

#### 说明

如下信息请按照实际值进行修改后，再执行上述命令。

- host和port为[步骤1](#)中获取到的GeminiDB Redis实例的负载均衡地址和端口。
  - password为GeminiDB Redis实例的密码。
- **使用集群方式连接GeminiDB Redis集群**

使用前需要设置：config set CompatibleMode ClusterClient

```
Python 3.7.4 (default, Jan 30 2021, 09:00:44)
[GCC 7.3.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> from redis.cluster import RedisCluster as Redis
>>> rc = Redis(host='127.0.0.1', port=6379, password='a')
>>> rc.set('key', 'Python test ok!')
True
>>> rc.get('key')
b'Python test ok!'
```

----结束

### 4.3.5.7 通过 Go 语言连接实例

本章节主要介绍使用Go语言访问GeminiDB Redis实例的方法。

#### 前提条件

- 已成功创建GeminiDB Redis实例，且实例状态为“正常”。
- 已创建弹性云服务器，创建弹性云服务器的方法，请参见《弹性云服务器快速入门》中“[购买弹性云服务器](#)”章节。
- 弹性云服务器上已经安装GCC等编译工具。
- 创建的弹性云服务器与GeminiDB Redis实例要保证区域、可用区、VPC和安全组一致。

#### 操作步骤

**步骤1** 获取GeminiDB Redis实例的负载均衡地址和端口信息。

- 负载均衡地址的查看方法请参见[查看负载均衡地址及端口](#)。
- 端口信息的获取方法请参见[查看实例节点端口](#)。

**步骤2** 登录弹性云服务器，具体操作请参见《弹性云服务器快速入门》中“[登录弹性云服务器](#)”。

**步骤3** 使用Golang客户端连接GeminiDB Redis实例，本文以使用go-redis SDK为例。

- go-redis的下载地址为：<https://github.com/go-redis/redis>。
- 使用单机方式连接GeminiDB Redis集群

```
package main
import (
 "fmt"
 "github.com/go-redis/redis"
 "os"
)
func main() {
 // 认证用的用户名和密码直接写到代码中有很大的安全风险，建议在配置文件或者环境变量中存放(密码应密文存放、使用时解密)，确保安全；
 // 本示例以用户名和密码保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量(环境变量名称请根据自身情况进行设置)EXAMPLE_USERNAME_ENV和EXAMPLE_PASSWORD_ENV。
 password = os.Getenv("EXAMPLE_PASSWORD_ENV")
 client := redis.NewClient(&redis.Options{
 Addr: "xx.xx.xx.xx:6379", //步骤1获取到的负载均衡地址
 Password: password,
 DB: 0, // 使用默认的0号数据库
 })
 pong, err := client.Ping().Result()
 fmt.Println(pong, err)
 err = client.Set("key1", "value1", 0).Err()
 if err != nil {
 panic(err)
 }
 val, err := client.Get("key1").Result()
 if err != nil {
 panic(err)
 }
 fmt.Println("key1", val)
}
```

预期输出为：

```
PONG
key1 value1
```

#### 说明

- 使用go-redis连接GeminiDB Redis时，需要使用普通模式，不能使用集群模式，如上示例代码所示。
  - 上述实例代码中的GeminiDB Redis地址和密码，请根据实际情况填写。
- 使用集群方式连接GeminiDB Redis集群

```
package main
import (
 "fmt"
 "github.com/go-redis/redis"
)
func main() {
 client := redis.NewClusterClient(&redis.ClusterOptions{
 Addrs: []string{ // 步骤1获取到的负载均衡地址
 "xx.xx.xx.xx:6379",
 },
 Password: "xx", // 密码
 })
 pong, err := client.Ping().Result()
 fmt.Println(pong, err)

 err = client.Set("key1", "value1", 0).Err()
 if err != nil {
 panic(err)
 }
 val, err := client.Get("key1").Result()
 if err != nil {
 panic(err)
 }
}
```

```
 fmt.Println("key1", val)
}
```

----结束

### 4.3.5.8 通过 C#语言连接实例

本章节主要介绍使用C# 语言访问GeminiDB Redis实例的方法。

#### 前提条件

- 已成功创建GeminiDB Redis实例，且实例状态为“正常”。
- 已创建弹性云服务器，创建弹性云服务器的方法，请参见《弹性云服务器快速入门》中“[购买弹性云服务器](#)”章节。
- 弹性云服务器上已经安装GCC等编译工具。
- 创建的弹性云服务器与GeminiDB Redis实例要保证区域、可用区、VPC和安全组一致。

#### 操作步骤

**步骤1** 获取GeminiDB Redis实例的负载均衡地址和端口信息。

- 负载均衡地址的查看方法请参见[查看负载均衡地址及端口](#)。
- 端口信息的获取方法请参见[查看实例节点端口](#)。
- 各实例地址的查看方法请参见[查看内网IP地址或弹性公网IP](#)。

**步骤2** 登录弹性云服务器，具体操作请参见《弹性云服务器快速入门》中“[登录弹性云服务器](#)”。

**步骤3** 安装.Net。Windows主机前往进入[.NET官网](#)下载，Linux主机需要安装 .NET Core 的密钥和存储库，再安装.Net运行时和SDK：

```
sudo rpm -Uvh https://packages.microsoft.com/config/centos/8/packages-microsoft-prod.rpm
sudo yum install dotnet-sdk-7.0
sudo yum install dotnet-runtime-7.0
```

运行

```
dotnet --version
```

您将会看到您的.Net版本信息。

**步骤4** 使用StackExchange.Redis客户端连接GeminiDB Redis实例。

- 创建项目

在命令行窗口中运行以下命令创建一个新的C#控制台应用程序；或者在Visual Studio中创建一个新的C#控制台应用程序。

```
dotnet new console -o restdemo
```

- 安装Redis的C#客户端 StackExchange.Redis包。Visual Studio中您可以从NuGet包管理器中安装StackExchange.Redis。或者dotnet项目所在的命令行窗口，运行以下命令：

```
dotnet add package StackExchange.Redis
```

- 使用单机方式连接GeminiDB Redis

```
using System;
using StackExchange.Redis;
namespace restdemo
{
```



```
class Program
{
 static void Main(string[] args)
 {
 // 创建一个连接到Redis服务器的ConnectionMultiplexer对象
 string redisConnectionString = "192.xx.xx.xx:6379"; // 步骤1获取到的负载均衡地址
 ConfigurationOptions options = ConfigurationOptions.Parse(redisConnectionString);
 // 认证用的用户名和密码直接写到代码中有很大的安全风险，建议在配置文件或者环境变量中存放
 // (密码应密文存放、使用时解密)，确保安全；
 // 本示例以用户名和密码保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量
 // (环境变量名称请根据自身情况进行设置)EXAMPLE_USERNAME_ENV和EXAMPLE_PASSWORD_ENV。
 string password = Environment.GetEnvironmentVariable("EXAMPLE_PASSWORD_ENV");
 options.Password = password;
 ConnectionMultiplexer redis = ConnectionMultiplexer.Connect(options);

 // 获取Redis数据库对象
 IDatabase redisDb = redis.GetDatabase();

 // 设置一个键值对
 string key = "mykey";
 string value = "myvalue";
 redisDb.StringSet(key, value);
 string valueGet = redisDb.StringGet(key);
 Console.WriteLine($"键 {key} 对应的值为: {valueGet}");
 }
}
```

#### 预期输出

键 mykey 对应的值为: myvalue

- 使用集群方式连接GeminiDB Redis集群

```
using System;
using StackExchange.Redis;

namespace redisdemo
{
 class Program
 {
 static void Main(string[] args)
 {
 ConfigurationOptions options = new ConfigurationOptions();
 options.EndPoints.Add("192.xx.xx.xx:6379"); // 步骤1获得的实例集群节点1的地址和端口号
 options.EndPoints.Add("192.xx.xx.xx:6379"); // 步骤1获得的实例集群节点2的地址和端口号

 options.Password = "your_password"; // 设置密码
 ConnectionMultiplexer redis = ConnectionMultiplexer.Connect(options);
 // 获取Redis数据库对象
 IDatabase redisDb = redis.GetDatabase();
 // 设置一个键值对
 string key = "mykey";
 string value = "myvalue";
 redisDb.StringSet(key, value);
 string valueGet = redisDb.StringGet(key);
 Console.WriteLine($"键 {key} 对应的值为: {valueGet}");
 }
 }
}
```

#### 预期输出

键 mykey 对应的值为: myvalue

----结束

### 4.3.5.9 使用 Sentinel 兼容模式连接实例

GeminiDB Redis采用自研的高可用服务HA组件，无需依赖Sentinel（哨兵）。但为了减少用户的代码改动，提高实例的兼容性，GeminiDB Redis兼容Redis Sentinel访问方式。开启Sentinel兼容模式，您可以像连接开源Redis Sentinel一样连接GeminiDB Redis实例。

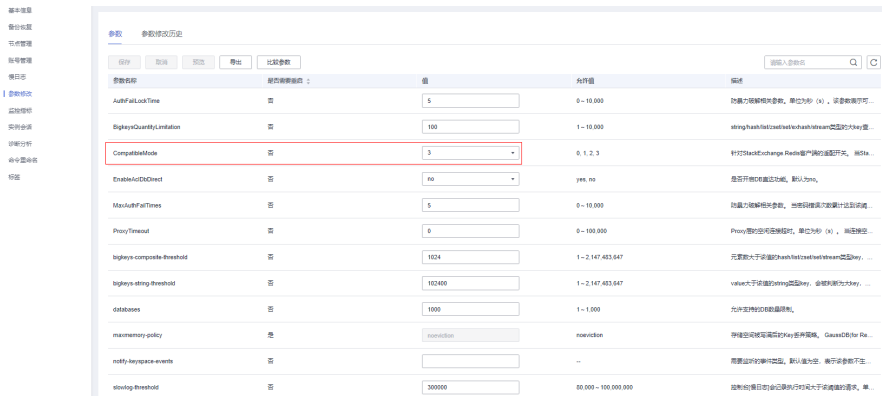
#### 前提条件

- 已成功创建GeminiDB Redis集群版或主备版实例，且实例状态为“正常”。
- 已创建弹性云服务器，创建弹性云服务器的方法，请参见《弹性云服务器快速入门》中“[购买弹性云服务器](#)”章节。
- 弹性云服务器上已经安装GCC等编译工具。
- 创建的弹性云服务器与GeminiDB Redis实例要保证区域、可用区、VPC和安全组一致。
- 使用Sentinel模式连接实例，必须要先开启Sentinel兼容模式。

#### 开启 Sentinel 兼容模式

- 步骤1** 登录[云数据库GeminiDB控制台](#)。
- 步骤2** 在“实例管理”页面，选择指定的实例，单击实例名称，进入“基本信息”页面。
- 步骤3** 在左侧导航树，单击“参数修改”。
- 步骤4** 检查修改参数“CompatibleMode”值，单击“保存”。
- 如果是集群版实例，修改参数“CompatibleMode”值为3。
  - 如果是主备版实例，修改参数“CompatibleMode”值为2。

图 4-50 设置参数



----结束

#### Sentinel 模式连接实例

本示例以Java为例，通过开源库Redisson和Jedis访问GeminiDB Redis实例。

## Redisson 代码示例

```
import org.redisson.Redisson;
import org.redisson.api.RedissonClient;
import org.redisson.config.Config;
import static org.redisson.config.ReadMode.MASTER;

public class SingleServerTests {

 public static void testSentinel() {
 Config config = new Config();
 // 认证用的用户名和密码直接写到代码中有很大的安全风险，建议在配置文件或者环境变量中存放(密码应密
 // 文存放、使用时解密)，确保安全；
 // 本示例以用户名和密码保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量(环境变量名
 // 称请根据自身情况进行设置)EXAMPLE_USERNAME_ENV和EXAMPLE_PASSWORD_ENV。
 String password = System.getenv("EXAMPLE_PASSWORD_ENV");
 config.useSentinelServers()
 .setMasterName("mymaster")
 .setCheckSentinelsList(false)
 .setReadMode(ReadMode.MASTER)
 .setPassword(password)
 .addSentinelAddress("redis://172.xx.xx.xx:6379");
 RedissonClient redisson = Redisson.create(config);
 execute(redisson); // send requests to database
 redisson.shutdown();
 }

 public static void main(String[] args) {
 testSentinel();
 }
}
```

## Jedis 代码示例

```
import java.util.HashSet;
import java.util.Set;
import org.apache.commons.pool2.impl.GenericObjectPoolConfig;
import redis.clients.jedis.Jedis;
import redis.clients.jedis.JedisSentinelPool;
public class TestJedisSentinel {
 public static void main(String[] args) {
 Set<String> sentinels = new HashSet<>();
 sentinels.add("192.xx.xx.xx:6379");
 GenericObjectPoolConfig<Jedis> poolConfig = new GenericObjectPoolConfig<>();
 poolConfig.setMaxIdle(100);
 poolConfig.setMaxWaitMillis(10000);
 poolConfig.setTestOnBorrow(true);
 int connectionTimeout = 5000;
 int soTimeout = 5000;
 int database = 0;
 // 认证用的用户名和密码直接写到代码中有很大的安全风险，建议在配置文件或者环境变量中存放(密
 // 码应密文存放、使用时解密)，确保安全；
 // 本示例以用户名和密码保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量(环境
 // 变量名称请根据自身情况进行设置)EXAMPLE_USERNAME_ENV和EXAMPLE_PASSWORD_ENV。
 String password = System.getenv("EXAMPLE_PASSWORD_ENV");
 try (JedisSentinelPool jspool = new JedisSentinelPool("mymaster", sentinels, poolConfig,
 connectionTimeout, soTimeout, password, database)) {
 Jedis jedis = jspool.getResource();
 jedis.mset("testkey", "AAA", "b", "BBB");
 } catch (Exception e) {
 e.printStackTrace();
 }
 }
}
```

- MasterName：为固定字符串“mymaster”。
- CheckSentinelsList：固定为“false”。

- ReadMode：使用ReadMode.MASTER。
- Password：实例对应的密码。
- SentinelAddress：为GeminiDB Redis实例的负载均衡地址，具体请以实际IP地址和Port为准。

您可以通过单击实例名称，进入“基本信息”页面，在连接信息区域获取“负载均衡地址”。

图 4-51 查看负载均衡地址



GeminiDB Redis只是实现了Sentinel模式的接入方式，不采用原生Sentinel的可用性功能。所以样例代码中，“master\_name”固定为“mymaster”，必须设置“CheckSentinelsList”为“false”，必须设置“ReadMode”为“MASTER”。

### 4.3.6 连接信息管理

#### 4.3.6.1 给 GeminiDB Redis 实例节点配置安全组规则

安全组是一个逻辑上的分组，为同一个虚拟私有云内具有相同安全保护需求，并相互信任的弹性云服务器和GeminiDB Redis实例提供访问策略。

为了保障数据库的安全性和稳定性，在使用GeminiDB Redis实例之前，您需要配置安全组，开通需访问数据库的IP地址和端口。

本节主要介绍在内网和公网连接GeminiDB Redis实例时，为GeminiDB Redis实例配置安全组规则的方法。

#### 使用须知

- 默认情况下，一个租户可以创建500条安全组规则。
- 为一个安全组配置过多的安全组规则会增加首包延时，因此，建议一个安全组内的安全组规则不超过50条。
- 目前一个GeminiDB Redis实例仅允许绑定一个安全组。
- 内网和公网连接实例时，需要配置的安全组规则请参见表4-26。

表 4-26 安全组规则说明

| 场景     | 配置的安全组规则说明                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
|--------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 内网连接实例 | <p>使用内网连接GeminiDB Redis实例时，配置安全组规则分为以下两种情况：</p> <ul style="list-style-type: none"><li>• 若用户有一个GeminiDB Redis实例，想用一个ECS连接时，此ECS与GeminiDB Redis实例在相同安全组时，默认ECS与GeminiDB Redis实例互通，无需配置安全组规则。</li><li>• ECS与GeminiDB Redis实例在不同安全组时，需要为GeminiDB Redis和ECS分别配置安全组规则。<ul style="list-style-type: none"><li>- 配置GeminiDB Redis安全组规则：为GeminiDB Redis所在安全组配置相应的入方向规则，具体操作请参见<a href="#">操作步骤</a>。</li><li>- 配置ECS安全组规则：安全组默认规则为出方向上数据报文全部放行，此时，无需对ECS配置安全组规则。当在ECS所在安全组为非默认安全组且出方向规则非全放通时，需要为ECS所在安全组配置相应的出方向规则。具体操作请参见《弹性云服务器用户指南》中“<a href="#">配置安全组规则</a>”章节。</li></ul></li></ul> |
| 公网连接实例 | <p>使用公网连接GeminiDB Redis实例时，需要为GeminiDB Redis所在安全组配置相应的入方向规则。具体操作请参见<a href="#">操作步骤</a>。</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |

操作步骤

- 步骤1 登录[云数据库GeminiDB控制台](#)。
- 步骤2 在“实例管理”页面，选择目标实例，单击实例名称，进入实例的“基本信息”页面。
- 步骤3 配置安全组规则。  
在“基本信息”页面，单击左侧导航栏中的“节点管理”，在右侧“节点安全组”区域，单击节点安全组名称，进入安全组页面。

图 4-52 节点安全组



- 步骤4 添加入方向规则。
1. 在安全组详情页面，选择页签。

图 4-53 入方向规则

基本设置入方向规则出方向规则关联规则

添加规则快速添加规则

删除一键放通所有规则

入方向规则 9

选择规则快速添加规则

选择规则快速添加规则

| 选择规则                     | 规则  | 类型 | 协议端口 | 源地址       | 描述      | 创建时间                          | 操作   |
|--------------------------|-----|----|------|-----------|---------|-------------------------------|------|
| <input type="checkbox"/> | 1   | 允许 | IPv4 | TCP: 8035 | 0.0.0.0 | 2024/05/29 16:09:08 GMT+08:00 | 修改删除 |
| <input type="checkbox"/> | 1   | 允许 | IPv4 | TCP: 2021 | 0.0.0.0 | 2024/05/29 16:08:47 GMT+08:00 | 修改删除 |
| <input type="checkbox"/> | 1   | 允许 | IPv4 | TCP: 80   | 0.0.0.0 | 2024/05/29 16:08:47 GMT+08:00 | 修改删除 |
| <input type="checkbox"/> | 1   | 允许 | IPv4 | ICMP: 全部  | 0.0.0.0 | 2024/05/29 16:08:47 GMT+08:00 | 修改删除 |
| <input type="checkbox"/> | 1   | 允许 | IPv4 | TCP: 3389 | 0.0.0.0 | 2024/05/29 16:08:47 GMT+08:00 | 修改删除 |
| <input type="checkbox"/> | 1   | 允许 | IPv4 | TCP: 22   | 0.0.0.0 | 2024/05/29 16:08:47 GMT+08:00 | 修改删除 |
| <input type="checkbox"/> | 1   | 允许 | IPv4 | TCP: 443  | 0.0.0.0 | 2024/05/29 16:08:47 GMT+08:00 | 修改删除 |
| <input type="checkbox"/> | 100 | 允许 | IPv4 | 全部        | default | 2022/08/10 15:13:25 GMT+08:00 | 修改删除 |
| <input type="checkbox"/> | 100 | 允许 | IPv6 | 全部        | default | 2022/08/10 15:13:25 GMT+08:00 | 修改删除 |

总条数: 919

2. 单击“添加规则”，弹出添加入方向规则窗口。

图 4-54 添加规则

添加入方向规则 教我设置

安全组入方向规则为白名单（允许），放通入方向网络流量。

安全组 dds-st-test-security-group

如您要添加多条规则，建议请单击导入规则以进行批量导入。

| 协议端口         | 类型   | 源地址       | 操作      |
|--------------|------|-----------|---------|
| TCP          | IPv4 | IP地址      | 复制   删除 |
| 例如: 22或22-30 |      | 0.0.0.0/0 |         |

增加1条规则

确定 取消

3. 根据界面提示配置安全组规则。

表 4-27 入方向安全组规则参数说明

| 参数   | 说明                                                                                                                                                                 | 取值示例 |
|------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------|------|
| 协议端口 | <ul style="list-style-type: none"><li>网络协议。GeminiDB Redis 目前只支持 TCP 连接。</li><li>端口：允许远端地址访问弹性云服务器指定端口，取值范围为：1 ~ 65535。常用端口请参见<a href="#">弹性云服务器常用端口</a>。</li></ul> | TCP  |
| 类型   | <p>IP地址类型。开通IPv6功能后可见。</p> <ul style="list-style-type: none"><li>IPv4</li><li>IPv6</li></ul>                                                                       | IPv4 |

| 参数  | 说明                                                                                                                                                                                                                                              | 取值示例      |
|-----|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------|
| 源地址 | 源地址：可以是IP地址、安全组、IP地址组。用于放通来自IP地址或另一安全组内的实例的访问。例如： <ul style="list-style-type: none"><li>- 192.168.10.10/32（IPv4地址）</li><li>- 192.168.1.0/24（子网）</li><li>- 0.0.0.0/0（任意地址）</li><li>- sg-abc（安全组）</li></ul> 更多IP地址组信息，请参见 <a href="#">IP地址组</a> 。 | 0.0.0.0/0 |
| 描述  | 安全组规则的描述信息，非必填项。<br>描述信息内容不能超过255个字符，且不能包含“<”和“>”。                                                                                                                                                                                              | -         |

步骤5 单击“确定”。

----结束

### 4.3.6.2 查看 GeminiDB Redis IP 地址和端口

本章节主要介绍如何在云数据库 GeminiDB管理控制台查找GeminiDB Redis实例的IP地址和端口等信息。

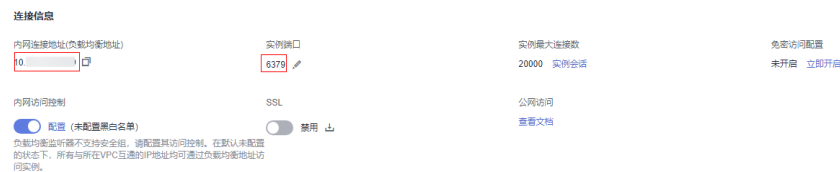
#### 查看负载均衡地址及端口

步骤1 登录[云数据库GeminiDB控制台](#)。

步骤2 在“实例管理”页面，单击目标实例名称，进入基本信息页面。

步骤3 在“连接信息”区域即可获取到“负载均衡地址”和对应的端口信息。

图 4-55 查看负载均衡地址和端口



----结束

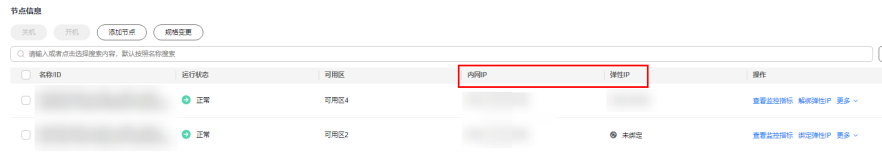
#### 查看内网 IP 地址或弹性公网 IP

步骤1 登录[云数据库GeminiDB控制台](#)。

步骤2 在“实例管理”页面，单击目标实例名称，进入基本信息页面。

单击实例“基本信息”左侧导航中的“节点管理”，即可查看到GeminiDB Redis实例的内网IP地址、绑定的弹性公网IP地址。

图 4-56 查看 IP



----结束

## 查看实例节点端口

- 步骤1** 登录[云数据库GeminiDB控制台](#)。
- 步骤2** 在“实例管理”页面，单击目标实例名称，进入基本信息页面。
- 在“基本信息 > 连接信息”页面可以查看到GeminiDB Redis实例的端口。

图 4-57 查看端口



----结束

## 4.3.6.3 修改 GeminiDB Redis 数据库端口

### 操作场景

GeminiDB Redis支持修改数据库端口，以确保安全性。

以下情况不可修改数据库端口：

- 冻结
- 重启中
- 节点扩容中
- 规格变更中
- 存储扩容中
- 删除节点中

### 操作步骤

- 步骤1** 登录[云数据库GeminiDB控制台](#)。
- 步骤2** 在“实例管理”页面，选择指定的实例，单击实例名称。
- 步骤3** 在左侧导航树，单击“连接管理”。



**步骤4** 在“基本信息”区域的“连接信息”栏的“实例端口”处，单击，修改数据库端口。

数据库端口修改范围为：1024~65535，其中，如下端口号为禁用端口：2180、2887、3887、6377、6378、6380、8018、8079、8091、8479、8484、8999、9864、9866、9867、12017、12333、50069。单击，提交修改。此过程约需1~5分钟。

- 单击，取消修改。

**步骤5** 稍后可在“基本信息”区域“连接信息”处，查看修改结果。

----结束

#### 4.3.6.4 给 GeminiDB Redis 实例节点绑定弹性公网 IP

##### 操作场景

GeminiDB Redis实例创建成功后，支持用户绑定弹性公网IP，通过公共网络访问数据库实例，绑定后也可根据需要解绑。

##### 使用须知

- 对于已绑定弹性公网IP的节点，需解绑后，才可重新绑定其他弹性公网IP。
- 用户需要评估业务所需带宽，购买带宽资源足够的弹性公网IP，**因公网网络性能造成的客户端访问异常不计入SLA**。
- Cluster集群版实例暂不支持该功能。

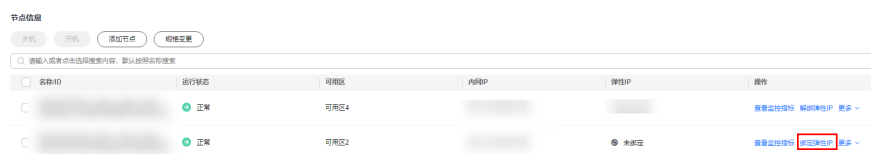
##### 绑定弹性公网 IP

**步骤1** 登录[云数据库GeminiDB控制台](#)。

**步骤2** 在“实例管理”页面，选择目标实例，单击实例名称，进入“基本信息”页面。

**步骤3** 在左侧导航栏单击“节点管理”，在“节点信息”区域的对应节点上，单击操作列的“绑定弹性IP”。

图 4-58 绑定弹性 IP



**步骤4** 在弹出框列表中，显示“未绑定”状态的弹性公网IP，选择所需绑定的弹性公网IP，单击“是”，提交绑定任务。如果没有可用的弹性公网IP，单击“查看弹性IP”，创建新的弹性IP。

图 4-59 选择弹性 IP



**步骤5** 在节点的“弹性IP”列, 查看绑定成功的弹性公网IP。

如需关闭, 请参见[解绑弹性公网IP](#)。

----结束

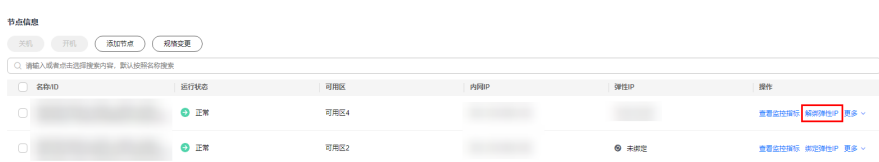
## 解绑弹性公网 IP

**步骤1** 登录[云数据库GeminiDB控制台](#)。

**步骤2** 对于已绑定弹性公网IP的节点, 在“实例管理”页面, 选择目标实例, 单击实例名称, 进入“基本信息”页面。

**步骤3** 在左侧导航栏单击“节点管理”, 在“节点信息”区域的对应节点上, 单击“解绑弹性IP”。

图 4-60 解绑弹性 IP



**步骤4** 在弹出框中, 单击“是”, 解绑弹性IP。

如需重新绑定, 请参见[绑定弹性公网IP](#)。

----结束

### 4.3.6.5 给 GeminiDB Redis 实例设置 SSL 数据加密

SSL (Secure Socket Layer, 安全套接层), 位于可靠的面向连接的网络层协议和应用层协议之间的一种协议层。SSL通过互相认证、使用数字签名确保完整性、使用加密确保私密性, 以实现客户端和服务器之间的安全通讯。

- 认证用户和服务端, 确保数据发送到正确的客户端和服务端。
- 加密数据以防止数据中途被窃取。
- 维护数据的完整性, 确保数据在传输过程中不被改变。

SSL连接开启后, 可以通过SSL方式连接实例, 提高数据安全性。

### 使用须知

- 开启或关闭SSL安全连接时需要重启实例, 此时已建立的旧连接都会中断
- 开启SSL会增加网络连接响应时间和CPU消耗, 请评估对业务的性能影响。
- GeminiDB Redis提供的SSL功能只支持TLS1.3及以上的协议, 不支持TLS1.0、TLS1.1、TLS1.2。

### 开启 SSL 安全连接

- 步骤1** 登录[云数据库GeminiDB控制台](#)。
- 步骤2** 在实例管理页面, 单击目标实例名称, 进入基本信息页面。
- 步骤3** 在连接信息区域, 单击SSL处的 , 开启SSL安全连接。

图 4-61 开启 SSL 安全连接



SSL安全连接开启成功后, 可以通过SSL的方式连接GeminiDB Redis实例, 详情请参见[通过SSL方式连接GeminiDB Redis实例](#)。

----结束

### 关闭 SSL 安全连接

- 步骤1** 登录[云数据库GeminiDB控制台](#)。
- 步骤2** 在实例管理页面, 单击目标实例名称, 进入基本信息页面。
- 步骤3** 在连接信息区域, 单击SSL处的 , 关闭SSL安全连接。

图 4-62 关闭 SSL 安全连接



SSL安全连接关闭后，可以通过非SSL的方式连接GeminiDB Redis实例，详情请参见[操作步骤](#)。

----结束

### 4.3.6.6 通过 SSL 方式连接 GeminiDB Redis 实例

GeminiDB Redis支持选用SSL方式通过Redis-cli客户端连接实例，实现了数据加密功能，具有更高的安全性。本章介绍使用SSL方式连接GeminiDB Redis实例的方法。

#### 使用须知

- 目标实例必须与弹性云服务器在同一个虚拟私有云和子网内才能访问。
- 弹性云服务器必须处于目标实例所属安全组允许访问的范围内，详情请参见[给 GeminiDB Redis实例节点配置安全组规则](#)。
- 开启SSL安全连接后，外部应用需要使用SSL证书并通过SSL的方式访问GeminiDB Redis实例。
- 使用SSL方式时需要确保所使用的Redis客户端具有SSL的连接能力，例如：Redis-cli 6.x版本。

#### 前提条件

创建弹性云服务器，以Linux操作系统为例。详情请参见《弹性云服务器快速入门》中[购买弹性云服务器](#)的内容。

#### 操作步骤

**步骤1** 登录弹性云服务器，详情请参见《弹性云服务器快速入门》中[登录弹性云服务器](#)的内容。

**步骤2** 获取Redis客户端。

**方法一：**

通过如下命令，下载Redis客户端。

```
wget https://download.redis.io/releases/redis-6.2.6.tar.gz
```

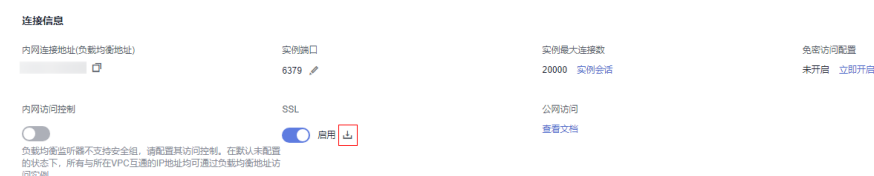
**方法二：**

下载[Redis客户端](#)，并将Redis客户端安装包上传到弹性云服务器。

**步骤3** 获取SSL安全证书。

单击目标实例名称，进入基本信息页面，在“连接信息 > SSL”处单击下载按钮，获取SSL证书。

图 4-63 获取 SSL 证书



**步骤4** 将SSL证书上传到弹性云服务器。

**步骤5** 通过如下命令查看ECS操作系统支持的openssl版本。

```
openssl version
```

#### 说明

- GeminiDB Redis提供的SSL功能只支持TLS1.3及以上的协议，不支持TLS1.0、TLS1.1、TLS1.2。
- 为了确保redis-cli支持TLS1.3及以上的协议，需要ECS的操作系统环境中的openssl版本高于或等于1.1.1。
- 如果操作系统版本低于1.1.1，则需要按照如下步骤安装openssl。

```
wget https://www.openssl.org/source/openssl-1.1.1m.tar.gz
tar -zxvf openssl-1.1.1m.tar.gz
cd openssl-1.1.1m/
./config --prefix=/usr/local/openssl-1.1.1m_install_dir
make
make install
```

待上述新版本的openssl安装成功后，继续执行**步骤8**。
- 如果操作系统版本高于或等于1.1.1，则继续执行**步骤6**。

**步骤6** 解压客户端工具包。

```
tar -xzf redis-6.2.6.tar.gz
```

**步骤7** 在“src”目录下，连接数据库实例。

- 当ECS操作系统中的openssl版本低于1.1.1，并且按照**步骤5**安装了符合要求的openssl版本时，您可以通过如下方式连接数据库实例。

```
cd redis-6.2.6
make BUILD_TLS=yes OPENSSL_PREFIX=/usr/local/openssl-1.1.1m_install_dir
cd src
LD_PRELOAD=/usr/local/openssl-1.1.1m_install_dir/lib/libssl.so.1.1:/usr/local/
openssl-1.1.1m_install_dir/lib/libcrypto.so.1.1 ./redis-cli -h <DB_HOST> -p <DB_PORT> -a
<DB_PWD> --tls --cacert <CACERT_PATH>
```

示例：

```
LD_PRELOAD=/usr/local/openssl-1.1.1m_install_dir/lib/libssl.so.1.1:/usr/local/
openssl-1.1.1m_install_dir/lib/libcrypto.so.1.1 ./redis-cli -h 192.168.0.208 -p 6379 -a <DB_PWD> --
tls --cacert ./cacert.crt
```

- 当ECS操作系统中的openssl版本高于或等于1.1.1时，您可以通过如下方式连接数据库实例。

```
cd redis-6.2.6
make BUILD_TLS=yes
cd src
./redis-cli -h <DB_HOST> -p <DB_PORT> -a <DB_PWD> --tls --cacert <CACERT_PATH>
```

示例：

```
./redis-cli -h 192.168.0.208 -p 6379 -a <DB_PWD> --tls --cacert ./cacert.crt
```

表 4-28 参数说明

| 参数            | 说明                                                                                                                                   |
|---------------|--------------------------------------------------------------------------------------------------------------------------------------|
| <DB_HOST>     | 待连接实例的内网IP。<br>您可以在“实例管理”页面，单击实例名称，进入“基本信息”页面，在左侧导航栏单击“节点管理”，在节点信息列表中获取“内网IP”。<br>如果您购买的实例有多个节点，选择其中任意一个节点的内网IP即可连接GeminiDB Redis实例。 |
| <DB_PORT>     | 待连接实例的端口，具体请以实际端口为准。<br>获取实例端口的方法如下：<br>在“实例管理”页面，单击实例名称，进入“基本信息”页面，在“连接信息 > 实例端口”处获取当前实例的端口信息。                                      |
| <DB_PWD>      | 购买GeminiDB Redis实例时设置的管理员密码。                                                                                                         |
| <CACERT_PATH> | SSL证书的路径。                                                                                                                            |

步骤8 出现如下信息，说明连接成功。

IP:port>

----结束

### 4.3.6.7 修改 GeminiDB Redis 实例节点安全组

#### 操作场景

GeminiDB Redis支持修改安全组。

#### 使用须知

- 对于进行节点扩容中的实例，不可修改安全组。

#### 操作步骤

步骤1 登录[云数据库GeminiDB控制台](#)。

步骤2 在“实例管理”页面，选择指定的实例，单击实例名称。

步骤3 在左侧导航树，单击“节点管理”。

步骤4 在“节点安全组”区域，单击，选择实例所属安全组。

- 单击，提交修改。此过程约需1~3分钟。
- 单击，取消修改。

步骤5 稍后可在“安全组”区域，查看修改结果。

----结束

### 4.3.6.8 给 GeminiDB Redis 实例设置负载均衡内网访问控制

#### 操作场景

GeminiDB Redis支持设置负载均衡内网访问控制。

#### 使用须知

- 负载均衡地址不支持安全组，创建完成后请配置IP访问控制，如未配置白名单，所有与VPC互通的IP地址均可访问实例。
- Cluster集群版实例暂不支持该功能。

#### 开启负载均衡地址黑白名单

- 步骤1** 登录[云数据库GeminiDB控制台](#)。
- 步骤2** 在“实例管理”页面，选择指定的实例，单击实例名称。
- 步骤3** 在连接信息区域，单击内网访问控制后的，开启负载均衡内网访问控制。

图 4-64 开启负载均衡内网访问控制



- 步骤4** 开启负载均衡地址黑白名单后，您可以根据需要选择黑名单或白名单。

图 4-65 修改访问控制



- 黑名单：仅支持设置白名单或者黑名单其中一种，切换后原设置失效。黑名单内的地址禁止访问，请谨慎操作。
- 白名单：仅支持设置白名单或者黑名单其中一种，切换后原设置失效。仅白名单内的地址允许访问，请谨慎操作。

----结束

## 关闭负载均衡内网访问控制

**步骤1** 登录[云数据库GeminiDB控制台](#)。

**步骤2** 在“实例管理”页面，选择指定的实例，单击实例名称。

**步骤3** 在连接信息区域，单击内网访问控制后的，在弹出框选择“是”，关闭负载均衡内网访问控制。

图 4-66 关闭负载均衡内网访问控制



**步骤4** 关闭负载均衡内网访问控制后，负载均衡地址无法生效。

----结束

## 4.4 数据迁移

### 4.4.1 Redis 数据迁移方案概览

本章节主要介绍如何将业务平迁到GeminiDB Redis，如有迁移问题，您可以在管理控制台右上角，选择“[工单 > 新建工单](#)”，提交工单获得技术支持。

#### 使用须知

阿里云Redis、腾讯云Redis等云服务不能使用华为云DRS服务进行数据迁移，通常有以下因素：

- 不支持pysync迁出协议：存在一些自研的 Redis-like 数据库没有兼容 PSync 协议。
- 架构限制：较多云厂商支持代理模式，即在用户与 Redis 服务之间增加 Proxy 组件。因为 Proxy 组件的存在，所以 PSync 协议无法支持。
- 安全限制：在原生 Redis 中 PSync 协议基本会触发 fork，会导致内存膨胀与用户请求延迟增加，较坏情况下甚至会发生 out of memory。
- 商业策略：较多用户使用 RedisShake 是为了下云或者换云，屏蔽了 PSync 协议。



迁移工具

- **DRS服务**：华为云数据复制服务DRS（Data Replication Service）提供数据库零停机的迁移上云体验，支持全量数据迁移和增量数据迁移，为您提供安全稳定高可靠的迁移链路，详情请参考[迁移方案概览](#)。
- **Redis-Shake工具**：Redis-Shake是一款开源迁移工具，支持全量扫描（rump）、数据恢复（restore）、增量同步（sync）等迁移方式，灵活方便。将工具下载到一台ECS上即可用命令行形式方便迁移。

权限要求

- 确保GeminiDB Redis实例安全组配置数据库端口。

迁移场景

表 4-29 迁移场景

| 序号 | 源端             | 目标端                           | 迁移方案                                                          |
|----|----------------|-------------------------------|---------------------------------------------------------------|
| 1  | GeminiDB Redis | 自建Redis/<br>GeminiDB<br>Redis | <a href="#">使用DRS服务将GeminiDB Redis迁移到Redis（推荐）</a>            |
| 2  | 阿里云Redis/Tair  | GeminiDB<br>Redis             | <a href="#">阿里云数据库Redis/Tair到GeminiDB Redis的迁移</a>            |
| 3  | 腾讯云Redis       | GeminiDB<br>Redis             | <a href="#">腾讯云Redis到GeminiDB Redis的迁移</a>                    |
| 4  | 自建Redis        | GeminiDB<br>Redis             | <a href="#">使用DRS服务将自建Redis迁移到GeminiDB Redis（推荐）</a>          |
| 5  | 自建Redis        | GeminiDB<br>Redis             | <a href="#">通过Redis-Shake迁移工具将自建Redis迁移到GeminiDB Redis</a>    |
| 6  | RDB文件          | GeminiDB<br>Redis             | <a href="#">使用Redis-Shake工具将RDB文件/AOF文件 导入到GeminiDB Redis</a> |
| 7  | RDB文件          | GeminiDB<br>Redis             | <a href="#">使用数据导入功能将RDB文件恢复到GeminiDB Redis(推荐)</a>           |
| 8  | 自建Kvrocks      | GeminiDB<br>Redis             | <a href="#">Kvrocks到GeminiDB Redis的迁移</a>                     |
| 9  | 自建Pika         | GeminiDB<br>Redis             | <a href="#">Pika到GeminiDB Redis的迁移</a>                        |
| 10 | 自建SSDB         | GeminiDB<br>Redis             | <a href="#">SSDB到GeminiDB Redis的迁移</a>                        |
| 11 | 自建LevelDB      | GeminiDB<br>Redis             | <a href="#">LevelDB到GeminiDB Redis的迁移</a>                     |
| 12 | 自建RocksDB      | GeminiDB<br>Redis             | <a href="#">RocksDB到GeminiDB Redis的迁移</a>                     |

| 序号 | 源端                         | 目标端            | 迁移方案                                                            |
|----|----------------------------|----------------|-----------------------------------------------------------------|
| 13 | AWS ElasticCache for Redis | GeminiDB Redis | <a href="#">AWS ElasticCache for Redis数据库到GeminiDB Redis的迁移</a> |

## 4.4.2 使用 DRS 服务将 GeminiDB Redis 迁移到 Redis（推荐）

华为云数据复制服务DRS（Data Replication Service）提供数据库零停机的迁移上云体验，支持全量数据迁移和增量数据迁移，为您提供安全稳定高可靠的迁移链路，详情请参考[迁移方案概览](#)。

如何使用DRS服务将GeminiDB Redis迁移到Redis，请参考[将GeminiDB Redis迁移到Redis](#)。

## 4.4.3 阿里云数据库 Redis/Tair 到 GeminiDB Redis 的迁移

本章节主要介绍阿里云数据库Redis/Tair到GeminiDB Redis的迁移方案。

### 迁移原理

- 使用阿里云的数据迁移工具DTS中的数据迁移功能，该功能支持将阿里云Redis服务的数据迁移至其他Redis服务中，利用该工具避开阿里云 Redis服务屏蔽sync和psync命令限制，实现数据从阿里云Redis服务迁移至华为云GeminiDB Redis中。

### 使用须知

- 阿里云侧源端需与华为侧目标端网络互通，需要开通专线或绑定公网IP。
- 阿里云DTS数据迁移功能实时收费，使用前请确保阿里云账户余额充足。
- 华为云GeminiDB Redis容量不得小于阿里云数据库Redis的内存容量。
- 注意放开源端和目标端安全组配置。
- 阿里侧部分Redis比较特殊，例如Tair混合存储型，无法实现在线全量加增量迁移，需要使用全量扫描方式来迁出，请参考[开源Redis全量扫描迁移到GeminiDB Redis](#)。

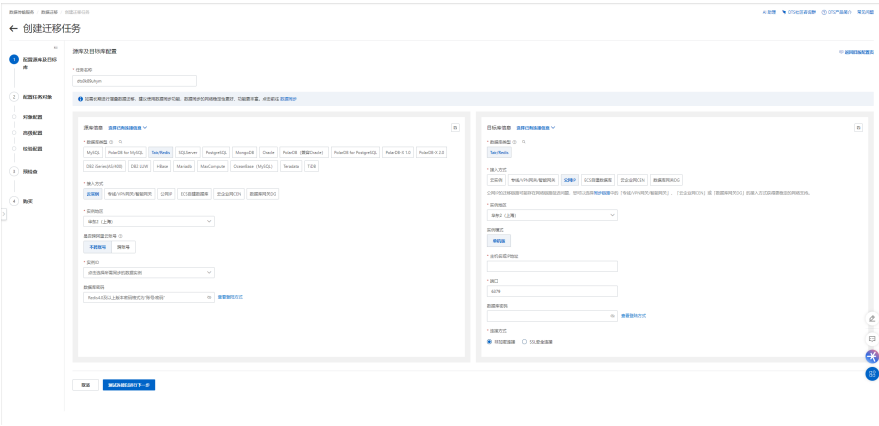
### 准备工作

- 通过公网IP迁移
  - 提前购买华为侧弹性公网IP，带宽需要大于源库流量。
  - 将弹性公网IP绑定到华为云GeminiDB Redis其中一个节点上。
  - 配置DTS时选择目标库接入方式为公网IP。
- 通过专线迁移
  - 提前购买阿里侧ECS，确保能连通到华为云GeminiDB Redis。
  - 配置数据转发，将本地端口接收的流量转发到目标端，实现从阿里云Redis迁移至GeminiDB Redis。  
`ssh -g -L (转发端口):(华为侧Redis 负载均衡IP):(华为侧Redis端口) -N -f root@ (本地ECS IP)`
  - 配置DTS时选择目标库接入方式为ECS自建数据库。

## 步骤 1：购买 DTS 数据同步功能

**步骤1** 源端选择阿里云侧 Redis服务，如果使用弹性公网IP迁移，目标端选择公网IP，主机名称填写弹性公网IP；如果使用专线迁移，目标端选择ECS自建Redis，主机名称填写ECS对应IP，端口号填写转发端口号，并输入数据库密码，单击测试链接，若测试无异常跳转至下一页面，否则检查整条链路是否正常，白名单配置是否正确。

图 4-67 源端和目标端配置



**步骤2** 根据需要选择迁移类型（全量迁移或全量迁移+增量迁移）。选择预检查并报错拦截，选择所需要迁移的DB。



若客户使用多DB功能，需要勾选需要迁移的DB。若没有使用多DB功能，则只选择DB0。

图 4-68 待迁移 DB

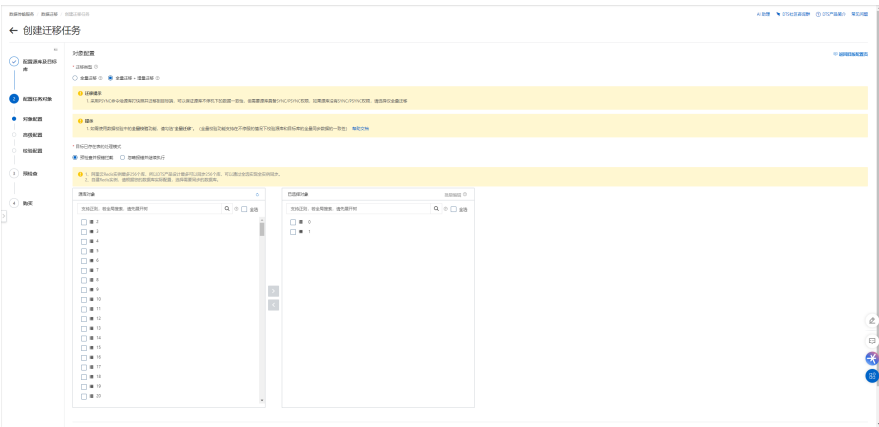
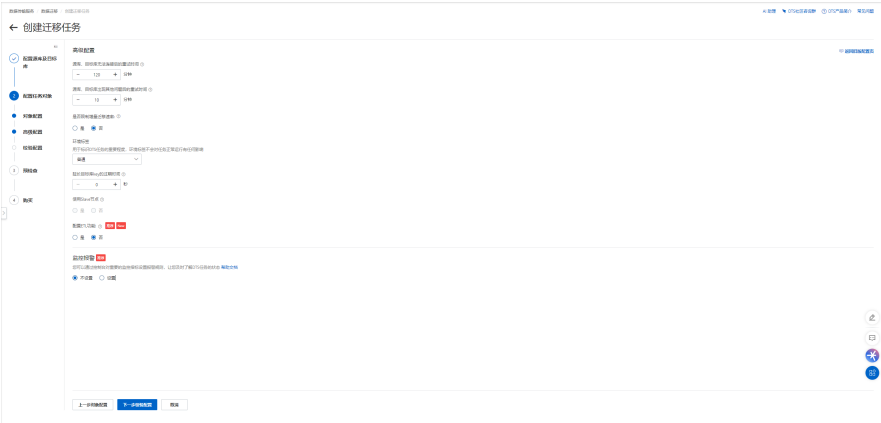
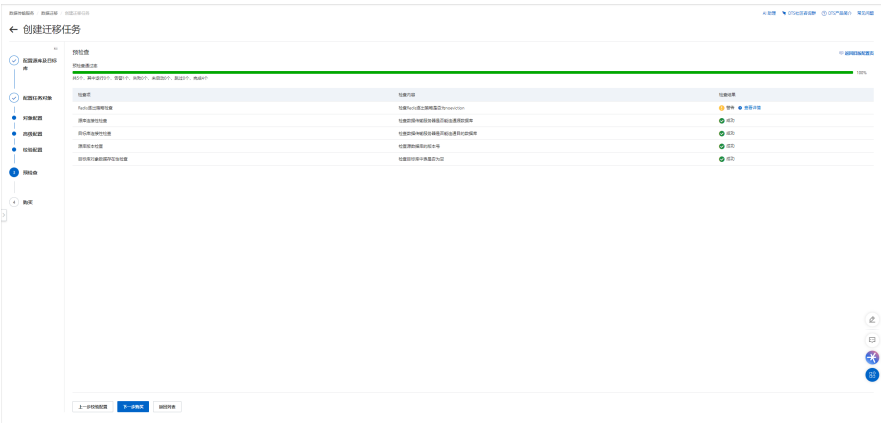


图 4-69 根据需要设置告警及重试时间等参数，根据需要设置校验配置



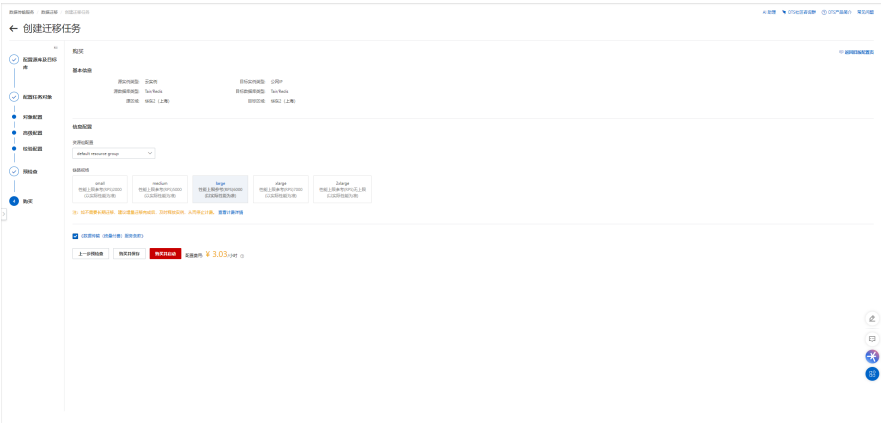
**步骤3** 待预检查完成后，单击“下一步购买”。

图 4-70 预检查



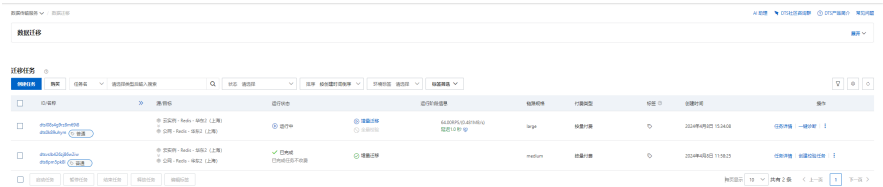
**步骤4** 选择迁移时的带宽，单击“购买并启动”。

图 4-71 带宽配置



**步骤5** 使用全量迁移+增量迁移时，迁移任务不会自动结束。当观察到无延迟（延迟0毫秒），则全量同步已完成。

图 4-72 任务状态

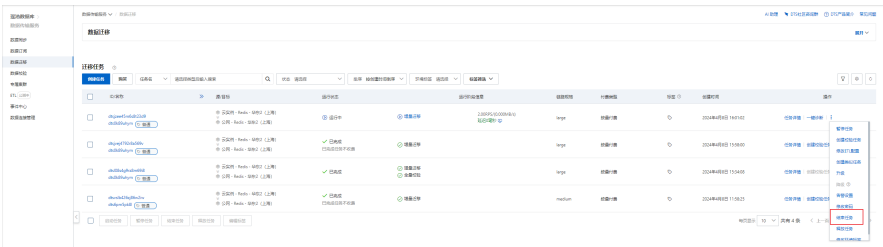


----结束

## 步骤 2：停止 DTS 数据同步服务

**步骤1** 待Redis业务割接后，结束数据同步任务。

图 4-73 结束数据迁移任务



----结束

## 步骤 3：迁移后 Redis 数据一致性校验

迁移完成后，您可以对Redis数据做一致性校验。

### 说明

- Redis迁移已完成或者已经进入增量状态。
- 需要ECS实例部署Redis-Full-Check开源工具，并确保与源端、目标端网络互通。
- 如果迁移任务处于增量状态，由于迁移源端和目标端间存在网络时延，无法确保校验时数据一致，如有条件，建议停止对源端的写入后再做校验。
- 业务使用Redis时往往存在给key设置过期时间，迁移时key过期行为会影响数据一致性，如果检验结果不一致，可能是由于key过期时间不一致导致。
- 云厂商DTS在迁移过程中会往目标端Redis写入临时探活key，校验时可能发现非业务数据，属于正常现象。

### 操作步骤：

**步骤1** 登录ECS实例，确保能连上源端和目标端Redis。

**步骤2** 部署redis-full-check工具。

**步骤3** 执行以下命令进行数据校验。

```
/redis-full-check -s {源端IP}:{源端口} -p {源端密码} -t {目标端IP}:{目标端口} -a {目标端密码} -m 1
```

表 4-30 参数说明

| 参数 | 参数                                                                                            | 参数示例             |
|----|-----------------------------------------------------------------------------------------------|------------------|
| -s | 源端Redis连接地址和端口。                                                                               | -s 10.0.0.1:6379 |
| -p | 源端Redis密码。                                                                                    | -                |
| -t | 目标端Redis连接地址和端口。                                                                              | -t 10.0.0.2:6379 |
| -a | 目标端Redis密码。                                                                                   | -                |
| -m | 校验模式：<br>1. 全量校验所有键值对。<br>2. 仅校验值的长度。<br>3. 仅校验key完整性。<br>4. 全量校验所有键值，但只校验大key的长度。<br>默认为模式2。 | -m 1             |
| -q | 最大qps限制，默认15000。                                                                              | -q 5000          |
| -d | 校验结果保存的文件名称，默认result.db。                                                                      | -d result.db     |

- 步骤4** 查看校验结果文件。
- 校验默认执行三轮，输出三个校验结果文件，一般只需要看最后一个校验结果文件。
- 执行sqlite3 result.db.3。
  - 执行select \* from key。
  - 查看是否存在异常的key。

```
Enter ".help" for usage hints.
sqlite> select * from key;
1|b|string|lack_target|0|1|0
2|c|string|lack_target|0|1|0
3|a|string|lack_target|0|1|0
sqlite>
```

----结束

4.4.4 腾讯云 Redis 到 GeminiDB Redis 的迁移

本章主要介绍腾讯云Redis到GeminiDB Redis的迁移方案。

## 迁移原理

- 使用腾讯云的数据迁移工具DTS中的数据同步功能，该功能支持将腾讯云 Redis服务的数据迁移至ECS自建Redis中，利用该工具避开腾讯云 Redis服务屏蔽分片实际IP，实现数据从腾讯云 Redis服务迁移至华为云GeminiDB Redis中。

## 使用须知

- 腾讯云侧源端需与华为侧目标端网络互通，需要开通专线或绑定公网IP。
- 腾讯云DTS数据同步功能实时收费，使用前请确保腾讯云账户余额充足。
- 华为云GeminiDB Redis容量不得小于腾讯云数据库Redis的内存容量。
- 注意放开源端和目标端安全组正确配置。
- 腾讯云不支持公网迁出，需要通过ECS做一层转发到华为云。
- 腾讯侧部分Redis比较特殊，无法实现在线全量加增量迁移，需要使用全量扫描方式来迁出，请参考[开源Redis全量扫描迁移到GeminiDB Redis](#)。

## 准备工作

腾讯云不支持公网迁出，需要通过ECS做数据转发实现迁移功能。

- 通过专线迁移
  - 需要在腾讯侧购买一台ECS，确保能连接到华为云GeminiDB Redis。
  - 配置数据转发，将本地端口接收的流量转发到目标端，实现从腾讯云Redis迁移至GeminiDB Redis。  
`ssh -g -L (转发端口):(华为侧Redis 负载均衡IP):(华为侧Redis端口) -N -f root@ (本地ECS IP)`
- 通过弹性公网IP迁移
  - 需要在腾讯侧购买一台ECS，购买并绑定腾讯侧弹性公网IP。
  - 提前购买华为侧弹性公网IP，确保带宽大于源库流量，并绑定在GeminiDB Redis其中一个节点上。
  - 配置数据转发，将本地端口接收的流量转发到目标端，实现从腾讯云Redis迁移至GeminiDB Redis。  
`ssh -g -L (转发端口):(华为侧弹性公网 IP):(华为侧Redis端口) -N -f root@ (本地ECS IP)`

## 步骤 1：购买 DTS 数据迁移功能

**步骤1** 选择源端以及目标端的region，选择迁移方式为下云。

图 4-74 DTS 配置

计费规则

实例类型

MySQL

Redis

MongoDB

MariaDB

PostgreSQL

Percona

SQL Server

TDSQL MySQL

TDSQL-C PostgreSQL

实例所在地域

华东地区

广州

深圳金融

深圳

上海

上海金融

杭州

南京

中国香港

多伦多

华北地区

北京

天津

北京金融

新加坡

曼谷

雅加达

硅谷

成都

重庆

华南地区

法兰克福

莫斯科

首尔

东京

孟买

弗吉尼亚

亚太东部

日本东部

美国西部

韩国东部

目标实例类型

Redis

目标实例地域

华南地区

广州

上海

杭州

南京

中国香港

多伦多

台北地区

北京

新加坡

曼谷

美国西部

华东地区

成都

重庆

法兰克福

莫斯科

首尔

东京

孟买

弗吉尼亚

亚太东部

日本东部

美国西部

韩国东部

迁移方式

上云

下云

规格

Xlarge

查看规格详情说明，请参看[购买须知文档](#)

续购

数量

-1+

按量配置费用 0元/小时

立即购买

**步骤2** 源端选择腾讯云侧 Redis服务，目标端选择ECS自建Redis，并输入数据库密码，单击测试链接，若测试无异常跳转至下一页面，否则检查整条链路是否正常，白名单配置是否正确。

图 4-75 DTS 任务列表

新增/编辑任务

编辑配置

数据库智能运维

多个过滤器将应用任务分组

| <input type="checkbox"/> 任务 ID/名称                                                  | 任务状态/进度  | 运行模式 | 策略     | 计费类型 | 最后一次检验结果 | 源库类型  | 目标库类型 | 源库类型 | 地址            | 创建时间                | 操作                                  |
|------------------------------------------------------------------------------------|----------|------|--------|------|----------|-------|-------|------|---------------|---------------------|-------------------------------------|
| <div><input type="checkbox"/> ds-0344c</div> <div>zds_test</div> <div>NewOTS</div> | 状态: 创建成功 | 立即执行 | Xlarge | 按量计费 |          | Redis | Redis | -    | 源: -<br>目标: - | 2022-06-16 14:42:00 | <div>立即删除 修改</div> <div>校验 更多</div> |

图 4-76 DTS 实例信息配置

源库设置

源库类型 • Redis

接入类型 • 云数据库 [类型说明](#)

所属地域 华东地区（上海）

数据库实例 • crs-qlp2x0fc (zsd\_test)

密码 \*\*\*\*\*

测试连通性 ✔ 测试通过

目标库设置

目标库类型 • Redis

所属地域 华东地区（上海）

接入类型 • 公网 **云主机自建** 专线接入 VPN 接入 云联网

节点类型 • **单节点迁移** Redis Cluster 迁移 代理集群迁移

在数据迁移之前，DTS 将尝试禁用目标集群的主动淘汰(通过命令 debug set-active-expire 0)。迁移完成后会启用主动淘汰。

云主机实例 • ins-rem5qq6n (未命名)

端口 • 6379

密码 \*\*\*\*\*

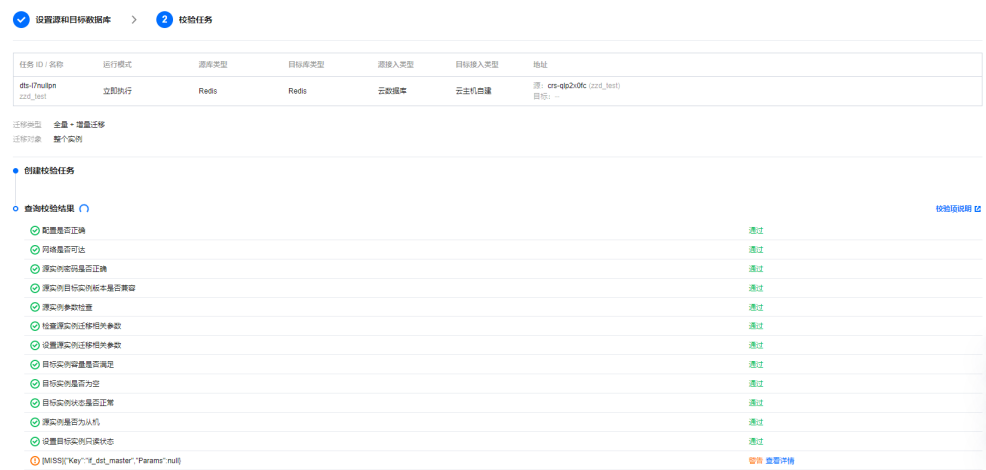
测试连通性 ✔ 测试通过

主机地址: 172.17.0.14

**步骤3** 待校验完成后，单击“下一步”。



图 4-77 DTS 预校验



----结束

## 步骤 2：停止 DTS 数据迁移服务

- 待Redis业务割接后，完成数据迁移任务。
- 若没有数据同步需求时，需要 DTS迁移任务（该功能会实时收费）。

## 步骤 3：迁移后 Redis 数据一致性校验

迁移完成后，您可以对Redis数据做一致性校验。

### 说明

- Redis迁移已完成或者已经进入增量状态。
- 需要ECS实例部署Redis-Full-Check开源工具，并确保与源端、目标端网络互通。
- 如果迁移任务处于增量状态，由于迁移源端和目标端间存在网络时延，无法确保校验时数据一致，如有条件，建议停止对源端的写入后再做校验。
- 业务使用Redis时往往存在给key设置过期时间，迁移时key过期行为会影响数据一致性，如果检验结果不一致，可能是由于key过期时间不一致导致。
- 云厂商DTS在迁移过程中会往目标端Redis写入临时探活key，校验时可能发现非业务数据，属于正常现象。

### 操作步骤：

**步骤1** 登录ECS实例，确保能连上源端和目标端Redis。

**步骤2** 部署redis-full-check工具。

**步骤3** 执行以下命令进行数据校验。

```
/redis-full-check -s {源端IP}:{源端口} -p {源端密码} -t {目标端IP}:{目标端口} -a {目标端密码} -m 1
```

表 4-31 参数说明

| 参数 | 参数                                                                                            | 参数示例             |
|----|-----------------------------------------------------------------------------------------------|------------------|
| -s | 源端Redis连接地址和端口。                                                                               | -s 10.0.0.1:6379 |
| -p | 源端Redis密码。                                                                                    | -                |
| -t | 目标端Redis连接地址和端口。                                                                              | -t 10.0.0.2:6379 |
| -a | 目标端Redis密码。                                                                                   | -                |
| -m | 校验模式：<br>1. 全量校验所有键值对。<br>2. 仅校验值的长度。<br>3. 仅校验key完整性。<br>4. 全量校验所有键值，但只校验大key的长度。<br>默认为模式2。 | -m 1             |
| -q | 最大qps限制，默认15000。                                                                              | -q 5000          |
| -d | 校验结果保存的文件名称，默认result.db。                                                                      | -d result.db     |

**步骤4** 查看校验结果文件。

校验默认执行三轮，输出三个校验结果文件，一般只需要看最后一个校验结果文件。

- 执行sqlite3 result.db.3。
- 执行select \* from key。
- 查看是否存在异常的key。

```
Enter ".help" for usage hints.
sqlite> select * from key;
1|b|string|lack_target|0|1|0
2|c|string|lack_target|0|1|0
3|a|string|lack_target|0|1|0
sqlite> |
```

----结束

## 4.4.5 使用 DRS 服务将自建 Redis 或者 Redis 集群迁移到 GeminiDB Redis（推荐）

华为云数据复制服务DRS（Data Replication Service）提供数据库零停机的迁移上云体验，支持全量数据迁移和增量数据迁移，为您提供安全稳定高可靠的迁移链路，详情请参考[迁移方案概览](#)。

如何使用DRS服务将Redis迁移到GeminiDB Redis，请参考[将Redis迁移到GeminiDB Redis](#)。

如何使用DRS服务将Redis集群迁移到GeminiDB Redis，请参考[将Redis集群迁移到GeminiDB Redis](#)。

## 4.4.6 通过 Redis-Shake 迁移工具将自建 Redis 迁移到 GeminiDB Redis

自建Redis到GeminiDB Redis的迁移可以使用Redis-Shake迁移工具。本章节主要介绍使用Redis-Shake迁移工具进行自建Redis到GeminiDB Redis的迁移方案。

### 迁移原理

使用Redis-Shake迁移工具进行源端Redis到目标端GeminiDB Redis的迁移。迁移过程支持全量+增量迁移，支持单机/主从/Cluster集群/RDB文件等多种Redis数据源。

- 全量迁移原理：工具模拟源端Redis的从节点，通过全量同步获取RDB文件，解析后将数据以命令形式发送给目标端GeminiDB Redis。此外，也可将RDB文件作为数据源，方便地导入某时刻的数据快照。
- 增量迁移原理：全量迁移完成后，工具继续将增量数据以命令形式发送给目标端GeminiDB Redis，直到手动停止工具。

### 使用须知

- 如遇到源端Redis同步脱节，此时需要停止迁移工具，清空目标端数据并重新迁移。为避免此场景，建议在写入流量低峰时段迁移，并可适当调大源端client-output-buffer-limit参数，以增大增量同步的环形缓冲区大小。
- 迁移工具理论上不会对源端数据造成影响，但可能会暂时影响性能。
- 多DB迁移场景，需注意源端与目标端的DB映射关系，避免造成非预期的数据覆盖。
- 暂时不支持stream类型数据的迁移。
- 保证迁移工具Redis-Shake、源端Redis和目标端GeminiDB Redis网络互通。
- 如需进行自建Redis到GeminiDB Redis的迁移支持，您可以在管理控制台右上角，选择“[工单 > 新建工单](#)”，联系客服进行处理。

## 开源 Redis 单机/主从版在线移到 GeminiDB Redis

开源Redis单机/主从版除了可以采用上述文件导入方式外，还可以在线迁移到GeminiDB Redis。

### 步骤1 部署迁移工具。

1. 获取开源工具[Redis-Shake](#)。

#### 说明

Redis-Shake工具可下载release版本，解压缩后即可使用。

2. 修改配置文件“Redis-Shake.conf”，修改时需要注意以下配置项。

**log.level = info** # 默认日志级别，info级别可以打印出相应的迁移进度，可以用来判断迁移是否结束。

**source.address = <host>:<port>** # 源端地址，开源redis所在主机及端口。

**source.password\_raw = \*\*\*\*\*** # 源端口令。

**source.type = standalone** # 根据源端类型选择。

**target.address = <host>:6379** # 目标端地址。

**target.password\_raw = \*\*\*\*\*** # 目标端口令。

**target.version = 5.0** # 目标端redis版本。

**target.type = standalone** # 目标端类型。

**target.db = -1** # 将所有数据迁移到目标端GeminiDB Redis的指定数据库上，若设置为-1，则迁移后的数据库和源实例的DB保持相同的映射关系。

3. 按需选配是否覆盖目标端数据。

**key\_exists = none**

 说明

当源端和目的端有重复key时需要选配是否覆盖目标端数据，可配置的值为：

- **rewrite**：源端覆盖目的端。
- **none**：一旦发生进程直接退出。
- **ignore**：保留目的端key，忽略源端的同步key。该值在rump模式下不会生效。

由于本次迁移源端是RDB文件，因此预期无重复数据，建议选择none。如迁移过程异常退出，您可以在管理控制台右上角，选择“[工单 > 新建工单](#)”，联系客服进行处理。

**步骤2** 进行数据迁移。

启动迁移命令：

**./redis-shake.linux -conf=redis-shake.conf -type=sync**

- 执行日志中出现如下信息，代表全量数据同步完成，进入增量同步阶段。  
sync rdb done
- 执行日志出现如下信息时，代表增量同步无新增内容，可以停止程序以中断增量同步。  
sync: +forwardCommands=0 +filterCommands=0 +writeBytes=0

**步骤3** 数据校验。

使[开源Redis单机/主从版在线移到GeminiDB Redis](#)用工具[RedisFullCheck](#)进行校验，下载并解压缩后即可使用。

**./redis-full-check -s SOURCE\_IP:SOURCE\_PORT -p SOURCE\_PWD -t TARGET\_IP:6379 -a TARGET\_PWD**

校验完毕出现如下信息表示迁移成功，源端与目的端数据相同。

all finish successfully, totally 0 key(s) and 0 field(s) conflict

----结束

## 开源 Redis 集群版迁移到 GeminiDB Redis

对于开源Redis集群版，需要注意配置文件中以下两项：

**source.address = <host1>:<port1>,<host2>:<port2>,<host2>:<port2>** # 源端各节点所在主机及端口

**source.type = cluster** # 源端为cluster集群

其他步骤与[开源Redis单机/主从版在线移到GeminiDB Redis](#)相同。

## 开源 Codis 集群版迁移到 GeminiDB Redis

对于Codis集群版，需要获取各个分片所在的主机以及端口，注意配置文件以下两项：

```
source.address = <host1>:<port1>,<host2>:<port2>,<host2>:<port2> # 源端各节点所在主机及端口
```

```
source.type = cluster # 源端为cluster集群
```

其他步骤与[开源Redis单机/主从版在线移到GeminiDB Redis](#)相同。

## 开源 Redis 全量扫描迁移到 GeminiDB Redis

如果以上场景都没法实现，可以采用Redis-Shake的全量扫描(rump)来逐个扫描来进行迁移。

### 步骤1 部署迁移工具。

1. 获取开源工具[Redis-Shake](#)。

#### 说明

Redis-Shake工具可下载release版本，解压缩后即可使用。

2. 修改配置文件“Redis-Shake.conf”，修改时需要注意以下配置项。

```
log.level = info # 默认日志级别，info级别可以打印出相应的迁移进度，可以用来判断迁移是否结束。
```

```
source.address = <host>:<port> # 源端地址，开源redis所在主机及端口。
```

```
source.password_raw = ***** # 源端口令。
```

```
source.type = standalone # 根据源端类型选择。
```

```
target.address = <host>:6379 # 目标端地址。
```

```
target.password_raw = ***** # 目标端口令。
```

```
target.version = 5.0 # 目标端redis版本。
```

```
target.type = standalone # 目标端类型。
```

```
target.db = -1 # 将所有数据迁移到目标端GeminiDB Redis的指定数据库上，若设置为-1，则迁移后的数据库和源实例的DB保持相同的映射关系。
```

3. 按需选配是否覆盖目标端数据。

```
key_exists = none
```

#### 说明

当源端和目的端有重复key时需要选配是否覆盖目标端数据，可配置的值为：

- rewrite：源端覆盖目的端。
- none：一旦发生进程直接退出。
- ignore：保留目的端key，忽略源端的同步key。该值在rump模式下不会生效。

由于本次迁移源端是RDB文件，因此预期无重复数据，建议选择none。如迁移过程异常退出，您可以在管理控制台右上角，选择“[工单 > 新建工单](#)”，联系客服进行处理。

### 步骤2 进行数据迁移。

启动迁移命令：

```
./redis-shake.linux -conf=redis-shake.conf -type=rump
```

- 执行日志中出现如下信息，代表全量数据同步完成。  
dbRumper[0] executor[0] finish

步骤3 数据校验。

使用工具[RedisFullCheck](#)进行校验，下载并解压缩后即可使用。

```
./redis-full-check -s SOURCE_IP:SOURCE_PORT -p SOURCE_PWD -t
TARGET_IP:6379 -a TARGET_PWD
```

校验完毕出现如下信息表示迁移成功，源端与目的端数据相同。

```
all finish successfully, totally 0 key(s) and 0 field(s) conflict
```

----结束

注：使用redis-shake v2迁移过程中，常见问题及解决方法请参考[redis-shake 2.x 常见问题](#)。

迁移后 Redis 数据一致性校验

迁移完成后，您可以对Redis数据做一致性校验。

说明

- Redis迁移已完成或者已经进入增量状态。
- 需要ECS实例部署Redis-Full-Check开源工具，并确保与源端、目标端网络互通。
- 如果迁移任务处于增量状态，由于迁移源端和目标端间存在网络时延，无法确保校验时数据一致，如有条件，建议停止对源端的写入后再做校验。
- 业务使用Redis时往往存在给key设置过期时间，迁移时key过期行为会影响数据一致性，如果检验结果不一致，可能是由于key过期时间不一致导致。
- 云厂商DTS在迁移过程中会往目标端Redis写入临时探活key，校验时可能发现非业务数据，属于正常现象。

操作步骤：

- 步骤1 登录ECS实例，确保能连上源端和目标端Redis。
- 步骤2 部署[redis-full-check](#)工具。
- 步骤3 执行以下命令进行数据校验。

```
/redis-full-check -s {源端IP}:{源端端口} -p {源端密码} -t {目标端IP}:{目标端端口} -a
{目标端密码} -m 1
```

表 4-32 参数说明

| 参数 | 参数               | 参数示例             |
|----|------------------|------------------|
| -s | 源端Redis连接地址和端口。  | -s 10.0.0.1:6379 |
| -p | 源端Redis密码。       | -                |
| -t | 目标端Redis连接地址和端口。 | -t 10.0.0.2:6379 |
| -a | 目标端Redis密码。      | -                |

| 参数 | 参数                                                                                            | 参数示例         |
|----|-----------------------------------------------------------------------------------------------|--------------|
| -m | 校验模式：<br>1. 全量校验所有键值对。<br>2. 仅校验值的长度。<br>3. 仅校验key完整性。<br>4. 全量校验所有键值，但只校验大key的长度。<br>默认为模式2。 | -m 1         |
| -q | 最大qps限制，默认15000。                                                                              | -q 5000      |
| -d | 校验结果保存的文件名称，默认result.db。                                                                      | -d result.db |

#### 步骤4 查看校验结果文件。

校验默认执行三轮，输出三个校验结果文件，一般只需要看最后一个校验结果文件。

- 执行sqlite3 result.db.3。
- 执行select \* from key。
- 查看是否存在异常的key。

```
Enter ".help" for usage hints.
sqlite> select * from key;
1|b|string|lack_target|0|1|0
2|c|string|lack_target|0|1|0
3|a|string|lack_target|0|1|0
sqlite>
```

----结束

## 4.4.7 使用 Redis-Shake 工具将 RDB 文件/AOF 文件 导入到 GeminiDB Redis

### RDB 导入 GeminiDB Redis

#### 步骤1 部署迁移工具。

1. 获取工具[Redis-Shake](#)。

##### 📖 说明

Redis-Shake工具可下载release版本，解压缩后即可使用。

2. 修改配置文件“Redis-Shake.conf”，修改时需要注意以下配置项。

**log.level = info** # 默认日志级别，info级别可以打印出相应的迁移进度，可以用来判断迁移是否结束。

**source.rdb.input = /xx/xx.rdb** # 数据源RDB文件绝对路径。

**target.address = <host>:6379** # 目标端地址。



**target.password\_raw = \*\*\*\*\*** # 目标端口令。

**target.version = 5.0** # 目标端redis版本。

**target.type = standalone** # 目标端类型。

**target.db = -1** # 将所有数据迁移到目标端GeminiDB Redis的指定数据库上，若设置为-1，则迁移后的数据库和源实例的DB保持相同的映射关系。

**target.dbmap = #** 配置db迁移映射关系，使用时target.db要为-1，例如 0-5;1-3 表示源端 db0 的数据会被写入目的端 db5，源端 db1 的数据会被写入目的端 db3

**big\_key\_threshold = 52428800** # 设置大key阈值，若key对应的value字节超过给定的阈值，会分批依次写入。

**resume\_from\_break\_point = false** # 关闭断点续传，该功能无法使用

3. 按需选配是否覆盖目标端数据。

**key\_exists = none**

#### 说明

当源端和目的端有重复key时需要选配是否覆盖目标端数据，可配置的值为：

- **rewrite**：源端覆盖目的端。
- **none**：一旦发生进程直接退出。
- **ignore**：保留目的端key，忽略源端的同步key。该值在rump模式下不会生效。

由于本次迁移源端是RDB文件，因此预期无重复数据，建议选择none。如迁移过程异常退出，您可以在管理控制台右上角，选择“[工单](#) > [新建工单](#)”，请联系客服处理。

### 步骤2 进行数据迁移。

启动迁移命令：

```
./redis-shake.linux -conf=redis-shake.conf -type=restore
```

#### 说明

由于数据源为RDB文件，因此使用restore模式。

迁移结束后停止进程。

### 步骤3 数据校验。

由于数据源为RDB文件，因此需要用户从业务角度对目标端GeminiDB Redis数据进行检查。

----结束

## AOF 导入 GeminiDB Redis

**步骤1** 用户上传数据源AOF文件到ECS上。

**步骤2** 在ECS上启动开源Redis 5.0单机进程，加载AOF并等待启动完成。开源Redis的启动目录需为AOF文件所在的目录。

**步骤3** 通过SAVE命令生成数据快照，等待RDB文件生成完毕。RDB文件在Redis的启动目录下。

**步骤4** 停止开源Redis 5.0单机进程。

**步骤5** 按照[RDB导入GeminiDB Redis](#)进行迁移。

----结束



注：使用redis-shake v2迁移过程中，常见问题及解决方法请参考[redis-shake 2.x 常见问题](#)。

## 4.4.8 使用数据导入功能将 RDB 文件恢复到 GeminiDB Redis(推荐)

### 操作场景

当前GeminiDB Redis支持将其他云厂商Redis、自建Redis的数据通过GeminiDB控制台导入到GeminiDB Redis。

您需要先将其他云厂商Redis、自建Redis的数据备份下载到本地，然后将备份数据文件上传与GeminiDB Redis实例同一租户下相同Region下的OBS桶中，最后在GeminiDB控制台创建数据导入任务，将数据导入到GeminiDB Redis实例中。

### 使用须知

- 数据导入会覆盖数据库数据，请谨慎操作。
- 将高版本Redis实例生成的备份文件导入低版本Redis实例可能失败。
- 备份导入前，请确保目标Redis已禁用高消耗命令，如FLUSHALL、KEYS、HGETALL等。
- 如果备份文件中包含多DB数据，其使用的DB数不能超过目标Redis支持的最大DB数。
- 只支持添加.rdb格式的备份文件。

### 创建 OBS 桶并上传备份文件

如果上传的备份文件大于5GB，请按照OBS服务提供的[超过5GB如何上传](#)操作指导执行。

如果上传的备份文件较小，且小于5GB，请执行如下步骤：

#### 步骤1 创建OBS桶。

在创建过程中，以下两个参数请按要求设置，其他详细的创建步骤，请参考《对象存储服务用户指南》的“[创建桶](#)”章节。

1. 选择“区域”。  
OBS桶所在区域必须跟Redis目标实例所在区域相同。
2. 设置“存储类别”，当前支持“标准存储”、“低频访问存储”和“归档存储”。  
请不要选择“归档存储”，否则会导致备份文件数据导入失败。
3. 设置完成后，单击“立即创建”，等待OBS桶创建完成。

**步骤2** 在OBS管理控制台的桶列表中，单击[步骤1](#)中创建的OBS桶名称，进入“概览”页面。

**步骤3** 在左侧导航栏，单击“对象”。

**步骤4** 在“对象”页签下，单击“上传对象”，系统弹出“上传对象”对话框。

**步骤5** 指定对象的存储类别。

请不要选择“归档存储”，否则会导致备份文件数据导入失败。

**步骤6** 上传对象。

您可以拖拽本地文件或文件夹至“上传对象”区域框内添加待上传的文件，也可以通过单击“上传对象”区域框内的“添加文件”，选择本地文件添加。

单次最多支持100个文件同时上传，总大小不超过5GB。

图 4-78 批量上传对象



- 步骤7** 可选：勾选“KMS加密”，用于加密上传文件。
- 步骤8** 单击“上传”，完成备份文件上传到OBS桶。
- 结束

备份文件数据导入

- 步骤1** 登录[云数据库GeminiDB控制台](#)。
- 步骤2** 在“实例管理”页面，选择目标实例，单击操作列“更多 > 数据导入”。

图 4-79 数据导入



- 步骤3** 在“数据导入”页面，在“OBS桶名”中选择已上传备份文件的OBS桶。
- 步骤4** 单击“添加备份文件”，选择需要导入的备份文件。

图 4-80 添加备份文件



- 如果需要导入多个备份文件，可以添加多个备份文件同时导入。一次最多可同时添加128个备份文件。
- 如果需要删除某个备份文件，单击目标备份文件操作列“删除”。
- 如果需要删除所有备份文件，在“备份文件”中单击“清空”。

**步骤5** 备份文件添加完成后，单击“立即创建”。

**步骤6** 确认进行数据导入，单击“确定”，开始数据导入任务。

**注意**

数据导入会覆盖数据库数据，请谨慎操作。

图 4-81 确定数据导入



----结束

4.4.9 Kvrocks 到 GeminiDB Redis 的迁移

Kvrocks是一款开源的兼容Redis生态的NoSQL key-value数据库，底层基于RocksDB实现，并提供namespace功能支持数据分区。Kvrocks集群管理功能相对薄弱，自建集群时需要与外部组件配合，Kvrocks支持的redis命令还不够全面，例如缺少在消息流和统计场景经常使用的stream及hyperloglog数据结构。

GeminiDB Redis接口是一款兼容Redis生态的云原生NoSQL数据库，基于共享存储池的多副本强一致机制，保证数据的安全可靠。GeminiDB Redis接口具有高兼容、高性价比、高可靠、弹性伸缩、高可用、无损扩容等特点。不亚于RedisCluster的兼容性，

使用户在应用时无需修改代码，可直接使用，100%兼容原生接口。GeminiDB Redis接口在适配Kvrocks业务的同时，还能克服管理能力弱、对Redis兼容度不高等缺点。

本章节主要介绍Kvrocks到GeminiDB Redis接口的迁移方案。

## 迁移原理

使用开源工具kvrocks2redis进行Kvrocks到GeminiDB Redis的迁移，在此基础上，从GeminiDB Redis源码层面对Kvrocks的namespace功能进行适配。

迁移过程分为全量和增量两个阶段：迁移开始后，先进行全量迁移，此时对kvrocks打快照，并记录对应的数据版本（seq）。然后解析全量数据文件成redis命令写入GeminiDB Redis。全量迁移完成后进入持续的增量迁移过程，迁移工具循环给Kvrocks发送PSYNC命令，将获取到的增量数据不断转发给GeminiDB Redis，完成增量迁移。

## 使用须知

- kvrocks2redis需要从Kvrocks提取数据到本地文件，并从中解析出命令发送到目标端GeminiDB Redis，该过程中可能影响源端性能，但理论上不会有数据受损风险。
- 迁移工具运行过程中，若出现问题，迁移工具会自动停止，方便问题定位。
- GeminiDB Redis接口从安全性角度出发，不提供清库语义命令，因此要在迁移开始前确保无数据。

## 前提条件

- 部署kvrocks2redis到独立主机。
- 确保源端、目标端、迁移工具之间网络互通。
- 源端Kvrocks实例提前做好数据备份。
- 目标端GeminiDB Redis实例清空全部数据。

## 操作步骤

如需进行Kvrocks到GeminiDB Redis的迁移，您可以在管理控制台右上角，选择“[工单 > 新建工单](#)”，联系客服进行处理。

## 迁移后 Redis 数据一致性校验

迁移完成后，您可以对Redis数据做一致性校验。

### 说明

- Redis迁移已完成或者已经进入增量状态。
- 需要ECS实例部署Redis-Full-Check开源工具，并确保与源端、目标端网络互通。
- 如果迁移任务处于增量状态，由于迁移源端和目标端间存在网络时延，无法确保校验时数据一致，如有条件，建议停止对源端的写入后再做校验。
- 业务使用Redis时往往存在给key设置过期时间，迁移时key过期行为会影响数据一致性，如果检验结果不一致，可能是由于key过期时间不一致导致。
- 云厂商DTS在迁移过程中会往目标端Redis写入临时探活key，校验时可能发现非业务数据，属于正常现象。

### 操作步骤：

**步骤1** 登录ECS实例，确保能连上源端和目标端Redis。

**步骤2** 部署redis-full-check工具。

**步骤3** 执行以下命令进行数据校验。

```
/redis-full-check -s {源端IP}:{源端端口} -p {源端密码} -t {目标端IP}:{目标端端口} -a {目标端密码} -m 1
```

表 4-33 参数说明

| 参数 | 参数                                                                                            | 参数示例             |
|----|-----------------------------------------------------------------------------------------------|------------------|
| -s | 源端Redis连接地址和端口。                                                                               | -s 10.0.0.1:6379 |
| -p | 源端Redis密码。                                                                                    | -                |
| -t | 目标端Redis连接地址和端口。                                                                              | -t 10.0.0.2:6379 |
| -a | 目标端Redis密码。                                                                                   | -                |
| -m | 校验模式：<br>1. 全量校验所有键值对。<br>2. 仅校验值的长度。<br>3. 仅校验key完整性。<br>4. 全量校验所有键值，但只校验大key的长度。<br>默认为模式2。 | -m 1             |
| -q | 最大qps限制，默认15000。                                                                              | -q 5000          |
| -d | 校验结果保存的文件名称，默认result.db。                                                                      | -d result.db     |

**步骤4** 查看校验结果文件。

校验默认执行三轮，输出三个校验结果文件，一般只需要看最后一个校验结果文件。

- 执行sqlite3 result.db.3。
- 执行select \* from key。
- 查看是否存在异常的key。

```
Enter ".help" for usage hints.
sqlite> select * from key;
1|b|string|lack_target|0|1|0
2|c|string|lack_target|0|1|0
3|a|string|lack_target|0|1|0
sqlite> |
```

----结束

## 4.4.10 Pika 到 GeminiDB Redis 的迁移

Pika是一个可持久化的大容量Redis存储服务，解决了Redis由于存储数据量巨大而导致内存不够用的容量瓶颈。但其集群管理功能较为薄弱，需要使用twemproxy或者codis实现静态数据分片。同时由于数据全部存储在磁盘中，相比于社区版Redis，性能明显下降。

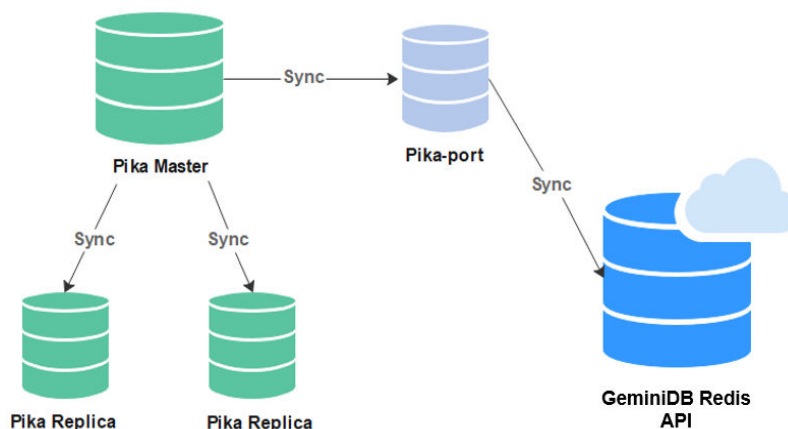
GeminiDB Redis接口是一款兼容Redis生态的云原生NoSQL数据库，基于共享存储池的多副本强一致机制，保证数据的安全可靠。GeminiDB Redis接口实现了冷热分离，解决了缓存（cache）与数据库（Data Base，DB）之间交互访问的问题，提高了程序可读性与程序运行效率。同时对RocksDB进行深度定制，实现秒级分裂弹性扩容，扩容无需搬迁数据，快速而平滑。通过proxy代理，使上层业务可以不感知内核处理扩容过程中的数据迁移。

本章节主要介绍Pika到GeminiDB Redis接口的迁移方案。

### 迁移原理

pika-port伪装成Pika的从节点运行，通过主从复制的方式进行数据迁移。Pika主节点通过比较pika-port和自己的binlog偏移量判断做全量迁移还是增量迁移。如果需要做全量迁移，Pika主节点会将全量数据快照发送给pika-port，pika-port将解析后的快照数据发送给GeminiDB Redis。全量迁移结束后进入增量迁移，pika-port将增量数据解析后以redis命令的形式发送给GeminiDB Redis。

图 4-82 迁移原理



Pika-migrate的迁移原理和pika-port相似，将工具虚拟为pika的从库，然后从主库获取到数据转发给目标redis，同时支持增量同步，实现在线热迁的功能。

pika-migrate 通过 dbsync 请求获取主库全量 DB 数据，以及当前 DB 数据所对应的 binlog 点位。获取到主库当前全量 DB 数据之后，扫描 DB，将 DB 中的数据打包转发给 Redis。通过之前获取的 binlog 的点位向主库进行增量同步，在增量同步的过程中，将从主库获取到的 binlog 重组成redis命令，转发给redis。

### 适用版本

Pika到Redis迁移工具Pika-Port适用于Pika v2.x和v3.0.x版本，具体使用步骤请参考[pika-port迁移工具使用说明](#)或者[pika到pika、redis迁移工具](#)。如果使用Pika v3.2及以上版本，需使用迁移工具Pika-Migrate，具体步骤请参考[pika-migrate迁移工具使用说明](#)。

## 使用须知

- Pika迁移工具伪装成源端Pika的从节点，只读取全量和增量数据，无数据受损风险。
- 源端增加了和Pika迁移工具的主从同步流程，可能会影响源端性能。
- 全量和增量结合迁移可以不停服，业务切入GeminiDB Redis时短暂停服。

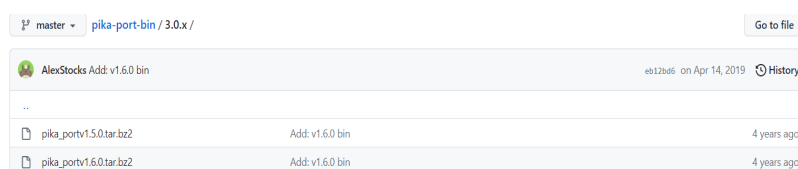
## pika-port 迁移工具使用说明

- 部署迁移工具pika-port

- 迁移工具下载

从Github上直接下载编译好的对应pika版本的pika-port开源迁移工具（如pika3.0.x版本就使用3.0.x版本的开源工具），网址为：<https://github.com/ipixiu/pika-port-bin>。

图 4-83 迁移工具下载



- 部署位置和目录结构

以pika\_portv1.6.0为例，将解压后的整个文件重命名为pika\_port3，复制在部署了pika的环境上（或者任一能与pika实例和GeminiDB Redis实例网络互通的服务器上），解压后的pika\_port3文件夹目录结构如下：

```
[root@pika_port3]# ls
lib log logs rsync_dump sbin start.sh
```

- 迁移前准备

由于sbin目录下的pika-port二进制文件需要引用lib目录下的库文件，迁移之前需在pika\_port3路径执行如下命令：

```
export LD_LIBRARY_PATH=./lib:${LD_LIBRARY_PATH} //链接库文件
rm -rf ./rsync_dump/ //删除文件夹
rm -rf ./sync_log/ //删除sync日志
```

- pika单机数据迁移

- 启动命令

执行如下命令，启动pika\_port迁移工具：

```
./sbin/pika_port -i master_ip -o master_port -m forward_ip -n
forward_port -x forward_thread_num -y forward_passwd -f filenum -s
offset -w password -r rsync_dump_path -l log_path
```

-h -- show this help

-t -- local host ip(OPTIONAL default: 127.0.0.1)，输入本机IP

-p -- local port(OPTIONAL)，默认即可

-i -- Pika主节点IP

-o -- Pika主节点port

-m -- GeminiDB Redis ELB IP

-n -- GeminiDB Redis port



-x -- 发送线程数量(OPTIONAL default: 1)  
-y -- GeminiDB Redis密码  
-f -- binlog文件数量, 默认即可(OPTIONAL default: local offset)  
-s -- binlog offset, 默认即可(OPTIONAL default: local offset)  
-w -- Pika主节点密码  
-r -- rsync dump文件到本地目录, 默认即可(OPTIONAL default: ./rsync\_dump)  
-l -- 产生的日志文件目录, 默认即可(OPTIONAL default: ./log)  
-b -- max batch number when port rsync dump data. 默认即可(OPTIONAL default: 512)  
-d -- 是否后台运行(OPTIONAL)

示例:

```
./sbin/pika_port -t 28.80.60.200 -p 12345 -i 28.80.60.200 -o 9221 -m 28.80.60.201 -n 6379 -x 7 -y a -w a
```

【注】上例中, 28.80.60.200为源端pika IP, 28.80.60.201为目标端redis IP。

#### - 等待结果

等待数据迁移, 若执行日志中出现如下信息, 代表全量数据同步完成, 此后数据迁移进入增量同步阶段。

```
trysync_thread.cc-Retransmit-280: Open strings DB ./rsync_dump/strings result OK
trysync_thread.cc-Retransmit-288: Open lists DB ./rsync_dump/lists result OK
trysync_thread.cc-Retransmit-296: Open hashes DB ./rsync_dump/strings result OK
trysync_thread.cc-Retransmit-304: Open sets DB ./rsync_dump/sets result OK
trysync_thread.cc-Retransmit-312: Open zsets DB ./rsync_dump/zsets result OK
migrator_thread.cc-ThreadMain-364: blackwidow::kStrings keys have been dispatched completly
migrator_thread.cc-ThreadMain-364: blackwidow::kHashes keys have been dispatched completly
migrator_thread.cc-ThreadMain-364: blackwidow::kZSets keys have been dispatched completly
migrator_thread.cc-ThreadMain-364: blackwidow::kSets keys have been dispatched completly
migrator_thread.cc-ThreadMain-364: blackwidow::kLists keys have been dispatched completly
```

#### • pika集群数据迁移

迁移步骤同pika单机数据迁移步骤, 为每个pika实例创建一个工作流即可。

## pika-migrate 迁移工具使用说明

#### • 迁移工具部署

- Pika-migrate工具下载: [https://github.com/OpenAtomFoundation/pika/tree/v3\\_2\\_7\\_migrate](https://github.com/OpenAtomFoundation/pika/tree/v3_2_7_migrate)
- 工具使用说明: [https://github.com/OpenAtomFoundation/pika/blob/v3\\_2\\_7\\_migrate/pika-migrate.md](https://github.com/OpenAtomFoundation/pika/blob/v3_2_7_migrate/pika-migrate.md)
- 也可以直接使用编译好的二进制文件。

将解压后的整个pika\_migrate\_output文件放置在部署了pika的环境上(或者任一能与pika实例和GeminiDB Redis实例网络互通的服务器上), 解压后的文件夹目录结构如下:

```
[root@csy output]# ls
bin conf db dbsync log
```

使用前清空db、dbsync和log目录下的内容。

#### • 迁移步骤

##### a. 修改配置文件

根据目标端(redis)的信息, 修改迁移工具的配置文件conf/pika.conf 中的如下参数:



```
target-redis-host :
target-redis-port : 6379
target-redis-pwd :
```

target-redis-host: Redis 的 IP 地址。

target-redis-port: Redis 的端口号。

target-redis-pwd: Redis 默认账号的密码。

masterauth: Pika主库的密码（如Pika主库没有设置密码则不填）。

---

 **注意**

- binlog文件为log/log\_db0目录下记录了所有操作的同步日志文件，这部分文件在增量同步数据阶段使用
- 将全量数据写入到Redis这段时间可能耗时很长，而导致Pika主库原先的binlog文件被清理。需要在Pika主库上保留足够的binlog文件个数，确保后续该工具请求增量同步的时候，对应的binlog文件还存在。
- binlog文件占用磁盘空间，可以根据观察Pika主库实际生成该文件的情况确定保留binlog的数量。

- 
- b. 在pika主库上执行“**config set expire-logs-nums 10000**”命令，让PIKA主库保留10000个 binlog 文件。

```
127.0.0.1:9221> config set expire-logs-nums 10000
OK
```

- c. 在工具包的路径下执行如下命令，启动 pika-migrate 工具，并查看控制台回显信息: bin/pika -c conf/pika.conf。

```
[root@~]# bin/pika -c conf/pika.conf
path : conf/pika.conf
-----Pika server 3.2.7 -----
-----Pika config list-----
 1 port 9222
 2 thread-num 1
 3 thread-pool-size 12
 4 sync-thread-num 6
 5 log-path ./log/
 6 db-path ./db/
 7 write-buffer-size 268435456
 8 timeout 60
 9 requirepass
10 masterauth
11 userpass
12 userblacklist
13 instance-mode classic
14 databases 1
15 default-slot-num 1024
16 dump-prefix
17 dump-path ./dump/
18 dump-expire 0
19 pidfile ./pika.pid
20 maxclients 20000
21 target-file-size-base 20971520
22 expire-logs-days 7
23 expire-logs-nums 10
24 root-connection-num 2
25 slowlog-write-errorlog no
26 slowlog-log-slower-than 10000
27 slowlog-max-len 128
28 db-sync-path ./dbsync/
29 db-sync-speed -1
30 slave-priority 100
31 server-id 1
32 sync-window-size 9000
33 target-redis-host
34 target-redis-port 6379
35 target-redis-pwd a
36 sync-batch-num 100
37 redis-sender-num 10
38 write-binlog yes
39 binlog-file-size 104857600
40 max-cache-statistic-keys 0
41 small-compaction-threshold 5000
42 max-write-buffer-size 10737418240
43 max-client-response-size 1073741824
44 compression snappy
45 max-background-flushes 1
46 max-background-compactions 2
47 max-cache-files 5000
48 max-bytes-for-level-multiplier 10
-----Pika config end-----
```

```
I20231010 17:53:17.996544 3193940 pika.cc:187] Server at: conf/pika.conf
I20231010 17:53:17.996986 3193940 pika_server.cc:188] Using Networker Interface: eth0
I20231010 17:53:17.997085 3193940 pika_server.cc:231] host: 26.84.49.223 port: 9222
I20231010 17:53:17.997107 3193940 pika_server.cc:89] Worker queue limit is 20100
I20231010 17:53:19.229496 3193940 pika_partition.cc:92] db0 DB Success
I20231010 17:53:19.229542 3193940 pika_binlog.cc:106] Binlog: Find the exist file.
I20231010 17:53:19.230276 3193987 redis_sender.cc:169] Start redis sender 0 thread...
I20231010 17:53:19.230368 3193989 redis_sender.cc:169] Start redis sender 2 thread...
I20231010 17:53:19.230389 3193990 redis_sender.cc:169] Start redis sender 3 thread...
I20231010 17:53:19.230382 3193988 redis_sender.cc:169] Start redis sender 1 thread...
I20231010 17:53:19.230484 3193991 redis_sender.cc:169] Start redis sender 4 thread...
I20231010 17:53:19.230520 3193992 redis_sender.cc:169] Start redis sender 5 thread...
I20231010 17:53:19.230556 3193993 redis_sender.cc:169] Start redis sender 6 thread...
I20231010 17:53:19.230618 3193994 redis_sender.cc:169] Start redis sender 7 thread...
I20231010 17:53:19.230659 3193995 redis_sender.cc:169] Start redis sender 8 thread...
I20231010 17:53:19.230665 3193940 pika_server.cc:291] Pika Server going to start
I20231010 17:53:19.230684 3193996 redis_sender.cc:169] Start redis sender 9 thread...
```

- d. 在pika-migrate客户端执行如下命令，将迁移工具伪装成 Slave，向主库请求同步，并观察控制台是否打印报错信息。

**slaveof [pika主库ip] [pika主库port]**

```
127.0.0.1:9222> slaveof 127.0.0.1 9221
OK
```

如果主从关系建立，此时回显信息为：

```
-----Pika config end-----
I20231010 17:53:17.996544 3193940 pika.cc:187] Server at: conf/pika.conf
I20231010 17:53:17.996986 3193940 pika_server.cc:188] Using Networker Interface: eth0
I20231010 17:53:17.997085 3193940 pika_server.cc:231] host: 26.84.49.223 port: 9222
I20231010 17:53:17.997107 3193940 pika_server.cc:89] Worker queue limit is 20100
I20231010 17:53:19.229496 3193940 pika_partition.cc:92] db0 DB Success
I20231010 17:53:19.229542 3193940 pika_binlog.cc:106] Binlog: Find the exist file.
I20231010 17:53:19.230276 3193987 redis_sender.cc:169] Start redis sender 0 thread...
I20231010 17:53:19.230368 3193989 redis_sender.cc:169] Start redis sender 2 thread...
I20231010 17:53:19.230389 3193990 redis_sender.cc:169] Start redis sender 3 thread...
I20231010 17:53:19.230382 3193988 redis_sender.cc:169] Start redis sender 1 thread...
I20231010 17:53:19.230484 3193991 redis_sender.cc:169] Start redis sender 4 thread...
I20231010 17:53:19.230520 3193992 redis_sender.cc:169] Start redis sender 5 thread...
I20231010 17:53:19.230556 3193993 redis_sender.cc:169] Start redis sender 6 thread...
I20231010 17:53:19.230618 3193994 redis_sender.cc:169] Start redis sender 7 thread...
I20231010 17:53:19.230659 3193995 redis_sender.cc:169] Start redis sender 8 thread...
I20231010 17:53:19.230665 3193940 pika_server.cc:291] Pika Server going to start
I20231010 17:53:19.230684 3193996 redis_sender.cc:169] Start redis sender 9 thread...
I20231010 17:54:13.573770 3193986 pika_rep_client.cc:145] Try Send Meta Sync Request to Master (26.84.49.223:9221)
I20231010 17:54:13.574483 3193942 pika_server.cc:570] Mark try connect finish
I20231010 17:54:13.574512 3193942 pika_rep_client_conn.cc:129] Finish to handle meta sync response
I20231010 17:54:13.674422 3193943 pika_rep_client_conn.cc:210] Partition: db0 TrySync Ok
I20231010 17:54:13.728996 3193950 pika_partition.cc:509] db0 Delete old db...
I20231010 17:54:13.860172 3193950 pika_partition.cc:523] db0 Open new db success
I20231010 17:54:13.860237 3193997 pika_server.cc:36] Delete dir: ./db/db0_deleting/ start
I20231010 17:54:13.861198 3193997 pika_server.cc:38] Delete dir: ./db/db0_deleting/ done
```

- e. 确认主从关系建立成功之后，pika-migrate开始向目标端Redis转发数据。在pika主库和从库执行“info Replication”命令，查看主从同步延迟。可在主库写入一个特殊的 key，然后在 Redis 侧查看是否可立即获取到该 key，判断数据同步完毕。

```
127.0.0.1:9221> info replication
Replication(MASTER)
role:master
connected_slaves:0
db0 binlog offset=0 1258,safety purge=none
```

```
127.0.0.1:9222> info replication
Replication(SLAVE)
role:slave
master_host:26.84.49.223
master_port:9221
master_link_status:up
slave_priority:100
slave_read_only:1
db0 binlog_offset=0 352,safety_purge=none
```

- 注意事项

- pika-migrate在线迁移只支持单DB场景，不支持多DB场景。
- pika-migrate在线迁移不支持集群模式：严格来说，pika进程有两种模式启动，一种是classic，一种是sharding，其中sharding模式不支持pika-migrate迁移。

具体模式可以在pika.conf里进行查看：

```
if this option is set to 'classic', that means pika support multiple DB, in
this mode, option databases enable
if this option is set to 'sharding', that means pika support multiple Table, you
can specify slot num for each table, in this mode, option default-slot-num enable
Pika Instance mode [classic | sharding]
instance-mode : classic
```

- 常见问题

- 如果从库一直未收到主库响应，会一直停留在Need To Sync阶段：

```
I1009 17:30:58.595940 1257316 pika_partition.cc:92] db0 DB Success
I1009 17:30:58.595970 1257316 pika_binlog.cc:106] Binlog: Find the exist file.
I1009 17:30:58.596279 1257316 pika_server.cc:291] Pika Server going to start
I1009 17:30:58.596341 1257491 redis_sender.cc:169] Start redis sender 0 thread...
I1009 17:31:18.413722 1257490 pika_repl_client.cc:145] Try Send Meta Sync Request to Master (10.288.46.91:5824)
I1009 17:31:18.422734 1257320 pika_server.cc:570] Mark try connect finish
I1009 17:31:18.422753 1257320 pika_repl_client_conn.cc:129] Finish to handle meta sync response
I1009 17:31:18.517016 1257321 pika_repl_client_conn.cc:166] Partition: (db0:0) Need Wait To Sync
```

- 可能原因

- pika主库未使用root启动导致rsync无权限，请参考网址：<https://github.com/OpenAtomFoundation/pika/issues/553>。
- rsync使用端口未开启监听，请开启监听。
- rsync使用端口已被其他进程占用，请解除端口占用。

```
tcp 0 0 0.0.0.0:10221 0.0.0.0:* LISTEN 3193295/rsync
tcp 0 0 0.0.0.0:10222 0.0.0.0:* LISTEN 3193959/rsync
```

## 迁移后 Redis 数据一致性校验

迁移完成后，您可以对Redis数据做一致性校验。



 说明

- Redis迁移已完成或者已经进入增量状态。
- 需要ECS实例部署Redis-Full-Check开源工具，并确保与源端、目标端网络互通。
- 如果迁移任务处于增量状态，由于迁移源端和目标端间存在网络时延，无法确保校验时数据一致，如有条件，建议停止对源端的写入后再做校验。
- 业务使用Redis时往往存在给key设置过期时间，迁移时key过期行为会影响数据一致性，如果检验结果不一致，可能是由于key过期时间不一致导致。
- 云厂商DTS在迁移过程中会往目标端Redis写入临时探活key，校验时可能发现非业务数据，属于正常现象。

操作步骤：

**步骤1** 登录ECS实例，确保能连上源端和目标端Redis。

**步骤2** 部署redis-full-check工具。

**步骤3** 执行以下命令进行数据校验。

```
/redis-full-check -s {源端IP}:{源端口} -p {源端密码} -t {目标端IP}:{目标端口} -a {目标端密码} -m 1
```

表 4-34 参数说明

| 参数 | 参数                                                                                            | 参数示例             |
|----|-----------------------------------------------------------------------------------------------|------------------|
| -s | 源端Redis连接地址和端口。                                                                               | -s 10.0.0.1:6379 |
| -p | 源端Redis密码。                                                                                    | -                |
| -t | 目标端Redis连接地址和端口。                                                                              | -t 10.0.0.2:6379 |
| -a | 目标端Redis密码。                                                                                   | -                |
| -m | 校验模式：<br>1. 全量校验所有键值对。<br>2. 仅校验值的长度。<br>3. 仅校验key完整性。<br>4. 全量校验所有键值，但只校验大key的长度。<br>默认为模式2。 | -m 1             |
| -q | 最大qps限制，默认15000。                                                                              | -q 5000          |
| -d | 校验结果保存的文件名称，默认result.db。                                                                      | -d result.db     |

**步骤4** 查看校验结果文件。

校验默认执行三轮，输出三个校验结果文件，一般只需要看最后一个校验结果文件。

- 执行sqlite3 result.db.3。
- 执行select \* from key。
- 查看是否存在异常的key。

```
Enter ".help" for usage hints.
sqlite> select * from key;
1|b|string|lack_target|0|1|0
2|c|string|lack_target|0|1|0
3|a|string|lack_target|0|1|0
sqlite> |
```

----结束

## 迁移性能参考

- 环境：Pika（单节点）和pika-port同时部署在华为云8U32GB的弹性云服务器上，目标端为8U16GB，3节点GeminiDB Redis实例。
- 预置数据：使用memtier\_benchmark工具预置200GB数据。
- 迁移性能：约50000qps。

### 4.4.11 SSDB 到 GeminiDB Redis 的迁移

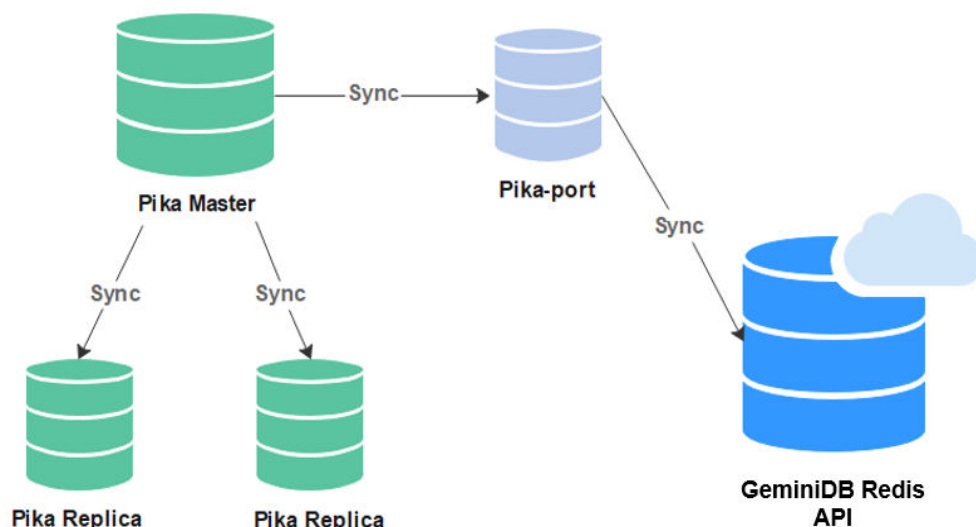
SSDB是一款使用C/C++语言开发的高性能NoSQL数据库，和Redis具有相似的API，支持KV，list，map(hash)，zset(sorted set)，qlist(队列)等数据结构，因此得到了广泛的应用。SSDB是一个持久化的KV存储系统，底层使用leveldb作为存储引擎。其业务直接与LevelDB交互，Compaction等操作会对业务读写造成直接的影响。GeminiDB Redis接口是一款兼容Redis生态的云原生NoSQL数据库，基于共享存储池的多副本强一致机制，以保证数据的安全性和可靠性。GeminiDB Redis接口使用RocksDB作为存储引擎，其性能与leveldb相比有了很大的提升，并解决了leveldb主动限制写的问题，同时实现了冷热分离，减小了存储层的操作对性能造成的影响。

本章节主要介绍SSDB到GeminiDB Redis接口的迁移方案。

## 迁移原理

ssdb-port作为源端SSDB数据库的主节点的从节点（replica）运行，通过主从复制的方式进行数据迁移。将获取到的数据解析、转换为Redis支持的格式，并发送到配置文件中指定的Redis实例，迁移过程如下图所示。全量同步完成后，SSDB中新增的数据也会同步到Redis实例中。

图 4-84 迁移原理



## 使用须知

- ssdb-port作为SSDB主节点的从节点，只读取全量和增量数据，无数据受损风险。
- 由于在源端使用ssdb-port迁移工具，源端SSDB性能会受到一定的影响。
- 全量迁移和增量迁移可以不停服，数据全部迁入GeminiDB Redis后需要短暂停服。

## 前提条件

在GeminiDB Redis实例所在的VPC网络中创建ECS实例，部署迁移工具ssdb-port，保证源端SSDB实例和目标端GeminiDB Redis实例网络互通。

## 操作步骤

如需进行SSDB到GeminiDB Redis的迁移，您可以在管理控制台右上角，选择“[工单 > 新建工单](#)”，联系客服进行处理。

## 迁移后 Redis 数据一致性校验

迁移完成后，您可以对Redis数据做一致性校验。

### 说明

- Redis迁移已完成或者已经进入增量状态。
- 需要ECS实例部署Redis-Full-Check开源工具，并确保与源端、目标端网络互通。
- 如果迁移任务处于增量状态，由于迁移源端和目标端间存在网络时延，无法确保校验时数据一致，如有条件，建议停止对源端的写入后再做校验。
- 业务使用Redis时往往存在给key设置过期时间，迁移时key过期行为会影响数据一致性，如果检验结果不一致，可能是由于key过期时间不一致导致。
- 云厂商DTS在迁移过程中会往目标端Redis写入临时探活key，校验时可能发现非业务数据，属于正常现象。

### 操作步骤：

**步骤1** 登录ECS实例，确保能连上源端和目标端Redis。

**步骤2** 部署redis-full-check工具。

**步骤3** 执行以下命令进行数据校验。

```
/redis-full-check -s {源端IP}:{源端端口} -p {源端密码} -t {目标端IP}:{目标端端口} -a {目标端密码} -m 1
```

表 4-35 参数说明

| 参数 | 参数                                                                                            | 参数示例             |
|----|-----------------------------------------------------------------------------------------------|------------------|
| -s | 源端Redis连接地址和端口。                                                                               | -s 10.0.0.1:6379 |
| -p | 源端Redis密码。                                                                                    | -                |
| -t | 目标端Redis连接地址和端口。                                                                              | -t 10.0.0.2:6379 |
| -a | 目标端Redis密码。                                                                                   | -                |
| -m | 校验模式：<br>1. 全量校验所有键值对。<br>2. 仅校验值的长度。<br>3. 仅校验key完整性。<br>4. 全量校验所有键值，但只校验大key的长度。<br>默认为模式2。 | -m 1             |
| -q | 最大qps限制，默认15000。                                                                              | -q 5000          |
| -d | 校验结果保存的文件名称，默认result.db。                                                                      | -d result.db     |

**步骤4** 查看校验结果文件。

校验默认执行三轮，输出三个校验结果文件，一般只需要看最后一个校验结果文件。

- 执行sqlite3 result.db.3。
- 执行select \* from key。
- 查看是否存在异常的key。

```
Enter ".help" for usage hints.
sqlite> select * from key;
1|b|string|lack_target|0|1|0
2|c|string|lack_target|0|1|0
3|a|string|lack_target|0|1|0
sqlite>
```

----结束



## 迁移性能参考

- 环境：源端SSDB和ssdb-port同时部署在华为云4U16GB的弹性云服务器上，目标端为8U16GB，3节点GeminiDB Redis实例。
- 预置数据：使用memtier\_benchmark工具预置100GB数据。
- 迁移性能：约3000qps。

### 4.4.12 LevelDB 到 GeminiDB Redis 的迁移

LevelDB是一个开源的持久化KV单机数据库引擎，具有很高的随机写，顺序读/写性能，适合应用在写多读少的场景。其内部没有设计成C/S网络结构，使用时必须和服务部署在同一台服务器，对于服务的部署、使用有较大的限制。相比于在LevelDB基础上开发的RocksDB，LevelDB存在较多缺点，如无法很好地使用多核服务器的计算性能，无法支撑TB级数据存储，不支持从HDFS读取数据等。

GeminiDB Redis接口采用RocksDB作为存储引擎，兼容Redis协议，具有丰富的数据类型，可以满足LevelDB的使用需求。同时GeminiDB Redis接口对RocksDB进行深度定制，实现秒级分裂弹性扩容，扩缩容无需搬迁数据，快速而平滑，为LevelDB业务转到Redis生态提供了便利。

本章节主要介绍LevelDB到GeminiDB Redis接口的迁移方案。

## 迁移原理

- 使用自研迁移工具leveldb-port，和LevelDB部署在相同机器上，准备好配置文件，启动迁移即可自动完成全量与增量的迁移。
- 全量迁移对LevelDB数据进行快照，然后扫描整个数据库，将数据打包成GeminiDB Redis接口识别的格式，发送到GeminiDB Redis接口，具有很高的迁移效率。
- 增量迁移解析LevelDB的wal文件，将LevelDB的操作解析出来，然后对其中的key进行分片，多线程进行发送。

## 使用须知

- 迁移工具需要部署在源端，对性能有一定消耗，可通过修改配置文件进行一定的控制。
- 迁移过程读取LevelDB的源数据文件，只读操作，理论上不会有数据受损风险。
- 迁移过程不需要停服。
- 若迁移过程出现故障，需要清理GeminiDB Redis实例，重新启动迁移。

## 操作步骤

如需进行LevelDB到GeminiDB Redis接口的迁移，您可以在管理控制台右上角，选择“[工单 > 新建工单](#)”，联系客服进行处理。

## 迁移性能参考

- 环境：源端LevelDB和leveldb-port部署在华为云4U16GB的弹性云服务器上，目标端为2U8GB，3节点的GeminiDB Redis实例。
- 全量迁移：预置10GB数据，迁移速度约8MB/s。
- 增量迁移：设置value值大小为1KB，迁移速度约为7000qps。

### 4.4.13 RocksDB 到 GeminiDB Redis 的迁移

RocksDB是FaceBook基于LevelDB开发的一个持久化KV单机数据库引擎，具有强大的顺序读写及随机写性能。相对于LevelDB，RocksDB做了许多优化，性能有了很大提升，而且解决了LevelDB主动限制写的问题。作为一个数据库引擎，RocksDB没有设计成C/S网络结构，直接使用需要和服务部署在同一台服务器，对于服务的部署、使用有较大的限制。

GeminiDB Redis接口采用RocksDB作为存储引擎，兼容Redis协议具有丰富的数据类型，可以满足RocksDB的使用需求。同时GeminiDB Redis接口对RocksDB进行深度定制，实现秒级分裂弹性扩容，扩缩容无需搬迁数据，快速而平滑，为RocksDB业务转到Redis生态提供了便利。

本章节主要介绍RocksDB到GeminiDB Redis接口的迁移方案。

#### 迁移原理

- 使用自研迁移工具rocksdb-port，和RocksDB部署在相同机器上，准备好配置文件，启动迁移即可自动完成全量与增量的迁移。
- 全量迁移对RocksDB数据进行快照，然后扫描整个数据库，将数据打包成GeminiDB Redis识别的格式，发送到GeminiDB Redis，具有很高的迁移效率。
- 增量迁移解析RocksDB的wal文件，将RocksDB的操作解析出来，然后对其中的key进行分片，多线程进行发送。

#### 使用须知

- 迁移工具需要部署在源端，对性能有一定消耗，可通过修改配置文件进行一定的控制。
- 迁移过程读取RocksDB的源数据文件，只读操作，理论上不会有数据受损风险。
- 迁移过程不需要停服。
- 若迁移过程出现故障，需要清理GeminiDB Redis实例，重新启动迁移。

#### 操作步骤

如需进行RocksDB到GeminiDB Redis的迁移，您可以在管理控制台右上角，选择“[工单 > 新建工单](#)”，联系客服进行处理。

### 4.4.14 AWS ElasticCache for Redis 数据库到 GeminiDB Redis 的迁移

#### 迁移原理

AWS ElasticCache for Redis通过备份导出RDB文件后，通过Redis-Shake迁移工具的数据恢复（restore）将数据迁移到GeminiDB Redis。

#### 使用须知

- AWS不支持psync/sync命令，无法增量迁出。
- 迁移前确保部署Redis-shake的ECS与目标端GeminiDB Redis网络通畅。
- 注意放开源端和目标端安全组配置。

## 操作步骤

### 步骤1 部署迁移工具。

1. 获取工具[Redis-Shake](#)。

#### 说明

Redis-Shake工具可下载release版本，解压缩后即可使用。

2. 修改配置文件“Redis-Shake.conf”，修改时需要注意以下配置项。

**log.level = info** # 默认日志级别，info级别可以打印出相应的迁移进度，可以用来判断迁移是否结束。

**source.rdb.input = /xx/xx.rdb** # 数据源RDB文件绝对路径。

**target.address = <host>:6379** # 目标端地址。

**target.password\_raw = \*\*\*\*\*** # 目标端口令。

**target.version = 5.0** # 目标端redis版本。

**target.type = standalone** # 目标端类型。

**target.db = 0** # 数据迁移到目标端GeminiDB Redis 的指定数据库上。默认是db0

**big\_key\_threshold = 1** # 设置大key阈值

3. 按需选配是否覆盖目标端数据。

**key\_exists = none**

#### 说明

当源端和目的端有重复key时需要选配是否覆盖目标端数据，可配置的值为：

- **rewrite**：源端覆盖目的端。
- **none**：一旦发生进程直接退出。
- **ignore**：保留目的端key，忽略源端的同步key。该值在rump模式下不会生效。

由于本次迁移源端是RDB文件，因此预期无重复数据，建议选择none。如迁移过程异常退出，您可以在管理控制台右上角，选择“[工单 > 新建工单](#)”，请联系客服处理。

### 步骤2 进行数据迁移。

启动迁移命令：

```
./redis-shake.linux -conf=redis-shake.conf -type=restore
```

#### 说明

由于数据源为RDB文件，因此使用restore模式。

迁移结束后停止进程。

### 步骤3 数据校验。

由于数据源为RDB文件，因此需要用户从业务角度对目标端GeminiDB Redis数据进行检查。

----结束

## 迁移后 Redis 数据一致性校验

迁移完成后，您可以对Redis数据做一致性校验。

 说明

- Redis迁移已完成或者已经进入增量状态。
- 需要ECS实例部署Redis-Full-Check开源工具，并确保与源端、目标端网络互通。
- 如果迁移任务处于增量状态，由于迁移源端和目标端间存在网络时延，无法确保校验时数据一致，如有条件，建议停止对源端的写入后再做校验。
- 业务使用Redis时往往存在给key设置过期时间，迁移时key过期行为会影响数据一致性，如果检验结果不一致，可能是由于key过期时间不一致导致。
- 云厂商DTS在迁移过程中会往目标端Redis写入临时探活key，校验时可能发现非业务数据，属于正常现象。

操作步骤：

**步骤1** 登录ECS实例，确保能连上源端和目标端Redis。

**步骤2** 部署redis-full-check工具。

**步骤3** 执行以下命令进行数据校验。

```
/redis-full-check -s {源端IP}:{源端口} -p {源端密码} -t {目标端IP}:{目标端口} -a {目标端密码} -m 1
```

表 4-36 参数说明

| 参数 | 参数                                                                                            | 参数示例             |
|----|-----------------------------------------------------------------------------------------------|------------------|
| -s | 源端Redis连接地址和端口。                                                                               | -s 10.0.0.1:6379 |
| -p | 源端Redis密码。                                                                                    | -                |
| -t | 目标端Redis连接地址和端口。                                                                              | -t 10.0.0.2:6379 |
| -a | 目标端Redis密码。                                                                                   | -                |
| -m | 校验模式：<br>1. 全量校验所有键值对。<br>2. 仅校验值的长度。<br>3. 仅校验key完整性。<br>4. 全量校验所有键值，但只校验大key的长度。<br>默认为模式2。 | -m 1             |
| -q | 最大qps限制，默认15000。                                                                              | -q 5000          |
| -d | 校验结果保存的文件名称，默认result.db。                                                                      | -d result.db     |

**步骤4** 查看校验结果文件。

校验默认执行三轮，输出三个校验结果文件，一般只需要看最后一个校验结果文件。

- 执行sqlite3 result.db.3。
- 执行select \* from key。
- 查看是否存在异常的key。

```
Enter ".help" for usage hints.
sqlite> select * from key;
1|b|string|lack_target|0|1|0
2|c|string|lack_target|0|1|0
3|a|string|lack_target|0|1|0
sqlite>
```

----结束

### 4.4.15 迁移后 Redis 数据一致性校验

迁移完成后，您可以对Redis数据做一致性校验。

#### 使用须知

- Redis迁移已完成或者已经进入增量状态。
- 需要ECS实例部署Redis-Full-Check开源工具，并确保与源端、目标端网络互通。
- 如果迁移任务处于增量状态，由于迁移源端和目标端间存在网络时延，无法确保校验时数据一致，如有条件，建议停止对源端的写入后再做校验。
- 业务使用Redis时往往存在给key设置过期时间，迁移时key过期行为会影响数据一致性，如果检验结果不一致，可能是由于key过期时间不一致导致。
- 云厂商DTS在迁移过程中会往目标端Redis写入临时探活key，校验时可能发现非业务数据，属于正常现象。

#### 操作步骤

**步骤1** 登录ECS实例，确保能连上源端和目标端Redis。

**步骤2** 部署[redis-full-check](#)工具。

**步骤3** 执行以下命令进行数据校验。

```
/redis-full-check -s {源端IP}:{源端口} -p {源端密码} -t {目标端IP}:{目标端口} -a {目标端密码} -m 1
```

表 4-37 参数说明

| 参数 | 参数               | 参数示例             |
|----|------------------|------------------|
| -s | 源端Redis连接地址和端口。  | -s 10.0.0.1:6379 |
| -p | 源端Redis密码。       | -                |
| -t | 目标端Redis连接地址和端口。 | -t 10.0.0.2:6379 |
| -a | 目标端Redis密码。      | -                |

| 参数 | 参数                                                                                            | 参数示例         |
|----|-----------------------------------------------------------------------------------------------|--------------|
| -m | 校验模式：<br>1. 全量校验所有键值对。<br>2. 仅校验值的长度。<br>3. 仅校验key完整性。<br>4. 全量校验所有键值，但只校验大key的长度。<br>默认为模式2。 | -m 1         |
| -q | 最大qps限制，默认15000。                                                                              | -q 5000      |
| -d | 校验结果保存的文件名称，默认result.db。                                                                      | -d result.db |

**步骤4** 查看校验结果文件。

校验默认执行三轮，输出三个校验结果文件，一般只需要看最后一个校验结果文件。

- 执行sqlite3 result.db.3。
- 执行select \* from key。
- 查看是否存在异常的key。

```
Enter ".help" for usage hints.
sqlite> select * from key;
1|b|string|lack_target|0|1|0
2|c|string|lack_target|0|1|0
3|a|string|lack_target|0|1|0
sqlite> |
```

----结束

## 4.5 实例管理

### 4.5.1 GeminiDB Redis 实例会话管理

#### 操作场景

GeminiDB Redis通用型支持实例会话，您可以登上现网查看各个节点的连接情况。

#### 使用须知

- 部分低版本的实例需要参考[升级内核小版本](#)升级内核小版本以支持实例会话功能。
- Redis Cluster集群版实例不支持实例会话功能。
- 节点状态为“正常”时，支持实例会话。
- 客户端连接实例后，显示实例会话的内容。

- Cluster集群版实例暂不支持该功能。

 注意

kill会话操作会导致业务断链，需要业务重连机制，请谨慎操作。

查看实例会话

- 步骤1 登录[云数据库GeminiDB控制台](#)。
- 步骤2 在“实例管理”页面，选择指定的实例，单击实例名称。
- 步骤3 在左侧导航树，单击“实例会话”。
- 步骤4 在“实例会话”页面，选择“节点”，查看节点对应的会话信息，具体参数请查看[表4-38](#)。

图 4-85 实例会话



表 4-38 实例会话

| 参数    | 描述                               |
|-------|----------------------------------|
| addr  | 客户端的地址和端口。                       |
| id    | 会话id。                            |
| name  | 连接名。                             |
| cmd   | 最近一次执行的命令。                       |
| age   | 以秒计算的已连接时长。                      |
| idle  | 以秒计算的空闲时长。                       |
| db    | 该客户端正在使用的数据库DB标识，例如DB0，DB1，DB2等。 |
| fd    | 套接字所使用的文件描述符。                    |
| sub   | 已订阅频道的数量。                        |
| psub  | 已订阅模式的数量。                        |
| multi | 在事务中被执行的命令数量。                    |

- 步骤5 您可以根据自己的需要在会话统计中选择“按照来源统计”或“按照DB统计”，具体参数请查看[表4-39](#)。
- 按照来源统计：节点连接的各个客户端连接数汇总，从大到小取前十个（可能出现并列），展示客户端的ip地址和连接总数。

- 按照DB统计：节点各数据库连接的客户端的ip和该ip连接节点的连接数，按连接数从高到低取前十个（可能出现并列）。

图 4-86 会话统计



表 4-39 会话统计

| 参数  | 描述                                                                                                                |
|-----|-------------------------------------------------------------------------------------------------------------------|
| 统计项 | 总client数和活跃client数。 <ul style="list-style-type: none"><li>总client数：总客户端连接数。</li><li>活跃client数：活跃客户端连接数。</li></ul> |
| 结果  | 统计结果。                                                                                                             |
| 来源  | 客户端ip地址。                                                                                                          |
| DB  | GeminiDB Redis数据库DB标识，例如DB0，DB1，DB2等。                                                                             |
| 总数  | 客户端连接数。                                                                                                           |

----结束

自动刷新

“自动刷新”开关按钮默认关闭，您可以通过单击按钮打开自动刷新。自动刷新打开后，默认每10s刷新一次数据，您可以根据自己的需要选择10s、30s或60s。

图 4-87 自动刷新



kill 会话

您可以单击“kill选中会话”、“kill当前节点全部会话”或者“kill实例全部会话”，结束选中的会话或者结束节点、实例全部的会话。

 **注意**

kill会话操作会导致业务断链，需要业务重连机制，请谨慎操作。



图 4-88 kill 会话



## 导出会话

您可以单击“导出节点全部会话”或者“导出实例全部会话”导出会话。

图 4-89 导出会话



## 4.5.2 GeminiDB Redis 命令重命名

### 操作场景

GeminiDB Redis通用型支持命令重命名，为了避免误操作导致数据丢失，实例重启，性能抖动等问题，您可以对高危命令进行重命名。

### 使用须知

- 部分低版本的实例需要参考[升级内核小版本](#)升级内核小版本以支持命令重命名功能。
- 实例状态为“正常”时，支持重命名高危命令。
- Redis Cluster集群版实例不支持命令重命名。
- Cluster集群版实例暂不支持该功能。

### 操作步骤

- 步骤1** 登录[云数据库GeminiDB控制台](#)。
- 步骤2** 在“实例管理”页面，选择指定的实例，单击“更多 > 命令重命名”，进入“命令重命名”页面。

图 4-90 命令重命名



- 步骤3** 您可以对命令进行重命名，并根据需要禁用相应的命令。参数请查看[表4-40](#)。

表 4-40 参数规则

| 参数       | 描述                                                                                                                            |
|----------|-------------------------------------------------------------------------------------------------------------------------------|
| flushall | 清除所有缓存区。                                                                                                                      |
| flushdb  | 清空当前数据库中的所有 key。                                                                                                              |
| hgetall  | 返回哈希表中所有的字段和值。                                                                                                                |
| hkeys    | 返回哈希表中所有的 key。                                                                                                                |
| hvals    | 返回哈希表中所有的值。                                                                                                                   |
| keys     | 查找所有符合给定模式 pattern 的 key 。                                                                                                    |
| smembers | 返回集合中的所有的成员。不存在的集合 key 被视为空集合。                                                                                                |
| 修改后的命令   | 当前生效的命令名称。命令名称在0位到30位之间，当长度为0时表示命令禁用，不区分大小写字母，可包含数字、字母和下划线的组合，不能包含其他特殊字符。修改成功后的命令不允许重复。                                       |
| 禁用       | 禁用默认关闭，您可以根据需要，单击  禁用命令。禁用的命令，修改后的命令会显示为空。 |

**步骤4** 修改完成后，单击“确定”，保存修改后的命令。

图 4-91 命令重命名

命令重命名

| 原始命令     | 修改后的命令                                | 禁用                       |
|----------|---------------------------------------|--------------------------|
| flushall | <input type="text" value="flushall"/> | <input type="checkbox"/> |
| flushdb  | <input type="text" value="flushdb"/>  | <input type="checkbox"/> |
| hgetall  | <input type="text" value="hgetall"/>  | <input type="checkbox"/> |
| hkeys    | <input type="text" value="hkeys"/>    | <input type="checkbox"/> |
| hvals    | <input type="text" value="hvals"/>    | <input type="checkbox"/> |
| keys     | <input type="text" value="keys"/>     | <input type="checkbox"/> |
| smembers | <input type="text" value="smembers"/> | <input type="checkbox"/> |

确定

取消

- 步骤5 检查命令重命名结果。
- 您可以在“命令重命名”页面查看修改后的命令。
  - 重命名成功后，原始命令失效，需要使用修改后的命令做操作。
- 结束

4.5.3 清除 GeminiDB Redis 数据

操作场景

GeminiDB Redis支持清除数据来释放实例容量空间，您可以根据需要选择清除实例中所有的数据，还可以选择清除指定数据库中的数据。

使用须知

- 清除数据不可恢复，请谨慎操作。建议您在清除数据前先创建备份，具体操作请参见[创建手动备份](#)。
- 清除指定DB中的数据，只会清除选中数据库的数据，其他数据库中的数据不会受到影响。
- Cluster集群版实例暂不支持该功能。

操作步骤

- 步骤1 登录[云数据库GeminiDB控制台](#)。
- 步骤2 在“实例管理”页面，选择目标实例，单击实例名称。
- 步骤3 在“基本信息”页面上方单击“清除数据”。

图 4-92 清除数据



- 如果您需要清除实例中的所有数据，清除方式选择“所有数据”，单击“确定”。

图 4-93 清除所有数据



- 如果您只需要清除一个特定的数据库数据，那么清除方式选择“指定DB”，输入需要清除数据的DB号，单击“确定”。

说明

DB号支持0~65535之间的整数。

图 4-94 清除指定 DB 数据



----结束

## 4.5.4 实例生命周期管理

### 4.5.4.1 重启 GeminiDB Redis 实例

#### 操作场景

出于维护目的，您可能需要重启数据库实例。

#### 使用须知

- 实例状态为“正常”、“异常”、“恢复检查中”，支持重启实例。
- 重启实例后，该实例下所有节点将会被重启。
- 重启实例会导致服务中断，请谨慎操作。请在业务低峰期重启，重启前请做好业务安排并确保应用有重连机制。
- 开启操作保护的用户，在进行敏感操作时，通过进行二次认证再次确认您的身份，进一步提高账号安全性，有效保护您安全使用云产品。关于如何开启操作保护，具体请参考《[统一身份认证服务用户指南](#)》的内容。

#### 操作步骤

**步骤1** 登录[云数据库GeminiDB控制台](#)。

**步骤2** 在“实例管理”页面，选择指定的实例，选择操作列“重启实例”或“更多 > 重启实例”。

您也可以在“实例管理”页面，单击指定实例的名称，在“基本信息”页面右上角，单击“重启实例”。

**步骤3** 若您已开启操作保护，在“重启实例”弹出框，单击“去验证”，跳转至验证页面，单击“免费获取验证码”，正确输入验证码并单击“认证”，页面自动关闭。

**步骤4** 在弹出框中，单击“是”或者“立即重启”重启实例。

- 经典部署模式实例

对于GeminiDB Redis经典部署模式实例实例，您可以根据业务需求，选择“节点同时重启”或者“节点逐个重启”。

 说明

“节点同时重启”会导致服务中断约10~20分钟，适用于短暂停服的维护操作；“节点逐个重启”每次闪断3~5秒，业务影响更小。

图 4-95 重启实例

重启实例

确定重启以下数据库实例吗？

☒ 节点同时重启

☐ 节点逐个重启

| 实例名称 | 运行状态            |
|------|-----------------|
|      | <div>→ 正常</div> |

重启时间

立即重启

可维护时间段内重启

?

⚠ 重启过程中，实例将不可用。

是

否

- 云原生部署模式实例  
对于GeminiDB Redis云原生部署模式实例，单击“是”或“立即重启”即可。

图 4-96 重启实例

重启实例

确定重启以下数据库实例吗？

☒ 节点同时重启

☐ 节点逐个重启

| 实例名称 | 运行状态            |
|------|-----------------|
|      | <div>→ 正常</div> |

重启时间

立即重启

可维护时间段内重启

?

⚠ 重启过程中，实例将不可用。

是

否

----结束

#### 4.5.4.2 导出实例

##### 操作场景

您可以导出所有实例，或根据一定条件筛选出来的目标实例，查看并分析实例信息。

##### 导出所有实例

**步骤1** 登录[云数据库GeminiDB控制台](#)。

**步骤2** 在“实例管理”页面，单击实例列表右上角，默认导出所有的数据库实例，在导出弹框勾选所需导出信息，单击“确定”。

**步骤3** 导出任务执行完成后，您可在本地查看到一个“.xls”文件。

----结束

##### 导出筛选的目标实例

**步骤1** 在“实例管理”页面，根据项目、兼容接口、实例名称、实例ID、标签搜索条件筛选实例，或勾选需要导出的实例，单击实例列表右上角，在导出弹框勾选所需导出信息，单击“确定”。

**步骤2** 导出任务执行完成后，您可在本地查看到一个“.xls”文件。

----结束

#### 4.5.4.3 删除按需实例

##### 操作场景

对于“按需计费”模式的实例，您可根据业务需要，在“实例管理”页面手动删除实例来释放资源。（对于包年/包月的实例，您需要进行订单退订后才可删除实例，详细操作请参见[如何退订包年/包月实例](#)）。

##### 使用须知

- 正在执行操作的实例不能手动删除，只有在实例操作完成后，才可删除实例。
- “按需计费”类型的实例删除后将不再产生费用，实例生成的自动备份会被同步删除，保留的手动备份会继续收取费用。
- 实例删除后，该实例上的数据以及相关的自动备份将全部被清除，且不可恢复，请谨慎操作。建议您在删除实例前先创建备份，具体操作请参见[创建手动备份](#)。
- 实例删除后，实例下所有节点将同步被删除。
- 删除的实例在释放后会在回收站保留一段时间，您可在回收站中对实例进行重建操作。

##### 操作步骤

**步骤1** 登录[云数据库GeminiDB控制台](#)。

**步骤2** 在“实例管理”页面，选择指定的实例，选择“删除”或“更多 > 删除实例”。

**步骤3** 若您已开启操作保护，在“删除实例”弹出框，单击“去验证”，跳转至验证页面，单击“免费获取验证码”，正确输入验证码并单击“认证”，页面自动关闭。

 **说明**

开启操作保护的用户，在进行敏感操作时，通过进行二次认证再次确认您的身份，进一步提高账号安全性，有效保护您安全使用云产品。关于如何开启操作保护，具体请参考[《统一身份认证服务用户指南》](#)的内容。

**步骤4** 在弹出框中，单击“是”。

实例被删除后，将不再显示在实例列表中。

----结束

### 4.5.4.4 GeminiDB Redis 实例回收站

GeminiDB Redis支持将退订后的包年包月实例和删除的按需实例，加入回收站管理。您可以在回收站中重建实例恢复数据。

### 使用须知

- 回收站策略机制默认开启，且不可关闭，默认保留天数为7天，该功能免费。
- 目前回收站允许加入100个实例，超过该配额的实例将无法添加至回收站中。
- 当实例存储空间满时，删除之后的实例不会放入回收站。
- 实例下发删除操作后，会保留一天前的最近一次自动全量备份（如果不存在一天前自动全量备份，则保留最新一次自动全量备份），并执行一次全量备份，可选择其中任一备份文件重建实例恢复数据。

### 设置回收站策略

 **说明**

修改回收站保留天数，仅对修改后新进入回收站的实例生效，对于修改前已经存在的实例，仍保持原来的回收策略，请您谨慎操作。

**步骤1** 登录[云数据库GeminiDB控制台](#)。

**步骤2** 在“实例回收站”页面，单击“回收站策略”，设置已删除实例保留天数，可设置范围为1~7天。单击“确定”，完成设置。

图 4-97 设置回收站策略

#### 回收站策略

保留天数

设置已删除实例保留天数，可设置范围1~7天。修改保留天数后删除的实例按照新的天数来保留，修改之前已在回收站的实例保留天数不变。

目前回收站允许加入100个实例，超过该配额的实例将无法添加至回收站中。

确定

取消

----结束



重建实例

在回收站保留期限内的实例可以通过重建实例恢复数据。

- 步骤1 登录[云数据库GeminiDB控制台](#)。
- 步骤2 在“实例回收站”页面，在实例列表中找到需要恢复的目标实例，单击操作列的“重建”。

图 4-98 重建实例

| 实例名称ID | 实例类型      | 兼容接口 | 计费模式 | 实例规格/配置信息                                                 | 实例内网IP地址段                                                 | 数据大小    | 企业项目    | 操作                 |
|--------|-----------|------|------|-----------------------------------------------------------|-----------------------------------------------------------|---------|---------|--------------------|
| 实例名称ID | Redis 6.2 | 兼容接口 | 按量计费 | 2025/06/05 15:49:08 GMT+08:00 ... 2025/06/12 16:12:00 ... | 2025/06/12 16:12:00 GMT+08:00 ... 2025/06/16 16:12:00 ... | 1.37 MB | default | <a href="#">重建</a> |

- 步骤3 在“重建新实例”页面，选填配置（建议与原实例规格保持一致）后，提交重建任务。
- 结束

4.6 变更实例

4.6.1 升级内核小版本

GeminiDB Redis支持补丁升级，补丁升级涉及性能提升、新功能或问题修复等。

当GeminiDB Redis发布新的涉及性能提升、新功能或问题修复等补丁版本时，客户可以根据自身的业务特点，选择合适的时机升级至最新版本。

华为云有新的补丁版本发布时，您可以在“实例管理”页面如[图4-99](#)“兼容接口”列看到补丁升级提示，单击“补丁升级”进行补丁版本升级。

图 4-99 补丁升级

| 实例名称ID                            | 实例类型 | 兼容接口              | 部署... | 运行状态                     | CPU规格/节点数         | 存储 | 负载均衡IP | 企业项目 | 计费模式    | 操作                                                                                          |
|-----------------------------------|------|-------------------|-------|--------------------------|-------------------|----|--------|------|---------|---------------------------------------------------------------------------------------------|
| <div><div></div><div></div></div> | 主备版  | Redis 6.2<br>补丁升级 | 经典    | <div><div></div>正常</div> | 2vCPUs 标准型<br>2节点 | 0% | 0/240B | --   | default | 按量计费<br>2024/06/07 09:00<br><a href="#">连接Redis</a> <a href="#">弹性伸缩</a> <a href="#">更多</a> |

如果当前实例的内核版本存在已知潜在风险、重大缺陷，或者已过期、已下线，系统会通过短信、邮件等渠道进行提前通知，并在可维护时间段内下发升级任务。

使用须知

- 当有对应的补丁更新时（定期同步开源社区问题、漏洞修复），请及时进行升级。
- 当数据库版本为风险版本时，会提醒用户进行数据库补丁升级。
- 补丁升级会采用滚动升级的方式，升级过程中会依次重启每一个节点，重启期间业务会由其他节点接管，每次接管会产生3-5s闪断，请在业务低峰变更，避免实例过载，并建议业务添加自动重连机制，确保重启后连接及时重建。
- 基础组件升级约需15分钟，数据组件升级与节点数量有关，每个节点升级时长约1-2min。
- 系统会自动检测实例的小版本，如果控制台的“补丁升级”按钮不存在，表示小版本已经是最新。

- GeminiDB Redis升级后，Redis兼容版本可能会提升：如5.0版本升级后会显示兼容6.2，高版本会兼容低版本，升级后，业务用法保持不变，还可以支持Redis 6.2相关命令。
- 并行升级过程中服务不可用，建议您在业务低峰期进行并行升级，并行升级时长总体约17-20min左右，与节点数量无关。

### 说明

开源Redis存在多个大版本，已有实例通常无法直接升级至高版本。而GeminiDB Redis仅有一个大版本，支持用户对实例进行小版本升级到最新稳定小版本。实现bug修复与功能扩展，且升级后版本完全兼容历史版本，业务用法无需调整。此外，新版本兼容Redis社区6.0、7.0及更高版本命令，能提供更丰富的功能与更优的运维体验。

## 操作步骤

- 步骤1** 登录[云数据库GeminiDB控制台](#)。
- 步骤2** 在“实例管理”页面，选择指定的实例，单击“兼容接口”列的“补丁升级”。

图 4-100 补丁升级

| 实例名称ID | 实例类型 | 兼容接口                     | 部署... | 运行状态 | CPU规格/节点数          | 存储 | 负载均衡IP | 企业项目    | 计费模式                    | 操作                                                              |
|--------|------|--------------------------|-------|------|--------------------|----|--------|---------|-------------------------|-----------------------------------------------------------------|
|        | 主备版  | Redis 6.2<br><b>补丁升级</b> | 经典    | 正常   | 2 vCPUs 标准型<br>2节点 | 0% | 0/24GB | default | 按量计费<br>2024/06/07 0... | <a href="#">连接Redis</a> <a href="#">释放实例</a> <a href="#">更多</a> |

您也可以单击实例名称，进入基本信息页面，在“规格信息 > 兼容接口”处单击“补丁升级”。

图 4-101 补丁升级

| 规格信息                                                |                               |
|-----------------------------------------------------|-------------------------------|
| 性能规格<br>2 vCPUs 标准型 <a href="#">规格变更</a>            | 节点数量<br>2                     |
| 实例总容量<br><a href="#">扩容/缩容</a> <a href="#">自动扩容</a> | 兼容接口<br>Redis 6.2 <b>补丁升级</b> |
| 使用状况 0.00/24 GB 0%                                  |                               |

- 步骤3** 在弹出框中，确认信息无误后，单击“确定”。
- 升级方式可以选择“节点逐个升级”串行升级或者“节点同时升级”并行升级。
  - 升级时间可以选择“立即升级”或者“可维护时间段内升级”，可维护时间段内预约的定时升级任务将于下一个时间窗执行。

图 4-102 确认信息

补丁升级

1 注意事项:

1、补丁升级会采用滚动升级的方式，升级过程中会依次重启每一个节点，重启期间业务会由其他节点接管，每次接管会产生3-5秒闪断，请在业务低峰变更，避免实例过载，并建议业务添加自动重连机制，确保重启后连接及时重建。  
2、基础组件升级约需15分钟，数据组件升级与节点数量有关，约1-2分钟每节点。

☒ 节点逐个升级

☐ 节点同时升级

| 实例名称 | 运行状态              |
|------|-------------------|
|      | <span>➔</span> 正常 |

升级时间

立即升级

可维护时间段内升级

?

确定

取消

**步骤4** 在“实例管理”页面，查看补丁升级情况。

- 升级过程中，实例运行状态为“补丁升级中”。
- 升级完成后，实例运行状态变为“正常”。

----结束

## 4.6.2 修改 GeminiDB Redis 实例名称

### 操作场景

GeminiDB Redis支持修改数据库实例名称，以方便您区分和识别实例。

### 方法一

**步骤1** 登录[云数据库GeminiDB控制台](#)。

**步骤2** 在“实例管理”页面，单击目标实例名称后的，修改实例名称。

- 单击“确认”，提交修改。
- 单击“取消”，取消修改。

#### 说明

实例名称需要满足如下规则：

- 允许和已有名称重复。
- 实例名称长度在4个到64个字节之间，必须以字母或中文字开头，区分大小写，可以包含字母、数字、中划线、下划线或中文（一个中文字符占用3个字节），不能包含其他特殊字符。

**步骤3** 在“实例管理”页面，查看修改结果。

----结束

## 方式二

**步骤1** 登录[云数据库GeminiDB控制台](#)。

**步骤2** 在“实例管理”页面，选择目标实例，单击实例名称，进入“基本信息”页面。

**步骤3** 在“基本信息 > 实例信息”区域，单击“实例名称”后的，修改实例名称。

- 单击，提交修改。
- 单击，取消修改。

### 说明

实例名称需要满足如下规则：

- 允许和已有名称重复。
- 实例名称长度在4个到64个字节之间，必须以字母或中文字开头，区分大小写，可以包含字母、数字、中划线、下划线或中文（一个中文字符占用3个字节），不能包含其他特殊字符。

**步骤4** 稍后在“基本信息”页面，查看修改结果。

----结束

## 4.6.3 修改 GeminiDB Redis 数据库管理员密码

### 操作场景

GeminiDB Redis支持重置数据库管理员密码，建议您定期修改密码，以提高系统安全性，防止出现密码被破解等安全风险。

### 使用须知

- **实例状态**为“正常”、“备份中”、“存储扩容中”时，支持重置密码。
- 开启操作保护的用户，在进行敏感操作时，通过进行二次认证再次确认您的身份，进一步提高账号安全性，有效保护您安全使用云产品。关于如何开启操作保护，具体请参考《[统一身份认证服务用户指南](#)》的内容。

## 方法一

**步骤1** 登录[云数据库GeminiDB控制台](#)。

**步骤2** 在“实例管理”页面，选择目标实例，单击操作列“更多 > 重置密码”。

**步骤3** 输入新管理员密码及确认密码，单击“确定”。

所设置的密码长度为8~32位，必须是大写字母、小写字母、数字、特殊字符~!@#%^\*-\_+=?&至少两种的组合。

**步骤4** 若您已开启操作保护，在弹出框，单击“去验证”，跳转至验证页面，单击“免费获取验证码”，正确输入验证码并单击“认证”，页面自动关闭。

----结束

方法二

- 步骤1 登录[云数据库GeminiDB控制台](#)。
- 步骤2 在“实例管理”页面，选择目标实例，单击实例名称，进入“基本信息”页面。
- 步骤3 在“数据库信息”区域，单击“管理员账户名”处的“重置密码”。
- 步骤4 输入新管理员密码及确认密码，单击“确定”。  
  
所设置的密码长度为8~32位，必须是大写字母、小写字母、数字、特殊字符~!@#%^\*-\_+=?\$( )&至少两种的组合。
- 步骤5 若您已开启操作保护，在弹出框，单击“去验证”，跳转至验证页面，单击“免费获取验证码”，正确输入验证码并单击“认证”，页面自动关闭。
- 结束

4.6.4 变更实例的 CPU 和内存规格

操作场景

规格变更用于扩大或者降低实例中所有节点的CPU、内存规格。当用户购买的实例的规格无法满足业务需求时，可以在控制台进行规格变更。若遇到实例过载，需要紧急增加计算资源，建议优先添加计算节点，速度更快。

使用须知

- 在线变更时，单个节点变更时会出现一次秒级闪断，因此，整个实例会出现数次秒级闪断，要求客户端有自动重连机制。建议在业务低峰期间变更。
- 正在进行变更的节点，其计算任务由其他节点分担，请在业务低峰变更，避免实例过载。
- 云原生部署模式的标准型实例规格变更后，系统会自动调整存储容量至分片数\*分片规格（GB）。

操作步骤

- 步骤1 登录[云数据库GeminiDB控制台](#)。
- 步骤2 在“实例管理”页面，选择指定的实例，在操作列单击“更多 > 规格变更”。
- 经典部署模式实例

图 4-103 规格变更



- 云原生部署模式实例

图 4-104 规格变更

| 实例名称ID          | 实例类型     | 兼容接口      | 部署模式 | 运行状态 | CPU规格/节点数 (总升数) | 存储            | 负载内网IP | 计费模式                           | 操作                                                           |
|-----------------|----------|-----------|------|------|-----------------|---------------|--------|--------------------------------|--------------------------------------------------------------|
| gemini-9551724e | Proxy集群版 | Redis 6.2 | 云原生  | 正常   | 4 GB<br>256核    | 0.03%<br>99GB | -      | 按量计费<br>20241204 11:15:57 GMT+ | <a href="#">规格变更</a> <a href="#">创建备份</a> <a href="#">更多</a> |

您也可以单击实例名称，在基本信息页面，“数据库信息”区域的“节点规格”处，单击“规格变更”。

- 经典部署模式实例

图 4-105 规格变更

| 规格信息                                                        |                        |           |
|-------------------------------------------------------------|------------------------|-----------|
| 节点规格                                                        | 节点数量                   | 兼容接口      |
| 2 vCPUs 标准型 <a href="#">规格变更</a>                            | 3 <a href="#">添加节点</a> | Redis 6.2 |
| 实例总容量                                                       |                        |           |
| <span>未加密</span> <a href="#">扩容/缩容</a> <a href="#">自动扩容</a> |                        |           |
| 使用状况 0/16 GB                                                | 0.01%                  |           |
| PITR占用存储空间 0 GB                                             |                        |           |

- 云原生容量型部署模式实例

图 4-106 规格变更

| 规格信息                               |      |           |
|------------------------------------|------|-----------|
| 性能规格                               | 节点数量 | 兼容接口      |
| 1 vCPU   4 GB <a href="#">规格变更</a> | 2    | Redis 6.2 |
| 实例总容量                              |      |           |
| <a href="#">磁盘扩容</a>               |      |           |
| 使用状况 0/200 GB                      | 0%   |           |

**步骤3** 在规格变更页面，选择所需规格变更方式和变更后的性能规格，单击“下一步”。

- 在线变更：变更过程中，实例节点依次滚动升级，对业务影响最小，变更时长跟节点数正相关，每个节点约需5~10分钟。若节点数较多，请耐心等待。
- 离线变更：离线变更时，全部节点将并行变更，会导致业务中断约10-20分钟，请谨慎操作！对于您的线上生产业务，请使用在线变更。
- 经典部署模式实例

图 4-107 规格变更

当前配置

实例名称

实例ID

计费模式

当前节点规格

当前节点数量

当前存储量

geminidb.redis.large.4 | 2 vCPUs 标准型

3

16 GB

规格变更方式

在线变更

离线变更

注意

建议在流量低峰期间变更，期间可能出现数次秒级闪断，要求客户端有自动重连机制。变更时长跟节点数正相关，每个节点约需5~10分钟。

节点规格

0.5 vCPUs (建议用于测试用途)

1 vCPU 特等型

2 vCPUs 增强型

4 vCPUs 标准型

4 vCPUs 增强型

8 vCPUs 标准型

8 vCPUs 增强型

16 vCPUs 增强型

16 vCPUs 标准型

32 vCPUs 标准型

32 vCPUs 增强型

规格名称

geminidb.redis.small.4

geminidb.redis.medium.2

geminidb.redis.large.8

geminidb.redis.xlarge.4

geminidb.redis.xlarge.8

geminidb.redis.2xlarge.4

geminidb.redis.2xlarge.8

geminidb.redis.4xlarge.8

geminidb.redis.4xlarge.4

geminidb.redis.8xlarge.4

geminidb.redis.8xlarge.8

变更后规格

geminidb.redis.small.4 | 0.5 vCPUs

如果开启自动扩容，规格变更后自动扩容至规格的最大存储容量。

变更时间

立即变更

可维护时间窗内变更

- 云原生容量型模式实例

图 4-108 规格变更

规格变更

当前配置

实例名称

实例ID

计费模式

当前节点规格

当前节点数量

当前存储

geminidb.redis-geminifs.medium.4 | 1 vCPU | 4 GB

2

200 GB

性能规格

规格名称

geminidb.redis-geminifs.large.4

geminidb.redis-geminifs.xlarge.4

geminidb.redis-geminifs.2xlarge.4

geminidb.redis-geminifs.4xlarge.4

geminidb.redis-geminifs.8xlarge.4

CPU内存

2 vCPUs | 8 GB

4 vCPUs | 16 GB

8 vCPUs | 32 GB

16 vCPUs | 64 GB

32 vCPUs | 128 GB

变更后规格

geminidb.redis-geminifs.large.4 | 2 vCPUs | 8 GB

建议在流量低峰期间变更，期间可能出现数次秒级闪断，要求客户端有自动重连机制。变更时长跟节点数正相关，每个节点约需5~10分钟。

实例规格变更后成功，部分参数会自动更新为新规格的默认值。

变更时间

立即变更

可维护时间窗内变更

- 云原生标准型部署模式实例

图 4-109 规格变更

规格变更

当前配置

实例名称

实例ID

计费模式

当前分片规格

当前分片数量

当前存储

4 GB

2

8 GB

分片规格

8 GB

建议在流量低峰期间变更，期间可能出现数次秒级闪断，要求客户端有自动重连机制。变更时长跟节点数正相关，每个节点约需5~10分钟。

实例规格变更后成功，部分参数会自动更新为新规格的默认值。

变更时间

立即变更

可维护时间窗内变更

步骤4 在规格确认页面，确认节点规格。

- 如需重新选择，单击“上一步”，修改规格。
- 核对无误后，单击“提交”，开始变更规格。

文档版本 01 (2025-11-13)

版权所有 © 华为云计算技术有限公司

209

**步骤5** 查看变更结果。

在实例“基本信息”页面的“数据库信息”区域，可查看变更后的实例规格。

----结束

## 4.6.5 设置可维护时间段

默认可维护时间段为10:00~14:00（GMT+08:00），您可以根据业务需求，设置可维护时间段。建议将可维护时间段设置在业务低峰期，避免业务在维护过程中异常中断。

### 使用须知

- 目前只有实例重启、规格变更和补丁升级支持可维护时间段。
- 已经选择可维护时间段的规格变更和补丁升级，不能再立即执行；实例重启可以立即执行。
- 您可以取消待执行的任务。
- 修改可维护时间段，不影响原有可维护时间段内定时任务的执行时间。
- 可维护时间段不要和备份的时间冲突，否则可能会导致定时任务失败。
- 在可维护时间段内，定时任务10分钟扫描一次，执行任务；临近可维护时间段结束时下发的任务，有可能扫描不到，取消执行。

### 设置可维护时间段

**步骤1** 登录[云数据库GeminiDB控制台](#)。

**步骤2** 在“实例管理”页面，选择指定的实例，单击实例名称，进入“基本信息”页面。

**步骤3** 在“基本信息”区域的“可维护时间段”处，单击“修改”。

图 4-110 修改



**步骤4** 在“修改可维护时间段”页面，选择要修改的可维护时间段，然后单击“确定”。

目前可维护时间段只支持“02:00-06:00”、“06:00-10:00”、“10:00-14:00”、“14:00-18:00”、“18:00-22:00”和“22:00-02:00”。

图 4-111 修改可维护时间段





- 步骤5 查看修改结果。
- 在实例“基本信息”页面的“基本信息”区域，可查看修改后的可维护时间段。
- 结束

查询执行任务

- 步骤1 登录[云数据库GeminiDB控制台](#)。
- 步骤2 在“任务中心”页面，选择“即时任务”或“定时任务”，查看执行任务。

图 4-112 查询任务

任务中心

即时任务

定时任务

任务名称/任务ID

任务状态

实例名称ID

创建时间

结束时间

创建实例

实例中  
已执行 11%

gemini-db-88613a31

2025/02/25 09:56:35 GMT+08:00

2025/02/25 09:56:38 GMT+08:00

修改端口

完成

gemini-db-88613a31

2025/02/25 09:47:52 GMT+08:00

2025/02/25 09:48:59 GMT+08:00

----结束

取消待执行任务

- 步骤1 登录[云数据库GeminiDB控制台](#)。
- 步骤2 在“任务中心”页面，选择“定时任务”，选择指定的任务，单击“取消”。

图 4-113 取消任务

任务中心

即时任务

定时任务

🔍

请输入搜索条件选择任务内容，默认为按实例ID搜索

任务名称/任务ID

任务状态

实例名称/ID

管理接口

创建时间

变更时间(最近10天)

操作

实例规格变更

d98bc05b-1696-4b95-ac39-abe022047dd

🔄 待执行

Redis

2024/06/25 17:56:00 Redis-08-00

2024/06/26 10:00:00 - 2024/06/26 14:00:00

🔗

- 步骤3 查看取消结果。
- 在实例“任务中心”页面，可查看取消的结果，取消成功后，“任务状态”变更为“取消”。

图 4-114 查看取消任务

即时任务

定时任务

请输入任务名称或选择管理方案，默认按照实例ID搜索

| 任务名称/任务ID                                     | 任务状态                                | 实例名称/ID | 部署接口  | 创建时间                          | 变更时间( GMT+08:00)                          | 操作 |
|-----------------------------------------------|-------------------------------------|---------|-------|-------------------------------|-------------------------------------------|----|
| 实例规格变更<br>d98bc05b-1696-4b95-ac39-abe022047dd | <div><div></div><div>取消</div></div> |         | Redis | 2024/06/25 17:56:00 GMT+08:00 | 2024/06/25 10:00:00 - 2024/06/25 14:00:00 | -- |

----结束

4.6.6 扩容和缩容实例

4.6.6.1 扩容和缩容实例概述

用户在购买GeminiDB Redis实例后，随着业务变化，对资源的需求也可能发生变化。面对这种场景，GeminiDB Redis实例提供了扩缩容功能。用户可以根据自己的需求，灵活地调整资源。目前实例的扩容和缩容主要通过以下几种方式来实现：

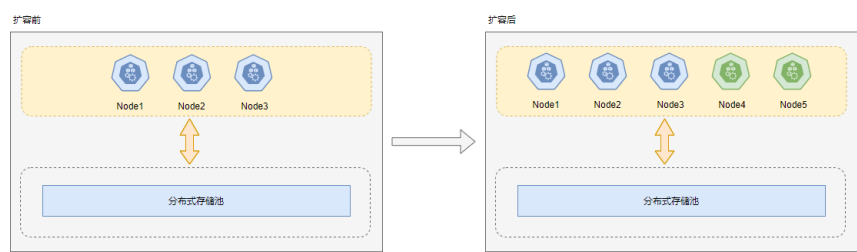
表 4-41 实例扩容和缩容方式

| 方式     | 支持的实例类型                                                                               |
|--------|---------------------------------------------------------------------------------------|
| 扩容实例节点 | <ul style="list-style-type: none"><li>经典部署模式的集群版实例</li><li>云原生部署模式的容量型集群版实例</li></ul> |
| 扩容实例分片 | 云原生部署模式的标准型集群版实例                                                                      |
| 缩容实例节点 | 经典部署模式集群版实例                                                                           |

扩容实例节点

以原实例配置为3节点为例，若新增节点个数为2个，则扩容后实例变更为5节点。具体操作请参见[扩容实例节点](#)。

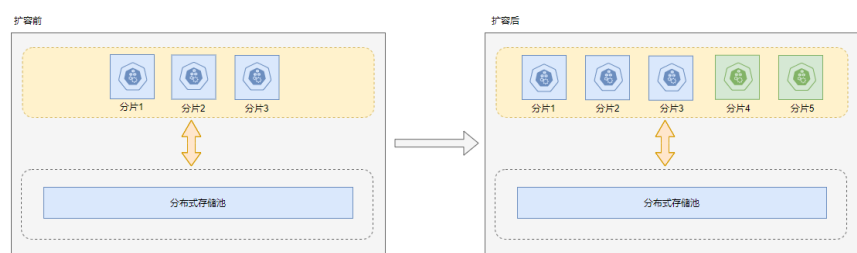
图 4-115 扩容实例节点示意图



扩容实例分片

以原实例配置为3分片为例，若新增分片个数为2个，则扩容后实例变更为5分片。具体操作请参见[扩容实例分片](#)。

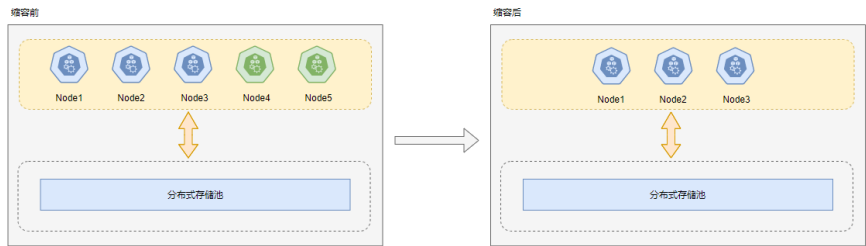
图 4-116 扩容实例分片示意图



缩容实例节点

以原实例配置为5节点为例，若缩容节点个数为2个，则缩容后实例变更为3节点。具体操作请参见[缩容实例节点](#)。

图 4-117 缩容实例节点示意图



### 4.6.6.2 扩容实例节点

#### 操作场景

随着业务数据的增加，原来申请的节点数量不能满足需求，这时，您可以为实例添加节点。添加成功后，您也可以根据需要删除节点，具体请参见[缩容实例节点](#)。

#### 使用须知

- 添加节点会触发快速负载均衡，可能出现请求超时，影响仅为秒级。建议业务侧有自动重试机制。
- 实例状态为正常、恢复检查中时可以添加节点。
- 实例进行添加节点时，该实例不可被删除。
- 添加节点可能导致当前磁盘容量不足以支撑实例运行，需要先扩容磁盘，再添加节点。不同规格实例支持的存储容量，请参考[实例规格说明](#)。
- 当有节点处于关机状态时，不能添加节点。
- 添加节点目前只支持Proxy集群版和Cluster集群版实例。
- 目前节点数上限为36个，如需更多，您可以在管理控制台右上角，选择“[工单 > 新建工单](#)”联系客服进行咨询。

#### 方法一

**步骤1** 登录[云数据库GeminiDB控制台](#)。

**步骤2** 在“实例管理”页面，选择目标实例，单击操作列“更多 > 添加节点”。

图 4-118 添加节点



**步骤3** 在“添加节点”页面，选择新增节点个数，查看实例的存储容量。

- 当存储容量支持添加节点时，单击“下一步”，执行[步骤7](#)。
- 当存储容量不支持添加节点时，单击“下一步”，执行[步骤4](#)。

添加节点 



实例名称 geminidb-redis-lss

实例ID eff016e6a1144ca896f3b022fd199f6bin12

当前节点规格 2 vCPUs 标准型

当前存储容量 16 GB

当前节点数量 2

增加节点数量  您最多可以增加34个节点数，当前还可以创建33个。  
当前扩容需要1个IP，实例所在子网剩余233个IP。

**注意** 添加节点会触发快速负载均衡，可能出现请求超时，影响仅为秒级。建议业务侧有自动重试机制。  
如果开启自动扩容，增加节点数量后自动扩容策略的容量上限将自动变更为实例节点数支持的最大存储容量。

总节点数量 3

新增节点规格默认与实例规格一致，不可修改。

**步骤4** 在容量变更页面，选择需要扩容存储容量大小，单击“下一步”。

图 4-119 容量变更



容量变更 

当前配置

实例名称

实例ID

计费模式 按量计费

当前节点规格 geminidb-redis.large.4 2 vCPUs 标准型

当前节点数量 2

当前存储容量 16 GB

当前存储容量 (GB)  GB

GeminiDB Redis接口支持快速扩容，仅需秒级完成，业务零感知。

**步骤5** 在确认页面，确认存储空间。

- 包年/包月
  - 如需重新选择，单击“上一步”，修改存储空间。
  - 核对无误后，单击“提交订单”，进入付款页面，选择支付方式，完成支付。
- 按需计费
  - 如需重新选择，单击“上一步”，修改存储空间。
  - 核对无误后，单击“提交”，开始扩容存储。

**步骤6** 存储容量扩容完成后，执行[步骤2](#)，重新进行添加节点。

**步骤7** 在确认页面，确认节点配置信息。

- 包年/包月
  - 如需重新选择，单击“上一步”，修改相关配置。
  - 核对无误后，单击“提交订单”，进入付款页面，选择支付方式，完成支付。

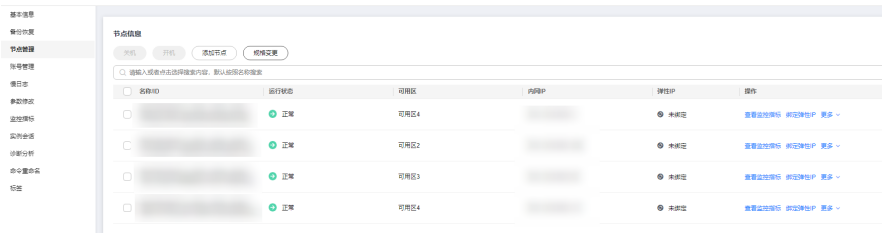
- 按需计费
  - 如需重新选择，单击“上一步”，修改相关配置。
  - 核对无误后，单击“提交”，开始添加节点。

----结束

## 方法二

- 步骤1** 登录[云数据库GeminiDB控制台](#)。
- 步骤2** 在“实例管理”页面，选择指定的实例，单击实例名称。
- 步骤3** 在左侧导航栏，单击“节点管理”。

图 4-120 节点管理



- 步骤4** 单击“添加节点”，在“添加节点”页面，选择新增节点个数，查看实例的存储容量。
- 当存储容量支持添加节点时，单击“下一步”，执行[步骤8](#)。
  - 当存储容量不支持添加节点时，单击“下一步”，执行[步骤5](#)。

### 添加节点

实例名称

geminidb-redis-lss

实例ID

eff016e6a1144ca896f3b022fd199f6bin12

当前节点规格

2 vCPUs 标准型

当前存储容量

16 GB

当前节点数量

2

增加节点数量

-

1

+

您最多可以增加34个节点数，当前还可以创建33个。  
当前扩容需要1个IP，实例所在子网剩余233个IP。

注意

添加节点会触发快速负载均衡，可能出现请求超时，影响仅为秒级，建议业务侧有自动重试机制。  
如果开启自动扩容，增加节点数量后自动扩容策略的容量上限将自动变更为实例节点数支持的最大存储容量。

总节点数量

3

新增节点规格默认与实例规格一致，不可修改。

- 步骤5** 在容量变更页面，选择需要扩容存储容量大小，单击“下一步”。

图 4-121 容量变更



**步骤6** 在确认页面，确认存储空间。

- 包年/包月
  - 如需重新选择，单击“上一步”，修改存储空间。
  - 核对无误后，单击“提交订单”，进入付款页面，选择支付方式，完成支付。
- 按需计费
  - 如需重新选择，单击“上一步”，修改存储空间。
  - 核对无误后，单击“提交”，开始扩容存储。

**步骤7** 存储容量扩容完成后，执行[步骤2](#)，重新进行添加节点。

**步骤8** 在确认页面，确认节点配置信息。

- 包年/包月
  - 如需重新选择，单击“上一步”，修改相关配置。
  - 核对无误后，单击“提交订单”，进入付款页面，选择支付方式，完成支付。
- 按需计费
  - 如需重新选择，单击“上一步”，修改相关配置。
  - 核对无误后，单击“提交”，开始添加节点。

----结束

### 方法三

**步骤1** 登录[云数据库GeminiDB控制台](#)。

**步骤2** 在“实例管理”页面，选择指定的实例，单击实例名称。

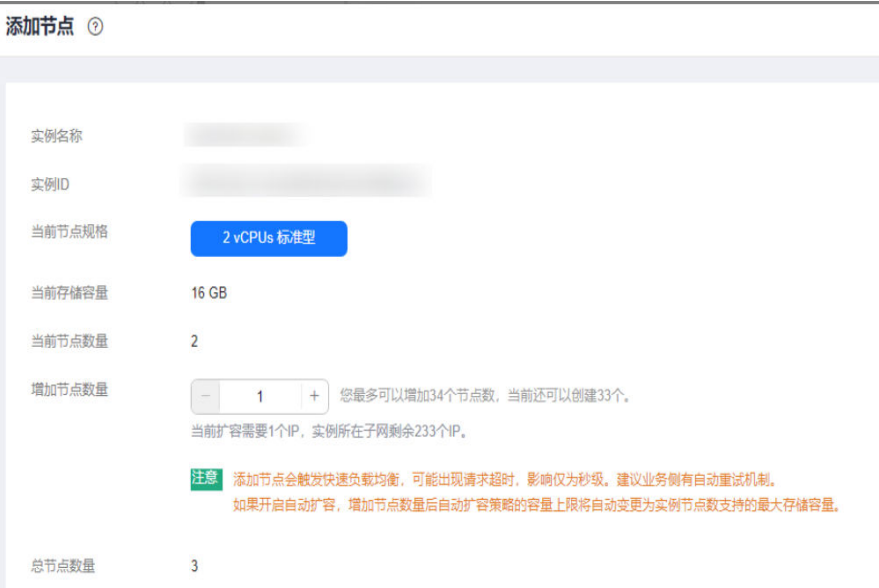
**步骤3** 在基本信息页面的“规格信息”区域，单击“添加节点”。

图 4-122 添加节点



**步骤4** 在“添加节点”页面，选择新增节点个数，查看实例的存储容量。

- 当存储容量支持添加节点时，单击“下一步”，执行**步骤8**。
- 当存储容量不支持添加节点时，单击“下一步”，执行**步骤5**。



新增节点规格默认与实例规格一致，不可修改。

**步骤5** 在容量变更页面，选择需要扩容存储容量大小，单击“下一步”。

图 4-123 容量变更



**步骤6** 在确认页面，确认存储空间。

- 包年/包月
  - 如需重新选择，单击“上一步”，修改存储空间。
  - 核对无误后，单击“提交订单”，进入付款页面，选择支付方式，完成支付。
- 按需计费
  - 如需重新选择，单击“上一步”，修改存储空间。
  - 核对无误后，单击“提交”，开始扩容存储。

**步骤7** 存储容量扩容完成后，执行[步骤2](#)，重新进行添加节点。

**步骤8** 在确认页面，确认节点配置信息。

- 包年/包月
  - 如需重新选择，单击“上一步”，修改相关配置。
  - 核对无误后，单击“提交订单”，进入付款页面，选择支付方式，完成支付。
- 按需计费
  - 如需重新选择，单击“上一步”，修改相关配置。
  - 核对无误后，单击“提交”，开始添加节点。

----结束

### 4.6.6.3 扩容实例分片

#### 操作场景

随着业务数据的增加，原来申请的分片数量不能满足需求，这时，您可以为实例添加分片。

#### 使用须知

- 添加分片目前只支持云原生部署模式的标准型实例。
- 添加分片会触发快速负载均衡，可能出现请求超时，影响仅为秒级。建议业务侧有自动重试机制。
- 实例状态为正常、恢复检查中时可以添加分片。
- 实例进行添加分片时，该实例不可被删除。



- 添加分片成功后，系统会自动扩容新增分片数\*分片规格（GB）的存储容量。
- 添加分片目前只支持Proxy集群版实例。

## 操作步骤

- 步骤1** 登录[云数据库GeminiDB控制台](#)。
- 步骤2** 在“实例管理”页面，选择指定的实例，单击实例名称。
- 步骤3** 在左侧导航栏，单击“分片管理”。

图 4-124 分片管理



- 步骤4** 单击“添加分片”，在“添加分片”页面，选择新增分片个数。



新增分片规格默认与实例规格一致，不可修改。

- 步骤5** 在确认页面，确认分片配置信息。
- 按需计费
    - 如需重新选择，单击“上一步”，修改相关配置。
    - 核对无误后，单击“提交”，开始添加分片。

----结束

### 4.6.6.4 缩容实例节点

## 操作场景

按需和包年/包月实例支持通过删除或缩减节点的方式来释放资源。

## 使用须知

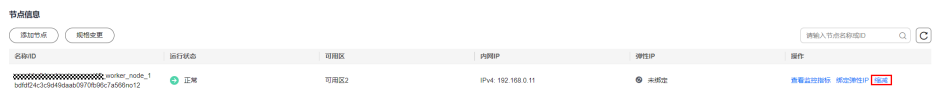
- 删除操作无法恢复，请谨慎操作。
- 开启操作保护的用户，在进行敏感操作时，通过进行二次认证再次确认您的身份，进一步提高账号安全性，有效保护您安全使用云产品。关于如何开启操作保护，具体请参考[《统一身份认证服务用户指南》](#)的内容。

## 包年/包月

- 步骤1** 登录[云数据库GeminiDB控制台](#)。
- 步骤2** 在“实例管理”页面，选择指定的实例，单击实例名称，进入“基本信息”页面。
- 步骤3** 在左侧导航栏单击“节点管理”，进入节点管理页面，确认需要删除的节点。
- 支持的节点删除方式：删除单个节点和批量删除多个节点。

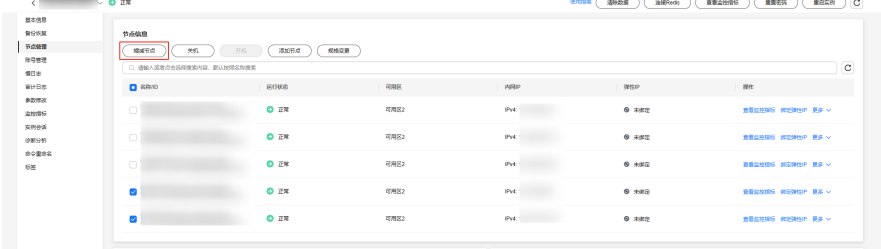
- 删除单个节点  
在“节点管理”页面的“节点信息”区域，在需要缩减的节点上，单击操作列“缩减”。

图 4-125 节点信息



- 批量删除多个节点  
在“节点管理”页面的“节点信息”区域，勾选需要删除的多个节点，单击“缩减节点”。

图 4-126 节点信息



- 步骤4** 若您已开启操作保护，在“缩减节点”或“删除节点”弹出框，单击“去验证”，跳转至验证页面，单击“免费获取验证码”，正确输入验证码并单击“认证”，页面自动关闭。
- 步骤5** 在弹出框中，单击“是”，缩减或删除节点。
- 缩减或删除过程中，实例运行状态显示为“节点缩容中”。
  - 缩减或删除完成后，实例运行状态变为“正常”。

----结束

## 按需计费

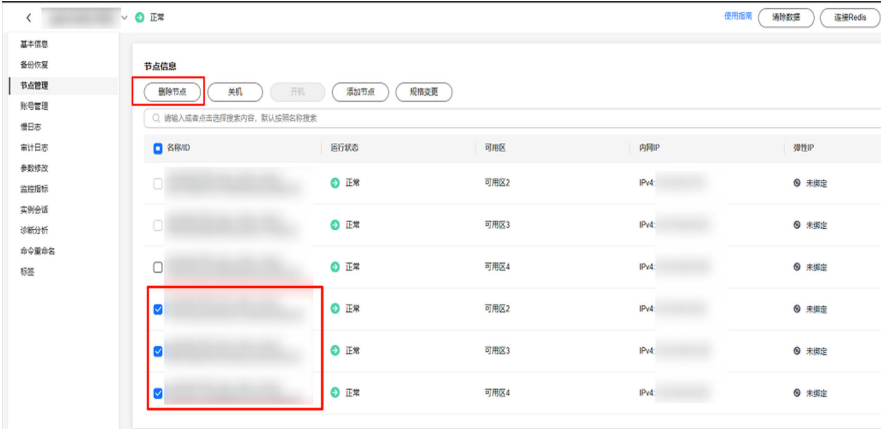
- 步骤1** 登录[云数据库GeminiDB控制台](#)。
- 步骤2** 在“实例管理”页面，选择指定的实例，单击实例名称，进入“基本信息”页面。
- 步骤3** 在左侧导航栏单击“节点管理”，进入节点管理页面，确认需要缩减或删除的节点。
- 支持的节点删除方式：删除单个节点和批量删除多个节点。
- 删除单个节点  
在“节点管理”页面的“节点信息”区域，在需要删除的节点上，单击操作列“更多 > 删除”。

图 4-127 节点信息



- 批量删除多个节点  
在“节点管理”页面的“节点信息”区域，勾选需要删除的多个节点，单击“删除节点”

图 4-128 节点信息



**步骤4** 若您已开启操作保护，在“缩减节点”或“删除节点”弹出框，单击“去验证”，跳转至验证页面，单击“免费获取验证码”，正确输入验证码并单击“认证”，页面自动关闭。

**步骤5** 在弹出框中，单击“是”，缩减或删除节点。

- 缩减或删除过程中，实例运行状态显示为“节点缩容中”。
- 缩减或删除完成后，实例运行状态变为“正常”。

----结束

## 4.6.7 扩容和缩容磁盘

### 4.6.7.1 扩容和缩容磁盘概述

GeminiDB Redis实例使用一段时间后业务攀升，原申请磁盘空间大小不足以支撑储存完整业务量。此时，您可以通过磁盘扩容功能扩容数据库实例的磁盘。随着业务下降，数据库节点利用率低，资源浪费严重，此时，您可以通过磁盘缩容功能减少数据库实例的磁盘。目前GeminiDB Redis支持的磁盘扩缩容方式见表1。

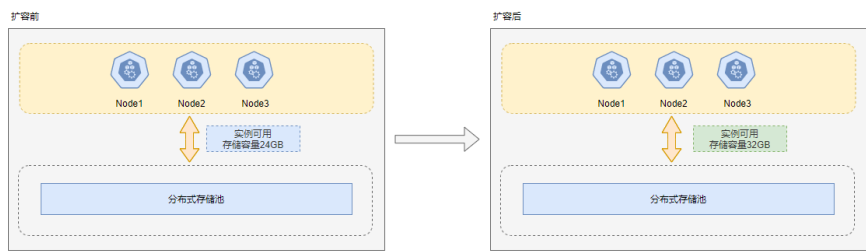
表 4-42 磁盘扩容和缩容方式

| 方式       | 支持的实例类型                                                                       | 说明                                                                                                               |
|----------|-------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------|
| 手动扩容实例磁盘 | <ul style="list-style-type: none"><li>经典部署模式实例</li><li>云原生部署模式容量型实例</li></ul> | 按照用户选择的存储空间大小进行实例级存储扩容。<br>存储空间的增量数值必须为1GB的整数倍，且不大于实例所支持的存储空间上限。                                                 |
| 自动扩容实例磁盘 | 经典部署模式集群版                                                                     | 按照用户设置的扩容步长进行实例级扩容，当实例的存储容量使用率百分比大于用户设置的阈值时，会触发自动扩容，扩容实例的存储容量。目前支持百分比扩容：<br>用户设置的扩容步长为百分比。存储空间的增量数值为实例存储空间容量*步长。 |
| 手动缩容实例磁盘 | 经典部署模式实例                                                                      | 按照用户选择的存储空间大小进行实例级存储缩容。<br>存储空间的缩容数值必须为1GB的整数倍，且不小于当前已使用存储容量的125%，向上取整。                                          |

手动扩容实例磁盘

以经典部署模式实例为例，假设扩容前实例存储为24GB，用户扩容8GB，则扩容后，实例存储空间为32GB。

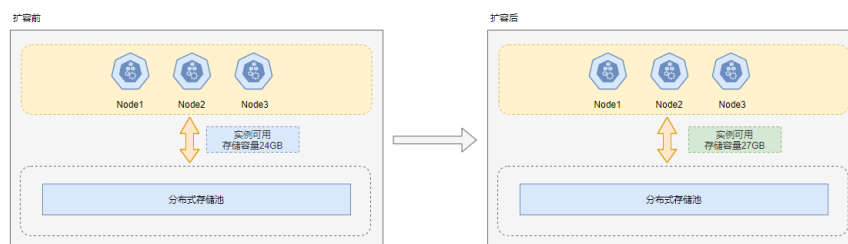
图 4-129 手动扩容实例磁盘示意图



自动扩容实例磁盘

以经典部署模式实例为例，假设扩容前实例存储容量为24GB，扩容策略配置存储容量使用率 $\geq 80\%$ ，扩容步长为10%。若实例的存储容量使用率 $\geq 80\%$ ，则会自动触发扩容，实例扩容的增量值为 $24 \times 10\% = 2.4\text{GB}$ ，向上取整为3GB，则扩容后的实例存储容量为 $24 + 3 = 27\text{GB}$ 。

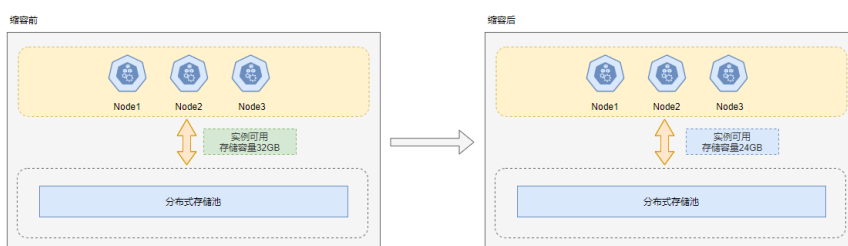
图 4-130 自动扩容实例存储示意图



## 手动缩容实例磁盘

以经典部署模式实例为例，假设缩容前实例存储为32GB，用户缩容8GB，则缩容后，实例存储空间为24GB。

图 4-131 手动缩容实例磁盘示意图



### 4.6.7.2 手动扩容实例磁盘

#### 操作场景

随着业务数据的增加，原来申请的数据库存储容量不能满足需求，这时，您可以扩容实例的磁盘容量。

#### 使用须知

- 为了避免影响业务访问，建议当存储容量使用率超过80%时，及时进行存储扩容。
- 容量变更无需重启实例，在此期间，不会对已有数据造成任何影响，服务不中断，不影响您正常使用数据库。
- 云原生部署模式的标准型实例不支持容量变更。如需扩容存储容量，可通过[添加分片](#)或[升级实例的规格](#)。

#### 实例设置只读状态说明

为保护GeminiDB Redis实例的正常运行，在存储空间即将被写满的时候，数据库会被设置成只读状态，该状态下只支持数据的读取，不支持写入和更新，可以通过扩容存储容量来解除该状态，重新获得写入和更新的能力。

表 4-43 实例设置只读状态说明

| 存储容量       | 说明                                                                                                               |
|------------|------------------------------------------------------------------------------------------------------------------|
| 小于600GB    | <ul style="list-style-type: none"><li>存储容量使用率达到97%时，实例状态被设置为只读。</li><li>存储容量使用率下降到85%，实例自动解除只读状态。</li></ul>      |
| 大于或等于600GB | <ul style="list-style-type: none"><li>存储容量剩余空间小于18GB，实例状态被设置为只读。</li><li>存储容量剩余空间大于或等于90GB，实例自动解除只读状态。</li></ul> |

方法一

- 步骤1 登录[云数据库GeminiDB控制台](#)。
- 步骤2 在“实例管理”页面，选择指定的实例，单击实例名称。
- 步骤3 在基本信息页面的“规格信息”区域，经典部署模式实例单击“扩容/缩容”，进入“容量变更”页面，云原生部署模式单击“磁盘扩容”，进入“磁盘扩容”页面。

图 4-132 容量变更（经典）



图 4-133 磁盘扩容（云原生）



- 步骤4 选择所需扩容磁盘空间大小，单击“下一步”。

图 4-134 容量扩容（经典）



图 4-135 磁盘扩容（云原生）



- 经典部署模式实例，用户每次至少选择1GB扩容量，且必须为整数。
- 云原生部署模式实例，用户每次至少选择10GB扩容量，且必须为10的整数倍。

步骤5 在确认页面，确认存储空间。

- 包年/包月
  - 如需重新选择，单击“上一步”，修改存储容量。
  - 核对无误后，单击“提交订单”，进入付款页面，选择支付方式，完成支付。
- 按需计费
  - 如需重新选择，单击“上一步”，修改存储容量。
  - 核对无误后，单击“提交”，开始变更存储容量。

步骤6 检查存储扩容结果。

- 扩容过程中，实例运行状态为“磁盘变更中”。
- 扩容完成后，实例运行状态变为“正常”。
- 单击实例名称，在实例“基本信息”页面的“规格信息”区域，可查看扩容后的实例存储总容量。

----结束

方法二

步骤1 登录[云数据库GeminiDB控制台](#)。

步骤2 在“实例管理”页面，选择指定的实例，单击“更多 > 容量变更”，进入“容量变更”页面。

图 4-136 容量变更



步骤3 选择所需扩容磁盘空间大小，单击“下一步”。

图 4-137 容量扩容（经典）



图 4-138 磁盘扩容（云原生）



- 经典部署模式实例，用户每次至少选择1GB扩容量，且必须为整数。
- 云原生部署模式实例，用户每次至少选择10GB扩容量，且必须为10的整数倍。

**步骤4** 在确认页面，确认存储空间。

- 包年/包月
  - 如需重新选择，单击“上一步”，修改存储容量。
  - 核对无误后，单击“提交订单”，进入付款页面，选择支付方式，完成支付。
- 按需计费
  - 如需重新选择，单击“上一步”，修改存储容量。
  - 核对无误后，单击“提交”，开始变更存储容量。

**步骤5** 检查存储扩容结果。

- 扩容过程中，实例运行状态为“磁盘变更中”。
- 扩容完成后，实例运行状态变为“正常”。
- 单击实例名称，在实例“基本信息”页面的“规格信息”区域，可查看扩容后的实例存储总容量。

----结束

4.6.7.3 自动扩容实例磁盘

GeminiDB Redis实例支持存储空间自动扩容，在实例存储空间达到阈值时，会触发自动扩容。

可以在创建实例时（详情请参见[购买实例](#)）和创建实例后设置存储空间自动扩容。

本章节介绍创建实例后如何设置存储空间自动扩容。



## 权限配置

如果您使用的是IAM用户，在使用存储空间自动扩容功能前需要配置GeminiDB服务和IAM服务相应的权限。具体操作如下：

1. 配置IAM服务的细粒度权限和GeminiDB服务的最小权限。

在IAM控制台配置如下IAM权限策略，具体操作请参见[创建自定义策略](#)。

```
{
 "Version": "1.1",
 "Statement": [
 {
 "Effect": "Allow",
 "Action": [
 "iam:permissions:listRolesForAgencyOnProject",
 "iam:permissions:grantRoleToGroupOnProject",
 "iam:agencies:createAgency",
 "iam:agencies:listAgencies",
 "iam:roles:listRoles",
 "iam:roles:createRole"
]
 }
]
}
```

2. [创建用户组并授权](#)

您可以在IAM控制台创建用户组，并授予该用户组1中创建的自定义权限和Security Administrator系统角色。

3. [将用户加入用户组](#)

以主账号或者具有IAM权限的IAM用户将需要设置存储空间自动扩容的实例所属的IAM用户，加入2中创建的用户组，此时该IAM用户享有该用户组配置的权限。详情请参考[在用户组中添加或删除IAM用户](#)。

## 使用须知

- 账户需要有足够的余额，才可进行自动扩容。
- 实例状态正常。
- 开启自动扩容，将会创建委托，且会自动扣费。
- 目前只支持GeminiDB Redis通用型实例。
- 当存储容量满(存储容量使用率 > 98%)时。
  - 存储总容量<600G，下一次自动扩容到当前存储容量使用率(已使用/总量)<85%。示例：目前存储总容量500GB，已使用495GB，则 495/下一次扩容后的总量<85%。
  - 存储总容量≥600G，下一次自动扩容到当前总容量+90G以上。示例：目前存储总容量700GB，则下一次扩容后的总量大于700+90=790GB。
- 变更实例规格或节点数量时，会影响实例存储容量上限。
- 容量变更无需重启实例，在此期间，不会对已有数据造成任何影响，服务不中断，不影响您正常使用数据库。

## 单个实例自动扩容

**步骤1** 登录[云数据库GeminiDB控制台](#)。

**步骤2** 在“实例管理”页面，选择目标实例，单击实例名称，进入“基本信息”页面。

步骤3 在“存储空间”区域，单击“自动扩容”。

图 4-139 自动扩容



步骤4 开启自动扩容，并设置自动扩容的触发条件。

图 4-140 设置自动扩容参数



表 4-44 参数说明

| 参数名称    | 描述                                                                  |
|---------|---------------------------------------------------------------------|
| 自动扩容    | 存储空间自动扩容开关。                                                         |
| 存储容量使用率 | 当存储容量使用率百分比大于等于该阈值时，会触发自动扩容。                                        |
| 扩容步长    | 自动扩容当前存储空间的百分比，目前支持选择10、15和20，您可以根据需要选择合适的扩容步长，非10倍数向上取整。每次至少扩容1GB。 |

| 参数名称     | 描述                                                                                                                                   |
|----------|--------------------------------------------------------------------------------------------------------------------------------------|
| 实例存储空间上限 | 自动扩容上限，单位：GB。 <ul style="list-style-type: none"><li>实例存储空间上限&gt;=当前存储容量+1GB。</li><li>最大实例存储空间上限不能超过实例<b>当前规格支持的最大存储容量</b>。</li></ul> |

步骤5 单击“确定”。

----结束

批量自动扩容

- 步骤1 登录[云数据库GeminiDB控制台](#)。
- 步骤2 在实例列表页面，选择目标实例，单击“自动扩容”。

图 4-141 自动扩容



步骤3 开启批量自动扩容，并设置自动扩容的触发条件。

图 4-142 设置自动扩容参数



表 4-45 参数说明

| 参数名称     | 描述                                                                  |
|----------|---------------------------------------------------------------------|
| 自动扩容     | 存储空间自动扩容开关。                                                         |
| 存储容量使用率  | 当存储容量使用率百分比大于等于该阈值时，会触发自动扩容。                                        |
| 扩容步长     | 自动扩容当前存储空间的百分比，目前支持选择10、15和20，您可以根据需要选择合适的扩容步长，非10倍数向上取整。每次至少扩容1GB。 |
| 实例存储空间上限 | 批量自动扩容时，不支持自定义存储自动扩容上限，默认扩容至所选实例对应的最大存储空间。                          |

步骤4 单击“确定”。

----结束

4.6.7.4 手动缩容实例磁盘

操作场景

随着业务下降，数据库节点利用率低，资源浪费严重，此时，您可以通过磁盘缩容功能减少数据库实例的磁盘。

使用须知

- 缩容场景下，待变更到的磁盘容量必须大于已用量的125%，向上取整。
- 容量变更无需重启实例，在此期间，不会对已有数据造成任何影响，服务不中断，不影响您正常使用数据库。
- 云原生部署模式的标准型实例不支持缩容实例磁盘。

实例设置只读状态说明

为保护GeminiDB Redis实例的正常运行，在存储空间即将被写满的时候，数据库会被设置成只读状态，该状态下只支持数据的读取，不支持写入和更新，可以通过扩容存储容量来解除该状态，重新获得写入和更新的能力。

表 4-46 实例设置只读状态说明

| 存储容量       | 说明                                                                                                                   |
|------------|----------------------------------------------------------------------------------------------------------------------|
| 小于600GB    | <ul style="list-style-type: none"><li>• 存储容量使用率达到97%时，实例状态被设置为只读。</li><li>• 存储容量使用率下降到85%，实例自动解除只读状态。</li></ul>      |
| 大于或等于600GB | <ul style="list-style-type: none"><li>• 存储容量剩余空间小于18GB，实例状态被设置为只读。</li><li>• 存储容量剩余空间大于或等于90GB，实例自动解除只读状态。</li></ul> |

方法一

- 步骤1 登录[云数据库GeminiDB控制台](#)。
- 步骤2 在“实例管理”页面，选择指定的实例，单击“更多 > 容量变更”，进入“容量变更”页面。

图 4-143 容量变更



- 步骤3 选择所需缩容磁盘空间大小，单击“下一步”。

图 4-144 容量变更



用户每次至少选择1GB缩容量，且必须为整数。

- 步骤4 在确认页面，确认存储空间。
- 包年/包月
    - 如需重新选择，单击“上一步”，修改存储容量。
    - 核对无误后，单击“提交订单”，进入付款页面，选择支付方式，完成支付。
  - 按需计费
    - 如需重新选择，单击“上一步”，修改存储容量。
    - 核对无误后，单击“提交”，开始变更存储容量。
- 步骤5 检查存储缩容结果。
- 缩容过程中，实例运行状态为“磁盘变更中”。
  - 缩容完成后，实例运行状态变为“正常”。
  - 单击实例名称，在实例“基本信息”页面的“规格信息”区域，可查看缩容后的实例存储总容量。

----结束

## 方法二

- 步骤1** 登录[云数据库GeminiDB控制台](#)。
- 步骤2** 在“实例管理”页面，选择指定的实例，单击实例名称。
- 步骤3** 在基本信息页面的“规格信息”区域，单击“扩容/缩容”，进入“容量变更”页面。

图 4-145 容量变更



- 步骤4** 选择所需缩容磁盘空间大小，单击“下一步”。

图 4-146 容量变更



用户每次至少选择1GB缩容量，且必须为整数。

- 步骤5** 在确认页面，确认存储空间。
- 包年/包月
    - 如需重新选择，单击“上一步”，修改存储容量。
    - 核对无误后，单击“提交订单”，进入付款页面，选择支付方式，完成支付。
  - 按需计费
    - 如需重新选择，单击“上一步”，修改存储容量。
    - 核对无误后，单击“提交”，开始变更存储容量。

- 步骤6** 检查存储缩容结果。
- 缩容过程中，实例运行状态为“磁盘变更中”。
  - 缩容完成后，实例运行状态变为“正常”。
  - 单击实例名称，在实例“基本信息”页面的“规格信息”区域，可查看缩容后的实例存储总容量。

----结束

## 4.6.8 GeminiDB Redis 主备切换

### 操作场景

GeminiDB Redis实例提供自动HA机制，通常无需手动执行主备切换。使用该功能可以进行容灾演练，验证HA场景客户端处理能力；也可以根据业务部署情况进行主备切换操作，满足业务就近连接需求。

### 前提条件

该功能目前只支持GeminiDB Redis主备版实例，且实例处于“正常”状态，才能执行此操作。

### 使用须知

- 主备切换后，实例IP地址不变，不需要改变业务连接地址。
- 主备切换时通常会发生小于10秒的连接闪断，会产生慢时延，或者命令执行失败，需要客户端有命令重试或者连接重试机制。为了避免影响业务，建议在业务低峰期操作。

### 操作步骤

**步骤1** 登录[云数据库GeminiDB控制台](#)。

**步骤2** 在“实例管理”页面，选择指定的实例，单击实例名称，进入“基本信息”页面。

**步骤3** 在左侧导航栏单击“节点管理”，在“节点信息”区域，单击“主备切换”，进入“主备切换”页面。

图 4-147 主备切换



**步骤4** 在“主备切换”页面，单击“确定”，完成实例主备切换。

图 4-148 主备切换



----结束

## 4.7 数据备份

### 4.7.1 备份概述

GeminiDB Redis支持数据库实例的备份，以保证数据可靠性。实例删除后，手动备份数据保留。自动备份的数据和实例一起释放，备份的数据不支持下载导出，即不支持备份导出。GeminiDB Redis当前仅支持全量备份。

#### 使用须知

数据备份期间，会消耗少部分CPU资源，备份文件上传到OBS占用计算节点带宽，对业务可能会产生轻微的时延抖动。

#### 备份方案

GeminiDB Redis实例支持自动备份和手动备份两种方案。

- 自动备份  
您可以在管理控制台[设置自动备份策略](#)，系统将会按照自动备份策略中设置的备份时间段和备份周期进行自动备份，并且会按照设置的备份保留天数对备份文件进行存放。  
自动备份的备份文件不支持手动删除，可通过[修改自动备份策略](#)调整备份保留天数，超出备份保留天数的已有备份文件会被自动删除。
- 手动备份  
手动备份是由用户根据自身业务特点随时启动的数据库实例的全量备份，会一直保存，直到用户手动删除。  
建议您定期对数据库进行备份，当数据库故障或数据损坏时，可以通过备份恢复数据库，从而保证数据可靠性。



表 4-47 备份方案

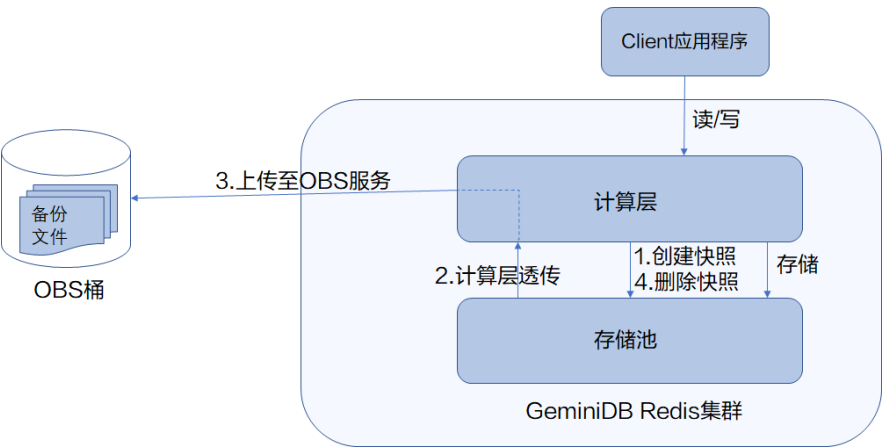
| 备份方案 | 使用场景                                     |
|------|------------------------------------------|
| 自动备份 | 系统按照自动备份策略，对数据库进行自动备份，您可以根据业务需求修改自动备份策略。 |
| 手动备份 | 根据自身业务特点手动启动数据库实例的全量备份。                  |

备份原理

GeminiDB Redis采用存算分离的架构，备份原理图4-149所示。GeminiDB Redis存储池中对数据进行秒级快照，通过计算层透传，以压缩文件的形式存储在OBS存储服务中，不会额外占据实例的存储空间。快照的创建和删除对计算层无影响，上传过程会临时消耗少量计算资源，实例的CPU和内存指标会略微升高，属正常现象。

相比于社区版Redis较慢的备份速度以及产生性能抖动的问题，GeminiDB Redis备份速度更快，对性能几乎不产生影响。

图 4-149 备份原理



备份存储

GeminiDB Redis的备份数据存储至对象存储服务（Object Storage Service，简称OBS），在提高数据容灾能力的同时有效降低磁盘空间占用。

购买实例存储空间后，GeminiDB Redis将同比例赠送备份存储空间，用于存储备份数据。例如，您购买的实例存储空间为100GB时，会得到赠送的100GB备份存储空间。当备份数据没有超出100GB，将免费存储在OBS上；当备份数据超出100GB，超出部分将根据OBS的计费规则收费，详情请参见[OBS计费详情](#)。

4.7.2 管理自动备份

GeminiDB Redis支持创建数据库实例的自动备份，以保证数据可靠性。当数据库或表被恶意或误删除，可依赖实例的备份保障数据安全。

自动备份策略

系统按照自动备份策略，对数据库进行自动备份，备份将以压缩包的形式存储在对象存储服务中，以保证用户数据的机密性和持久性。建议您定期对数据库进行备份，当

数据库故障或数据损坏时，可以通过备份恢复数据库。由于开启备份会损耗数据库读写性能，建议您选择业务低峰时间段启动自动备份。

创建数据库实例时，系统默认开启自动备份策略，默认开启的自动备份策略设置如下：

图 4-150 修改备份策略

**修改备份策略**

自动备份 ☒

保留天数  设置备份保留天数，可设置范围为1~3660天。

时区 GMT+08:00

备份时间段 03:00-04:00

备份周期 ☒ 全选

☒ 周一 ☒ 周二 ☒ 周三 ☒ 周四

☒ 周五 ☒ 周六 ☒ 周日

备份周期至少选择一天。

- **保留天数：**自动备份可保留天数默认为7天。可设置保留天数范围为1~3660天。新的全量备份未超过保留天数前系统会一直保留，直至新的全量备份超过保留天数后才会删除。
  - 增加保留天数，可提升数据可靠性，请根据需要设置。
  - 减少保留天数，会针对已有的备份文件生效，即超出备份保留天数的已有备份文件（包括全量备份和增量备份）会被自动删除，但手动备份不会自动删除，请您谨慎选择。

#### 说明

- 保留天数小于7天，系统每天都会进行自动备份。
- 系统会自动检测已有的自动备份文件，若备份文件超过用户自定义的数据保留天数，则将其删除。
- **备份时间段：**默认为24小时中，间隔一小时的随机的一个时间段，例如04:00~05:00。备份时间段以GMT时区保存。如果碰到夏令时或冬令时切换，备份时间段会因时区变化而改变。

假如保留天数设置为“2”，表示超过两天的全量备份和增量备份会被自动删除。即周一产生的备份会在周三删除，同理，周二产生的备份会在周四删除。

#### 全量备份文件自动删除策略：

已有备份文件超出备份天数后会自动删除，考虑到数据完整性，自动删除时仍然会保留最近的一次超过保留天数的全量备份，保证在保留天数内的数据可正常恢复。

假如备份周期选择“周一”、“周二”，保留天数设置为“2”，备份文件的删除策略如下：

- 本周一产生的全量备份，会在本周四当天自动删除。原因如下：  
本周二的全量备份在本周四当天超过保留天数，按照全量备份文件自动删除策略，会保留最近的一个超过保留天数的全量备份（即本周二的备份会被保留），因此周四当天删除本周一产生的全量备份文件。
- 本周二产生的全量备份，会在下周三当天自动删除。原因如下：  
下周一产生的全量备份在下周三超过保留天数，按照全量备份文件自动删除策略，会保留最近的一个超过保留天数的全量备份（即下周一的备份会被保留），因此下周三当天删除本周二产生的全量备份。
- **备份周期：**默认为全选。
  - 全选：选择一周内的每一天。系统每天都会进行自动备份。
  - 选择周期：选择一周内的一天或几天。系统会在所选时间进行自动备份。

#### 📖 说明

备份周期对应的备份开始时间1小时内，系统会自动触发全量备份。备份所需时间由备份数据量决定，备份数据量越大，备份所需时间越长。

- 实例创建成功后，您可根据业务需要设置自动备份策略。系统将按照您设置的自动备份策略对数据库进行备份。
- 关闭自动备份策略后，自动备份将会立即停止。

## 修改自动备份策略

**步骤1** 登录[云数据库GeminiDB控制台](#)。

**步骤2** 在“实例管理”页面，单击目标实例，进入实例的“基本信息”页面。

**步骤3** 在左侧导航栏中选择“备份恢复”页签，单击“修改备份策略”，设置备份策略。备份策略设置完成后，单击“确定”，保存修改。

设置备份策略的方法可参考[自动备份策略](#)。

图 4-151 修改备份策略

#### 修改备份策略

自动备份 ☒

保留天数

设置备份保留天数，可设置范围为1~3660天。

时区 GMT+08:00

备份时间段 03:00-04:00

备份周期 ☒ 全选

☒ 周一

☒ 周二

☒ 周三

☒ 周四

☒ 周五

☒ 周六

☒ 周日

备份周期至少选择一天。

确定

取消

**步骤4** 备份策略修改成功后，您可在“备份管理”页面或“备份恢复”页签，查看或管理已经生成的备份文件。

----结束

## 关闭自动备份策略

**步骤1** 登录[云数据库GeminiDB控制台](#)。

**步骤2** 在“实例管理”页面，单击目标实例，进入实例的“基本信息”页面。

**步骤3** 在左侧导航栏中选择“备份恢复”页签，单击“修改备份策略”。

**步骤4** 在“修改备份策略”弹出框中单击 然后单击“是”，关闭自动备份策略。

图 4-152 关闭备份策略



**修改备份策略**

自动备份 ☐ 关闭自动备份后，自动备份将会立即停止。

☐ 删除全部自动备份

保留天数  设置备份保留天数，可设置范围为1~3660天。

时区 GMT+08:00

备份时间段 03:00-04:00

备份周期 ☒ 全选

☒ 周一 ☒ 周二 ☒ 周三 ☒ 周四

☒ 周五 ☒ 周六 ☒ 周日

确定 取消

关闭自动备份策略时，您可选择是否同时删除自动备份。

- 勾选，删除当前还在保留天数内的备份文件。备份列表中不会有自动备份，直到您再次开启自动备份策略。
- 不勾选，将保存当前还在保留天数内的备份文件，后期可手动删除，请参见[删除自动备份](#)。

关闭自动备份策略后，自动备份将会立即停止。

----结束

## 删除自动备份

自动备份策略关闭后，支持用户删除已保存的自动备份，从而释放相关存储空间。

自动备份策略开启后，对于过期的自动备份，系统会检测并删除，用户不可删除自动备份。

### ⚠ 注意

用户如需删除已保存的自动备份，需先关闭自动备份策略，具体参见[关闭自动备份策略](#)。

备份删除后，不可恢复，请谨慎操作。

- 方式一
  - a. 登录[云数据库GeminiDB控制台](#)。
  - b. 在“实例管理”页面，单击目标实例，进入实例的“基本信息”页面。
  - c. 在左侧导航栏中选择“备份恢复”页签，单击目标备份对应操作列中的“删除”。
  - d. 在“删除备份”弹出框中，确认目标备份信息，单击“是”。
- 方式二
  - a. 登录[云数据库GeminiDB控制台](#)。
  - b. 在“备份管理”页面，单击目标备份对应操作列中的“删除”。
  - c. 在“删除备份”弹出框中，确认目标备份信息，单击“是”。

## 设置恢复到指定时间点策略

**步骤1** 登录[云数据库GeminiDB控制台](#)。

**步骤2** 在“实例管理”页面，单击目标实例，进入实例的“基本信息”页面。

**步骤3** 在左侧导航栏中选择“备份恢复”页签，单击“设置恢复到指定时间点策略”，设置恢复到指定时间点备份策略。设置完成后，单击“是”，保存修改。

图 4-153 设置恢复到指定时间点策略

设置恢复到指定时间点策略

是否开启 ☒

备份时间间隔

保留天数

确定 取消

- 是否开启：此开关控制开启或关闭备份。
- 备份时间间隔：执行自动备份的时间间隔，单位：分钟，可设置备份时间间隔范围为5~120分钟。

例如：备份时间间隔设为5分钟，首次备份为04:00，则下一次为04:05。

- 保留天数：自动备份可保留的时间，单位：天，可设置保留天数范围为1~7天。新的全量备份未超过保留天数前系统会一直保留，直至新的全量备份超过保留天数后才会删除

----结束

### 4.7.3 管理手动备份

GeminiDB Redis支持对“运行状态”为“正常”的实例创建手动备份，以保证数据可靠性。当数据库或表被恶意或误删除，可依赖实例的备份保障数据安全。手动备份为全量备份。

#### 使用须知

手动备份会收费。

#### 创建手动备份

步骤1 登录[云数据库GeminiDB控制台](#)。

步骤2 创建手动备份。

##### 方式一

在“实例管理”页面，选择目标实例，单击操作列的“更多 > 创建备份”。

图 4-154 创建手动备份

创建备份

实例名称

\* 备份名称 backup-3968

描述

0/256

确定 取消

##### 方式二

1. 在“实例管理”页面，单击目标实例，进入实例的“基本信息”页面。
2. 在左侧导航栏中选择“备份恢复”页签，单击“创建备份”。

图 4-155 创建手动备份

×

创建备份

实例名称

★ 备份名称

backup-3968

?

描述

0/256

?

确定

取消

方式三

在左侧导航树单击“备份管理”，进入“备份管理”页面，单击“创建备份”。

图 4-156 创建手动备份

^

创建备份

★ 兼容接口

Cassandra

MongoDB

InfluxDB

Redis

★ 实例类型

集群版

性能版

主备版

★ 实例名称

?

★ 备份名称

backup-b194

?

描述

0/256

?

确定

取消

**步骤3** 在“创建备份”弹出框中，输入备份名称及描述，单击“确定”。

表 4-48 创建手动备份参数说明

| 参数   | 说明                                                             |
|------|----------------------------------------------------------------|
| 实例名称 | 默认为目标实例名称，不可修改。                                                |
| 备份名称 | 备份名称在4~64位之间，必须以英文字母开头，不区分大小写，可以包含英文字母、数字、中划线或者下划线，不能包含其他特殊字符。 |
| 描述   | 描述不能超过256位，且不能包含回车和 > ! < " & ' =特殊字符。                         |

**步骤4** 创建手动备份任务下发成功后，可查看备份状态。

- 在“备份管理”或“备份恢复”页面，可查看正在创建的手动备份的状态显示为“备份中”。
- 手动备份创建成功的状态显示为“备份完成”。

----结束

删除手动备份

如果不再需要已经生成的手动备份，可在“备份管理”页面或“备份恢复”页签进行删除。

手动备份被删除后，将不再显示在备份列表中。

 **注意**

备份删除后，不可恢复，请谨慎操作。

方式一

- 登录[云数据库GeminiDB控制台](#)。
- 在“实例管理”页面，单击目标实例，进入实例的“基本信息”页签。
- 在左侧导航栏中选择“备份恢复”页签，单击目标备份对应操作列中的“删除”。
- 在删除备份弹出框中，确认目标备份的信息，单击“是”。

方式二

- 登录[云数据库GeminiDB控制台](#)。
- 在“备份管理”页面，单击目标备份对应操作列中的“删除”。
- 在删除备份弹出框中，确认目标备份的信息，单击“是”。

4.8 数据恢复

4.8.1 恢复方案

GeminiDB Redis支持数据恢复，您可以根据业务需要选择合适的恢复方法。



表 4-49 恢复方案

| 恢复方案            | 使用场景                                                                    |
|-----------------|-------------------------------------------------------------------------|
| 重建实例            | 当实例被误删除后，在回收站保留期限内的实例，支持通过重建实例恢复数据。                                     |
| 恢复备份到新实例        | 使用已有的备份文件恢复实例数据到新建实例。                                                   |
| 通过PITR回档恢复到当前实例 | 支持PITR(point-in-time recovery)数据回档功能，客户可将数据库恢复到故障发生前的某一个正常时刻，实现数据的快速恢复。 |

4.8.2 恢复备份到新实例

操作场景

GeminiDB Redis支持使用已有的备份，将备份数据恢复到新实例，您可根据业务需要进行恢复。

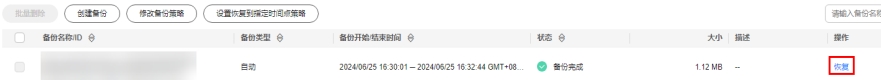
操作步骤

- 步骤1 登录云数据库GeminiDB控制台。
- 步骤2 恢复备份。

方法一

1. 在“实例管理”页面，单击目标实例的名称。
2. 在左侧导航栏中选择“备份恢复”页签，单击目标备份对应操作列中的“恢复”。

图 4-157 恢复



方法二

在“备份管理”页面，单击目标备份对应操作列中的“恢复”。

图 4-158 恢复



- 步骤3 在“恢复实例”弹出框中确认当前实例信息及恢复方式，单击“确定”，跳转到“恢复到新数据库实例”的服务选型页面。

图 4-159 恢复到新实例



- 新实例的接口类型和版本，默认与原实例相同，不可修改。
- 新实例的节点数必须大于等于之前实例的节点数。
- 系统会根据所选择的备份文件大小自动去计算恢复新实例所需的最小存储空间，用户选择容量大小必须为1的整数倍，可根据不同的性能规格选择对应的存储空间。
- 数据库密码需重新设置。
- 其他参数，用户可修改，具体请参见各接口快速入门中购买实例的内容。

#### 步骤4 查看恢复结果。

为用户重新创建一个和该备份数据相同的实例。可看到实例由“创建中”变为“正常”，说明恢复成功。

创建或恢复完成后，系统会自动执行一次全量备份。

恢复成功的新实例是一个独立的实例，与原有实例没有关联。

----结束

### 4.8.3 通过 PITR 回档恢复到当前实例

在实际业务场景中，客户数据库难免会出现数据损毁、数据丢失、数据误删除等故障场景。GeminiDB Redis支持PITR(point-in-time recovery)数据回档功能，客户可将数据库恢复到故障发生前的某一个正常时刻，实现数据的快速恢复。

#### 功能概述

PITR ( Point-in-Time Recovery )，是指数据库的“时间点恢复”功能。它是一种数据库恢复技术，通常用于恢复误删除的数据或者误操作导致损坏的数据，将其恢复到一个指定时间点的数据状态。

以游戏场景为例，在游戏运行期间，有玩家利用游戏漏洞复制装备、货币，使游戏公平性遭到破坏。传统数据库备份频率一般是一天全备一次，备份间隔即一整天，不仅恢复时间长、时间粒度大，甚至无法恢复到想要时间点等。而GeminiDB Redis接口具备的PITR特性能够让游戏数据快速回档，可根据客户自定的备份粒度，最低支持5分钟粒度，自行选择需要恢复的时间点，实现数据的快速恢复。

#### 约束限制

- 此功能目前只支持Gemini Redis集群版实例，不支持灾备/容灾实例。

- 此功能仅支持恢复到当前实例，且恢复期间数据库不可用。
- 此功能目前处于公测中，如需使用，您可以在管理控制台右上角，选择“[工单 > 新建工单](#)”，联系客服进行处理。

## 设置恢复到指定时间点策略

**步骤1** 登录[云数据库GeminiDB控制台](#)。

**步骤2** 在“实例管理”页面，单击目标实例，进入实例的“基本信息”页面。

**步骤3** 在左侧导航栏中选择“备份恢复”页签，单击“数据闪回”，单击“设置数据归档策略”，设置恢复到指定时间点备份策略。设置完成后，单击“是”，保存修改。

图 4-160 设置数据归档策略



- 是否开启：此开关控制开启或关闭备份。
- 备份时间间隔：执行自动备份的时间间隔，单位：分钟，可设置备份时间间隔范围为5~120分钟。  
例如：备份时间间隔设为5分钟，首次备份为04:00，则下一次为04:05。
- 保留天数：自动备份可保留的时间，单位：天，可设置保留天数范围为1~7天。新的备份未超过保留天数前系统会一直保留，直至新的备份超过保留天数后才会删除。

**注意**

如果备份时间间隔设置过短，数据膨胀较快；建议在测试环境实测，以结合业务设置合适的间隔时间。

----结束

## 恢复备份到当前实例

**步骤1** 登录[云数据库GeminiDB控制台](#)。

**步骤2** 在“实例管理”页面，单击目标实例的名称。

**步骤3** 在左侧导航栏中选择“备份恢复”页签，单击“数据闪回”，单击“恢复到指定时间点”。

图 4-161 备份恢复

基本信息

备份恢复

节点管理

账号管理

慢日志



**步骤4** 可以根据需求选择需要恢复到的日期和时间点。

图 4-162 恢复到指定时间点

恢复到指定时间点

恢复日期 2024/05/23

要恢复到的时间点 15:03:53

恢复到 当前实例

确定

取消

**步骤5** 单击“确定”，完成恢复备份到当前实例操作。

----结束

## 4.9 诊断分析

## 4.9.1 大 Key 诊断

### 功能介绍

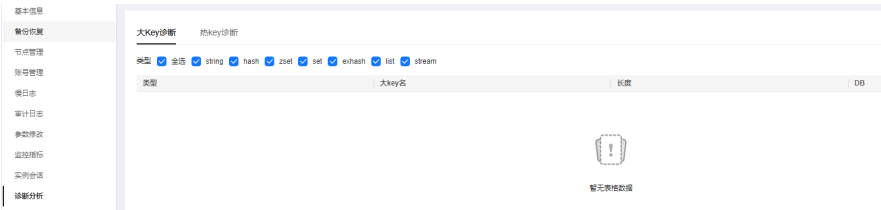
单个Key占用数据量过大时，便被认定为大Key，GeminiDB Redis提供大Key诊断分析功能，可以统计当前实例中的大Key信息。

GeminiDB Redis采用共享存储的设计，大Key存储并不会带来数据倾斜或分片OOM风险。不过大Key访问依然是Redis使用中的常见问题，大key诊断功能直接提取后台分析结果，对业务影响较小。

### 如何查看大 Key 诊断

- 步骤1 登录[云数据库GeminiDB控制台](#)。
- 步骤2 在“实例管理”页面，选择指定的实例，单击实例名称。
- 步骤3 在左侧导航栏，单击“诊断分析”。
- 步骤4 选择“大Key诊断”，勾选对应的类型，查看对应类型的大Key信息。

图 4-163 大 Key 诊断



- 步骤5 您可以查看大Key的信息。参数请查看[表4-50](#)。

表 4-50 大 Key 的参数

| 参数    | 描述                                                                                                                                               |
|-------|--------------------------------------------------------------------------------------------------------------------------------------------------|
| 类型    | 大Key的类型。 <ul style="list-style-type: none"><li>string</li><li>hash</li><li>zset</li><li>set</li><li>exhash</li><li>list</li><li>stream</li></ul> |
| 大Key名 | 大Key的名字。                                                                                                                                         |
| 长度    | Value的长度。                                                                                                                                        |
| DB    | 大Key所在的DB。                                                                                                                                       |

----结束

## 如何设置大 Key 诊断相关参数

string类型key以value大小为判断标准，hash/list/zset/set/stream类型key以元素数量为判断标准。

相关的配置参数有2个：

- bigkeys-string-threshold：value大于该值的string类型key，会被判断为大key，单位为byte。默认为102400（即1MB）。
- bigkeys-composite-threshold：元素数大于该值的hash/list/zset/set类型key，会被判断为大key，单位为个。默认为10240。
- 图 4-164 大 Key 诊断相关参数

| 参数名称                        | 是否必填 | 值          | 允许值                    | 描述                                          |
|-----------------------------|------|------------|------------------------|---------------------------------------------|
| AuthFailLockTime            | 否    | 5          | 0 ~ 10,000             | 认证失败锁定时间，单位为秒（s）。该参数表示可...                  |
| BigkeysQuantityLimitation   | 否    | 100        | 1 ~ 10,000             | string/hash/list/zset/set/stream类型的大key量... |
| CompatibleMode              | 否    | 3          | 0, 1, 2, 3             | 针对BackExchange Redis客户端的适配开关。当Sta...        |
| EnableAclDirect             | 否    | no         | yes, no                | 是否开启ACL直接访问功能。默认为no。                        |
| MaxAuthFailTimes            | 否    | 5          | 0 ~ 10,000             | 认证失败最大尝试次数。当连续失败次数达到该值...                   |
| ProxyTimeout                | 否    | 0          | 0 ~ 100,000            | Proxy端与客户端连接超时时间，单位为秒（s）。当连接...             |
| Bigkeys-composite-threshold | 否    | 1024       | 1 ~ 2,147,483,647      | 元素数大于该值的hash/list/zset/set/stream类型key...   |
| Bigkeys-string-threshold    | 否    | 102400     | 1 ~ 2,147,483,647      | value大于该值的string类型key，会被判断为大key...          |
| databases                   | 否    | 1000       | 1 ~ 1,000              | 允许支持的数据库数量。                                 |
| maxmemory-policy            | 是    | noeviction | noeviction             | 当内存使用量超过配置的最大内存限制时，GaussDB for Red...       |
| notify-keyspace-events      | 否    |            | -                      | 键空间事件通知策略。默认为空，表示该参数不生效...                  |
| slowlog-threshold           | 否    | 300000     | 0, 0.001 ~ 100,000,000 | 控制慢日志记录已执行的耗时大于该阈值的命令。单...                  |

以上两个参数设置不宜过小，以免过多无效结果占据网络带宽，影响业务访问。

大Key诊断相关参数配置方法请参考[修改当前实例的参数](#)。

## 4.9.2 热 key 诊断

当key的访问频次非常高时，该key已被认定为热key，GeminiDB Redis提供热key诊断分析功能。

### 使用须知

- GeminiDB Redis Proxy集群版和Cluster集群版实例，热key诊断最多统计前30个热key；主备版实例，热key诊断最多统计前20个。
- 如果开启审计日志功能，历史的热key诊断会记录在审计日志中。
- 热key诊断QPS值 $\geq 1000$ 时，认定为热key。当QPS值 $> 6000$ 时，不再统计精确的QPS值。

### 操作步骤

步骤1 登录[云数据库GeminiDB控制台](#)。

步骤2 在“实例管理”页面，选择指定的实例，单击实例名称。

- 步骤3** 在左侧导航栏，单击“诊断分析”。
- 步骤4** 选择“热key诊断”，查看实例的热key信息。

图 4-165 热 key 诊断



表 4-51 热 key 诊断参数说明

| 参数名称   | 参数说明                                                            |
|--------|-----------------------------------------------------------------|
| 名称     | 热key的名称。                                                        |
| 类型     | 热key的数据类型，包括string、hash、list、set、sorted set等数据类型。               |
| 最新执行命令 | 热key的最新执行命令。                                                    |
| QPS    | 热key的每秒访问数。<br><b>说明</b><br>QPS统计的峰值是6000，超过6000时，不会再统计精确的频次信息。 |
| DB     | 热key所在的DB。                                                      |

----结束

## 4.10 账号与安全

### 4.10.1 开启免密访问

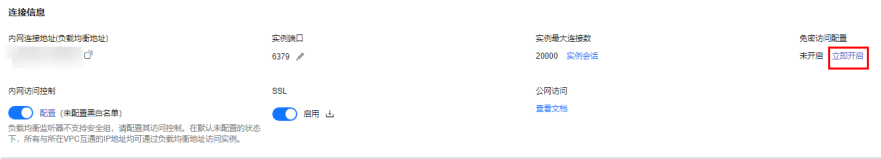
#### 约束限制

- 配置开启免密访问时，最多可开启30个免密网段。
- Cluster集群版实例暂不支持该功能。

#### 操作步骤

- 步骤1** 登录[云数据库GeminiDB控制台](#)。
- 步骤2** 单击实例名称，进入“基本信息”页面。
- 步骤3** 在“连接信息”区域的免密访问配置，单击“立即开启”。

图 4-166 连接信息



步骤4 在“免密访问配置”框，填写免密网段。

图 4-167 免密访问配置



- 单击“+”，可以添加多个免密网段。
- 单击“-”，可以删除免密网段。

步骤5 单击“确定”，完成免密访问配置。

----结束

## 常见问题

对于免密IP，是否还支持密码访问方式？

支持。不论密码是否正确，免密IP都具备访问权限。

## 4.10.2 ACL 账号管理

### 操作场景

GeminiDB Redis接口提供企业级的多租户能力，支持添加只读账号、读写账号，并且可约束每个账号可访问的数据库（DB）范围，避免误操作其他租户数据。本文介绍账号管理使用方法。

### 使用须知

- 每个GeminiDB Redis实例最多能创建200个ACL账号。
- 执行账号变更后，需10s生效。
- 在通过备份恢复到新实例时，原实例的账号信息不会继承。
- 创建账号需要满足表4-52规则。



表 4-52 参数规则

| 参数   | 规则                                                                                                                                                                                                                                         | 示例           |
|------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------|
| 账号名称 | <ul style="list-style-type: none"><li>不能为空。</li><li>长度不超过36个字符。</li><li>以字母开头，只能包含数字、英文字母、下划线、中划线。</li></ul>                                                                                                                               | Organization |
| 权限   | <ul style="list-style-type: none"><li>读写。</li><li>只读。</li></ul>                                                                                                                                                                            | 读写           |
| 数据库  | <ul style="list-style-type: none"><li>授权所有数据库。</li><li>未授权数据库。</li><li>已授权数据库。</li></ul> <p>说明</p> <ul style="list-style-type: none"><li>用户可以根据自己的需要在数据库右边“添加”数据库。</li><li>用户可以根据自己的需要选择需要授权的数据库。</li><li>此处“数据库”对应开源redis的“DB”。</li></ul> | 授权所有数据库      |
| 密码   | <ul style="list-style-type: none"><li>不能为空。</li><li>长度为8~32位。</li><li>密码需包含大写字母、小写字母、数字和特殊字符中的至少两种，支持的特殊字符为~!@#%^*-_ = +? \$ () &amp;。</li></ul>                                                                                           | test123456   |

创建 ACL 账号

- 步骤1 登录[云数据库GeminiDB控制台](#)。
- 步骤2 在“实例管理”页面，选择目标实例，单击实例名称，进入“基本信息”页面。
- 步骤3 在左侧导航树，单击“账号管理”。
- 步骤4 在“账号管理”页面，单击“创建账号”。

图 4-168 创建账号



- 步骤5 在弹出框中，输入账号，并选择权限，给需要的数据库（DB）进行授权，输入密码并确认密码，单击“确定”。

图 4-169 创建账号

创建账号

账号名称

权限

读写

只读

数据库

授权所有数据库

输入其他未授权库名

添加

未授权数据库

0 / 0

暂无数据

已授权数据库

0 / 1

授权所有数据库

密码

\*\*\*\*\*

确认密码

\*\*\*\*\*

确定

取消

步骤6 创建成功后，您可在当前实例的所有账号中，查询并管理自己的账号。

----结束

重置密码

步骤1 登录云数据库GeminiDB控制台。

步骤2 在“实例管理”页面，选择目标实例，单击实例名称，进入“基本信息”页面。

步骤3 在左侧导航树，单击“账号管理”。

步骤4 在“账号管理”页面，选择需要重置密码的账号，单击“重置密码”。

图 4-170 重置密码

| 账号名称            | 账号类型  | 权限 | 已授权数据库 | 操作           |
|-----------------|-------|----|--------|--------------|
| 管理员（只能使用密码即可登录） | 管理类用户 | 读写 | all    | ...          |
|                 | 普通用户  | 读写 | all    | 重置密码 修改权限 删除 |

步骤5 在弹出框中输入新密码，并确认密码，单击“确定”。

文档版本 01 (2025-11-13)

版权所有 © 华为云计算技术有限公司

252

图 4-171 重置密码

### 重置密码

实例名称

账号名称

密码

确认密码

确定

取消

**步骤6** 重置成功后，原先已建立起的长链接需要重新连接。连接命令参考：**auth USER PWD** 或 **auth USER:PWD**。

----结束

修改权限

- 步骤1** 登录[云数据库GeminiDB控制台](#)。

**步骤2** 在“实例管理”页面，选择目标实例，单击实例名称，进入“基本信息”页面。

**步骤3** 在左侧导航树，单击“账号管理”。

**步骤4** 在“账号管理”页面，选择需要修改权限的账号，单击“修改权限”。

图 4-172 修改权限

| 账号名称            | 账号类型  | 权限 | 已授权数据库 | 操作                                                           |
|-----------------|-------|----|--------|--------------------------------------------------------------|
| 管理员（只能使用密码即可登录） | 管理员用户 | 读写 | all    | ...                                                          |
|                 | 普通用户  | 读写 | all    | <a href="#">重置密码</a> <a href="#">修改权限</a> <a href="#">删除</a> |

**步骤5** 在弹出框中选择需要修改的权限和数据库，单击“确定”。

图 4-173 修改权限

修改权限

账号名称

权限

读写

只读

数据库

授权所有数据库

输入其他未授权库名

添加

未授权数据库0 / 0

暂无数据

已授权数据库0 / 1

授权所有数据库

确定

取消

**步骤6** 重置成功后，原先已建立起的长链接需要重新连接。连接命令参考：**auth USER PWD** 或 **auth USER:PWD**。

----结束

删除 ACL 账号

- 步骤1

登录云数据库GeminiDB控制台。
- 步骤2

在“实例管理”页面，选择目标实例，单击实例名称，进入“基本信息”页面。
- 步骤3

在左侧导航树，单击“账号管理”。
- 步骤4

在“账号管理”页面，选择需要删除的账号，单击“删除”，在弹出框中单击“确定”。

图 4-174 删除

| 账号名称            | 账号类型  | 权限 | 已授权数据库 | 操作           |
|-----------------|-------|----|--------|--------------|
| 管理员（只能使用密码即可登录） | 管理员用户 | 读写 | all    | —            |
|                 | 普通用户  | 读写 | all    | 重置密码 修改权限 删除 |

**步骤5** 删除成功后，该账号将不再显示在实例的所有账号中。

----结束

如何使用新账号访问数据库

1. 命令行访问：**auth USER PWD** 或 **auth USER:PWD**

2. 业务程序使用SDK访问：分别将USER、PWD作为账号、密码参数值，或将"USER:PWD"作为密码参数。

使用`auth argc`登录时，`argc`中不能含有冒号，若输入的错误密码含有冒号时，返回值与开源输入`auth argc1 argc2`的返回值相同。

### 4.10.3 开启 ACL 账号自动定向 DB 功能

#### 操作场景

GeminiDB Redis实例支持创建多个ACL账号，实现多业务共用实例，从而帮助DBA（数据库管理员）有效降本。不同ACL账号之间可以配置DB（数据库DB标识，例如DB0，DB1，DB2等）级隔离，防止误操作。通用的账号管理功能仅支持使用`auth <user> <pwd>`或者`auth <user:pwd>`的鉴权方式，对于用户不输入`<user>`或者不方便在代码中修改`<pwd>`为`<user:pwd>`的场景，可以开启账号自动定向DB功能，即通过`auth <pwd>`鉴权来达到多用户DB隔离的目的。

示例：

1. 为业务A分配一组账号/密码(`user1/p1`)，并配置该密码仅可访问“DB 10”。
2. 为业务B分配另一组账号/密码(`user2/p2`)，并配置该密码仅可访问“DB 18”。

业务往往希望减少应用程序改动，例如业务B不想配置客户端的DB参数为18，而是保持默认。这个场景下就可以开启账号自动定向DB功能，实现仅用密码即可自动定向指定DB 18的效果。此时即使业务B误操作`select`到无权限的DB 10，也没有在DB10上操作数据的权限，不用担心业务A的数据受到影响。使用效果如下所示：

```
redis:6379> auth p2
OK
redis:6379> set k v
OK
redis:6379> get k
"v"
redis:6379> select 10
OK
redis:6379[10]> get k
(error) ERR db not allowed for your account
redis:6379[10]> set k v2
(error) ERR db not allowed for your account
```

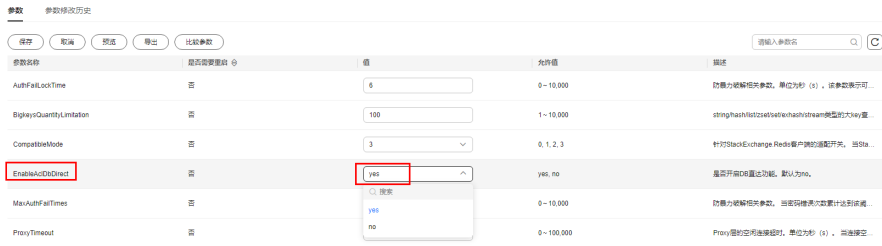
#### 使用须知

- 每个自动定向的ACL账号仅支持设置1个DB，否则鉴权失败。
- 新账号的密码不能与已知密码重复，否则鉴权失败。
- Cluster集群版实例暂不支持该功能。

#### 开启 ACL 账号自动定向 DB 功能

通过设置实例参数“`EnableAclDbDirect`”值为“yes”，开启直达DB功能。设置方法请参考[修改当前实例的参数](#)。

图 4-175 参数



| 参数名称                      | 是否可配置 | 值   | 允许值         | 描述                                            |
|---------------------------|-------|-----|-------------|-----------------------------------------------|
| AuthFailLockTime          | 否     | 5   | 0 ~ 10,000  | 认证失败后锁定的时间，单位为秒（s），该参数表示可...                  |
| BigkeysQuantityLimitation | 否     | 100 | 1 ~ 10,000  | stringhashlistzsetsetsethashstream模型的大key量... |
| CompatibleMode            | 否     | 3   | 0, 1, 2, 3  | 针对StackExchange Redis客户端的兼容开关，当Sta...         |
| EnableAuthDeny            | 否     | yes | yes, no     | 是否开启对暴力破解功能，默认为no。                            |
| MaxAuthFailTimes          | 否     | yes | 0 ~ 10,000  | 认证失败后锁定的次数，当认证失败次数累计达到该值...                   |
| ProxyTimeout              | 否     | no  | 0 ~ 100,000 | Proxy超时时间，单位为秒（s），当连接空...                     |

如何使用新账号访问数据库

- 1. 命令行访问：**auth PWD**
  - 2. 业务程序使用SDK访问：将PWD作为密码参数值。
- 使用**auth argc**登录时，argc中不能含有冒号，若输入的错误密码含有冒号时，返回值与开源输入**auth argc1 argc2**的返回值相同。

4.10.4 防暴力破解

- 防暴力破解安全机制  
GeminiDB Redis默认开启auth认证的防暴力破解功能，当auth认证错误次数累计超过5次，该IP会被后台锁定，不能继续进行auth认证。
- 自动解除机制  
当一个IP被锁定超过5秒后，该IP会被自动解锁，此后可以重新进行认证。
- 人工解锁  
如需人工解锁IP，或关闭实例的防暴力破解功能，您可以在管理控制台右上角，选择“[工单 > 新建工单](#)”联系客服提交工单，进行授权确认。



注意

为进一步提高安全性，可根据实际情况提交工单，由技术人员协助进行认证次数及锁定时间的调节。

为保证安全性，请您充分评估风险，谨慎关闭或调整该安全策略，关闭后由此带来的安全风险及事故，将不计入服务SLA中，由客户自行承担。

4.11 参数管理

4.11.1 修改 GeminiDB Redis 实例参数

为确保GeminiDB Redis发挥出更优性能，用户可根据业务需求对实例参数进行修改调整。

使用须知

- 您可以直接在实例的参数修改页面修改参数。

- 您可以修改用户创建的数据库参数模板中的参数值，但不能更改默认数据库参数模板中的参数值。
- 系统提供的默认参数模板不允许修改，只可单击参数模板名进行查看。
- 当用户参数设置不合理导致数据库无法启动时，可参考默认参数模板重新配置。

 注意

您在修改参数相关参数值时，请谨慎，避免因设置参数值不当造成实例或业务异常。

修改当前实例的参数

- 步骤1 登录[云数据库GeminiDB控制台](#)。
- 步骤2 在左侧导航栏选择“实例管理”，单击指定实例名称，进入基本信息页面。
- 步骤3 单击左侧导航栏中的“参数修改”，在“参数”页签下根据需要修改相关参数值，具体参数详解参见[表4-53](#)。

图 4-176 修改当前实例的参数

| 参数 参数修改历史                   |        |            |                 |                                            |
|-----------------------------|--------|------------|-----------------|--------------------------------------------|
| 保存                          | 取消     | 预览         | 导出              | 忘記参数                                       |
| 请输入参数名                      |        |            |                 |                                            |
| 参数名称                        | 是否需要重启 | 值          | 允许值             | 描述                                         |
| AuthFailLockTime            | 否      | 5          | 0~10,000        | 防暴力破解相关参数。单位为秒（s）。该参数表示可...                |
| BigkeyQuantityLimitation    | 否      | 100        | 1~10,000        | stringhashlistzsetbsethaskstream类型的key量... |
| CompatibMode                | 否      | 3          | 0, 1, 2, 3      | 针对StackExchange.Redis客户端适配开关。当Sta...       |
| EnableAclDirect             | 否      | no         | yes, no         | 是否开启ACL直连功能。默认为no。                         |
| MaxAuthFailTimes            | 否      | 5          | 0~10,000        | 防暴力破解相关参数。当前连续失败次数达到该阈...                  |
| ProxyTimeout                | 否      | 0          | 0~100,000       | Proxy需的应用连接超时。单位为秒（s）。当连接迟...              |
| bigkeys-composite-threshold | 否      | 1024       | 1~2,147,483,647 | 元素数大于该值的hashlistzsetbsetstream类型key...     |
| bigkeys-string-threshold    | 否      | 102400     | 1~2,147,483,647 | value大于该值的string类型key，会重新为大key...          |
| databases                   | 否      | 1000       | 1~1,000         | 允许连接的数据库数量限制。                              |
| mem-memoty-policy           | 是      | noeviction | noeviction      | 存储空间耗尽可采取的key清理策略。GaussDB for Red...       |
| notify-keyspace-events      | 否      |            | -               | 需要监听的事件类型。默认为空，表示该参数不生...                  |

- 单击“保存”，在弹出框中单击“确认”，保存修改。
- 单击“取消”，放弃本次设置。
- 单击“预览”，可对比参数修改前和修改后的值。

表 4-53 GeminiDB Redis 实例参数

| 参数名称             | 是否需要重启 | 值 | 允许值      | 描述                                                | 例外场景                |
|------------------|--------|---|----------|---------------------------------------------------|---------------------|
| AuthFailLockTime | 否      | 5 | 0~10,000 | 防暴力破解相关参数。单位为秒（s）。该参数表示可疑IP被锁定禁止访问的时长，超过该时长后自动解锁。 | Cluster集群版实例不支持该参数。 |

| 参数名称                        | 是否需要重启 | 值    | 允许值               | 描述                                                                                                                         | 例外场景                 |
|-----------------------------|--------|------|-------------------|----------------------------------------------------------------------------------------------------------------------------|----------------------|
| BigkeysQuantityLimitation   | 否      | 100  | 1 ~ 10,000        | string/hash/list/zset/set/exhash/stream类型的大key查询数量的限制，默认每种类型最多查询100个。                                                      | Cluster集群版实例不支持该参数。  |
| CompatibleMode              | 否      | 3    | 0, 1, 2, 3        | 针对StackExchange.Redis客户端的适配开关。当StackExchange.Redis客户端报错"Multiple databases are not supported on this server"时，请将该参数修改为"0"。 | Cluster集群版实例不支持该参数。  |
| EnableAclDbDirect           | 否      | no   | yes, no           | 是否开启DB直达功能。默认为no。                                                                                                          | Cluster集群版实例不支持该参数。  |
| MaxAuthFailTimes            | 否      | 5    | 0 ~ 10,000        | 防暴力破解相关参数。当密码错误次数累计达到该阈值后，实例会短时间内禁止该可疑IP访问。备注：配置为0表示关闭防暴力破解功能。                                                             | Cluster集群版实例不支持该参数。  |
| ProxyTimeout                | 否      | 0    | 0 ~ 100,000       | Proxy层的空闲连接超时。单位为秒（s）。当连接空闲时长达到该阈值时，proxy层主动关闭连接。备注：配置为0表示proxy层不会主动断开客户端连接。                                               | Cluster集群版实例不支持该参数。  |
| enable-acl-direct           | 否      | no   | yes, no           | 是否开启DB直达功能。默认为no。                                                                                                          | Proxy集群版、主备版实例不支持该参数 |
| bigkeys-composite-threshold | 否      | 1024 | 1 ~ 2,147,483,647 | 元素数大于该值的hash/list/zset/set/stream类型key，会被判断为大key，单位为个。默认为1024。                                                             | -                    |



| 参数名称                     | 是否需要重启 | 值          | 允许值                             | 描述                                                                                | 例外场景 |
|--------------------------|--------|------------|---------------------------------|-----------------------------------------------------------------------------------|------|
| bigkeys-string-threshold | 否      | 102400     | 1 ~ 2,147,483,647               | value大于该值的string类型key，会被判断为大key，单位为byte。默认为102400。                                | -    |
| databases                | 否      | 1000       | 1 ~ 1,000                       | 允许支持的DB数量限制。                                                                      | -    |
| key-scan-batch           | 否      | 5000       | 1 ~ 2,147,483,647               | 每次 key scan 扫描的 key 数量。                                                           | -    |
| maxmemory-policy         | 是      | noeviction | noeviction                      | 存储空间被写满后的Key丢弃策略。<br>GaussDB(for Redis)支持秒级无感扩容，写满后进入只读保护，不丢弃业务数据。备注：后续会开放自动扩容策略。 | -    |
| notify-keyspace-events   | 否      | -          | 取值为A,K,E,g,\$,l,s,h,z,x,e,t的组合。 | 需要监听的事件类型。默认值为空，表示该参数不生效。取值为A,K,E,g,\$,l,s,h,z,x,e,t的组合。                          | -    |
| slowlog-threshold        | 否      | 30000      | 80,000 ~ 100,000,000            | 控制台[慢日志]会记录执行时间大于该阈值的请求。单位为微秒（us）。备注：该参数配置过小可能影响实例性能，建议保持默认值。                     | -    |

**步骤4** 参数修改完成后，您可在“参数修改历史”页面，查看参数的修改详情。

查看参数修改详情的具体操作请参见[查看参数修改历史](#)。

 **说明**

参数修改后，会立即应用到当前实例。

根据参数列表中“是否需要重启”提示，进行相应操作：

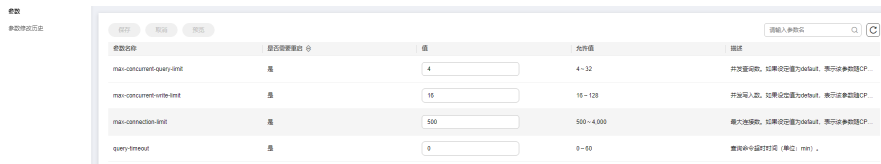
- 是：在实例列表中，查看“运行状态”，如果显示参数变更，等待重启，则需重启实例使之生效。
- 否：无需重启，立即生效。

----结束

修改自定义参数模板并应用到实例

- 步骤1 登录[云数据库GeminiDB控制台](#)。
- 步骤2 在左侧导航树，单击“参数模板管理”。
- 步骤3 在“参数模板管理”页面的“自定义”页签，选择目标参数模板，单击参数模板名称。
- 步骤4 在“参数”页面，根据需要修改相关参数值。

图 4-177 编辑参数模板



- 单击“保存”，在弹出框中单击“确认”，保存修改。
- 单击“取消”，放弃本次设置。
- 单击“预览”，可对比参数修改前和修改后的值。

图 4-178 预览修改参数



- 步骤5 参数修改完成后，您可在“参数修改历史”页面，查看参数的修改详情。
- 查看参数修改详情的具体操作请参见[查看参数修改历史](#)。

说明

- 参数模板修改后，不会立即应用到当前使用的实例，您需要进行应用操作才可生效，具体操作请参见[应用参数模板](#)。
- 参数修改历史页面仅显示7天之内的参数修改历史。

----结束

4.11.2 创建参数模板

您可以使用数据库参数模板中的参数来管理数据库接口配置。数据库参数模板就像是接口配置值的容器，这些值可应用于一个或多个数据库实例。

每个用户可以创建100个参数模板，同一项目下所有类型的实例可以共享该配额。

操作步骤

- 步骤1 登录[云数据库GeminiDB控制台](#)。
- 步骤2 在左侧导航树，单击“参数模板管理”。
- 步骤3 在“参数模板管理”页面，单击“创建参数模板”。
- 步骤4 选择兼容接口数据库版本，命名参数组，并添加参数组描述，单击“确定”，创建参数组模板。

图 4-179 创建参数模板

创建参数模板

兼容接口

Cassandra

InfluxDB

MongoDB

Redis

数据库版本

5.0

参数模板名

paramsGroup-d4f7

描述

请输入参数模板描述信息

0/256

您还可以创建98个参数模板。当前项目下所有实例类型共享参数模板配额。

确定

取消

- 选择该数据库接口参数模板所需兼容的接口类型。
- 选择数据库版本，例如：5.0。
- 参数模板名称在1位到64位之间，区分大小写，可包含字母、数字、中划线、下划线或句点，不能包含其他特殊字符。
- 描述不能超过256位，且不能包含回车和> ! < " & ' =特殊字符。

- 步骤5 参数组模板创建成功后，您可在“参数模板管理”页面，查看并管理创建完成的参数模板。

----结束

### 4.11.3 查看参数修改历史

#### 操作场景

您可以查看目标参数模板和当前实例的参数修改历史，以满足业务需要。

#### 使用须知

用户创建或导出的新参数模板，在未进行参数修改前，无修改历史。

#### 查看目标参数模板的参数修改历史

- 步骤1 登录[云数据库GeminiDB控制台](#)。
- 步骤2 在“参数模板管理”页面的“自定义”页签，选择目标参数模板，单击参数模板名称。
- 步骤3 单击左侧导航栏中的“参数修改历史”，您可查看参数对应的参数名称、修改前参数值、修改后参数值、修改状态和修改时间。

图 4-180 查看目标参数模板的参数修改历史

| 参数修改历史           |        |        |      |                               |  |
|------------------|--------|--------|------|-------------------------------|--|
| 只能查看7天之内的事件修改历史  |        |        |      |                               |  |
| 参数名称             | 修改前参数值 | 修改后参数值 | 修改状态 | 修改时间                          |  |
| AuthFailLockTime | 5      | 6      | 成功   | 2024/06/25 20:08:57 GMT+08:00 |  |

如果修改后参数模板未应用，请根据业务需要，参考[应用参数模板](#)，将其应用到对应实例。

----结束

#### 查看当前实例的参数修改历史

- 步骤1 登录[云数据库GeminiDB控制台](#)。
- 步骤2 在“实例管理”页面，选择指定实例，单击实例名称，进入“基本信息”页面。
- 步骤3 单击左侧导航栏中的“参数修改”，单击“参数修改历史”，您可查看参数对应的参数名称、修改前参数值、修改后参数值、修改状态和修改时间。

图 4-181 查看当前实例的参数修改历史

| 参数修改历史           |        |        |      |                               |      |                               |
|------------------|--------|--------|------|-------------------------------|------|-------------------------------|
| 当前仅显示7天之内的事件修改历史 |        |        |      |                               |      |                               |
| 参数名称             | 修改前参数值 | 修改后参数值 | 修改状态 | 修改时间                          | 是否应用 | 应用时间                          |
| AuthFailLockTime | 5      | 6      | 成功   | 2024/06/25 20:10:27 GMT+08:00 | 是    | 2024/06/25 20:10:27 GMT+08:00 |
| MaxAuthFailTimes | 5      | 6      | 成功   | 2024/06/25 20:10:51 GMT+08:00 | 是    | 2024/06/25 20:10:51 GMT+08:00 |

----结束

### 4.11.4 导出参数模板

- 您可以导出您创建的数据库实例参数列表，生成一个新的参数模板，供您后期使用。请参考[应用参数模板](#)将导出的参数模板应用到新的实例。

- 您可以将该实例对应的参数模板信息（参数名称，值，描述）导出到CSV中，方便查看并分析。

操作步骤

- 步骤1** 登录[云数据库GeminiDB控制台](#)。
- 步骤2** 在左侧导航栏选择“实例管理”，单击指定实例名称，进入基本信息页面。
- 步骤3** 在左侧导航栏选择“参数修改”，单击参数页签下的“导出”。

图 4-182 导出参数模板



- 导出到参数模板。将该实例对应参数列表导出并生成一个参数模板，供您后期使用。  
在弹出框中，填写新参数模板名称和描述，单击“确定”。

说明

- 参数模板名称在1位到64位之间，区分大小写，可包含字母、数字、中划线、下划线或句点，不能包含其他特殊字符。
- 参数模板的描述长度不能超过256个字符，且不能包含回车和>!"<'=特殊字符。

创建完成后，会生成一个新的参数模板，您可在“参数模板管理”页面的对其进行管理。

- 导出到文件。将该实例对应的参数模板信息（参数名称，值，描述）导出到CSV表中，方便用户查看并分析。  
在弹出框中，填写文件名称，单击“确定”。

说明

文件名称在4位到81位之间，必须以字母开头，可以包含字母、数字、中划线或下划线，不能包含其他特殊字符。

----结束

### 4.11.5 比较参数模板

您可以比较相同实例类型和兼容接口的参数模板，以了解该参数模板当前的配置情况。

#### 比较目标参数模板

- 步骤1 登录[云数据库GeminiDB控制台](#)。
- 步骤2 在左侧的导航栏，单击“参数模板管理”。
- 步骤3 在参数模板列表中，选择对应实例类型下用户创建的参数模板，单击“比较”。
- 步骤4 选择相同实例类型和兼容接口的不同参数模板，单击“确定”，比较两个参数模板之间的配置参数差异项。

图 4-183 比较参数模板



#### 比较当前实例的参数模板

- 步骤1 登录[云数据库GeminiDB控制台](#)。
- 步骤2 在左侧导航栏，单击“实例管理”。
- 步骤3 在实例列表中，选择指定的实例，单击实例名称，进入基本信息页面。
- 步骤4 在左侧导航栏中的“参数修改”，选择“参数”页签，单击“比较参数”。
- 步骤5 在弹出框中选择当前实例同数据库类型的参数模板，单击“确定”，比较两个参数模板的差异项。

图 4-184 比较当前实例的参数模板



- 存在差异项，则会显示差异参数的如下信息：参数名称、当前实例的参数值和被比较参数模板的参数值。
- 不存在差异项，则不显示。

----结束

## 4.11.6 复制参数模板

您可以复制您创建的自定义数据库参数模板。当您已创建一个数据库参数模板，并且想在新的数据库参数模板中包含该组中的大部分自定义参数和值时，复制参数模板是一个方便的解决方案。您还可以导出某数据库实例应用的参数列表，生成一个新的参数模板，供您后期使用。

您无法复制默认参数模板。不过，您可以创建基于默认参数模板的新参数模板。

### 操作步骤

**步骤1** 登录[云数据库GeminiDB控制台](#)。

**步骤2** 在左侧导航栏，单击“参数模板管理”。

**步骤3** 在参数模板管理页面的“自定义”页签，选择需要复制的参数模板，单击“复制”。

除了上述操作，您还可以在“实例管理”页面，单击实例名称，在左侧导航栏，单击“参数修改”，单击“导出”，将该实例对应参数列表导出并生成一个参数模板，供您后期使用。

**步骤4** 在弹出框中，填写新参数模板名称及描述，单击“确定”。

图 4-185 复制参数模板

复制参数模板

★ 源参数模板 paramsGroup-61c9

★ 新参数模板名 paramsGroup-8a65

描述 请输入参数模板描述信息

0/256

您还可以复制96个参数模板。当前项目下所有实例类型共享参数模板配额。

确定 取消

- 参数模板名称在1位到64位之间，区分大小写，可包含字母、数字、中划线、下划线或句点，不能包含其他特殊字符。

- 描述不能超过256位，且不能包含回车和> ! < " & ' =特殊字符。

创建完成后会生成一个新的参数模板，您可在参数模板列表中对其进行管理。

----结束

### 4.11.7 重置参数模板

您可根据自己的业务需求，重置自己创建的参数模板对应的所有参数，使其恢复到默认值。

#### 操作步骤

**步骤1** 登录[云数据库GeminiDB控制台](#)。

**步骤2** 在左侧导航栏，单击“参数模板管理”。

**步骤3** 在参数模板管理页面的“自定义”页签，选择需要重置的参数模板，单击“更多 > 重置”。

**步骤4** 单击“是”，将当前参数模板中的所有参数恢复到默认值。

----结束

### 4.11.8 应用参数模板

GeminiDB Redis支持应用参数模板。参数模板编辑修改后，不会立即应用到实例，您可以根据业务需要应用到实例中。

#### 操作步骤

**步骤1** 登录[云数据库GeminiDB控制台](#)。

**步骤2** 在左侧导航栏，单击“参数模板管理”。

**步骤3** 在“参数模板管理”页面，根据参数模板类型不同进行如下操作。

- 若需要将默认参数模板应用到实例，在“系统默认”页签的目标参数模板单击“应用”。
- 若需要将用户自己创建的参数模板应用到实例，在“自定义”页签的目标参数模板单击“更多 > 应用”。

一个参数模板可被应用到一个或多个实例。

**步骤4** 在弹出框中，选择需要应用的实例，单击“确定”。

参数模板应用成功后，您可[查看参数模板应用记录](#)。

----结束

### 4.11.9 查看参数模板应用记录

GeminiDB Redis支持查看参数模板所应用到实例的记录。

#### 操作步骤

**步骤1** 登录[云数据库GeminiDB控制台](#)。



**步骤2** 在左侧导航栏，单击“参数模板管理”。

**步骤3** 在“参数模板管理”页面，选择指定的参数模板，单击“更多 > 应用记录”。

您可以查看到当前参数模板应用到的实例名称或ID、应用状态、应用时间和失败原因。

----结束

### 4.11.10 修改参数模板描述

对于创建成功后的自定义参数模板，用户可以根据需要修改参数模板描述。

#### 操作步骤

**步骤1** 登录[云数据库GeminiDB控制台](#)。

**步骤2** 在左侧的导航栏，单击“参数模板管理”。

**步骤3** 在参数模板管理的“自定义”页面，选择指定的参数模板，单击“描述”列的.

**步骤4** 输入新的描述信息，单击, 提交修改，单击, 取消修改。

- 修改成功后，新的描述信息，可在参数模板列表的“描述”列查看。
- 参数模板的描述长度不能超过256个字符，且不能包含>!<"&'=特殊字符。

----结束

### 4.11.11 删除参数模板

您可以删除不需要的参数模板。

#### 使用须知

- 删除后的参数模板无法恢复，请谨慎操作。
- 默认参数模板不可进行删除。

#### 操作步骤

**步骤1** 登录[云数据库GeminiDB控制台](#)。

**步骤2** 在左侧导航栏，单击“参数模板管理”。

**步骤3** 在“参数模板管理”的“自定义”页面，选择需要删除的参数模板，单击“更多 > 删除”。

**步骤4** 单击“是”，即可删除当前参数模板。

----结束

## 4.12 日志与审计

## 4.12.1 日志配置管理

### 操作场景

配置访问日志后，GeminiDB Redis实例新生成的日志记录会上传到云日志服务（Log Tank Service，简称LTS）进行管理。

### 注意事项

- 经典部署模式实例如需使用该功能，您可以在管理控制台右上角，选择“[工单 > 新建工单](#)”，联系客服申请权限。
- 配置成功后，会产生一定费用。
- 确保与GeminiDB Redis实例相同region下的LTS服务已有日志组和日志流。  
更多日志组和日志流信息，请参见[日志管理](#)。

### 批量配置访问日志

**步骤1** 登录[云数据库GeminiDB控制台](#)。

**步骤2** 在左侧导航栏选择“日志配置管理”。

**步骤3** 选择一个或多个实例，单击“配置访问日志”。

**步骤4** 在下拉框分别选择LTS日志组和日志流，单击“确定”。

#### 说明

- 配置完成后不会立即生效，存在10分钟左右的时延，请知悉。
- 配置成功后，会产生一定费用，具体计费可参考LTS的[定价详情](#)。

图 4-186 配置访问日志

### 配置访问日志

**①**

- 访问日志提供了实例所请求的所有详细日志，日志存在LTS云日志服务中。
- 配置完成后不会立即生效，存在10分钟左右的时延，请知悉。
- 配置成功后，会产生一定费用。
- 在您进行LTS审计日志配置后，会默认上传所有审计策略。
- 若您的审计策略为打开的状态，则将会复用您之前的审计策略，且会在原有的审计日志计费基础上增加您LTS的审计日志收费。（原有的审计日志费用会在您关闭原有审计策略后终止）
- 出于对您实例的正常运行考虑，若您原有审计策略开启的情况下，配置了LTS审计日志，建议保留原有审计策略一段时间，待您LTS审计日志运转正常后，再行关闭。

选择日志类型 ☒ 慢日志 ☐ 审计日志

LTS慢日志配置

★ 配置日志组  [查看日志组](#)

★ 配置日志流

确定 取消

**步骤5** 如需解除LTS配置，选择一个或多个实例，单击“解除配置”。

**步骤6** 在弹框中，单击“确定”。

图 4-187 解除配置

### 解除配置

**⚠**

- 日志配置解除后，所生产的日志记录不会再向LTS服务上传。
- 日志配置解除后，不会立即生效，会有10分钟左右的时延，请知悉。

选择日志类型 ☒ 慢日志 ☐ 审计日志

确定 取消

----结束

4.12.2 查看和导出慢日志

GeminiDB Redis的日志管理功能支持查看数据库级别的慢日志，执行时间的单位为ms。通过该日志，可查找出执行效率低的语句，以便优化。

LTS 慢日志配置

如需使用该功能，您可以在管理控制台右上角，选择“工单 > 新建工单”联系客服申请权限。

- 步骤1 登录[云数据库GeminiDB控制台](#)。
- 步骤2 在左侧导航树，选择“日志配置管理”。
- 步骤3 选择指定的实例，单击LTS慢日志配置开关，配置日志记录上传LTS。
- 步骤4 在下拉框分别选择LTS日志组和日志流，单击“确定”。

说明

配置成功后，会产生一定费用，具体计费可参考LTS的[定价详情](#)。

图 4-188 LTS 慢日志配置



查看和导出日志明细

- 步骤1 登录[云数据库GeminiDB控制台](#)。
- 步骤2 在“实例管理”页面，选择指定的实例，单击实例名称。
- 步骤3 在左侧导航树，单击“慢日志”。
- 步骤4 在“慢日志”页面，设置查询条件，查看日志信息。

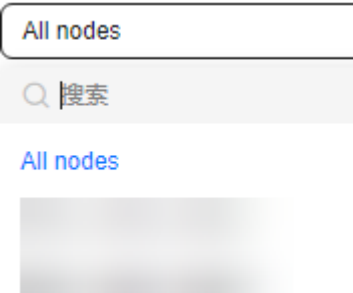
图 4-189 查看慢日志

| 节点名称     | 命令   | 命令参数key名称         | 最大执行时间(ms) | 大Key所在DB | 请求大小    | 请求参数个数 | 响应大小    | 响应元素个数                   | 执行结果时间                        |
|----------|------|-------------------|------------|----------|---------|--------|---------|--------------------------|-------------------------------|
| shardovr | open | "default#30e71... | 907.73     | 0        | 0.23 KB | 3      | 0.24 KB | 1, resp: "S230vnrdefa... | 2024/06/25 16:29:34 GMT+08:00 |
| shardovr | open | "default#3a3dc... | 871.78     | 0        | 0.24 KB | 3      | 0.25 KB | 1, resp: "S246vnrdefa... | 2024/06/25 16:29:34 GMT+08:00 |

 说明

- 部分低版本的实例需要参考[升级内核小版本](#)章节升级内核小版本以支持大Key所在DB, 请求大小, 请求参数个数, 回包大小, 回包元素个数这5种指标。
- 慢日志的默认阈值是300毫秒。
- 节点类型默认 “All nodes”，可查看实例下所有节点的慢日志信息，您也可以选择查看某个节点的慢日志信息。

图 4-190 设置查询节点



- 可选择查看全部语句类型或查看以下指定类型的慢查询语句：
  - SET
  - GET
  - DEL
  - INCR
  - INCRBY
  - INCRBYFLOAT
  - DECR
  - DECRBY
  - GETSET
  - APPEND
  - MGET
  - .....

 说明

常用的Redis语句类型均支持慢日志查询。

- 可查看对应节点在不同时间段的慢日志。

- 步骤5** 在“日志明细”页签下，您也可以单击日志列表右上方，导出日志明细。
- 导出完成后，您可以在本地查看生成的 “.csv” 文件。
  - 支持单次最多导出2000条日志明细。

图 4-191 导出慢日志



----结束

### 4.12.3 查看审计日志

GeminiDB Redis的日志管理功能支持查看数据库级别的审计日志。通过云日志服务进行分析日志、搜索日志、日志可视化、下载日志和查看实时日志等操作。

#### 使用须知

- 内部连接的操作不记录审计日志。
- 始终审计的命令：BigKeys keys flushall flushdb script client config 。
- 只审计大批量的命令（单次命令操作参数较多）：  
bitop msetnx pfcount pfmerge hdel hmget hmset hset lpush lpushx sadd srem  
zadd geoadd geohash bfininsert bfmadd bfmexists 。
- Exec 只审计单次事务过大的操作（超过100个命令）。
- Redis Cluster集群版实例不支持审计日志功能。

#### LTS 审计日志配置

如需使用该功能，您可以在管理控制台右上角，选择“[工单 > 新建工单](#)”联系客服申请权限。

- 步骤1** 登录[云数据库GeminiDB控制台](#)。
- 步骤2** 在左侧导航栏选择“日志配置管理”，
- 步骤3** 选择指定的实例，单击LTS审计日志配置开关，配置日志记录上传LTS。
- 步骤4** 在下拉框分别选择LTS日志组和日志流，单击“确定”。

#### 说明

配置成功后，会产生一定费用，具体计费可参考LTS的[定价详情](#)。

图 4-192 配置审计日志

#### 配置审计日志



- 访问日志提供了实例所请求的所有详细日志，日志存在LTS云日志服务中。
- 配置完成后不会立即生效，存在10分钟左右的时延，请知悉。
- 配置成功后，会产生一定费用。
- 在您进行LTS审计日志配置后，会默认上传所有审计策略。
- 若您的审计策略为打开的状态，则将会复用您之前的审计策略，且会在原有的审计日志计费基础上增加您LTS的审计日志收费。（原有的审计日志费用会在您关闭原有审计策略后终止）
- 出于对您实例的正常运行考虑，若您原有审计策略开启的情况下，配置了LTS审计日志，建议保留原有审计策略一段时间，待您LTS审计日志运转正常后，再行关闭。

\* 配置日志组

 [查看日志组](#)

\* 配置日志流



确定

取消

----结束

查看日志明细

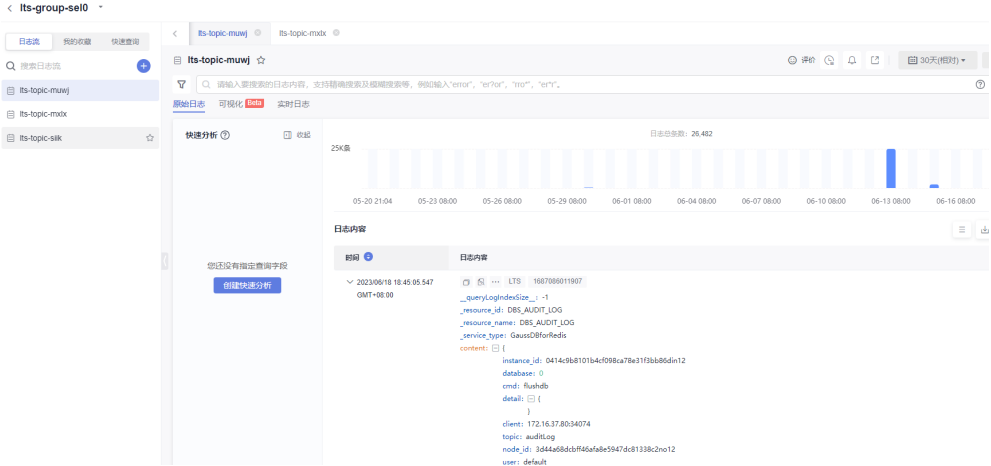
- 步骤1 登录[云日志服务LTS控制台](#)。
- 步骤2 在“日志管理”页面的“日志组列表”中，选择日志组和日志流，单击选择的日志流，进入日志流页面。

图 4-193 选择日志流



- 步骤3 相对时间默认选择15分钟。您可以根据需要，在右上角选择相对时间。
- 步骤4 在“原始日志”区域查看相对时间内的审计日志。

图 4-194 审计日志



----结束

4.12.4 CTS 审计

4.12.4.1 支持审计的 GeminiDB Redis 关键操作列表

通过云审计服务，您可以记录与GeminiDB Redis相关的操作事件，便于日后的查询、审计和回溯。

表 4-54 GeminiDB Redis 的关键操作列表

| 操作名称     | 资源类型           | 事件名称                              |
|----------|----------------|-----------------------------------|
| 创建实例     | instance       | NoSQLCreateInstance               |
| 删除实例     | instance       | NoSQLDeleteInstance               |
| 扩容节点     | instance       | NoSQLEnlargeInstance              |
| 缩容节点     | instance       | NoSQLReduceInstance               |
| 重启实例     | instance       | NoSQLRestartInstance              |
| 恢复到新实例   | instance       | NoSQLRestoreNewInstance           |
| 磁盘扩容     | instance       | NoSQLExtendInstanceVolume         |
| 重置密码     | instance       | NoSQLResetPassword                |
| 修改实例名称   | instance       | NoSQLRenameInstance               |
| 绑定弹性公网IP | instance       | NoSQLResizeInstance               |
| 解绑弹性公网IP | instance       | NoSQLBindEIP                      |
| 规格变更     | instance       | NoSQLUnBindEIP                    |
| 实例冻结     | instance       | NoSQLFreezeInstance               |
| 实例解冻     | instance       | NoSQLUnfreezeInstance             |
| 创建备份     | backup         | NoSQLCreateBackup                 |
| 删除备份     | backup         | NoSQLDeleteBackup                 |
| 设置备份策略   | backup         | NoSQLSetBackupPolicy              |
| 添加实例标签   | tag            | NoSQLAddTags                      |
| 修改实例标签   | tag            | NoSQLModifyInstanceTag            |
| 删除实例标签   | tag            | NoSQLDeleteInstanceTag            |
| 创建参数模板   | parameterGroup | NoSQLCreateConfigurations         |
| 修改参数模板   | parameterGroup | NoSQLUpdateConfigurations         |
| 修改实例参数   | parameterGroup | NoSQLUpdateInstanceConfigurations |
| 复制参数模板   | parameterGroup | NoSQLCopyConfigurations           |
| 重置参数模板   | parameterGroup | NoSQLResetConfigurations          |
| 应用参数模板   | parameterGroup | NoSQLApplyConfigurations          |
| 删除参数模板   | parameterGroup | NoSQLDeleteConfigurations         |



| 操作名称      | 资源类型     | 事件名称                       |
|-----------|----------|----------------------------|
| 删除扩容失败的节点 | instance | NoSQLDeleteEnlargeFailNode |
| 切换SSL     | instance | NoSQLSwitchSSL             |
| 修改实例安全组   | instance | NoSQLModifySecurityGroup   |
| 回收站策略     | instance | NoSQLModifyRecyclePolicy   |

#### 4.12.4.2 查看 GeminiDB Redis 追踪事件

##### 操作场景

在您开启了云审计服务后，系统开始记录云服务资源的操作。云审计服务管理控制台保存最近7天的操作记录。

本节介绍如何在云审计服务管理控制台查看最近7天的操作记录。

##### 操作步骤

- 步骤1** 登录[云审计服务CTS控制台](#)。
- 步骤2** 单击管理控制台左上角的，选择区域和项目。
- 步骤3** 在左侧导航树，单击“事件列表”，进入事件列表信息页面。
- 步骤4** 事件列表支持通过筛选来查询对应的操作事件。当前事件列表支持五个维度的组合查询，详细信息如下：
- 事件类型：可选项为“管理事件”、“数据时间”，只可选择其中一项。
  - 事件来源、资源类型和筛选类型。  
在下拉框中选择查询条件。  
其中，筛选类型选择“按事件名称”时，还需选择某个具体的事件名称。  
选择“按资源ID”时，还需选择或者手动输入某个具体的资源ID。  
选择“按资源名称”时，还需选择或手动输入某个具体的资源名称。
  - 操作用户：在下拉框中选择某一具体的操作用户，此操作用户指用户级别，而非租户级别。
  - 事件级别：可选项为“所有事件级别”、“normal”、“warning”、“incident”，只可选择其中一项。
  - 起始时间、结束时间：可通过选择时间段查询操作事件。
- 步骤5** 在需要查看的记录左侧，单击展开该记录的详细信息。
- 步骤6** 在需要查看的记录右侧，单击“查看事件”，在弹出框中显示该操作事件结构的详细信息。
- 结束

## 4.13 查看监控指标与配置告警

### 4.13.1 GeminiDB Redis 支持的监控指标

#### 功能说明

本节定义了GeminiDB Redis上报云监控服务的监控指标的命名空间，监控指标列表和维度定义，用户可以通过云监控服务提供的API接口来检索GeminiDB Redis产生的监控指标和告警信息。

 说明

云监控服务最大支持4个层级维度，维度编号从0开始，编号3为最深层级。例如监控指标中的维度信息为“redis\_cluster\_id,redis\_node\_id”时，表示对应的监控指标的维度存在层级关系，且“redis\_cluster\_id”为0层，“redis\_node\_id”为1层。

GeminiDB Redis支持的监控指标分为实例级和节点级（如下表4-55所示），监控指标均需要在实例节点上查看，具体方法请参见[查看GeminiDB Redis监控指标](#)。其中，实例级的指标各实例节点展示结果一致，均为实例的监控结果。

表 4-55 指标分类介绍

| 指标级 | 指标类型                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
|-----|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 实例级 | <a href="#">实例级指标</a>                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| 节点级 | <a href="#">基础指标</a><br><a href="#">String命令类型指标</a><br><a href="#">Hash命令类型指标</a><br><a href="#">List命令类型指标</a><br><a href="#">Set命令类型指标</a><br><a href="#">Zset命令类型指标</a><br><a href="#">Bitmap命令类型指标</a><br><a href="#">Stream命令类型指标</a><br><a href="#">Geo命令类型指标</a><br><a href="#">Hyperloglog命令类型指标</a><br><a href="#">Pub/Sub命令类型指标</a><br><a href="#">Scripting命令类型指标</a><br><a href="#">Transactions命令类型指标</a><br><a href="#">ExHash命令类型指标</a><br><a href="#">通用命令指标</a> |

#### 命名空间

SYS.NoSQL

支持的实例级指标

| 指标ID                                 | 指标名称           | 指标含义                                                                                                       | 取值范围            | 单位       | 进制  | 维度               | 监控周期（原始指标） |
|--------------------------------------|----------------|------------------------------------------------------------------------------------------------------------|-----------------|----------|-----|------------------|------------|
| redis823_cluster_persist_key_num     | 实例不带过期时间的key数量 | 实例不带过期时间的key数量（全部DB）。                                                                                      | ≥ 0<br>Counts   | Counts   | 不涉及 | redis_cluster_id | 1分钟        |
| redis822_cluster_cmd_err_num         | 实例命令返回ERROR次数  | 实例每秒命令执行错误的次数。例如，命令不存在、参数错误等情况。                                                                            | ≥ 0<br>Counts/s | Counts/s | 不涉及 | redis_cluster_id | 1分钟        |
| redis821_cluster_out_bandwidth_usage | 实例ELB出带宽使用率    | 实例发送的总流量和ELB最大带宽的百分比。                                                                                      | 0~100%          | %        | 不涉及 | redis_cluster_id | 1分钟        |
| redis820_cluster_in_bandwidth_usage  | 实例ELB入带宽使用率    | 实例接收的总流量和ELB最大带宽的百分比。<br>若入带宽使用率达到100%，说明已经超出ELB规格最大带宽。请工单联系后台升级ELB带宽，若ELB带宽已达到最大10Gbit/s，请审视业务是否合理，或进行拆分。 | 0~100%          | %        | 不涉及 | redis_cluster_id | 1分钟        |
| redis816_cluster_disk_usage          | 实例的存储容量利用率     | 该指标为实例的存储容量利用率。                                                                                            | 0~100%          | %        | 不涉及 | redis_cluster_id | 1分钟        |
| redis815_cluster_cpu_usage           | 实例的CPU利用率      | 该指标为实例的CPU利用率。                                                                                             | 0~100%          | %        | 不涉及 | redis_cluster_id | 1分钟        |
| redis814_cluster_mem_usage           | 实例的内存利用率       | 该指标为实例的内存利用率。                                                                                              | 0~100%          | %        | 不涉及 | redis_cluster_id | 1分钟        |
| redis813_cluster_slow_query_count    | 实例的慢查询数量       | 该指标为实例的慢查询数量。                                                                                              | ≥ 0<br>Counts   | Counts   | 不涉及 | redis_cluster_id | 1分钟        |

| 指标ID                                     | 指标名称       | 指标含义                       | 取值范围       | 单位     | 进制  | 维度               | 监控周期（原始指标） |
|------------------------------------------|------------|----------------------------|------------|--------|-----|------------------|------------|
| redis812_cluster_processed_command_count | 实例的处理命令总数量 | 该指标为实例的处理命令总数量。            | ≥ 0 Counts | Counts | 不涉及 | redis_cluster_id | 1分钟        |
| redis811_cluster_max_connect_count       | 实例的最大连接数   | 该指标为实例的最大连接数（节点数*1w）。      | ≥ 0 Counts | Counts | 不涉及 | redis_cluster_id | 1分钟        |
| redis808_cluster_new_client_connection   | 实例的每秒新建连接数 | 该指标为实例的每秒新建连接数。            | ≥ 0 Counts | Counts | 不涉及 | redis_cluster_id | 1分钟        |
| redis807_cluster_all_connection_count    | 实例的总连接数    | 该指标为实例的总连接数(所有节点已使用连接数之和)。 | ≥ 0 Counts | Counts | 不涉及 | redis_cluster_id | 1分钟        |
| redis806_cluster_max_connection_usage    | 实例的连接数使用率  | 该指标为实例中最大的节点连接数使用率。        | 0~100%     | %      | 不涉及 | redis_cluster_id | 1分钟        |
| redis805_cluster_avg_hit_rate            | 实例的平均命中率   | 该指标为实例的平均命中率。              | 0~100%     | %      | 不涉及 | redis_cluster_id | 1分钟        |
| redis804_cluster_all_p99_usec            | 实例的p99时延   | 该指标为实例的p99时延。              | ≥ 0 us     | us     | 不涉及 | redis_cluster_id | 1分钟        |
| redis803_cluster_all_avg_usec            | 实例的平均时延    | 该指标为实例的平均时延。               | ≥ 0 us     | us     | 不涉及 | redis_cluster_id | 1分钟        |

| 指标ID                                | 指标名称                | 指标含义                     | 取值范围         | 单位       | 进制        | 维度               | 监控周期（原始指标） |
|-------------------------------------|---------------------|--------------------------|--------------|----------|-----------|------------------|------------|
| redis802_cluster_max_response_argc  | 实例的客户端单次请求获取的最大元素个数 | 该指标为实例的客户端单次请求获取的最大元素个数。 | ≥ 0 Counts   | Counts   | 不涉及       | redis_cluster_id | 1分钟        |
| redis801_cluster_max_response_bytes | 实例的客户端单次请求获取的最大字节数  | 该指标为实例的客户端单次请求获取的最大字节数。  | ≥ 0 Bytes    | Bytes    | 1024(IEC) | redis_cluster_id | 1分钟        |
| redis800_cluster_max_request_argc   | 实例的客户端单次请求发送的最大参数个数 | 该指标为实例的客户端单次请求发送的最大参数个数。 | ≥ 0 Counts   | Counts   | 不涉及       | redis_cluster_id | 1分钟        |
| redis799_cluster_max_request_bytes  | 实例的客户端单次请求发送的最大字节数  | 该指标为实例的客户端单次请求发送的最大字节数。  | ≥ 0 Bytes    | Bytes    | 1024(IEC) | redis_cluster_id | 1分钟        |
| redis798_cluster_expire_key_counts  | 实例的设置过期时间Key数量      | 该指标为实例的设置过期时间Key数量。      | ≥ 0 Counts   | Counts   | 不涉及       | redis_cluster_id | 1分钟        |
| redis689_qps_receive_total          | 实例级接收总流量            | 描述该实例所接收的总流量，反映应用侧流量大小。  | ≥ 0 Bytes/s  | Bytes/s  | 1024(IEC) | redis_cluster_id | 1分钟        |
| redis688_qps_send_total             | 实例级发送总流量            | 描述该实例所发送的总流量，反映应用侧流量大小。  | ≥ 0 Bytes/s  | Bytes/s  | 1024(IEC) | redis_cluster_id | 1分钟        |
| redis668_cluster_key_counts         | 实例的key数量            | 该指标为实例的key总数。            | ≥ 0 Counts   | Counts   | 不涉及       | redis_cluster_id | 1分钟        |
| redis667_cluster_qps                | 实例的QPS              | 该指标为实例的QPS。              | ≥ 0 Counts/s | Counts/s | 不涉及       | redis_cluster_id | 1分钟        |

| 指标ID                                       | 指标名称             | 指标含义                                                         | 取值范围         | 单位       | 进制        | 维度               | 监控周期（原始指标） |
|--------------------------------------------|------------------|--------------------------------------------------------------|--------------|----------|-----------|------------------|------------|
| redis834_cluster_dbcache_recv_bps          | dbcache同步速率      | dbcache功能中 GeminiDB Redis接收到的总流量速率。                          | ≥ 0 Byte s/s | Byte s/s | 1024(IEC) | redis_cluster_id | 1分钟        |
| redis833_cluster_dbcache_replication_delay | dbcache主从同步时延    | dbcache功能中 RDS(主库)和 GeminiDB Redis(从库)同步的时延, 值越小, 代表同步越及时。   | ≥ 0 ms       | ms       | 不涉及       | redis_cluster_id | 1分钟        |
| redis832_cluster_dbcache_binlog_delay      | dbcache主从同步位点延迟量 | dbcache功能中 RDS(主库)和 GeminiDB Redis(从库)同步位点的差值, 值越小, 代表同步越及时。 | ≥ 0 Counts   | Counts   | 不涉及       | redis_cluster_id | 1分钟        |
| redis831_cluster_dbcache_total_ops         | dbcache每秒总执行数    | dbcache每秒总执行数。                                               | ≥ 0 Counts/s | Counts/s | 不涉及       | redis_cluster_id | 1分钟        |

节点级指标

基础指标

表 4-56 基础指标

| 指标ID                       | 指标名称     | 指标含义          | 取值范围        | 单位      | 进制  | 维度                             | 监控周期（原始指标） |
|----------------------------|----------|---------------|-------------|---------|-----|--------------------------------|------------|
| redis817_slow_query_count  | 节点的慢查询数量 | 该指标为节点的慢查询数量。 | ≥ 0 Count s | Count s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis810_max_connect_count | 节点的最大连接数 | 该指标为节点的最大连接数。 | ≥ 0 Count s | Count s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |

| 指标ID                                      | 指标名称              | 指标含义                              | 取值范围               | 单位     | 进制  | 维度                             | 监控周期<br>(原始指标) |
|-------------------------------------------|-------------------|-----------------------------------|--------------------|--------|-----|--------------------------------|----------------|
| redis809_processed_command_count          | 节点的处理命令总数         | 该指标为节点的处理命令总数。                    | $\geq 0$<br>Counts | Counts | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis687_client_network_packet_return_p99 | redis客户端网络回包p99耗时 | 描述 proxy到客户端网络回包 p99耗时，代表应用侧网络质量。 | $\geq 0$ us        | us     | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis686_client_network_packet_return_max | redis客户端网络回包最大耗时  | 描述 proxy到客户端网络回包最大耗时，代表应用侧网络质量。   | $\geq 0$ us        | us     | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis685_client_network_packet_return_avg | redis客户端网络回包平均耗时  | 描述 proxy到客户端网络回包平均耗时，代表应用侧网络质量。   | $\geq 0$ us        | us     | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis684_client_packet_return_queue_p99   | redis客户端回包排队p99耗时 | 描述 proxy到客户端回包排队 p99耗时，代表应用侧网络质量。 | $\geq 0$ us        | us     | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |

| 指标ID                                    | 指标名称              | 指标含义                           | 取值范围            | 单位     | 进制        | 维度                             | 监控周期<br>(原始指标) |
|-----------------------------------------|-------------------|--------------------------------|-----------------|--------|-----------|--------------------------------|----------------|
| redis683_client_packet_return_queue_max | redis客户端回包排队最大耗时  | 描述proxy到客户端回包排队最大耗时，代表应用侧网络质量。 | $\geq 0$ us     | us     | 不涉及       | redis_cluster_id,redis_node_id | 1分钟            |
| redis682_client_packet_return_queue_avg | redis客户端回包排队平均耗时  | 描述proxy到客户端回包排队平均耗时，代表应用侧网络质量。 | $\geq 0$ us     | us     | 不涉及       | redis_cluster_id,redis_node_id | 1分钟            |
| redis678_max_response_argc              | 客户端单次请求获取的最大元素个数  | 描述客户端单次请求获取的最大元素个数。            | $\geq 0$ Counts | Counts | 不涉及       | redis_cluster_id,redis_node_id | 1分钟            |
| redis677_max_response_bytes             | 客户端单次请求获取的最大字节数   | 描述客户端单次请求获取的最大字节数。             | $\geq 0$ Bytes  | Bytes  | 1024(IEC) | redis_cluster_id,redis_node_id | 1分钟            |
| redis676_max_request_argc               | 客户端单次请求发送的最大参数个数  | 描述客户端单次请求发送的最大参数个数。            | $\geq 0$ Counts | Counts | 不涉及       | redis_cluster_id,redis_node_id | 1分钟            |
| redis675_max_request_bytes              | 客户端单次请求发送的最大字节数   | 描述客户端单次请求发送的最大字节数。             | $\geq 0$ Bytes  | Bytes  | 1024(IEC) | redis_cluster_id,redis_node_id | 1分钟            |
| redis674_max_pipeline                   | 客户端pipeline最大命令数量 | 描述客户端使用pipeline批量发送命令总数量。      | $\geq 0$ Counts | Counts | 不涉及       | redis_cluster_id,redis_node_id | 1分钟            |



| 指标ID                           | 指标名称         | 指标含义                                       | 取值范围                 | 单位       | 进制  | 维度                             | 监控周期<br>(原始指标) |
|--------------------------------|--------------|--------------------------------------------|----------------------|----------|-----|--------------------------------|----------------|
| redis673_wrong_auth            | 每秒AUTH认证失败次数 | 描述节点上每秒AUTH认证失败次数。                         | $\geq 0$<br>Counts   | Counts   | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis672_no_auth               | 每秒未认证导致请求失败数 | 描述节点上每秒未认证导致请求失败数。                         | $\geq 0$<br>Counts   | Counts   | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis671_new_client_connection | 每秒新建连接数      | 描述节点上每秒新建连接数。                              | $\geq 0$<br>Counts   | Counts   | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis670_hit_rate              | 命中率          | 采集周期内的命中率, 计算方法为: Key命中数÷(Key命中数+Key未命中数)。 | 0~100 %              | %        | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis669_connection_usage      | 连接数使用率       | 采集周期内的连接数使用率, 计算方法为: 已使用连接数÷总连接数。          | 0~100 %              | %        | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis319_all_qps               | 节点Proxy层的QPS | 该指标为节点Proxy层的QPS。                          | $\geq 0$<br>Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis318_all_p99               | 访问p99时延      | 该指标为节点执行所有命令的p99时延。                        | $\geq 0$ us          | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |

| 指标ID                           | 指标名称         | 指标含义                      | 取值范围               | 单位        | 进制        | 维度                             | 监控周期<br>(原始指标) |
|--------------------------------|--------------|---------------------------|--------------------|-----------|-----------|--------------------------------|----------------|
| redis317_all_max_usec          | 访问最大时延       | 该指标为节点执行所有命令的最大时延。        | $\geq 0$ us        | us        | 不涉及       | redis_cluster_id,redis_node_id | 1分钟            |
| redis316_all_avg_usec          | 访问平均时延       | 该指标为节点执行所有命令的平均时延。        | $\geq 0$ us        | us        | 不涉及       | redis_cluster_id,redis_node_id | 1分钟            |
| redis032_shard_qps             | 节点Shard层的QPS | 该指标为节点Shard层的QPS。         | $\geq 0$ Count/s/s | Count/s/s | 不涉及       | redis_cluster_id,redis_node_id | 1分钟            |
| redis021_proxy_send_client_bps | 节点发送流量速率     | 该指标为节点发送客户端的流量速率。<br>/s   | $\geq 0$ Bytes/s   | Bytes/s   | 1024(IEC) | redis_cluster_id,redis_node_id | 1分钟            |
| redis020_proxy_recv_client_bps | 节点接收流量速率     | 该指标为节点接收客户端的流量速率。<br>/s   | $\geq 0$ Bytes/s   | Bytes/s   | 1024(IEC) | redis_cluster_id,redis_node_id | 1分钟            |
| gemini004_bytes_in             | 网络输入吞吐量      | 统计平均每秒从测量对象的所有网络适配器输入的流量。 | $\geq 0$ Bytes/s   | Bytes/s   | 1024(IEC) | redis_cluster_id,redis_node_id | 1分钟            |
| gemini003_bytes_out            | 网络输出吞吐量      | 统计平均每秒从测量对象的所有网络适配器输出的流量。 | $\geq 0$ Bytes/s   | Bytes/s   | 1024(IEC) | redis_cluster_id,redis_node_id | 1分钟            |

| 指标ID                       | 指标名称         | 指标含义                  | 取值范围                     | 单位           | 进制        | 维度                             | 监控周期<br>(原始指标) |
|----------------------------|--------------|-----------------------|--------------------------|--------------|-----------|--------------------------------|----------------|
| redis019_proxy_response_ps | proxy的返回请求速率 | 该指标为proxy返回请求给客户端的速率。 | $\geq 0$<br>Count<br>s/s | Count<br>s/s | 不涉及       | redis_cluster_id,redis_node_id | 1分钟            |
| redis018_proxy_request_ps  | proxy的接收请求速率 | 该指标为proxy接收客户端请求速率。   | $\geq 0$<br>Count<br>s/s | Count<br>s/s | 不涉及       | redis_cluster_id,redis_node_id | 1分钟            |
| redis017_proxy_accept      | 节点的客户端连接数    | 该指标为节点的客户端连接数。        | $\geq 0$<br>Count<br>s   | Count<br>s   | 不涉及       | redis_cluster_id,redis_node_id | 1分钟            |
| nosql007_disk_used_size    | 存储容量使用量      | 该指标为实例的存储容量使用量。       | $\geq 0$<br>GB           | GB           | 1024(IEC) | redis_cluster_id,redis_node_id | 1分钟            |
| nosql006_disk_total_size   | 存储容量总容量      | 该指标为实例的存储容量总容量。       | $\geq 0$<br>GB           | GB           | 1024(IEC) | redis_cluster_id,redis_node_id | 1分钟            |
| nosql005_disk_usage        | 存储容量使用率      | 该指标为存储容量使用率。          | 0~100<br>%               | %            | 不涉及       | redis_cluster_id,redis_node_id | 1分钟            |
| nosql002_mem_usage         | 内存利用率        | 该指标为从系统层面采集的内存使用率。    | 0~100<br>%               | %            | 不涉及       | redis_cluster_id,redis_node_id | 1分钟            |
| nosql001_cpu_usage         | CPU利用率       | 该指标为从系统层面采集的CPU使用率。   | 0~100<br>%               | %            | 不涉及       | redis_cluster_id,redis_node_id | 1分钟            |

String 命令类型指标

表 4-57 String 命令类型指标

| 指标ID                   | 指标名称           | 指标含义                    | 取值范围            | 单位       | 进制  | 维度                             | 监控周期<br>(原始指标) |
|------------------------|----------------|-------------------------|-----------------|----------|-----|--------------------------------|----------------|
| redis107_decr_qps      | decr命令的QPS     | 该指标为节点执行decr命令的QPS。     | ≥ 0<br>Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis106_decr_p99      | decr命令的p99时延   | 该指标为节点执行decr命令的p99时延。   | ≥ 0 us          | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis104_decr_avg_usec | decr命令的平均时延    | 该指标为节点执行decr命令的平均时延。    | ≥ 0 us          | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis103_incr_qps      | incr命令的QPS     | 该指标为节点执行incr命令的QPS。     | ≥ 0<br>Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis102_incr_p99      | incr命令的p99时延   | 该指标为节点执行incr命令的p99时延。   | ≥ 0 us          | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis100_incr_avg_usec | incr命令的平均时延    | 该指标为节点执行incr命令的平均时延。    | ≥ 0 us          | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis099_strlen_qps    | strlen命令的QPS   | 该指标为节点执行strlen命令的QPS。   | ≥ 0<br>Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis098_strlen_p99    | strlen命令的p99时延 | 该指标为节点执行strlen命令的p99时延。 | ≥ 0 us          | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |

| 指标ID                       | 指标名称             | 指标含义                      | 取值范围              | 单位       | 进制  | 维度                             | 监控周期<br>(原始指标) |
|----------------------------|------------------|---------------------------|-------------------|----------|-----|--------------------------------|----------------|
| redis096_strlen_avg_usec   | strlen命令的平均时延    | 该指标为节点执行strlen命令的平均时延。    | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis095_substr_qps        | substr命令的QPS     | 该指标为节点执行substr命令的QPS。     | $\geq 0$ Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis094_substr_p99        | substr命令的p99时延   | 该指标为节点执行substr命令的p99时延。   | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis092_substr_avg_usec   | substr命令的平均时延    | 该指标为节点执行substr命令的平均时延。    | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis091_setrange_qps      | setrange命令的QPS   | 该指标为节点执行setrange命令的QPS。   | $\geq 0$ Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis090_setrange_p99      | setrange命令的p99时延 | 该指标为节点执行setrange命令的p99时延。 | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis088_setrange_avg_usec | setrange命令的平均时延  | 该指标为节点执行setrange命令的平均时延。  | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis087_getrange_qps      | getrange命令的QPS   | 该指标为节点执行getrange命令的QPS。   | $\geq 0$ Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |

| 指标ID                       | 指标名称             | 指标含义                      | 取值范围              | 单位       | 进制  | 维度                             | 监控周期<br>(原始指标) |
|----------------------------|------------------|---------------------------|-------------------|----------|-----|--------------------------------|----------------|
| redis086_getrange_p99      | getrange命令的p99时延 | 该指标为节点执行getrange命令的p99时延。 | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis084_getrange_avg_usec | getrange命令的平均时延  | 该指标为节点执行getrange命令的平均时延。  | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis083_mset_qps          | mset命令的QPS       | 该指标为节点执行mset命令的QPS。       | $\geq 0$ Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis082_mset_p99          | mset命令的p99时延     | 该指标为节点执行mset命令的p99时延。     | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis080_mset_avg_usec     | mset命令的平均时延      | 该指标为节点执行mset命令的平均时延。      | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis079_mget_qps          | mget命令的QPS       | 该指标为节点执行mget命令的QPS。       | $\geq 0$ Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis078_mget_p99          | mget命令的p99时延     | 该指标为节点执行mget命令的p99时延。     | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis076_mget_avg_usec     | mget命令的平均时延      | 该指标为节点执行mget命令的平均时延。      | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |

| 指标ID                     | 指标名称           | 指标含义                    | 取值范围              | 单位       | 进制  | 维度                             | 监控周期<br>(原始指标) |
|--------------------------|----------------|-------------------------|-------------------|----------|-----|--------------------------------|----------------|
| redis075_append_qps      | append命令的QPS   | 该指标为节点执行append命令的QPS。   | $\geq 0$ Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis074_append_p99      | append命令的p99时延 | 该指标为节点执行append命令的p99时延。 | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis072_append_avg_usec | append命令的平均时延  | 该指标为节点执行append命令的平均时延。  | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis071_getset_qps      | getset命令的QPS   | 该指标为节点执行getset命令的QPS。   | $\geq 0$ Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis070_getset_p99      | getset命令的p99时延 | 该指标为节点执行getset命令的p99时延。 | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis068_getset_avg_usec | getset命令的平均时延  | 该指标为节点执行getset命令的平均时延。  | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis067_get_qps         | get命令的QPS      | 该指标为节点执行get命令的QPS。      | $\geq 0$ Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis066_get_p99         | get命令的p99时延    | 该指标为节点执行get命令的p99时延。    | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |

| 指标ID                  | 指标名称        | 指标含义                 | 取值范围              | 单位       | 进制  | 维度                             | 监控周期<br>(原始指标) |
|-----------------------|-------------|----------------------|-------------------|----------|-----|--------------------------------|----------------|
| redis064_get_avg_usec | get命令的平均时延  | 该指标为节点执行get命令的平均时延。  | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis063_set_qps      | set命令的QPS   | 该指标为节点执行set命令的QPS。   | $\geq 0$ Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis062_set_p99      | set命令的p99时延 | 该指标为节点执行set命令的p99时延。 | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis060_set_avg_usec | set命令的平均时延  | 该指标为节点执行set命令的平均时延。  | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |

Hash 命令类型指标

表 4-58 Hash 命令类型指标

| 指标ID                    | 指标名称          | 指标含义                   | 取值范围              | 单位       | 进制  | 维度                             | 监控周期<br>(原始指标) |
|-------------------------|---------------|------------------------|-------------------|----------|-----|--------------------------------|----------------|
| redis159_hscan_qps      | hscan命令的QPS   | 该指标为节点执行hscan命令的QPS。   | $\geq 0$ Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis158_hscan_p99      | hscan命令的p99时延 | 该指标为节点执行hscan命令的p99时延。 | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis156_hscan_avg_usec | hscan命令的平均时延  | 该指标为节点执行hscan命令的平均时延。  | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |



| 指标ID                      | 指标名称            | 指标含义                     | 取值范围              | 单位       | 进制  | 维度                             | 监控周期<br>(原始指标) |
|---------------------------|-----------------|--------------------------|-------------------|----------|-----|--------------------------------|----------------|
| redis155_hvals_qps        | hvals命令的QPS     | 该指标为节点执行hvals命令的QPS。     | $\geq 0$ Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis154_hvals_p99        | hvals命令的p99时延   | 该指标为节点执行hvals命令的p99时延。   | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis152_hvals_avg_usec   | hvals命令的平均时延    | 该指标为节点执行hvals命令的平均时延。    | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis151_hstrlen_qps      | hstrlen命令的QPS   | 该指标为节点执行hstrlen命令的QPS。   | $\geq 0$ Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis150_hstrlen_p99      | hstrlen命令的p99时延 | 该指标为节点执行hstrlen命令的p99时延。 | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis148_hstrlen_avg_usec | hstrlen命令的平均时延  | 该指标为节点执行hstrlen命令的平均时延。  | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis147_hlen_qps         | hlen命令的QPS      | 该指标为节点执行hlen命令的QPS。      | $\geq 0$ Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis146_hlen_p99         | hlen命令的p99时延    | 该指标为节点执行hlen命令的p99时延。    | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis144_hlen_avg_usec    | hlen命令的平均时延     | 该指标为节点执行hlen命令的平均时延。     | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis143_hkeys_qps        | hkeys命令的QPS     | 该指标为节点执行hkeys命令的QPS。     | $\geq 0$ Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis142_hkeys_p99        | hkeys命令的p99时延   | 该指标为节点执行hkeys命令的p99时延。   | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |

| 指标ID                      | 指标名称            | 指标含义                     | 取值范围                 | 单位       | 进制  | 维度                             | 监控周期<br>(原始指标) |
|---------------------------|-----------------|--------------------------|----------------------|----------|-----|--------------------------------|----------------|
| redis140_hkeys_avg_usec   | hkeys命令的平均时延    | 该指标为节点执行hkeys命令的平均时延。    | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis139_hincrby_qps      | hincrby命令的QPS   | 该指标为节点执行hincrby命令的QPS。   | $\geq 0$<br>Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis138_hincrby_p99      | hincrby命令的p99时延 | 该指标为节点执行hincrby命令的p99时延。 | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis136_hincrby_avg_usec | hincrby命令的平均时延  | 该指标为节点执行hincrby命令的平均时延。  | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis135_hexists_qps      | hexists命令的QPS   | 该指标为节点执行hexists命令的QPS。   | $\geq 0$<br>Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis134_hexists_p99      | hexists命令的p99时延 | 该指标为节点执行hexists命令的p99时延。 | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis132_hexists_avg_usec | hexists命令的平均时延  | 该指标为节点执行hexists命令的平均时延。  | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis131_hgetall_qps      | hgetall命令的QPS   | 该指标为节点执行hgetall命令的QPS。   | $\geq 0$<br>Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis130_hgetall_p99      | hgetall命令的p99时延 | 该指标为节点执行hgetall命令的p99时延。 | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis128_hgetall_avg_usec | hgetall命令的平均时延  | 该指标为节点执行hgetall命令的平均时延。  | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |

| 指标ID                    | 指标名称          | 指标含义                   | 取值范围                 | 单位       | 进制  | 维度                             | 监控周期<br>(原始指标) |
|-------------------------|---------------|------------------------|----------------------|----------|-----|--------------------------------|----------------|
| redis127_hdel_qps       | hdel命令的QPS    | 该指标为节点执行hdel命令的QPS。    | $\geq 0$<br>Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis126_hdel_p99       | hdel命令的p99时延  | 该指标为节点执行hdel命令的p99时延。  | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis124_hdel_avg_usec  | hdel命令的平均时延   | 该指标为节点执行hdel命令的平均时延。   | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis123_hmget_qps      | hmget命令的QPS   | 该指标为节点执行hmget命令的QPS。   | $\geq 0$<br>Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis122_hmget_p99      | hmget命令的p99时延 | 该指标为节点执行hmget命令的p99时延。 | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis120_hmget_avg_usec | hmget命令的平均时延  | 该指标为节点执行hmget命令的平均时延。  | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis119_hmset_qps      | hmset命令的QPS   | 该指标为节点执行hmset命令的QPS。   | $\geq 0$<br>Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis118_hmset_p99      | hmset命令的p99时延 | 该指标为节点执行hmset命令的p99时延。 | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis116_hmset_avg_usec | hmset命令的平均时延  | 该指标为节点执行hmset命令的平均时延。  | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis115_hget_qps       | hget命令的QPS    | 该指标为节点执行hget命令的QPS。    | $\geq 0$<br>Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis114_hget_p99       | hget命令的p99时延  | 该指标为节点执行hget命令的p99时延。  | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |

| 指标ID                   | 指标名称         | 指标含义                  | 取值范围         | 单位       | 进制  | 维度                             | 监控周期（原始指标） |
|------------------------|--------------|-----------------------|--------------|----------|-----|--------------------------------|------------|
| redis112_hget_avg_usec | hget命令的平均时延  | 该指标为节点执行hget命令的平均时延。  | ≥ 0 us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis111_hset_qps      | hset命令的QPS   | 该指标为节点执行hset命令的QPS。   | ≥ 0 Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis110_hset_p99      | hset命令的p99时延 | 该指标为节点执行hset命令的p99时延。 | ≥ 0 us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis108_hset_avg_usec | hset命令的平均时延  | 该指标为节点执行hset命令的平均时延。  | ≥ 0 us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |

List 命令类型指标

表 4-59 List 命令类型指标

| 指标ID                    | 指标名称          | 指标含义                   | 取值范围         | 单位       | 进制  | 维度                             | 监控周期（原始指标） |
|-------------------------|---------------|------------------------|--------------|----------|-----|--------------------------------|------------|
| redis207_ltrim_qps      | ltrim命令的QPS   | 该指标为节点执行ltrim命令的QPS。   | ≥ 0 Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis206_ltrim_p99      | ltrim命令的p99时延 | 该指标为节点执行ltrim命令的p99时延。 | ≥ 0 us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis204_ltrim_avg_usec | ltrim命令的平均时延  | 该指标为节点执行ltrim命令的平均时延。  | ≥ 0 us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis203_lset_qps       | lset命令的QPS    | 该指标为节点执行lset命令的QPS。    | ≥ 0 Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |

| 指标ID                     | 指标名称            | 指标含义                     | 取值范围               | 单位        | 进制  | 维度                             | 监控周期（原始指标） |
|--------------------------|-----------------|--------------------------|--------------------|-----------|-----|--------------------------------|------------|
| redis202_lset_p99        | lset命令的p99时延    | 该指标为节点执行lset命令的p99时延。    | $\geq 0$ us        | us        | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis200_lset_avg_usec   | lset命令的平均时延     | 该指标为节点执行lset命令的平均时延。     | $\geq 0$ us        | us        | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis199_lrem_qps        | lrem命令的QPS      | 该指标为节点执行lrem命令的QPS。      | $\geq 0$ Count/s/s | Count/s/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis198_lrem_p99        | lrem命令的p99时延    | 该指标为节点执行lrem命令的p99时延。    | $\geq 0$ us        | us        | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis196_lrem_avg_usec   | lrem命令的平均时延     | 该指标为节点执行lrem命令的平均时延。     | $\geq 0$ us        | us        | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis195_lrange_qps      | lrange命令的QPS    | 该指标为节点执行lrange命令的QPS。    | $\geq 0$ Count/s/s | Count/s/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis194_lrange_p99      | lrange命令的p99时延  | 该指标为节点执行lrange命令的p99时延。  | $\geq 0$ us        | us        | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis192_lrange_avg_usec | lrange命令的平均时延   | 该指标为节点执行lrange命令的平均时延。   | $\geq 0$ us        | us        | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis191_linsert_qps     | linsert命令的QPS   | 该指标为节点执行linsert命令的QPS。   | $\geq 0$ Count/s/s | Count/s/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis190_linsert_p99     | linsert命令的p99时延 | 该指标为节点执行linsert命令的p99时延。 | $\geq 0$ us        | us        | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |

| 指标ID                        | 指标名称              | 指标含义                       | 取值范围              | 单位       | 进制  | 维度                             | 监控周期（原始指标） |
|-----------------------------|-------------------|----------------------------|-------------------|----------|-----|--------------------------------|------------|
| redis188_linsert_avg_usec   | linsert命令的平均时延    | 该指标为节点执行linsert命令的平均时延。    | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis187_lindex_qps         | lindex命令的QPS      | 该指标为节点执行lindex命令的QPS。      | $\geq 0$ Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis186_lindex_p99         | lindex命令的p99时延    | 该指标为节点执行lindex命令的p99时延。    | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis184_lindex_avg_usec    | lindex命令的平均时延     | 该指标为节点执行lindex命令的平均时延。     | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis183_llen_qps           | llen命令的QPS        | 该指标为节点执行llen命令的QPS。        | $\geq 0$ Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis182_llen_p99           | llen命令的p99时延      | 该指标为节点执行llen命令的p99时延。      | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis180_llen_avg_usec      | llen命令的平均时延       | 该指标为节点执行llen命令的平均时延。       | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis179_rpoplpush_qps      | rpoplpush命令的QPS   | 该指标为节点执行rpoplpush命令的QPS。   | $\geq 0$ Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis178_rpoplpush_p99      | rpoplpush命令的p99时延 | 该指标为节点执行rpoplpush命令的p99时延。 | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis176_rpoplpush_avg_usec | rpoplpush命令的平均时延  | 该指标为节点执行rpoplpush命令的平均时延。  | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |

| 指标ID                    | 指标名称          | 指标含义                   | 取值范围               | 单位        | 进制  | 维度                             | 监控周期（原始指标） |
|-------------------------|---------------|------------------------|--------------------|-----------|-----|--------------------------------|------------|
| redis175_rpop_qps       | rpop命令的QPS    | 该指标为节点执行rpop命令的QPS。    | $\geq 0$ Count s/s | Count s/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis174_rpop_p99       | rpop命令的p99时延  | 该指标为节点执行rpop命令的p99时延。  | $\geq 0$ us        | us        | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis172_rpop_avg_usec  | rpop命令的平均时延   | 该指标为节点执行rpop命令的平均时延。   | $\geq 0$ us        | us        | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis171_rpush_qps      | rpush命令的QPS   | 该指标为节点执行rpush命令的QPS。   | $\geq 0$ Count s/s | Count s/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis170_rpush_p99      | rpush命令的p99时延 | 该指标为节点执行rpush命令的p99时延。 | $\geq 0$ us        | us        | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis168_rpush_avg_usec | rpush命令的平均时延  | 该指标为节点执行rpush命令的平均时延。  | $\geq 0$ us        | us        | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis167_lpop_qps       | lpop命令的QPS    | 该指标为节点执行lpop命令的QPS。    | $\geq 0$ Count s/s | Count s/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis166_lpop_p99       | lpop命令的p99时延  | 该指标为节点执行lpop命令的p99时延。  | $\geq 0$ us        | us        | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis164_lpop_avg_usec  | lpop命令的平均时延   | 该指标为节点执行lpop命令的平均时延。   | $\geq 0$ us        | us        | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis163_lpush_qps      | lpush命令的QPS   | 该指标为节点执行lpush命令的QPS。   | $\geq 0$ Count s/s | Count s/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |

| 指标ID                         | 指标名称               | 指标含义                        | 取值范围               | 单位        | 进制  | 维度                             | 监控周期（原始指标） |
|------------------------------|--------------------|-----------------------------|--------------------|-----------|-----|--------------------------------|------------|
| redis162_lpush_p99           | lpush命令的p99时延      | 该指标为节点执行lpush命令的p99时延。      | $\geq 0$ us        | us        | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis160_lpush_avg_usec      | lpush命令的平均时延       | 该指标为节点执行lpush命令的平均时延。       | $\geq 0$ us        | us        | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis470_brpoplpush_qps      | brpoplpush命令的QPS   | 该指标为节点执行brpoplpush命令的QPS。   | $\geq 0$ Count/s/s | Count/s/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis468_brpoplpush_p99      | brpoplpush命令的p99时延 | 该指标为节点执行brpoplpush命令的p99时延。 | $\geq 0$ us        | us        | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis466_brpoplpush_avg_usec | brpoplpush命令的平均时延  | 该指标为节点执行brpoplpush命令的平均时延。  | $\geq 0$ us        | us        | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis465_brpop_qps           | brpop命令的QPS        | 该指标为节点执行brpop命令的QPS。        | $\geq 0$ Count/s/s | Count/s/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis463_brpop_p99           | brpop命令的p99时延      | 该指标为节点执行brpop命令的p99时延。      | $\geq 0$ us        | us        | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis461_brpop_avg_usec      | brpop命令的平均时延       | 该指标为节点执行brpop命令的平均时延。       | $\geq 0$ us        | us        | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis460_blpop_qps           | blpop命令的QPS        | 该指标为节点执行blpop命令的QPS。        | $\geq 0$ Count/s/s | Count/s/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis458_blpop_p99           | blpop命令的p99时延      | 该指标为节点执行blpop命令的p99时延。      | $\geq 0$ us        | us        | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |



| 指标ID                    | 指标名称         | 指标含义                  | 取值范围   | 单位 | 进制  | 维度                             | 监控周期（原始指标） |
|-------------------------|--------------|-----------------------|--------|----|-----|--------------------------------|------------|
| redis456_blpop_avg_usec | blpop命令的平均时延 | 该指标为节点执行blpop命令的平均时延。 | ≥ 0 us | us | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |

Set 命令类型指标

表 4-60 Set 命令类型指标

| 指标ID                          | 指标名称                | 指标含义                         | 取值范围         | 单位       | 进制  | 维度                             | 监控周期（原始指标） |
|-------------------------------|---------------------|------------------------------|--------------|----------|-----|--------------------------------|------------|
| redis247_srandmember_qps      | srandmember命令的QPS   | 该指标为节点执行srandmember命令的QPS。   | ≥ 0 Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis246_srandmember_p99      | srandmember命令的p99时延 | 该指标为节点执行srandmember命令的p99时延。 | ≥ 0 us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis244_srandmember_avg_usec | srandmember命令的平均时延  | 该指标为节点执行srandmember命令的平均时延。  | ≥ 0 us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis243_sdiff_qps            | sdiff命令的QPS         | 该指标为节点执行sdiff命令的QPS。         | ≥ 0 Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis242_sdiff_p99            | sdiff命令的p99时延       | 该指标为节点执行sdiff命令的p99时延。       | ≥ 0 us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis240_sdiff_avg_usec       | sdiff命令的平均时延        | 该指标为节点执行sdiff命令的平均时延。        | ≥ 0 us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |

| 指标ID                        | 指标名称              | 指标含义                       | 取值范围         | 单位       | 进制  | 维度                             | 监控周期<br>(原始指标) |
|-----------------------------|-------------------|----------------------------|--------------|----------|-----|--------------------------------|----------------|
| redis239_sismember_qps      | sismember命令的QPS   | 该指标为节点执行sismember命令的QPS。   | ≥ 0 Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis238_sismember_p99      | sismember命令的p99时延 | 该指标为节点执行sismember命令的p99时延。 | ≥ 0 us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis236_sismember_avg_usec | sismember命令的平均时延  | 该指标为节点执行sismember命令的平均时延。  | ≥ 0 us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis235_sinter_qps         | sinter命令的QPS      | 该指标为节点执行sinter命令的QPS。      | ≥ 0 Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis234_sinter_p99         | sinter命令的p99时延    | 该指标为节点执行sinter命令的p99时延。    | ≥ 0 us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis232_sinter_avg_usec    | sinter命令的平均时延     | 该指标为节点执行sinter命令的平均时延。     | ≥ 0 us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis231_sunion_qps         | sunion命令的QPS      | 该指标为节点执行sunion命令的QPS。      | ≥ 0 Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis230_sunion_p99         | sunion命令的p99时延    | 该指标为节点执行sunion命令的p99时延。    | ≥ 0 us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis228_sunion_avg_usec    | sunion命令的平均时延     | 该指标为节点执行sunion命令的平均时延。     | ≥ 0 us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis227_srem_qps           | srem命令的QPS        | 该指标为节点执行srem命令的QPS。        | ≥ 0 Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |

| 指标ID                       | 指标名称             | 指标含义                      | 取值范围                 | 单位       | 进制  | 维度                             | 监控周期<br>(原始指标) |
|----------------------------|------------------|---------------------------|----------------------|----------|-----|--------------------------------|----------------|
| redis226_srem_p99          | srem命令的p99时延     | 该指标为节点执行srem命令的p99时延。     | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis224_srem_avg_usec     | srem命令的平均时延      | 该指标为节点执行srem命令的平均时延。      | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis223_smembers_qps      | smembers命令的QPS   | 该指标为节点执行smembers命令的QPS。   | $\geq 0$<br>Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis222_smembers_p99      | smembers命令的p99时延 | 该指标为节点执行smembers命令的p99时延。 | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis220_smembers_avg_usec | smembers命令的平均时延  | 该指标为节点执行smembers命令的平均时延。  | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis219_scard_qps         | scard命令的QPS      | 该指标为节点执行scard命令的QPS。      | $\geq 0$<br>Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis218_scard_p99         | scard命令的p99时延    | 该指标为节点执行scard命令的p99时延。    | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis216_scard_avg_usec    | scard命令的平均时延     | 该指标为节点执行scard命令的平均时延。     | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis215_spop_qps          | spop命令的QPS       | 该指标为节点执行spop命令的QPS。       | $\geq 0$<br>Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis214_spop_p99          | spop命令的p99时延     | 该指标为节点执行spop命令的p99时延。     | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |

| 指标ID                          | 指标名称                | 指标含义                         | 取值范围                 | 单位       | 进制  | 维度                             | 监控周期<br>(原始指标) |
|-------------------------------|---------------------|------------------------------|----------------------|----------|-----|--------------------------------|----------------|
| redis212_spop_avg_usec        | spop命令的平均时延         | 该指标为节点执行spop命令的平均时延。         | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis211_sadd_qps             | sadd命令的QPS          | 该指标为节点执行sadd命令的QPS。          | $\geq 0$<br>Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis210_sadd_p99             | sadd命令的p99时延        | 该指标为节点执行sadd命令的p99时延。        | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis208_sadd_avg_usec        | sadd命令的平均时延         | 该指标为节点执行sadd命令的平均时延。         | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis485_sdiffstore_qps       | sdiffstore命令的QPS    | 该指标为节点执行sdiffstore命令的QPS。    | $\geq 0$<br>Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis483_sdiffstore_p99       | sdiffstore命令的p99时延  | 该指标为节点执行sdiffstore命令的p99时延。  | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis481_sdiffstore_avg_usec  | sdiffstore命令的平均时延   | 该指标为节点执行sdiffstore命令的平均时延。   | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis480_sinterstore_qps      | sinterstore命令的QPS   | 该指标为节点执行sinterstore命令的QPS。   | $\geq 0$<br>Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis478_sinterstore_p99      | sinterstore命令的p99时延 | 该指标为节点执行sinterstore命令的p99时延。 | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis476_sinterstore_avg_usec | sinterstore命令的平均时延  | 该指标为节点执行sinterstore命令的平均时延。  | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |

| 指标ID                          | 指标名称                | 指标含义                         | 取值范围         | 单位       | 进制  | 维度                             | 监控周期（原始指标） |
|-------------------------------|---------------------|------------------------------|--------------|----------|-----|--------------------------------|------------|
| redis475_sunionstore_qps      | sunionstore命令的QPS   | 该指标为节点执行sunionstore命令的QPS。   | ≥ 0 Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis473_sunionstore_p99      | sunionstore命令的p99时延 | 该指标为节点执行sunionstore命令的p99时延。 | ≥ 0 us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis471_sunionstore_avg_usec | sunionstore命令的平均时延  | 该指标为节点执行sunionstore命令的平均时延。  | ≥ 0 us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |

Zset 命令类型指标

表 4-61 Zset 命令类型指标

| 指标ID                             | 指标名称                   | 指标含义                            | 取值范围         | 单位       | 进制  | 维度                             | 监控周期（原始指标） |
|----------------------------------|------------------------|---------------------------------|--------------|----------|-----|--------------------------------|------------|
| redis315_zremrangebylex_qps      | zremrangebylex命令的QPS   | 该指标为节点执行zremrangebylex命令的QPS。   | ≥ 0 Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis314_zremrangebylex_p99      | zremrangebylex命令的p99时延 | 该指标为节点执行zremrangebylex命令的p99时延。 | ≥ 0 us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis312_zremrangebylex_avg_usec | zremrangebylex命令的平均时延  | 该指标为节点执行zremrangebylex命令的平均时延。  | ≥ 0 us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |

| 指标ID                               | 指标名称                     | 指标含义                              | 取值范围                 | 单位       | 进制  | 维度                             | 监控周期<br>(原始指标) |
|------------------------------------|--------------------------|-----------------------------------|----------------------|----------|-----|--------------------------------|----------------|
| redis311_zremrangebyscore_qps      | zremrangebyscore命令的QPS   | 该指标为节点执行zremrangebyscore命令的QPS。   | $\geq 0$<br>Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis310_zremrangebyscore_p99      | zremrangebyscore命令的p99时延 | 该指标为节点执行zremrangebyscore命令的p99时延。 | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis308_zremrangebyscore_avg_usec | zremrangebyscore命令的平均时延  | 该指标为节点执行zremrangebyscore命令的平均时延。  | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis307_zremrangebyrank_qps       | zremrangebyrank命令的QPS    | 该指标为节点执行zremrangebyrank命令的QPS。    | $\geq 0$<br>Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis306_zremrangebyrank_p99       | zremrangebyrank命令的p99时延  | 该指标为节点执行zremrangebyrank命令的p99时延。  | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis304_zremrangebyrank_avg_usec  | zremrangebyrank命令的平均时延   | 该指标为节点执行zremrangebyrank命令的平均时延。   | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis303_zpopmin_qps               | zpopmin命令的QPS            | 该指标为节点执行zpopmin命令的QPS。            | $\geq 0$<br>Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis302_zpopmin_p99               | zpopmin命令的p99时延          | 该指标为节点执行zpopmin命令的p99时延。          | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis300_zpopmin_avg_usec          | zpopmin命令的平均时延           | 该指标为节点执行zpopmin命令的平均时延。           | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |

| 指标ID                        | 指标名称                 | 指标含义                           | 取值范围                 | 单位       | 进制  | 维度                             | 监控周期<br>(原始指标) |
|-----------------------------|----------------------|--------------------------------|----------------------|----------|-----|--------------------------------|----------------|
| redis299_zpopmax_qps        | zpopmax 命令的 QPS      | 该指标为节点执行 zpopmax 命令的 QPS。      | $\geq 0$<br>Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis298_zpopmax_p99        | zpopmax 命令的 p99 时延   | 该指标为节点执行 zpopmax 命令的 p99 时延。   | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis296_zpopmax_avg_usec   | zpopmax 命令的平均时延      | 该指标为节点执行 zpopmax 命令的平均时延。      | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis295_zlexcount_qps      | zlexcount 命令的 QPS    | 该指标为节点执行 zlexcount 命令的 QPS。    | $\geq 0$<br>Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis294_zlexcount_p99      | zlexcount 命令的 p99 时延 | 该指标为节点执行 zlexcount 命令的 p99 时延。 | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis292_zlexcount_avg_usec | zlexcount 命令的平均时延    | 该指标为节点执行 zlexcount 命令的平均时延。    | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis291_zrevrank_qps       | zrevrank 命令的 QPS     | 该指标为节点执行 zrevrank 命令的 QPS。     | $\geq 0$<br>Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis290_zrevrank_p99       | zrevrank 命令的 p99 时延  | 该指标为节点执行 zrevrank 命令的 p99 时延。  | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis288_zrevrank_avg_usec  | zrevrank 命令的平均时延     | 该指标为节点执行 zrevrank 命令的平均时延。     | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |

| 指标ID                     | 指标名称           | 指标含义                    | 取值范围                 | 单位       | 进制  | 维度                             | 监控周期<br>(原始指标) |
|--------------------------|----------------|-------------------------|----------------------|----------|-----|--------------------------------|----------------|
| redis287_zrank_qps       | zrank命令的QPS    | 该指标为节点执行zrank命令的QPS。    | $\geq 0$<br>Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis286_zrank_p99       | zrank命令的p99时延  | 该指标为节点执行zrank命令的p99时延。  | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis284_zrank_avg_usec  | zrank命令的平均时延   | 该指标为节点执行zrank命令的平均时延。   | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis283_zscore_qps      | zscore命令的QPS   | 该指标为节点执行zscore命令的QPS。   | $\geq 0$<br>Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis282_zscore_p99      | zscore命令的p99时延 | 该指标为节点执行zscore命令的p99时延。 | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis280_zscore_avg_usec | zscore命令的平均时延  | 该指标为节点执行zscore命令的平均时延。  | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis279_zrem_qps        | zrem命令的QPS     | 该指标为节点执行zrem命令的QPS。     | $\geq 0$<br>Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis278_zrem_p99        | zrem命令的p99时延   | 该指标为节点执行zrem命令的p99时延。   | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis276_zrem_avg_usec   | zrem命令的平均时延    | 该指标为节点执行zrem命令的平均时延。    | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis275_zcount_qps      | zcount命令的QPS   | 该指标为节点执行zcount命令的QPS。   | $\geq 0$<br>Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |



| 指标ID                        | 指标名称              | 指标含义                       | 取值范围                 | 单位       | 进制  | 维度                             | 监控周期<br>(原始指标) |
|-----------------------------|-------------------|----------------------------|----------------------|----------|-----|--------------------------------|----------------|
| redis274_zcount_p99         | zcount命令的p99时延    | 该指标为节点执行zcount命令的p99时延。    | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis272_zcount_avg_usec    | zcount命令的平均时延     | 该指标为节点执行zcount命令的平均时延。     | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis271_zrange_qps         | zrange命令的QPS      | 该指标为节点执行zrange命令的QPS。      | $\geq 0$<br>Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis270_zrange_p99         | zrange命令的p99时延    | 该指标为节点执行zrange命令的p99时延。    | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis268_zrange_avg_usec    | zrange命令的平均时延     | 该指标为节点执行zrange命令的平均时延。     | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis267_zrevrange_qps      | zrevrange命令的QPS   | 该指标为节点执行zrevrange命令的QPS。   | $\geq 0$<br>Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis266_zrevrange_p99      | zrevrange命令的p99时延 | 该指标为节点执行zrevrange命令的p99时延。 | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis264_zrevrange_avg_usec | zrevrange命令的平均时延  | 该指标为节点执行zrevrange命令的平均时延。  | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis263_zincrby_qps        | zincrby命令的QPS     | 该指标为节点执行zincrby命令的QPS。     | $\geq 0$<br>Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis262_zincrby_p99        | zincrby命令的p99时延   | 该指标为节点执行zincrby命令的p99时延。   | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |

| 指标ID                      | 指标名称           | 指标含义                    | 取值范围                 | 单位       | 进制  | 维度                             | 监控周期<br>(原始指标) |
|---------------------------|----------------|-------------------------|----------------------|----------|-----|--------------------------------|----------------|
| redis260_zincrby_avg_usec | zincrby命令的平均时延 | 该指标为节点执行zincrby命令的平均时延。 | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis259_zscan_qps        | zscan命令的QPS    | 该指标为节点执行zscan命令的QPS。    | $\geq 0$<br>Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis258_zscan_p99        | zscan命令的p99时延  | 该指标为节点执行zscan命令的p99时延。  | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis256_zscan_avg_usec   | zscan命令的平均时延   | 该指标为节点执行zscan命令的平均时延。   | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis255_zcard_qps        | zcard命令的QPS    | 该指标为节点执行zcard命令的QPS。    | $\geq 0$<br>Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis254_zcard_p99        | zcard命令的p99时延  | 该指标为节点执行zcard命令的p99时延。  | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis252_zcard_avg_usec   | zcard命令的平均时延   | 该指标为节点执行zcard命令的平均时延。   | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis251_zadd_qps         | zadd命令的QPS     | 该指标为节点执行zadd命令的QPS。     | $\geq 0$<br>Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis250_zadd_p99         | zadd命令的p99时延   | 该指标为节点执行zadd命令的p99时延。   | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis248_zadd_avg_usec    | zadd命令的平均时延    | 该指标为节点执行zadd命令的平均时延。    | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |

| 指标ID                          | 指标名称                | 指标含义                         | 取值范围         | 单位       | 进制  | 维度                             | 监控周期（原始指标） |
|-------------------------------|---------------------|------------------------------|--------------|----------|-----|--------------------------------|------------|
| redis495_zinterstore_qps      | zinterstore命令的QPS   | 该指标为节点执行zinterstore命令的QPS。   | ≥ 0 Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis493_zinterstore_p99      | zinterstore命令的p99时延 | 该指标为节点执行zinterstore命令的p99时延。 | ≥ 0 us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis491_zinterstore_avg_usec | zinterstore命令的平均时延  | 该指标为节点执行zinterstore命令的平均时延。  | ≥ 0 us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis490_zunionstore_qps      | zunionstore命令的QPS   | 该指标为节点执行zunionstore命令的QPS。   | ≥ 0 Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis488_zunionstore_p99      | zunionstore命令的p99时延 | 该指标为节点执行zunionstore命令的p99时延。 | ≥ 0 us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis486_zunionstore_avg_usec | zunionstore命令的平均时延  | 该指标为节点执行zunionstore命令的平均时延。  | ≥ 0 us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |

Bitmap 命令类型指标

表 4-62 Bitmap 命令类型指标

| 指标ID                       | 指标名称             | 指标含义                      | 取值范围              | 单位       | 进制  | 维度                             | 监控周期<br>(原始指标) |
|----------------------------|------------------|---------------------------|-------------------|----------|-----|--------------------------------|----------------|
| redis440_bitfield_qps      | bitfield命令的QPS   | 该指标为节点执行bitfield命令的QPS。   | $\geq 0$ Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis438_bitfield_p99      | bitfield命令的p99时延 | 该指标为节点执行bitfield命令的p99时延。 | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis436_bitfield_avg_usec | bitfield命令的平均时延  | 该指标为节点执行bitfield命令的平均时延。  | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis435_bitop_qps         | bitop命令的QPS      | 该指标为节点执行bitop命令的QPS。      | $\geq 0$ Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis433_bitop_p99         | bitop命令的p99时延    | 该指标为节点执行bitop命令的p99时延。    | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis431_bitop_avg_usec    | bitop命令的平均时延     | 该指标为节点执行bitop命令的平均时延。     | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis430_bitpos_qps        | bitpos命令的QPS     | 该指标为节点执行bitpos命令的QPS。     | $\geq 0$ Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis428_bitpos_p99        | bitpos命令的p99时延   | 该指标为节点执行bitpos命令的p99时延。   | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis426_bitpos_avg_usec   | bitpos命令的平均时延    | 该指标为节点执行bitpos命令的平均时延。    | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |

| 指标ID                       | 指标名称             | 指标含义                      | 取值范围              | 单位       | 进制  | 维度                             | 监控周期<br>(原始指标) |
|----------------------------|------------------|---------------------------|-------------------|----------|-----|--------------------------------|----------------|
| redis425_bitcount_qps      | bitcount命令的QPS   | 该指标为节点执行bitcount命令的QPS。   | $\geq 0$ Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis423_bitcount_p99      | bitcount命令的p99时延 | 该指标为节点执行bitcount命令的p99时延。 | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis421_bitcount_avg_usec | bitcount命令的平均时延  | 该指标为节点执行bitcount命令的平均时延。  | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis420_getbit_qps        | getbit命令的QPS     | 该指标为节点执行getbit命令的QPS。     | $\geq 0$ Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis418_getbit_p99        | getbit命令的p99时延   | 该指标为节点执行getbit命令的p99时延。   | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis416_getbit_avg_usec   | getbit命令的平均时延    | 该指标为节点执行getbit命令的平均时延。    | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis415_setbit_qps        | setbit命令的QPS     | 该指标为节点执行setbit命令的QPS。     | $\geq 0$ Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis413_setbit_p99        | setbit命令的p99时延   | 该指标为节点执行setbit命令的p99时延。   | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis411_setbit_avg_usec   | setbit命令的平均时延    | 该指标为节点执行setbit命令的平均时延。    | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |

Stream 命令类型指标

表 4-63 Stream 命令类型指标

| 指标ID                         | 指标名称               | 指标含义                        | 取值范围         | 单位       | 进制  | 维度                             | 监控周期（原始指标） |
|------------------------------|--------------------|-----------------------------|--------------|----------|-----|--------------------------------|------------|
| redis590_xreadgroup_qps      | xreadgroup命令的QPS   | 该指标为节点执行xreadgroup命令的QPS。   | ≥ 0 Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis588_xreadgroup_p99      | xreadgroup命令的p99时延 | 该指标为节点执行xreadgroup命令的p99时延。 | ≥ 0 us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis586_xreadgroup_avg_usec | xreadgroup命令的平均时延  | 该指标为节点执行xreadgroup命令的平均时延。  | ≥ 0 us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis585_xread_qps           | xread命令的QPS        | 该指标为节点执行xread命令的QPS。        | ≥ 0 Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis583_xread_p99           | xread命令的p99时延      | 该指标为节点执行xread命令的p99时延。      | ≥ 0 us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis581_xread_avg_usec      | xread命令的平均时延       | 该指标为节点执行xread命令的平均时延。       | ≥ 0 us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis580_xinfo_qps           | xinfo命令的QPS        | 该指标为节点执行xinfo命令的QPS。        | ≥ 0 Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis578_xinfo_p99           | xinfo命令的p99时延      | 该指标为节点执行xinfo命令的p99时延。      | ≥ 0 us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis576_xinfo_avg_usec      | xinfo命令的平均时延       | 该指标为节点执行xinfo命令的平均时延。       | ≥ 0 us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |

| 指标ID                        | 指标名称              | 指标含义                       | 取值范围               | 单位       | 进制  | 维度                             | 监控周期<br>(原始指标) |
|-----------------------------|-------------------|----------------------------|--------------------|----------|-----|--------------------------------|----------------|
| redis575_xpending_qps       | xpending命令的QPS    | 该指标为节点执行xpending命令的QPS。    | $\geq 0$ Count/s/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis573_xpending_p99       | xpending命令的p99时延  | 该指标为节点执行xpending命令的p99时延。  | $\geq 0$ us        | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis571_xpending_avg_usec  | xpending命令的平均时延   | 该指标为节点执行xpending命令的平均时延。   | $\geq 0$ us        | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis570_xclaim_qps         | xclaim命令的QPS      | 该指标为节点执行xclaim命令的QPS。      | $\geq 0$ Count/s/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis568_xclaim_p99         | xclaim命令的p99时延    | 该指标为节点执行xclaim命令的p99时延。    | $\geq 0$ us        | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis566_xclaim_avg_usec    | xclaim命令的平均时延     | 该指标为节点执行xclaim命令的平均时延。     | $\geq 0$ us        | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis565_xrevrange_qps      | xrevrange命令的QPS   | 该指标为节点执行xrevrange命令的QPS。   | $\geq 0$ Count/s/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis563_xrevrange_p99      | xrevrange命令的p99时延 | 该指标为节点执行xrevrange命令的p99时延。 | $\geq 0$ us        | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis561_xrevrange_avg_usec | xrevrange命令的平均时延  | 该指标为节点执行xrevrange命令的平均时延。  | $\geq 0$ us        | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis560_xrange_qps         | xrange命令的QPS      | 该指标为节点执行xrange命令的QPS。      | $\geq 0$ Count/s/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |

| 指标ID                     | 指标名称           | 指标含义                    | 取值范围              | 单位       | 进制  | 维度                             | 监控周期<br>(原始指标) |
|--------------------------|----------------|-------------------------|-------------------|----------|-----|--------------------------------|----------------|
| redis558_xrange_p99      | xrange命令的p99时延 | 该指标为节点执行xrange命令的p99时延。 | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis556_xrange_avg_usec | xrange命令的平均时延  | 该指标为节点执行xrange命令的平均时延。  | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis555_xlen_qps        | xlen命令的QPS     | 该指标为节点执行xlen命令的QPS。     | $\geq 0$ Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis553_xlen_p99        | xlen命令的p99时延   | 该指标为节点执行xlen命令的p99时延。   | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis551_xlen_avg_usec   | xlen命令的平均时延    | 该指标为节点执行xlen命令的平均时延。    | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis550_xtrim_qps       | xtrim命令的QPS    | 该指标为节点执行xtrim命令的QPS。    | $\geq 0$ Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis548_xtrim_p99       | xtrim命令的p99时延  | 该指标为节点执行xtrim命令的p99时延。  | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis546_xtrim_avg_usec  | xtrim命令的平均时延   | 该指标为节点执行xtrim命令的平均时延。   | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis545_xdel_qps        | xdel命令的QPS     | 该指标为节点执行xdel命令的QPS。     | $\geq 0$ Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis543_xdel_p99        | xdel命令的p99时延   | 该指标为节点执行xdel命令的p99时延。   | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |



| 指标ID                     | 指标名称           | 指标含义                    | 取值范围              | 单位       | 进制  | 维度                             | 监控周期<br>(原始指标) |
|--------------------------|----------------|-------------------------|-------------------|----------|-----|--------------------------------|----------------|
| redis541_xdel_avg_usec   | xdel命令的平均时延    | 该指标为节点执行xdel命令的平均时延。    | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis540_xgroup_qps      | xgroup命令的QPS   | 该指标为节点执行xgroup命令的QPS。   | $\geq 0$ Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis538_xgroup_p99      | xgroup命令的p99时延 | 该指标为节点执行xgroup命令的p99时延。 | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis536_xgroup_avg_usec | xgroup命令的平均时延  | 该指标为节点执行xgroup命令的平均时延。  | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis535_xack_qps        | xack命令的QPS     | 该指标为节点执行xack命令的QPS。     | $\geq 0$ Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis533_xack_p99        | xack命令的p99时延   | 该指标为节点执行xack命令的p99时延。   | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis531_xack_avg_usec   | xack命令的平均时延    | 该指标为节点执行xack命令的平均时延。    | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis530_xadd_qps        | xadd命令的QPS     | 该指标为节点执行xadd命令的QPS。     | $\geq 0$ Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis528_xadd_p99        | xadd命令的p99时延   | 该指标为节点执行xadd命令的p99时延。   | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |
| redis526_xadd_avg_usec   | xadd命令的平均时延    | 该指标为节点执行xadd命令的平均时延。    | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟            |

Geo 命令类型指标

表 4-64 Geo 命令类型指标

| 指标ID                      | 指标名称            | 指标含义                     | 取值范围                 | 单位       | 进制  | 维度                             | 监控周期（原始指标） |
|---------------------------|-----------------|--------------------------|----------------------|----------|-----|--------------------------------|------------|
| redis525_geopos_qps       | geopos命令的QPS    | 该指标为节点执行geopos命令的QPS。    | $\geq 0$<br>Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis523_geopos_p99       | geopos命令的p99时延  | 该指标为节点执行geopos命令的p99时延。  | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis521_geopos_avg_usec  | geopos命令的平均时延   | 该指标为节点执行geopos命令的平均时延。   | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis520_geodist_qps      | geodist命令的QPS   | 该指标为节点执行geodist命令的QPS。   | $\geq 0$<br>Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis518_geodist_p99      | geodist命令的p99时延 | 该指标为节点执行geodist命令的p99时延。 | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis516_geodist_avg_usec | geodist命令的平均时延  | 该指标为节点执行geodist命令的平均时延。  | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis515_geohash_qps      | geohash命令的QPS   | 该指标为节点执行geohash命令的QPS。   | $\geq 0$<br>Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |

| 指标ID                                | 指标名称                      | 指标含义                               | 取值范围              | 单位       | 进制  | 维度                             | 监控周期（原始指标） |
|-------------------------------------|---------------------------|------------------------------------|-------------------|----------|-----|--------------------------------|------------|
| redis513_geohash_p99                | geohash命令的p99时延           | 该指标为节点执行geohash命令的p99时延。           | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis511_geohash_avg_usec           | geohash命令的平均时延            | 该指标为节点执行geohash命令的平均时延。            | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis510_georadius_qps              | georadius命令的QPS           | 该指标为节点执行georadius命令的QPS。           | $\geq 0$ Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis508_georadius_p99              | georadius命令的p99时延         | 该指标为节点执行georadius命令的p99时延。         | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis506_georadius_avg_usec         | georadius命令的平均时延          | 该指标为节点执行georadius命令的平均时延。          | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis505_georadiusbymember_qps      | georadiusbymember命令的QPS   | 该指标为节点执行georadiusbymember命令的QPS。   | $\geq 0$ Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis503_georadiusbymember_p99      | georadiusbymember命令的p99时延 | 该指标为节点执行georadiusbymember命令的p99时延。 | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis501_georadiusbymember_avg_usec | georadiusbymember命令的平均时延  | 该指标为节点执行georadiusbymember命令的平均时延。  | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |

| 指标ID                     | 指标名称           | 指标含义                    | 取值范围                 | 单位       | 进制  | 维度                             | 监控周期（原始指标） |
|--------------------------|----------------|-------------------------|----------------------|----------|-----|--------------------------------|------------|
| redis500_geoadd_qps      | geoadd命令的QPS   | 该指标为节点执行geoadd命令的QPS。   | $\geq 0$<br>Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis498_geoadd_p99      | geoadd命令的p99时延 | 该指标为节点执行geoadd命令的p99时延。 | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis496_geoadd_avg_usec | geoadd命令的平均时延  | 该指标为节点执行geoadd命令的平均时延。  | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |

Hyperloglog 命令类型指标

表 4-65 Hyperloglog 命令类型指标

| 指标ID                      | 指标名称            | 指标含义                     | 取值范围                 | 单位       | 进制  | 维度                             | 监控周期（原始指标） |
|---------------------------|-----------------|--------------------------|----------------------|----------|-----|--------------------------------|------------|
| redis455_pfmerge_qps      | pfmerge命令的QPS   | 该指标为节点执行pfmerge命令的QPS。   | $\geq 0$<br>Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis453_pfmerge_p99      | pfmerge命令的p99时延 | 该指标为节点执行pfmerge命令的p99时延。 | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis451_pfmerge_avg_usec | pfmerge命令的平均时延  | 该指标为节点执行pfmerge命令的平均时延。  | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |

| 指标ID                     | 指标名称            | 指标含义                     | 取值范围                 | 单位       | 进制  | 维度                             | 监控周期（原始指标） |
|--------------------------|-----------------|--------------------------|----------------------|----------|-----|--------------------------------|------------|
| redis450_pfcoun_qps      | pfcount命令的QPS   | 该指标为节点执行pfcount命令的QPS。   | $\geq 0$<br>Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis448_pfcoun_p99      | pfcount命令的p99时延 | 该指标为节点执行pfcount命令的p99时延。 | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis446_pfcoun_avg_usec | pfcount命令的平均时延  | 该指标为节点执行pfcount命令的平均时延。  | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis445_pfadd_qps       | pfadd命令的QPS     | 该指标为节点执行pfadd命令的QPS。     | $\geq 0$<br>Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis443_pfadd_p99       | pfadd命令的p99时延   | 该指标为节点执行pfadd命令的p99时延。   | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis441_pfadd_avg_usec  | pfadd命令的平均时延    | 该指标为节点执行pfadd命令的平均时延。    | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |

Pub/Sub 命令类型指标

表 4-66 Pub/Sub 命令类型指标

| 指标ID                | 指标名称         | 指标含义                  | 取值范围                 | 单位       | 进制  | 维度                             | 监控周期（原始指标） |
|---------------------|--------------|-----------------------|----------------------|----------|-----|--------------------------------|------------|
| redis620_pubsub_qps | pubsub命令的QPS | 该指标为节点执行pubsub命令的QPS。 | $\geq 0$<br>Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |

| 指标ID                           | 指标名称                 | 指标含义                          | 取值范围              | 单位       | 进制  | 维度                             | 监控周期（原始指标） |
|--------------------------------|----------------------|-------------------------------|-------------------|----------|-----|--------------------------------|------------|
| redis618_pubsub_p99            | pubsub命令的p99时延       | 该指标为节点执行pubsub命令的p99时延。       | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis616_pubsub_avg_usec       | pubsub命令的平均时延        | 该指标为节点执行pubsub命令的平均时延。        | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis615_punsubscribe_qps      | punsubscribe命令的QPS   | 该指标为节点执行punsubscribe命令的QPS。   | $\geq 0$ Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis613_punsubscribe_p99      | punsubscribe命令的p99时延 | 该指标为节点执行punsubscribe命令的p99时延。 | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis611_punsubscribe_avg_usec | punsubscribe命令的平均时延  | 该指标为节点执行punsubscribe命令的平均时延。  | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis610_psubscribe_qps        | psubscribe命令的QPS     | 该指标为节点执行psubscribe命令的QPS。     | $\geq 0$ Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis608_psubscribe_p99        | psubscribe命令的p99时延   | 该指标为节点执行psubscribe命令的p99时延。   | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis606_psubscribe_avg_usec   | psubscribe命令的平均时延    | 该指标为节点执行psubscribe命令的平均时延。    | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis605_unsubscribe_qps       | unsubscribe命令的QPS    | 该指标为节点执行unsubscribe命令的QPS。    | $\geq 0$ Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis603_unsubscribe_p99       | unsubscribe命令的p99时延  | 该指标为节点执行unsubscribe命令的p99时延。  | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |

| 指标ID                          | 指标名称               | 指标含义                        | 取值范围              | 单位       | 进制  | 维度                             | 监控周期（原始指标） |
|-------------------------------|--------------------|-----------------------------|-------------------|----------|-----|--------------------------------|------------|
| redis601_unsubscribe_avg_usec | unsubscribe命令的平均时延 | 该指标为节点执行unsubscribe命令的平均时延。 | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis600_subscribe_qps        | subscribe命令的QPS    | 该指标为节点执行subscribe命令的QPS。    | $\geq 0$ Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis598_subscribe_p99        | subscribe命令的p99时延  | 该指标为节点执行subscribe命令的p99时延。  | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis596_subscribe_avg_usec   | subscribe命令的平均时延   | 该指标为节点执行subscribe命令的平均时延。   | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis595_publish_qps          | publish命令的QPS      | 该指标为节点执行publish命令的QPS。      | $\geq 0$ Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis593_publish_p99          | publish命令的p99时延    | 该指标为节点执行publish命令的p99时延。    | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis591_publish_avg_usec     | publish命令的平均时延     | 该指标为节点执行publish命令的平均时延。     | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |

Scripting 命令类型指标

表 4-67 Scripting 命令类型指标

| 指标ID                      | 指标名称            | 指标含义                     | 取值范围            | 单位       | 进制  | 维度                             | 监控周期（原始指标） |
|---------------------------|-----------------|--------------------------|-----------------|----------|-----|--------------------------------|------------|
| redis635_script_qps       | script命令的QPS    | 该指标为节点执行script命令的QPS。    | ≥ 0<br>Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis633_script_p99       | script命令的p99时延  | 该指标为节点执行script命令的p99时延。  | ≥ 0<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis631_script_avg_usec  | script命令的平均时延   | 该指标为节点执行script命令的平均时延。   | ≥ 0<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis630_evalsha_qps      | evalsha命令的QPS   | 该指标为节点执行evalsha命令的QPS。   | ≥ 0<br>Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis628_evalsha_p99      | evalsha命令的p99时延 | 该指标为节点执行evalsha命令的p99时延。 | ≥ 0<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis626_evalsha_avg_usec | evalsha命令的平均时延  | 该指标为节点执行evalsha命令的平均时延。  | ≥ 0<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis625_eval_qps         | eval命令的QPS      | 该指标为节点执行eval命令的QPS。      | ≥ 0<br>Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis623_eval_p99         | eval命令的p99时延    | 该指标为节点执行eval命令的p99时延。    | ≥ 0<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis621_eval_avg_usec    | eval命令的平均时延     | 该指标为节点执行eval命令的平均时延。     | ≥ 0<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |



Transactions 命令类型指标

表 4-68 Transactions 命令类型指标

| 指标ID                      | 指标名称            | 指标含义                     | 取值范围            | 单位       | 进制  | 维度                             | 监控周期（原始指标） |
|---------------------------|-----------------|--------------------------|-----------------|----------|-----|--------------------------------|------------|
| redis660_discard_qps      | discard命令的QPS   | 该指标为节点执行discard命令的QPS。   | ≥ 0<br>Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis658_discard_p99      | discard命令的p99时延 | 该指标为节点执行discard命令的p99时延。 | ≥ 0<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis656_discard_avg_usec | discard命令的平均时延  | 该指标为节点执行discard命令的平均时延。  | ≥ 0<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis655_exec_qps         | exec命令的QPS      | 该指标为节点执行exec命令的QPS。      | ≥ 0<br>Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis653_exec_p99         | exec命令的p99时延    | 该指标为节点执行exec命令的p99时延。    | ≥ 0<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis651_exec_avg_usec    | exec命令的平均时延     | 该指标为节点执行exec命令的平均时延。     | ≥ 0<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis650_multi_qps        | multi命令的QPS     | 该指标为节点执行multi命令的QPS。     | ≥ 0<br>Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis648_multi_p99        | multi命令的p99时延   | 该指标为节点执行multi命令的p99时延。   | ≥ 0<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis646_multi_avg_usec   | multi命令的平均时延    | 该指标为节点执行multi命令的平均时延。    | ≥ 0<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |

| 指标ID                      | 指标名称            | 指标含义                     | 取值范围                 | 单位       | 进制  | 维度                             | 监控周期（原始指标） |
|---------------------------|-----------------|--------------------------|----------------------|----------|-----|--------------------------------|------------|
| redis645_unwatch_qps      | unwatch命令的QPS   | 该指标为节点执行unwatch命令的QPS。   | $\geq 0$<br>Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis643_unwatch_p99      | unwatch命令的p99时延 | 该指标为节点执行unwatch命令的p99时延。 | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis641_unwatch_avg_usec | unwatch命令的平均时延  | 该指标为节点执行unwatch命令的平均时延。  | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis640_watch_qps        | watch命令的QPS     | 该指标为节点执行watch命令的QPS。     | $\geq 0$<br>Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis638_watch_p99        | watch命令的p99时延   | 该指标为节点执行watch命令的p99时延。   | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis636_watch_avg_usec   | watch命令的平均时延    | 该指标为节点执行watch命令的平均时延。    | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |

ExHash 命令类型指标

表 4-69 ExHash 命令类型指标

| 指标ID                        | 指标名称              | 指标含义                       | 取值范围         | 单位       | 进制  | 维度                             | 监控周期（原始指标） |
|-----------------------------|-------------------|----------------------------|--------------|----------|-----|--------------------------------|------------|
| redis794_exhdel_avg_usec    | exhdel命令的平均时延     | 该指标为节点执行exhdel命令的平均时延。     | ≥ 0 us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis796_exhdel_p99         | exhdel命令的p99时延    | 该指标为节点执行exhdel命令的p99时延。    | ≥ 0 us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis797_exhdel_qps         | exhdel命令的QPS      | 该指标为节点执行exhdel命令的QPS。      | ≥ 0 Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis766_exhexists_avg_usec | exhexists命令的平均时延  | 该指标为节点执行exhexists命令的平均时延。  | ≥ 0 us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis768_exhexists_p99      | exhexists命令的p99时延 | 该指标为节点执行exhexists命令的p99时延。 | ≥ 0 us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis769_exhexists_qps      | exhexists命令的QPS   | 该指标为节点执行exhexists命令的QPS。   | ≥ 0 Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis722_exhexpire_avg_usec | exhexpire命令的平均时延  | 该指标为节点执行exhexpire命令的平均时延。  | ≥ 0 us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis724_exhexpire_p99      | exhexpire命令的p99时延 | 该指标为节点执行exhexpire命令的p99时延。 | ≥ 0 us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |

| 指标ID                          | 指标名称                | 指标含义                         | 取值范围                 | 单位       | 进制  | 维度                             | 监控周期（原始指标） |
|-------------------------------|---------------------|------------------------------|----------------------|----------|-----|--------------------------------|------------|
| redis725_exhexpire_qps        | exhexpire命令的QPS     | 该指标为节点执行exhexpire命令的QPS。     | $\geq 0$<br>Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis718_exhexpireat_avg_usec | exhexpireat命令的平均时延  | 该指标为节点执行exhexpireat命令的平均时延。  | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis720_exhexpireat_p99      | exhexpireat命令的p99时延 | 该指标为节点执行exhexpireat命令的p99时延。 | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis721_exhexpireat_qps      | exhexpireat命令的QPS   | 该指标为节点执行exhexpireat命令的QPS。   | $\geq 0$<br>Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis702_exhget_avg_usec      | exhget命令的平均时延       | 该指标为节点执行exhget命令的平均时延。       | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis704_exhget_p99           | exhget命令的p99时延      | 该指标为节点执行exhget命令的p99时延。      | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis705_exhget_qps           | exhget命令的QPS        | 该指标为节点执行exhget命令的QPS。        | $\geq 0$<br>Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis782_exhgetall_avg_usec   | exhgetall命令的平均时延    | 该指标为节点执行exhgetall命令的平均时延。    | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis784_exhgetall_p99        | exhgetall命令的p99时延   | 该指标为节点执行exhgetall命令的p99时延。   | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |

| 指标ID                             | 指标名称                   | 指标含义                            | 取值范围                 | 单位       | 进制  | 维度                             | 监控周期（原始指标） |
|----------------------------------|------------------------|---------------------------------|----------------------|----------|-----|--------------------------------|------------|
| redis785_exhgetall_qps           | exhgetall命令的QPS        | 该指标为节点执行exhgetall命令的QPS。        | $\geq 0$<br>Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis750_exhgetwithver_avg_usec  | exhgetwithver命令的平均时延   | 该指标为节点执行exhgetwithver命令的平均时延。   | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis752_exhgetwithver_p99       | exhgetwithver命令的p99时延  | 该指标为节点执行exhgetwithver命令的p99时延。  | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis753_exhgetwithver_qps       | exhgetwithver命令的QPS    | 该指标为节点执行exhgetwithver命令的QPS。    | $\geq 0$<br>Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis742_exhincrby_avg_usec      | exhincrby命令的平均时延       | 该指标为节点执行exhincrby命令的平均时延。       | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis744_exhincrby_p99           | exhincrby命令的p99时延      | 该指标为节点执行exhincrby命令的p99时延。      | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis745_exhincrby_qps           | exhincrby命令的QPS        | 该指标为节点执行exhincrby命令的QPS。        | $\geq 0$<br>Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis746_exhincrbyfloat_avg_usec | exhincrbyfloat命令的平均时延  | 该指标为节点执行exhincrbyfloat命令的平均时延。  | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis748_exhincrbyfloat_p99      | exhincrbyfloat命令的p99时延 | 该指标为节点执行exhincrbyfloat命令的p99时延。 | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |

| 指标ID                        | 指标名称                 | 指标含义                          | 取值范围                 | 单位       | 进制  | 维度                             | 监控周期（原始指标） |
|-----------------------------|----------------------|-------------------------------|----------------------|----------|-----|--------------------------------|------------|
| redis749_exhincrbyfloat_qps | exhincrbyfloat命令的QPS | 该指标为节点执行exhincrbyfloat命令的QPS。 | $\geq 0$<br>Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis774_exhkeys_avg_usec   | exhkeys命令的平均时延       | 该指标为节点执行exhkeys命令的平均时延。       | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis776_exhkeys_p99        | exhkeys命令的p99时延      | 该指标为节点执行exhkeys命令的p99时延。      | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis777_exhkeys_qps        | exhkeys命令的QPS        | 该指标为节点执行exhkeys命令的QPS。        | $\geq 0$<br>Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis762_exhlen_avg_usec    | exhlen命令的平均时延        | 该指标为节点执行exhlen命令的平均时延。        | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis764_exhlen_p99         | exhlen命令的p99时延       | 该指标为节点执行exhlen命令的p99时延。       | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis765_exhlen_qps         | exhlen命令的QPS         | 该指标为节点执行exhlen命令的QPS。         | $\geq 0$<br>Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis754_exhmget_avg_usec   | exhmget命令的平均时延       | 该指标为节点执行exhmget命令的平均时延。       | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis756_exhmget_p99        | exhmget命令的p99时延      | 该指标为节点执行exhmget命令的p99时延。      | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |

| 指标ID                             | 指标名称                   | 指标含义                            | 取值范围                 | 单位       | 进制  | 维度                             | 监控周期（原始指标） |
|----------------------------------|------------------------|---------------------------------|----------------------|----------|-----|--------------------------------|------------|
| redis757_exhmget_t_qps           | exhmget命令的QPS          | 该指标为节点执行exhmget命令的QPS。          | $\geq 0$<br>Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis758_exhmgetwithver_avg_usec | exhmgetwithver命令的平均时延  | 该指标为节点执行exhmgetwithver命令的平均时延。  | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis760_exhmgetwithver_p99      | exhmgetwithver命令的p99时延 | 该指标为节点执行exhmgetwithver命令的p99时延。 | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis761_exhmgetwithver_qps      | exhmgetwithver命令的QPS   | 该指标为节点执行exhmgetwithver命令的QPS。   | $\geq 0$<br>Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis706_exhmset_avg_usec        | exhmset命令的平均时延         | 该指标为节点执行exhmset命令的平均时延。         | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis708_exhmset_p99             | exhmset命令的p99时延        | 该指标为节点执行exhmset命令的p99时延。        | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis709_exhmset_qps             | exhmset命令的QPS          | 该指标为节点执行exhmset命令的QPS。          | $\geq 0$<br>Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis714_exhpexpire_avg_usec     | exhpexpire命令的平均时延      | 该指标为节点执行exhpexpire命令的平均时延。      | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |

| 指标ID                           | 指标名称                 | 指标含义                          | 取值范围              | 单位       | 进制  | 维度                             | 监控周期（原始指标） |
|--------------------------------|----------------------|-------------------------------|-------------------|----------|-----|--------------------------------|------------|
| redis716_exhpexpire_p99        | exhpexpire命令的p99时延   | 该指标为节点执行exhpexpire命令的p99时延。   | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis717_exhpexpire_qps        | exhpexpire命令的QPS     | 该指标为节点执行exhpexpire命令的QPS。     | $\geq 0$ Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis710_exhpexpireat_avg_usec | exhpexpireat命令的平均时延  | 该指标为节点执行exhpexpireat命令的平均时延。  | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis712_exhpexpireat_p99      | exhpexpireat命令的p99时延 | 该指标为节点执行exhpexpireat命令的p99时延。 | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis713_exhpexpireat_qps      | exhpexpireat命令的QPS   | 该指标为节点执行exhpexpireat命令的QPS。   | $\geq 0$ Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis726_exhpttl_avg_usec      | exhpttl命令的平均时延       | 该指标为节点执行exhpttl命令的平均时延。       | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis728_exhpttl_p99           | exhpttl命令的p99时延      | 该指标为节点执行exhpttl命令的p99时延。      | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis729_exhpttl_qps           | exhpttl命令的QPS        | 该指标为节点执行exhpttl命令的QPS。        | $\geq 0$ Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis786_exhscan_avg_usec      | exhscan命令的平均时延       | 该指标为节点执行exhscan命令的平均时延。       | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |



| 指标ID                             | 指标名称                   | 指标含义                            | 取值范围                 | 单位       | 进制  | 维度                             | 监控周期（原始指标） |
|----------------------------------|------------------------|---------------------------------|----------------------|----------|-----|--------------------------------|------------|
| redis788_exhscan_p99             | exhscan命令的p99时延        | 该指标为节点执行exhscan命令的p99时延。        | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis789_exhscan_qps             | exhscan命令的QPS          | 该指标为节点执行exhscan命令的QPS。          | $\geq 0$<br>Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis790_exhscanunorder_avg_usec | exhscanunorder命令的平均时延  | 该指标为节点执行exhscanunorder命令的平均时延。  | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis792_exhscanunorder_p99      | exhscanunorder命令的p99时延 | 该指标为节点执行exhscanunorder命令的p99时延。 | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis793_exhscanunorder_qps      | exhscanunorder命令的QPS   | 该指标为节点执行exhscanunorder命令的QPS。   | $\geq 0$<br>Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis698_exhset_avg_usec         | exhset命令的平均时延          | 该指标为节点执行exhset命令的平均时延。          | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis700_exhset_p99              | exhset命令的p99时延         | 该指标为节点执行exhset命令的p99时延。         | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis701_exhset_qps              | exhset命令的QPS           | 该指标为节点执行exhset命令的QPS。           | $\geq 0$<br>Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis738_exhsetver_avg_usec      | exhsetver命令的平均时延       | 该指标为节点执行exhsetver命令的平均时延        | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |

| 指标ID                        | 指标名称              | 指标含义                       | 取值范围                 | 单位       | 进制  | 维度                             | 监控周期（原始指标） |
|-----------------------------|-------------------|----------------------------|----------------------|----------|-----|--------------------------------|------------|
| redis740_exhsetver_p99      | exhsetver命令的p99时延 | 该指标为节点执行exhsetver命令的p99时延。 | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis741_exhsetver_qps      | exhsetver命令的QPS   | 该指标为节点执行exhsetver命令的QPS。   | $\geq 0$<br>Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis770_exhstrlen_avg_usec | exhstrlen命令的平均时延  | 该指标为节点执行exhstrlen命令的平均时延。  | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis772_exhstrlen_p99      | exhstrlen命令的p99时延 | 该指标为节点执行exhstrlen命令的p99时延。 | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis773_exhstrlen_qps      | exhstrlen命令的QPS   | 该指标为节点执行exhstrlen命令的QPS。   | $\geq 0$<br>Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis730_exhttl_avg_usec    | exhttl命令的平均时延     | 该指标为节点执行exhttl命令的平均时延。     | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis732_exhttl_p99         | exhttl命令的p99时延    | 该指标为节点执行exhttl命令的p99时延。    | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis733_exhttl_qps         | exhttl命令的QPS      | 该指标为节点执行exhttl命令的QPS。      | $\geq 0$<br>Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis778_exhvals_avg_usec   | exhvals命令的平均时延    | 该指标为节点执行exhvals命令的平均时延。    | $\geq 0$<br>us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |

| 指标ID                     | 指标名称            | 指标含义                     | 取值范围         | 单位       | 进制  | 维度                             | 监控周期（原始指标） |
|--------------------------|-----------------|--------------------------|--------------|----------|-----|--------------------------------|------------|
| redis780_exhvals_p99     | exhvals命令的p99时延 | 该指标为节点执行exhvals命令的p99时延。 | ≥ 0 us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis781_exhvals_qps     | exhvals命令的QPS   | 该指标为节点执行exhvals命令的QPS。   | ≥ 0 Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis734_exhver_avg_usec | exhver命令的平均时延   | 该指标为节点执行exhver命令的平均时延。   | ≥ 0 us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis736_exhver_p99      | exhver命令的p99时延  | 该指标为节点执行exhver命令的p99时延。  | ≥ 0 us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis737_exhver_qps      | exhver命令的QPS    | 该指标为节点执行exhver命令的QPS。    | ≥ 0 Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |

通用命令指标

表 4-70 通用命令指标

| 指标ID                      | 指标名称            | 指标含义                     | 取值范围         | 单位               | 进制          | 维度                                         | 监控周期（原始指标） |
|---------------------------|-----------------|--------------------------|--------------|------------------|-------------|--------------------------------------------|------------|
| redis059_scan_qps         | scan命令的QPS      | 该指标为节点执行scan命令的QPS。      | ≥ 0 Counts/s | Cou<br>nts/<br>s | 不<br>涉<br>及 | redis_clu<br>ster_id,re<br>dis_node<br>_id | 1分<br>钟    |
| redis058_scan_p99         | scan命令的p99时延    | 该指标为节点执行scan命令的p99时延。    | ≥ 0 us       | us               | 不<br>涉<br>及 | redis_clu<br>ster_id,re<br>dis_node<br>_id | 1分<br>钟    |
| redis056_scan_avg_usec    | scan命令的平均时延     | 该指标为节点执行scan命令的平均时延。     | ≥ 0 us       | us               | 不<br>涉<br>及 | redis_clu<br>ster_id,re<br>dis_node<br>_id | 1分<br>钟    |
| redis055_persist_qps      | persist命令的QPS   | 该指标为节点执行persist命令的QPS。   | ≥ 0 Counts/s | Cou<br>nts/<br>s | 不<br>涉<br>及 | redis_clu<br>ster_id,re<br>dis_node<br>_id | 1分<br>钟    |
| redis054_persist_p99      | persist命令的p99时延 | 该指标为节点执行persist命令的p99时延。 | ≥ 0 us       | us               | 不<br>涉<br>及 | redis_clu<br>ster_id,re<br>dis_node<br>_id | 1分<br>钟    |
| redis052_persist_avg_usec | persist命令的平均时延  | 该指标为节点执行persist命令的平均时延。  | ≥ 0 us       | us               | 不<br>涉<br>及 | redis_clu<br>ster_id,re<br>dis_node<br>_id | 1分<br>钟    |
| redis051_ttl_qps          | ttl命令的QPS       | 该指标为节点执行ttl命令的QPS。       | ≥ 0 Counts/s | Cou<br>nts/<br>s | 不<br>涉<br>及 | redis_clu<br>ster_id,re<br>dis_node<br>_id | 1分<br>钟    |
| redis050_ttl_p99          | ttl命令的p99时延     | 该指标为节点执行ttl命令的p99时延。     | ≥ 0 us       | us               | 不<br>涉<br>及 | redis_clu<br>ster_id,re<br>dis_node<br>_id | 1分<br>钟    |

| 指标ID                     | 指标名称           | 指标含义                    | 取值范围              | 单位       | 进制  | 维度                             | 监控周期（原始指标） |
|--------------------------|----------------|-------------------------|-------------------|----------|-----|--------------------------------|------------|
| redis048_ttl_avg_usec    | ttl命令的平均时延     | 该指标为节点执行ttl命令的平均时延。     | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis047_del_qps         | del命令的QPS      | 该指标为节点执行del命令的QPS。      | $\geq 0$ Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis046_del_p99         | del命令的p99时延    | 该指标为节点执行del命令的p99时延。    | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis044_del_avg_usec    | del命令的平均时延     | 该指标为节点执行del命令的平均时延。     | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis043_expire_qps      | expire命令的QPS   | 该指标为节点执行expire命令的QPS。   | $\geq 0$ Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis042_expire_p99      | expire命令的p99时延 | 该指标为节点执行expire命令的p99时延。 | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis040_expire_avg_usec | expire命令的平均时延  | 该指标为节点执行expire命令的平均时延。  | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis039_exists_qps      | exists命令的QPS   | 该指标为节点执行exists命令的QPS。   | $\geq 0$ Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis038_exists_p99      | exists命令的p99时延 | 该指标为节点执行exists命令的p99时延。 | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |

| 指标ID                        | 指标名称              | 指标含义                       | 取值范围              | 单位       | 进制  | 维度                             | 监控周期（原始指标） |
|-----------------------------|-------------------|----------------------------|-------------------|----------|-----|--------------------------------|------------|
| redis036_exists_avg_usec    | exists命令的平均时延     | 该指标为节点执行exists命令的平均时延。     | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis410_randomkey_qps      | randomkey命令的QPS   | 该指标为节点执行randomkey命令的QPS。   | $\geq 0$ Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis408_randomkey_p99      | randomkey命令的p99时延 | 该指标为节点执行randomkey命令的p99时延。 | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis406_randomkey_avg_usec | randomkey命令的平均时延  | 该指标为节点执行randomkey命令的平均时延。  | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis405_type_qps           | type命令的QPS        | 该指标为节点执行type命令的QPS。        | $\geq 0$ Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis403_type_p99           | type命令的p99时延      | 该指标为节点执行type命令的p99时延。      | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis401_type_avg_usec      | type命令的平均时延       | 该指标为节点执行type命令的平均时延。       | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis400_info_qps           | info命令的QPS        | 该指标为节点执行info命令的QPS。        | $\geq 0$ Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis398_info_p99           | info命令的p99时延      | 该指标为节点执行info命令的p99时延。      | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |

| 指标ID                   | 指标名称         | 指标含义                  | 取值范围              | 单位       | 进制  | 维度                             | 监控周期（原始指标） |
|------------------------|--------------|-----------------------|-------------------|----------|-----|--------------------------------|------------|
| redis396_info_avg_usec | info命令的平均时延  | 该指标为节点执行info命令的平均时延。  | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis395_ping_qps      | ping命令的QPS   | 该指标为节点执行ping命令的QPS。   | $\geq 0$ Counts/s | Counts/s | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis393_ping_p99      | ping命令的p99时延 | 该指标为节点执行ping命令的p99时延。 | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |
| redis391_ping_avg_usec | ping命令的平均时延  | 该指标为节点执行ping命令的平均时延。  | $\geq 0$ us       | us       | 不涉及 | redis_cluster_id,redis_node_id | 1分钟        |

对于有多层测量维度的测量对象，使用接口查询监控指标时，需要代入具体指标的维度层级关系。

例如，需要查询节点的慢查询数量（redis817\_slow\_query\_count），该指标的维度信息为“redis\_cluster\_id,redis\_node\_id”，表示redis\_cluster\_id为0层，redis\_node\_id为1层。

- 通过API查询单个监控指标时，redis\_node\_id的维度信息代入样例如下：  
dim.0=redis\_cluster\_id,4dd5708e1730498e92d314fc7eb2117ein12&dim.1=redis\_node\_id,0c45432dccee461994bcee6a93807943no12  
其中，4dd5708e1730498e92d314fc7eb2117ein12和0c45432dccee461994bcee6a93807943no12分别为redis\_cluster\_id和redis\_node\_id的维度值，具体获取方法请参见“[维度](#)”表格中的获取指导。
- 通过API批量查询监控指标时，redis\_node\_id的维度信息代入样例如下：

```
"dimensions": [
 {
 "name": "redis_cluster_id",
 "value": "4dd5708e1730498e92d314fc7eb2117ein12"
 },
 {
 "name": "redis_node_id",
 "value": "0c45432dccee461994bcee6a93807943no12"
 }
]
```

```
}
]
```

其中，4dd5708e1730498e92d314fc7eb2117ein12和0c45432dccee461994bcee6a93807943no12分别为redis\_cluster\_id和redis\_node\_id的维度值，具体获取方法请参见“[维度](#)”表格中的获取指导。

维度

| Key              | Value                                                                        |
|------------------|------------------------------------------------------------------------------|
| redis_cluster_id | GeminiDB Redis数据库实例的集群ID。<br>该取值可通过云监控服务的“ <a href="#">查询主机监控维度指标信息</a> ”获取。 |
| redis_node_id    | GeminiDB Redis数据库实例的节点ID。<br>该取值可通过云监控服务的“ <a href="#">查询主机监控维度指标信息</a> ”获取。 |

4.13.2 设置告警规则

操作场景

通过设置告警规则，用户可自定义监控目标与通知策略，及时了解实例的运行状况，从而起到预警作用。

设置告警规则包括设置告警规则名称、监控对象、监控指标、告警阈值、监控周期和是否发送通知等参数。本节介绍了设置告警规则的具体方法。

操作步骤

- 步骤1** 登录[云监控服务控制台](#)。
- 步骤2** 在左侧导航树，选择“告警 > 告警规则”。
- 步骤3** 在“告警规则”页面，单击“创建告警规则”。

图 4-195 创建告警规则



- 步骤4** 根据界面提示配置告警参数。
1. 配置告警基本信息。



图 4-196 告警规则基本信息

\* 名称

alarm-cag2

描述

0/256

表 4-71 规则基本信息

| 参数名称 | 描述                    | 示例         |
|------|-----------------------|------------|
| 名称   | 系统会随机产生一个名称，您也可以进行修改。 | alarm-cag2 |
| 描述   | 告警规则描述（此参数非必填项）。      | -          |

2. 选择监控对象，配置告警范围。

表 4-72 参数说明

| 参数名称 | 说明                                                                                                                                                                                                                                          | 示例   |
|------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------|
| 告警类型 | 告警规则适用的告警类型，可选择指标或者事件告警。                                                                                                                                                                                                                    | 指标   |
| 资源类型 | 配置告警规则监控的服务名称。<br>此处请选择“云数据库 GeminiDB”。                                                                                                                                                                                                     | -    |
| 维度   | 用于指定告警规则对应指标的维度名称。<br>此处请选择“Redis-Redis节点”。                                                                                                                                                                                                 | -    |
| 监控范围 | 告警规则适用的资源范围。<br><b>说明</b> <ul style="list-style-type: none"><li>- 选择“全部资源”时，则任何实例满足告警策略时，都会发送告警通知，同时新购资源将自动绑定到告警规则。</li><li>- 选择“资源分组”时，该分组下任何资源满足告警策略时，都会触发告警通知。</li><li>- 选择“指定资源”时，单击“选择指定资源”，勾选具体的监控对象，单击“确定”将监控对象同步到“指定资源”中。</li></ul> | 全部资源 |
| 分组   | 当监控范围为资源分组时需配置此参数。                                                                                                                                                                                                                          | -    |

3. 设置告警策略。

图 4-197 设置告警策略



表 4-73 参数说明

| 参数名称 | 说明                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    | 示例                                                                              |
|------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------|
| 触发规则 | 根据需要可选择从关联模板、导入已有模板或自定义创建。<br><b>说明</b><br>当监控范围为指定资源时可选择从模板导入。                                                                                                                                                                                                                                                                                                                                                                                                                                       | 自定义创建                                                                           |
| 模板   | 选择需要导入的模板。<br>该项仅为从模板导入时需要配置。                                                                                                                                                                                                                                                                                                                                                                                                                                                                         | -                                                                               |
| 告警策略 | 触发告警规则的告警策略。您可以根据实际需求调整阈值、连续周期、告警间隔、告警级别等配置。 <ul style="list-style-type: none"><li>指标名称：告警规则对应的指标名称。<br/>推荐设置如下常用指标：<br/><b>存储容量使用率（整个实例级）</b><br/>用于监控GeminiDB Redis实例容量使用情况。如果存储容量使用率&gt;80%时，建议及时进行<b>扩容</b>。<br/><b>CPU利用率、内存利用率（节点级）</b><br/>用于监控GeminiDB Redis实例各节点计算资源的使用情况。如果CPU利用率或者内存利用率&gt;80%时，建议及时<b>添加节点</b>或<b>升级节点规格</b>。<br/>更多支持的监控指标请参见<b>GeminiDB Redis支持的监控指标</b>。</li><li>告警级别：根据告警的严重程度不同等级，可选择紧急、重要、次要、提示。<br/><b>说明</b><br/>告警规则内最多可添加50条告警策略，若其中一条告警策略达到条件都会触发告警。</li></ul> | 以CPU利用率为例如，如 <b>图 4-197</b> 配置的告警策略表示：CPU利用率的原始值连续3次大于等于80%时，则每10分钟向用户发送一次重要告警。 |

4. 配置告警通知信息。

图 4-198 配置告警通知信息

发送通知

\* 通知方式

通知组

主题订阅

\* 通知组

--请选择--

C

您可以选择通知组，若没有您想要选择的通知组，您可以单击 [创建通知组](#)。创建通知组后，您需要点击通知组列表操作栏的添加通知对象按钮，添加组内成员及通知方式。

\* 生效时间

每日

00:00

-

23:59

GMT+08:00

?

\* 触发条件

☒ 出现告警

☒ 恢复正常

表 4-74 参数说明

| 参数名称 | 说明                                                                                                                                                                                                     | 示例     |
|------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------|
| 发送通知 | 配置是否发送邮件、短信、HTTP 和HTTPS通知用户。<br>此处推荐您开启发送通知，当监控指标触发设定的告警策略时，云监控服务会在第一时间通过消息通知服务实时告知您云上资源异常，以免因此造成业务损失。                                                                                                 | 开启发送通知 |
| 通知方式 | 根据需要可选择通知组或主题订阅两种方式。                                                                                                                                                                                   | -      |
| 通知组  | 需要发送告警通知的通知组。创建通知组请参见 <a href="#">创建通知组</a> 。                                                                                                                                                          | -      |
| 通知对象 | 需要发送告警通知的对象，可选择云账号联系人或主题。 <ul style="list-style-type: none"><li>云账号联系人：注册账号时的手机和邮箱。</li><li>主题：消息发布或客户端订阅通知的特定事件类型，若此处没有需要的主题，需先创建主题并订阅该主题。详细操作请参见<a href="#">创建主题</a>和<a href="#">添加订阅</a>。</li></ul> | -      |
| 生效时间 | 该告警规则仅在生效时间内发送通知消息。<br>如生效时间为00:00-8:00，则该告警规则仅在00:00-8:00发送通知消息。                                                                                                                                      | -      |
| 触发条件 | 可以选择“出现告警”、“恢复正常”两种状态，作为触发告警通知的条件。                                                                                                                                                                     | -      |

5. 高级配置。

图 4-199 高级配置

高级配置 ▲

归属企业项目 | 标签

★ 归属企业项目

default ▼

创建企业项目

告警规则所属企业项目，非实例所属企业项目。

标签

如果您需要使用同一标签标识多种云资源，即所有服务均可在标签输入框下拉选择同一标签，建议在TMS中创建预定义标签。[查看预定义标签](#)

在下方键/值输入框输入内容后单击“添加”，即可将标签加入此处

请输入标签键

请输入标签值

添加

您还可以添加20个标签。

表 4-75 参数说明

| 参数名称   | 说明                                                                                                                                                                              | 示例      |
|--------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------|
| 归属企业项目 | 告警规则所属的企业项目。只有拥有该企业项目权限的用户才可以查看和管理该告警规则。创建企业项目请参考： <a href="#">创建企业项目</a> 。                                                                                                     | default |
| 标签     | 标签由键值对组成，用于标识云资源，可对云资源进行分类和搜索。建议在TMS中创建预定义标签。创建预定义标签请参考： <a href="#">创建预定义标签</a> 。 <ul style="list-style-type: none"><li>键的长度最大128字符，值的长度最大255字符。</li><li>最多可创建20个标签。</li></ul> | -       |

**步骤5** 配置完成后，单击“立即创建”，完成告警规则的创建。

告警规则创建完成后，当监控指标触发设定的告警策略时，云监控服务会在第一时间通过消息通知服务实时告知您云上资源异常，以免因此造成业务损失。

 说明

更多关于告警规则的信息，请参见《[云监控用户指南](#)》。

----结束

4.13.3 GeminiDB Redis 指标告警策略推荐配置

本章节主要介绍在实际业务中，GeminiDB Redis推荐用户配置的实例级告警和节点级告警策略。

表 4-76 GeminiDB Redis 实例级告警策略

| 指标名称                        | 指标维度 | 取值范围   | 告警策略                                                                        |
|-----------------------------|------|--------|-----------------------------------------------------------------------------|
| 实例级发送总流量                    | 实例级  | >=0    | 告警等级：重要<br>连续触发次数：3<br>告警阈值：<br>>=875000000byte<br>s/s<br>告警周期：每5分<br>钟告警一次 |
| 实例级接收总流量                    | 实例级  | >=0    | 告警等级：重要<br>连续触发次数：3<br>告警阈值：<br>>=875000000byte<br>s/s<br>告警周期：每5分<br>钟告警一次 |
| 实例的平均时延                     | 实例级  | >=0    | 告警等级：重要<br>连续触发次数：5<br>告警阈值：<br>>=15000us<br>告警周期：每15分<br>钟告警一次             |
| 实例的P99时延                    | 实例级  | >=0    | 告警等级：重要<br>连续触发次数：5<br>告警阈值：<br>>=30000us<br>告警周期：每15分<br>钟告警一次             |
| 实例的存储容量利<br>用率              | 实例级  | 0~100% | 告警等级：重要<br>连续触发次数：3<br>告警阈值：>=70%<br>告警周期：每1天<br>告警一次                       |
| 实例的客户端单次<br>请求发送的最大参<br>数个数 | 实例级  | >=0    | 告警等级：重要<br>连续触发次数：1<br>告警阈值：<br>>=10000<br>告警周期：每15分<br>钟告警一次               |

| 指标名称                | 指标维度 | 取值范围     | 告警策略                                                                 |
|---------------------|------|----------|----------------------------------------------------------------------|
| 实例的客户端单次请求获取的最大元素个数 | 实例级  | $\geq 0$ | 告警等级：重要<br>连续触发次数：1<br>告警阈值：<br>$\geq 10000$<br>告警周期：每15分钟告警一次       |
| 实例的客户端单次请求发送的最大字节数  | 实例级  | $\geq 0$ | 告警等级：重要<br>连续触发次数：1<br>告警阈值：<br>$\geq 1\text{MiB}$<br>告警周期：每15分钟告警一次 |
| 实例的客户端单次请求获取的最大字节数  | 实例级  | $\geq 0$ | 告警等级：重要<br>连续触发次数：1<br>告警阈值：<br>$\geq 1\text{MiB}$<br>告警周期：每15分钟告警一次 |

表 4-77 GeminiDB Redis 节点级告警策略

| 指标名称   | 指标维度 | 取值范围   | 告警策略                                                       |
|--------|------|--------|------------------------------------------------------------|
| CPU使用率 | 节点级  | 0~100% | 告警等级：重要<br>连续触发次数：3<br>告警阈值： $\geq 70\%$<br>告警周期：每15分钟告警一次 |
| 内存使用率  | 节点级  | 0~100% | 告警等级：提示<br>连续触发次数：3<br>告警阈值： $\geq 70\%$<br>告警周期：每15分钟告警一次 |
| 连接数使用率 | 节点级  | 0~100% | 告警等级：重要<br>连续触发次数：3<br>告警阈值： $\geq 50\%$<br>告警周期：每5分钟告警一次  |

| 指标名称     | 指标维度 | 取值范围     | 告警策略                                                                           |
|----------|------|----------|--------------------------------------------------------------------------------|
| 节点接收流量速率 | 节点级  | $\geq 0$ | 告警等级：重要<br>连续触发次数：3<br>告警阈值：<br>$\geq 87500000\text{bytes/s}$<br>告警周期：每5分钟告警一次 |
| 节点发送流量速率 | 节点级  | $\geq 0$ | 告警等级：重要<br>连续触发次数：3<br>告警阈值：<br>$\geq 87500000\text{bytes/s}$<br>告警周期：每5分钟告警一次 |

## 4.13.4 查看 GeminiDB Redis 监控指标

### 操作场景

云监控可以对GeminiDB Redis的运行状态进行日常监控。您可以通过管理控制台，直观地查看GeminiDB Redis的各项监控指标。

由于监控数据的获取与传输会花费一定时间，因此，云监控显示的是当前时间5~10分钟前的监控状态。如果您的实例刚刚创建完成，请等待5~10分钟后查看监控数据。

### 前提条件

- 实例正常运行。  
故障或已删除的实例，无法在云监控中查看其监控指标。当实例再次启动或恢复后，即可正常查看。
- 实例已正常运行一段时间（约10分钟）。  
对于新创建的实例，需要等待一段时间，才能查看上报的监控数据和监控视图。

### 方法一

- 步骤1** 登录[云数据库GeminiDB控制台](#)。
- 步骤2** 在“实例管理”页面，单击目标实例名称，进入实例的“基本信息”页面。
- 查看实例级监控指标：在右上角，单击“查看监控指标”。

图 4-200 查看实例级监控



- 查看节点级监控指标：在左侧导航栏选择“节点管理”页面，在“节点信息”区域选择指定节点，单击操作列“查看监控指标”。

图 4-201 查看节点级监控

节点信息

节点 节点池 添加节点 规格变更

请输入节点的名称或ID进行检索，默认按照名称检索

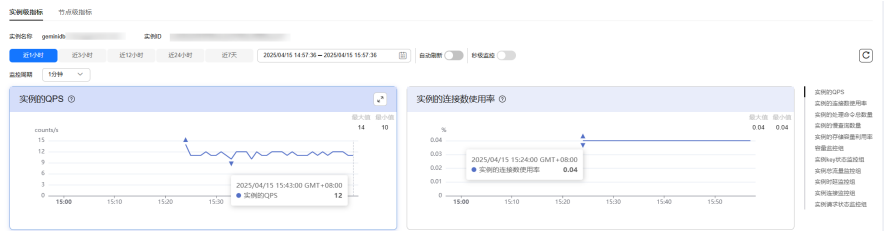
| 名称ID | 运行状态 | 可用区  | 内网IP | 弹性IP | 操作               |
|------|------|------|------|------|------------------|
|      | 正常   | 可用区4 |      |      | 查看监控指标 编辑弹性IP 更多 |
|      | 正常   | 可用区2 |      | 未绑定  | 查看监控指标 绑定弹性IP 更多 |

**步骤3** 在监控指标页面，您可以通过选择时长，查看对应时间的监控数据。  
当前页支持查看近1小时、近3小时和近12小时的监控数据。  
如需查看更长时间范围监控曲线，请在监控视图中单击  进入大图模式查看。  
----结束

## 方法二

- 步骤1** 登录云数据库GeminiDB控制台。
- 步骤2** 在“实例管理”页面，单击目标实例名称，进入实例的“基本信息”页面。
- 步骤3** 在左侧导航栏，单击“监控指标”。
- 步骤4** 在“监控指标”页面，查看实时监控数据。
- 选择“实例级指标”页签，查看实例的QPS、实例的平均命中率、实例的连接数使用率等实时监控数据。

图 4-202 实例级指标



- 选择“节点级指标”页签，查看CPU利用率、内存利用率、连接数使用率等实时监控数据。

图 4-203 节点级指标



- 支持的指标监控时间窗包括：近1小时、近3小时、近12小时、近24小时、近7天和自定义时间段。



- 您还可以打开自动刷新（自动刷新周期60s）和秒级监控(暂不支持CPU数小于4的实例。)



- 监控周期包括：1分钟和5分钟。

----结束

### 4.13.5 配置 GeminiDB Redis 监控看板

监控看板为您提供自定义查看监控数据的功能，将您关注的监控指标集中呈现在一张监控看板里，为您定制一个立体化的监控平台。

本章节主要介绍如何为GeminiDB Redis配置监控看板的方法。

#### 操作步骤

- 步骤1 登录[云监控服务控制台](#)。
- 步骤2 创建监控看板。

1. 在左侧导航栏选择“我的看板 > 自定义监控看板”，单击“创建看板”。

图 4-204 创建监控看板



2. 在“创建看板”弹出框中配置参数。

图 4-205 配置参数

创建监控看板

\*

名称

请输入监控看板名称

\*

归属企业项目

default

创建企业项目

取消

确定

表 4-78 参数说明

| 参数名称 | 说明                                             |
|------|------------------------------------------------|
| 名称   | 表示监控看板名称，该参数只能由中文、英文字母、数字、下划线、中划线组成，且长度不超过128。 |

| 参数名称   | 说明                                                                          |
|--------|-----------------------------------------------------------------------------|
| 归属企业项目 | 将监控看板关联给到某个企业项目时，只有拥有该企业项目权限的用户才可以查看和管理该监控看板。<br><b>说明</b><br>企业项目仅在部分区域上线。 |

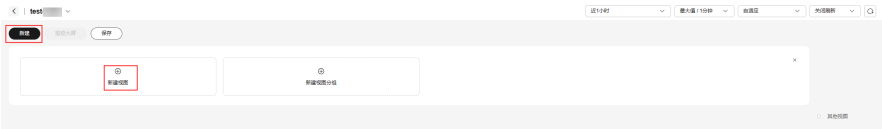
3. 单击“确定”，完成创建监控看板。

**步骤3** 为监控看板新建视图。

在完成监控看板的创建后，您就可以添加监控视图对GeminiDB Redis实例进行监控。

- 在“我的看板”页面，单击目标看板名称，进入监控看板详情页，单击“新建”来创建您的“视图”或“视图分组”。
  - 视图：通过不同的图表来展示指标的趋势或瞬时值。
  - 视图分组：类似文件目录，可以将看板中的视图归属到不同的视图分组。

**图 4-206** 新建视图



- 新建一个“视图”，在“添加监控视图”界面，完成以下配置。以下配置参数，详细参见[添加监控视图](#)。
  - 在右侧图表配置页签，根据需要选择“单指标一个视图”或“多指标一个视图”，在视图分组下面选择已有的分组，或单击“新建分组”重新创建。
  - 在右侧图表配置页签，图表类型可以选择“折线图”、“面积图”、“柱状图”、“条形图”、“环形图”或者“表格”。

图 4-207 图标配置

图表配置

多指标一个视图

单指标一个视图

视图分组

--请选择--

图表类型

折线图

面积图

柱状图

条形图

环形图

表格

视图

视图名称

^ 基础信息

备注 (可选)

^ 图例

位置

☐ 不显示

☐ 底部

☐ 右侧

图例值

--请选择--

^ 阈值及标注线

阈值线1

阈值

- c. 旧版：在左下角的“监控项配置”区域，设置监控范围、同比或环比、展示数量。  
新版：在左下角的“请选择监控指标”区域，设置监控指标、监控范围（监控范围支持选择全部资源、指定资源）、是否开启聚合及聚合规则、同比或环比、展示规则。
- d. 在监控范围区域的右上角可设置“左Y轴”或“右Y轴”，在视图预览区域可查看设置后的图表。

图 4-208 监控项配置（旧版）

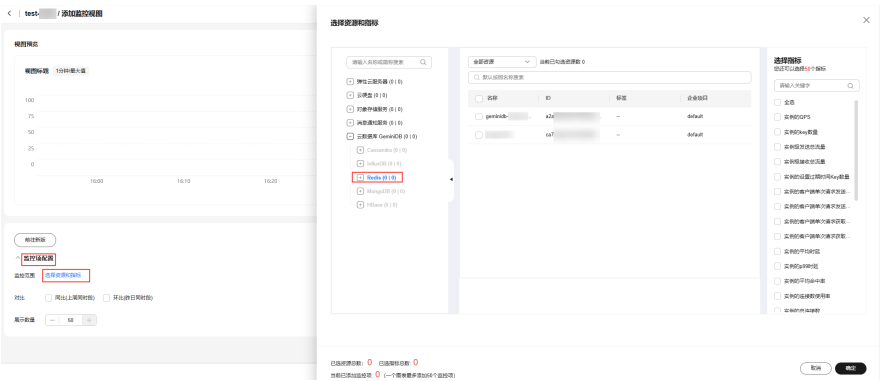
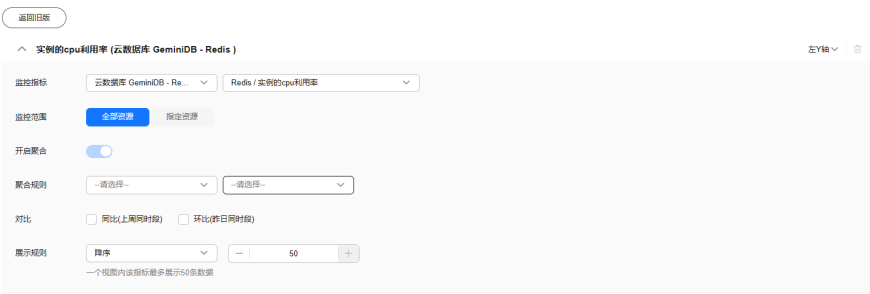


图 4-209 请选择监控指标（新版）

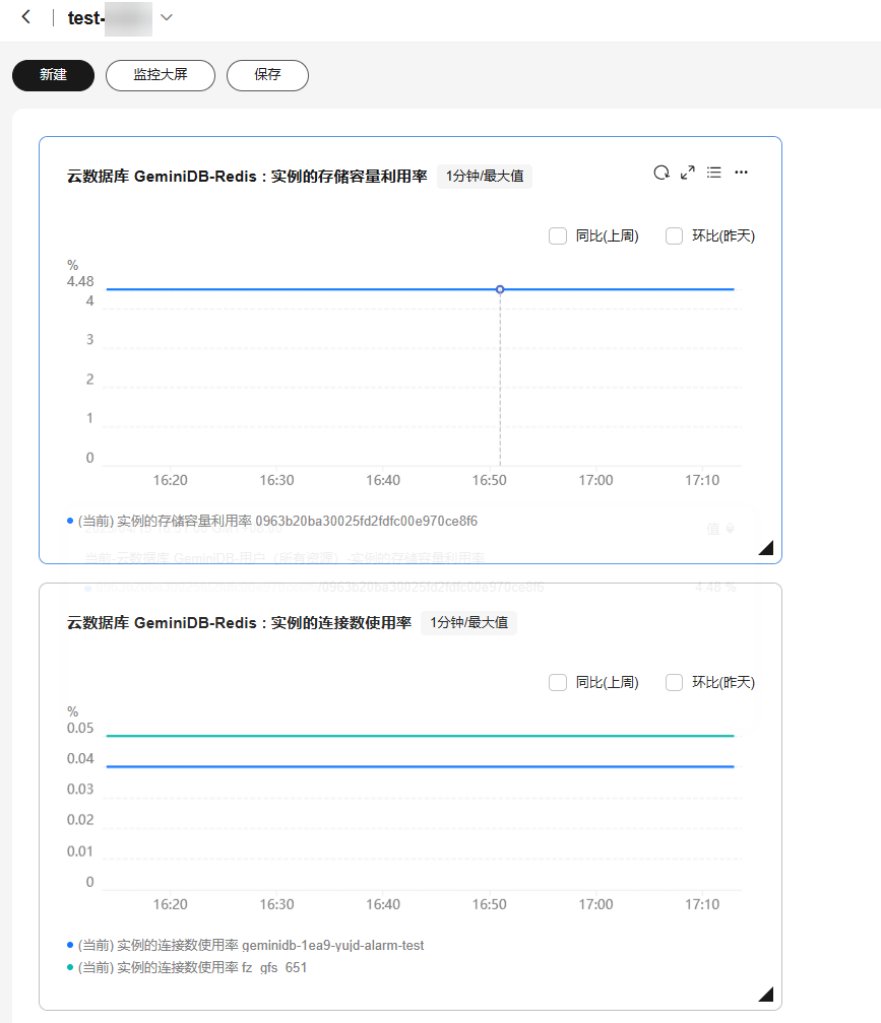


说明

添加监控视图时，建议选择单指标一个视图，这样每个监控指标就可以独立生成一个监控视图，方便您观察和分析监控数据。如果您需要添加多个监控指标，可通过多次添加监控视图来添加对应监控指标。

3. 在所选的监控看板上可以查看新建的监控视图的监控走势图。

图 4-210 查看新建的监控视图



----结束

4.13.6 事件监控

4.13.6.1 GeminiDB Redis 事件监控简介

事件监控提供了事件类型数据上报、查询和告警的功能。方便您将业务中的各类重要事件或对云资源的操作事件收集到云监控服务，并在事件发生时进行告警。

事件即云监控服务保存并监控的GeminiDB Redis资源的关键操作，您可以通过“事件”了解到谁在什么时间对系统哪些资源做了什么操作，如修改实例名称、规格变更等。

事件监控为您提供上报自定义事件的接口，方便您将业务产生的异常事件或重要变更事件采集上报到云监控服务。

事件监控默认开通，您可以在事件监控中查看系统事件和自定义事件的监控详情，目前支持的系统事件请参见[事件监控支持的事件说明](#)。

说明

如果您不创建事件监控的告警通知，默认不会收到告警通知。

4.13.6.2 GeminiDB Redis 事件监控支持的事件说明

表 4-79 云数据库 GeminiDB 事件监控支持的事件说明

| 事件来源  | 事件名称     | 事件ID                       | 事件级别 | 事件说明                     | 处理建议                                                                                | 事件影响       |
|-------|----------|----------------------------|------|--------------------------|-------------------------------------------------------------------------------------|------------|
| NoSQL | 创建实例业务失败 | NoSQL CreateInstanceFailed | 重要   | 一般是由于实例配额不足或底层资源不足等原因导致。 | 先释放不再使用的实例再尝试重新发放，或者您可以在管理控制台右上角，选择“ <a href="#">工单 &gt; 新建工单</a> ”，提交工单调整配额上限。     | 无法创建数据库实例。 |
|       | 变更规格失败   | NoSQL ResizeInstanceFailed | 重要   | 一般是由于底层资源不足等原因导致。        | 您可以在管理控制台右上角，选择“ <a href="#">工单 &gt; 新建工单</a> ”，提交工单让运维在后台协调资源再重试规格变更操作。            | 业务中断。      |
|       | 添加节点失败   | NoSQL AddNodesFailed       | 重要   | 一般是由于底层资源不足等原因导致。        | 提您可以在管理控制台右上角，选择“ <a href="#">工单 &gt; 新建工单</a> ”，交工单让运维在后台协调资源，删除添加失败的节点，重新尝试添加新节点。 | 无          |
|       | 删除节点失败   | NoSQL DeleteNodesFailed    | 重要   | 一般是由于底层释放资源失败导致。         | 重新尝试删除节点。                                                                           | 无          |

| 事件来源 | 事件名称     | 事件ID                                 | 事件级别 | 事件说明                                | 处理建议                                                                   | 事件影响           |
|------|----------|--------------------------------------|------|-------------------------------------|------------------------------------------------------------------------|----------------|
|      | 扩卷失败     | NoSQL ScaleUpStorageFailed           | 重要   | 一般是由于底层资源不足等原因导致。                   | 您可以在管理控制台右上角，选择“ <a href="#">工单 &gt; 新建工单</a> ”，提交工单让运维在后台协调资源再重试扩卷操作。 | 如果磁盘满，会导致业务中断。 |
|      | 重置密码失败   | NoSQL ResetPasswordFailed            | 重要   | 一般是由于重置密码命令超时导致。                    | 重新尝试重置密码操作。                                                            | 无              |
|      | 修改参数模板失败 | NoSQL UpdateInstanceParamGroupFailed | 重要   | 一般是由于修改参数模板命令超时导致。                  | 重新尝试修改参数模板操作。                                                          | 无              |
|      | 设置备份策略失败 | NoSQL SetBackupPolicyFailed          | 重要   | 一般是由于数据库连接异常导致。                     | 重新重试设置备份策略操作。                                                          | 无              |
|      | 创建手动备份失败 | NoSQL CreateManualBackupFailed       | 重要   | 一般是备份文件导出失败或上传失败等原因导致。              | 您可以在管理控制台右上角，选择“ <a href="#">工单 &gt; 新建工单</a> ”，提交工单让运维处理。             | 无法备份数据。        |
|      | 创建自动备份失败 | NoSQL CreateAutomatedBackupFailed    | 重要   | 一般是备份文件导出失败或上传失败等原因导致。              | 您可以在管理控制台右上角，选择“ <a href="#">工单 &gt; 新建工单</a> ”，提交工单让运维处理。             | 无法备份数据。        |
|      | 实例运行状态异常 | NoSQL FaultyDBInstance               | 重要   | 由于灾难或者物理机故障导致实例故障时，会上报该事件，属于关键告警事件。 | 您可以在管理控制台右上角，选择“ <a href="#">工单 &gt; 新建工单</a> ”，提交工单让运维处理。             | 可能导致数据库服务不可用。  |

| 事件来源 | 事件名称        | 事件ID                         | 事件级别 | 事件说明                                                       | 处理建议                                                                     | 事件影响                       |
|------|-------------|------------------------------|------|------------------------------------------------------------|--------------------------------------------------------------------------|----------------------------|
|      | 实例运行状态异常已恢复 | NoSQL DBInstanceRecovered    | 重要   | 针对灾难性的故障，NoSQL有高可用工具会自动进行恢复或者手动恢复，执行完成后会上报该事件。             | 不需要处理。                                                                   | 无                          |
|      | 节点运行状态异常    | NoSQL FaultyDBNode           | 重要   | 由于灾难或者物理机故障导致数据库节点故障时，会上报该事件，属于关键告警事件。                     | 检查数据库服务是否可以正常使用，并在管理控制台右上角，选择“ <a href="#">工单 &gt; 新建工单</a> ”，提交工单让运维处理。 | 可能导致数据库服务不可用。              |
|      | 节点运行状态异常已恢复 | NoSQL DBNodeRecovered        | 重要   | 针对灾难性的故障，NoSQL有高可用工具会自动进行恢复或者手动恢复，执行完成后会上报该事件。             | 不需要处理。                                                                   | 无                          |
|      | 实例主备切换      | NoSQL PrimaryStandbySwitched | 重要   | 在手动触发的主备倒换或节点故障自动触发的故障倒换场景下，会上报该事件。                        | 不需要处理。                                                                   | 无                          |
|      | 出现热点分区键     | HotKey Occurs                | 重要   | 客观上是因为主键设置不合理，使得热点数据集中分布在一个分区。客户端不合理的应用程序设计，造成对某一key的频繁读写。 | 1. 选择合理的分区键。<br>2. 业务增加缓存，业务应用先从缓存中读取热点数据。                               | 影响业务请求成功率，存在影响集群性能及稳定性的风险。 |



| 事件来源 | 事件名称          | 事件ID                         | 事件级别 | 事件说明                                                    | 处理建议                                                                        | 事件影响                          |
|------|---------------|------------------------------|------|---------------------------------------------------------|-----------------------------------------------------------------------------|-------------------------------|
|      | 出现超大分区键       | BigKey Occurs                | 重要   | 主键设计不合理，单个分区的记录数或数据量过大，引起了节点负载不均。                       | 1. 选择合理的分区键。<br>2. 基于现有分区键，增加分区键散列。                                         | 随着数据量增长，集群稳定性会下降。             |
|      | 数据盘空间不足       | NoSQL RiskyDataDiskUsage     | 重要   | 数据盘空间不足，产生此告警。                                          | 请参见对应服务用户指南中“扩容磁盘”的内容，进行磁盘扩容。                                               | 实例被设为只读模式，数据无法写入。             |
|      | 数据盘空间已扩容并恢复可写 | NoSQL DataDiskUsageRecovered | 重要   | 数据盘空间已扩容并恢复可写，产生此事件。                                    | 无需处理。                                                                       | 无                             |
|      | 创建索引失败        | NoSQL CreateIndexFailed      | 重要   | 业务负载超过实例规格瓶颈，此时再创建索引会耗费更多实例资源，导致响应变慢甚至卡顿，最终超时，引起索引创建失败。 | 1、根据业务负载，选择匹配的实例规格。<br>2、在业务低峰期创建索引。<br>3、使用后台方式创建索引。<br>4、索引字段，结合业务进行合理选择。 | 索引创建失败或不完整，导致索引无效，需要删掉索引重新创建。 |

| 事件来源 | 事件名称      | 事件ID                                  | 事件级别 | 事件说明                                                 | 处理建议                                                                      | 事件影响              |
|------|-----------|---------------------------------------|------|------------------------------------------------------|---------------------------------------------------------------------------|-------------------|
|      | 发生写入降速    | NoSQL Stalling Occurs                 | 重要   | 写入速度快，接近集群规模和实例规格范围允许最大写能力，从而触发数据库自身的限流机制，会发生请求失败情况  | 1. 根据业务的最大写请求速率，调整集群规模或者节点规格。<br>2. 衡量业务的最大写请求速率，分散业务写峰值速率。               | 影响业务的请求的成功率。      |
|      | 发生写入停止    | NoSQL Stopping Occurs                 | 重要   | 写入速度过快，达到集群规模和实例规格范围允许最大写能力，从而触发数据库自身的限流机制，会发生请求失败情况 | 1. 根据业务的最大写请求速率，调整集群规模或者节点规格。<br>2. 衡量业务的最大写请求速率，分散业务写峰值速率。               | 影响业务的请求的成功率。      |
|      | 重启数据库失败   | NoSQL Restart DBFailed                | 重要   | 一般是由于实例状态异常等原因导致。                                    | 您可以在管理控制台右上角，选择“ <a href="#">工单 &gt; 新建工单</a> ”，提交工单让运维处理。                | 数据库实例状态可能存在异常。    |
|      | 恢复到新实例失败  | NoSQL Restore ToNew Instance Failed   | 重要   | 一般是由于底层资源不足等原因导致。                                    | 您可以在管理控制台右上角，选择“ <a href="#">工单 &gt; 新建工单</a> ”，提交工单让运维在后台协调资源，重新尝试添加新节点。 | 无法恢复到新的数据库实例。     |
|      | 恢复到已有实例失败 | NoSQL Restore ToExist Instance Failed | 重要   | 一般是由于备份文件下载或恢复失败等原因导致。                               | 您可以在管理控制台右上角，选择“ <a href="#">工单 &gt; 新建工单</a> ”，提交工单让运维处理。                | 当前数据库实例可能处于不可用状态。 |

| 事件来源 | 事件名称        | 事件ID                               | 事件级别 | 事件说明                               | 处理建议                                                                                         | 事件影响           |
|------|-------------|------------------------------------|------|------------------------------------|----------------------------------------------------------------------------------------------|----------------|
|      | 删除备份文件失败    | NoSQL DeleteBackupFailed           | 重要   | 一般是由于备份文件从obs删除失败导致。               | 重新尝试删除备份文件。                                                                                  | 无              |
|      | 切换慢日志明文开关失败 | NoSQL SwitchSlowlogPlainTextFailed | 重要   | 一般是由于接口不支持切换等原因导致。                 | 请查阅 GeminiDB 用户指南，确认接口支持打开慢日志明文开关。您可以在管理控制台右上角，选择“ <a href="#">工单 &gt; 新建工单</a> ”，提交工单让运维处理。 | 无              |
|      | 绑定EIP失败     | NoSQL BindEipFailed                | 重要   | 一般是由于节点状态不正常，节点已经绑定EIP或EIP非法等原因导致。 | 检查节点是否正常，EIP是否合法。                                                                            | 无法通过公网访问数据库实例。 |
|      | 解绑EIP失败     | NoSQL UnbindEipFailed              | 重要   | 一般是由于节点状态不正常，节点已经未绑定EIP等原因导致。      | 检查节点和EIP状态是否正常。                                                                              | 无              |
|      | 修改参数失败      | NoSQL ModifyParameterFailed        | 重要   | 一般是由于参数取值非法等原因导致。                  | 排查参数值是否符合在合法范围内，您可以在管理控制台右上角，选择“ <a href="#">工单 &gt; 新建工单</a> ”，提交工单让运维处理。                   | 无              |
|      | 参数模板应用失败    | NoSQL ApplyParameterGroupFailed    | 重要   | 一般是由于实例状态异常导致参数模板无法应用等原因导致。        | 您可以在管理控制台右上角，选择“ <a href="#">工单 &gt; 新建工单</a> ”，提交工单让运维处理。                                   | 无              |

| 事件来源 | 事件名称        | 事件ID                  | 事件级别 | 事件说明                                  | 处理建议                                                                               | 事件影响                        |
|------|-------------|-----------------------|------|---------------------------------------|------------------------------------------------------------------------------------|-----------------------------|
|      | 开启或关闭SSL失败  | NoSQL SwitchSSLFailed | 重要   | 一般是由于修改SSL命令超时导致。                     | 重新提交一次或者您可以在管理控制台右上角，选择“ <a href="#">工单 &gt; 新建工单</a> ”，提交工单处理，并先保持切换之前使用SSL的连接方式。 | 是否使用SSL连接。                  |
|      | 单行数据量太大     | LargeRowOccurs        | 重要   | 用户单行数据量过大，可能会导致查询超时，进而节点OOM异常等各种故障发生。 | 1. 对每列和每行的写入长度做限制，遵从规范，使得单行的key和value长度和不超过阈值。<br>2. 排查业务是否出现异常写入和异常编码，导致写入大row。   | 过大的单行记录，随着数据量增长，集群稳定性会下降。   |
|      | 用户计划删除KMS密钥 | planDeleteKmsKey      | 重要   | 由于用户计划删除KMS密钥导致。                      | 及时检查此密钥关联的GeminiDB实例是否已删除或不再使用，密钥删除后将影响实例业务的正常运行。                                  | 密钥到期后会自动删除，删除后将影响实例业务的正常运行。 |
|      | 查询墓碑数过多     | TooManyQueryTombs     | 重要   | 查询墓碑数过多，可能会导致查询超时，影响查询性能。             | 使用合理的查询删除方式，避免大批次的范围查询                                                             | 可能会导致查询超时，影响查询性能。           |

| 事件来源 | 事件名称  | 事件ID                     | 事件级别 | 事件说明                    | 处理建议                                                   | 事件影响        |
|------|-------|--------------------------|------|-------------------------|--------------------------------------------------------|-------------|
|      | 超大集合列 | TooLargeCollectionColumn | 重要   | 当集合列元素个数太多时，会出现集合列查询失败。 | 对集合列的元素个数做限制，遵从规范不超过阈值。<br>排查业务是否出现异常写入和异常编码，导致写入大集合列。 | 会出现集合列查询失败。 |

4.13.6.3 创建 GeminiDB Redis 事件监控的告警通知

操作场景

本章节指导用户创建事件监控的告警通知。

使用须知

如果您不创建事件监控的告警通知，默认不会收到告警通知。

操作步骤

- 步骤1 登录[云监控服务控制台](#)。
- 步骤2 在左侧导航栏选择“事件监控”，进入“事件监控”页面。
- 步骤3 在事件列表页面，单击页面右上角的“创建告警规则”。
- 步骤4 在“创建告警规则”界面，配置参数。

表 4-80 告警内容参数说明

| 参数     | 参数说明                                                                      |
|--------|---------------------------------------------------------------------------|
| 名称     | 系统会随机产生一个名称，用户也可以进行修改。                                                    |
| 描述     | 告警规则描述（此参数非必填项）。                                                          |
| 归属企业项目 | 可选择已有的企业项目，或单击“创建企业项目”进行创建，具体请参见《企业管理用户指南》中“ <a href="#">创建企业项目</a> ”的内容。 |
| 告警类型   | 用于指定告警规则对应的告警类型。                                                          |
| 事件类型   | 用于指定告警规则对应指标的事件类型。                                                        |
| 事件来源   | 事件来源的云服务名称。<br>选择云数据库 GeminiDB。                                           |
| 监控范围   | 创建事件监控针对的资源范围。                                                            |

| 参数   | 参数说明                                                                                                          |
|------|---------------------------------------------------------------------------------------------------------------|
| 选择类型 | 选择自定义创建。                                                                                                      |
| 告警策略 | 事件名称：用户操作系统资源的动作，如用户登录，用户登出，为一个瞬间的操作动作。<br>事件监控支持的操作事件请参见 <a href="#">事件监控支持的事件说明</a> 。<br>用户根据需要选择触发方式、告警级别。 |

单击  开启“发送通知”，生效时间默认为全天，若没有您想要选择的主题，可以单击下一行的“创建主题”进行添加。

表 4-81 发送通知

| 参数   | 参数说明                                                                                                                                                                                                          |
|------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 发送通知 | 配置是否发送邮件、短信、HTTP和HTTPS通知用户。                                                                                                                                                                                   |
| 通知对象 | 需要发送告警通知的对象，可选择“云账号联系人”或主题。 <ul style="list-style-type: none"><li>云账号联系人：注册账号时的手机和邮箱。</li><li>主题：消息发布或客户端订阅通知的特定事件类型，若此处没有需要的主题，需先创建主题并订阅该主题。<br/>详细操作请参见<a href="#">创建主题</a>和<a href="#">添加订阅</a>。</li></ul> |
| 生效时间 | 该告警规则仅在生效时间内发送通知消息。<br>如生效时间为08:00-20:00，则该告警规则仅在08:00-20:00发送通知消息。                                                                                                                                           |
| 触发条件 | 出现告警                                                                                                                                                                                                          |

步骤5 配置完成后，单击“立即创建”，完成告警规则的创建。

----结束

4.13.6.4 查看 GeminiDB Redis 事件监控数据

操作场景

事件监控提供了事件类型数据上报、查询和告警的功能。方便您将业务中的各类重要事件或对云资源的操作事件收集到云监控服务，并在事件发生时进行告警。

事件监控默认开通，您可以在事件监控中查看系统事件和自定义事件的监控详情。

本章节指导用户查看事件监控的监控数据。

操作步骤

步骤1 登录[云数据库GeminiDB控制台](#)。

**步骤2** 您可以通过如下方式查看事件监控数据：

- 在“实例管理”页面，单击目标实例名称进入“基本信息”页面，在左侧导航栏，单击“监控指标”，根据需要可以选择“实例级指标”或“节点级指标”，查看事件监控数据。
- 在“实例管理”页面，单击目标实例名称进入“基本信息”页面，在左侧导航栏，单击“节点管理”，在节点信息对应的操作列下，单击“查看监控指标”，跳转到云监控页面，查看事件监控数据。

**步骤3** 单击上方的  返回云监控服务主界面。

**步骤4** 单击业务左侧导航栏的“事件监控”。

进入“事件监控”页面。在“事件监控”页面，默认展示近24小时的所有系统事件。

您也可以根据需要选择“近1小时”“近3小时”“近12小时”“近24小时”“近7天”“近30天”，分别查看不同时段的事件。

**步骤5** 展开对应的事件类型，单击具体事件右侧的操作列的“查看事件”，可查看具体事件的内容。

----结束

## 4.14 GeminiDB Redis 标签管理

### 操作场景

标签管理服务（Tag Management Service，简称TMS）用于用户在云平台，通过统一的标签管理各种资源。标签管理服务与各服务共同实现标签管理能力，标签管理服务提供全局标签管理能力，各服务维护自身标签管理。

为GeminiDB Redis实例添加标签，可以方便用户识别和管理拥有的GeminiDB Redis资源。您可以在创建实例时添加标签，也可以在实例创建完成后，在实例详情页添加标签。

标签添加成功后，您可以通过搜索标签键或值，快速查询关联的资源信息。

### 使用须知

- 建议您先在标签管理服务系统中设置预定义标签。
- 标签由“键”和“值”组成，每个标签中的一个“键”只能对应一个“值”。关于标签键和标签值的命名规则，请参见[表4-82](#)。
- 每个实例默认最多支持20个标签配额。
- 标签命名需要满足[表4-82](#)规则。

表 4-82 命名规则

| 参数  | 规则                                                                                                                                                                             | 示例           |
|-----|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------|
| 标签键 | <ul style="list-style-type: none"><li>不能为空。</li><li>对于每个实例，每个标签的键唯一。</li><li>长度不超过128个字符。</li><li>不能以“_sys_”开头和以空格开头、结尾，且只能包含数字、英文字母、下划线、点、中划线、中文、空格、冒号、加号、等号、符号@和/。</li></ul> | Organization |
| 标签值 | <ul style="list-style-type: none"><li>可以为空。</li><li>长度不超过255个字符。</li><li>只能包含数字、英文字母、下划线、点、中划线、中文、空格、冒号、加号、等号、符号@和/。</li></ul>                                                 | nosql_01     |

添加标签

- 步骤1 登录[云数据库GeminiDB控制台](#)。
- 步骤2 在“实例管理”页面，选择目标实例，单击实例名称，进入“基本信息”页面。
- 步骤3 在左侧导航树，单击“标签”。
- 步骤4 在“标签”页面，单击“添加标签”，在弹出框中，输入标签键和标签值，单击“确定”。
- 步骤5 添加成功后，您可在当前实例的所有关联的标签集合中，查询并管理自己的标签。
- 结束

编辑标签

- 步骤1 登录[云数据库GeminiDB控制台](#)。
- 步骤2 在“实例管理”页面，选择目标实例，单击实例名称，进入“基本信息”页面。
- 步骤3 在左侧导航树，单击“标签”。
- 步骤4 在“标签”页面，选择需要编辑的标签，单击“编辑”，在弹出框中修改标签值，单击“确定”。
- 编辑标签时，不能修改标签的键，只能修改标签的值。
- 步骤5 编辑成功后，您可在当前实例的所有关联的标签集合中，查询并管理自己的标签。
- 结束

删除标签

- 步骤1 登录[云数据库GeminiDB控制台](#)。



- 步骤2** 在“实例管理”页面，选择目标实例，单击实例名称，进入“基本信息”页面。
- 步骤3** 在左侧导航树，单击“标签”。
- 步骤4** 在“标签”页面，选择需要删除的标签，单击“删除”，在弹出框中单击“是”。
- 步骤5** 删除成功后，该标签将不再显示在实例的所有关联的标签集合中。

----结束

## 标签搜索

- 步骤1** 登录[云数据库GeminiDB控制台](#)。
- 步骤2** 在“实例管理”页面，在搜索框中单击“标签”。

图 4-211 选择标签



- 步骤3** 勾选需要查询的标签，单击“确定”，可以查询到与该标签关联的实例信息。

图 4-212 标签搜索



----结束

## 4.15 GeminiDB Redis 用户资源配额

### 操作场景

为防止资源滥用，平台限定了各服务资源的配额，对用户的资源数量和容量做了限制。如您最多可以创建多少个云数据库 GeminiDB实例。

如果当前资源配额限制无法满足使用需要，您可以申请扩大配额。

### 查看配额

**步骤1** 登录[云数据库GeminiDB控制台](#)。

**步骤2** 单击管理控制台左上角的📍，选择区域和项目。

**步骤3** 在页面右上角，选择“资源 > 我的配额”，进入“服务配额”页面。  
系统进入“服务配额”页面。

图 4-213 我的配额



**步骤4** 您可以在“服务配额”页面，查看各项资源GeminiDB的总配额以及使用情况。

----结束

## 申请扩大配额

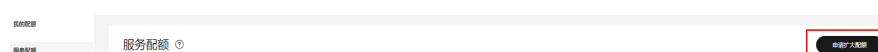
**步骤1** 登录[云数据库GeminiDB控制台](#)。

**步骤2** 单击管理控制台左上角的，选择区域和项目。

**步骤3** 在页面右上角，选择“资源 > 我的配额”，进入“服务配额”页面。

**步骤4** 在页面右上角单击“申请扩大配额”，进入配额申请页面。

图 4-214 申请扩大配额



**步骤5** 在“新建工单”页面，根据您的需求，填写相关参数。

其中，“问题描述”项请填写需要调整的内容和申请原因。

**步骤6** 填写完毕后，勾选协议并单击“提交”。

----结束

## 4.16 内存加速

### 4.16.1 通过 GeminiDB Redis 实现 RDS 内存加速

#### 4.16.1.1 内存加速概述

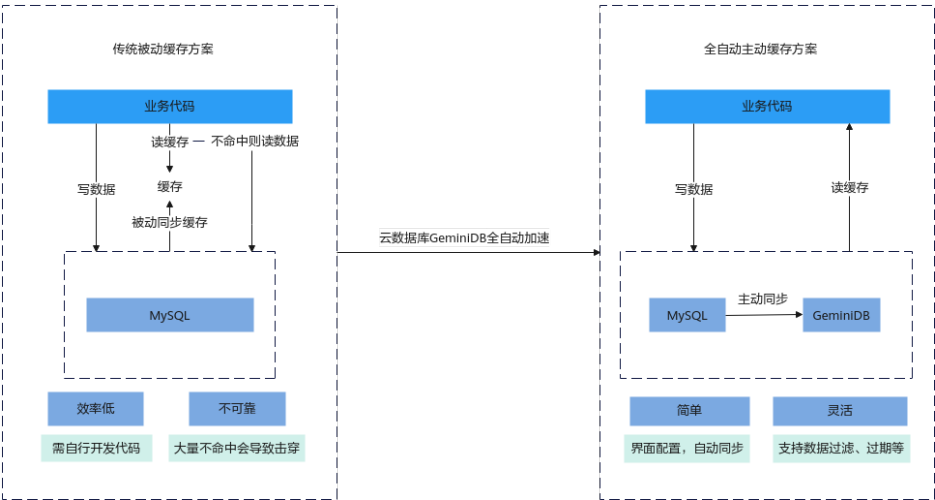
内存加速是GeminiDB Redis为了优化“传统被动缓存方案”而推出的功能，它可以让用户通过界面配置规则的形式，自动缓存MySQL的数据，加速MySQL的访问。

##### 说明

- 通过内存加速功能所创建的GeminiDB Redis实例需单独收费，具体收费规则为：
  - 如果创建的GeminiDB Redis实例规格为16GB、24GB、32GB、48GB或64GB，则实例限期免费3个月，到期后按正常计费规则收费。
  - 其他规格的实例将直接按计费规则收费，计费详情请参见[计费说明](#)。
- 1个IAM账号在1个区域最多创建3个限期3个月免费试用的GeminiDB Redis实例（实例规格为16GB、24GB、32GB、48GB或64GB）。
- 支持创建的GeminiDB Redis实例规格变更，但是变更后GeminiDB Redis实例开始收费。
- 解除映射后，需要及时删除GeminiDB Redis实例，否则会对GeminiDB Redis实例收费。

如下图[图4-215](#)所示，“传统被动缓存方案”需要用户自行开发代码把MySQL中的数据写入到缓存中，存在效率低、不可靠的缺点。而采用云数据内存加速的“全自动主动缓存方案”，支持界面可视化配置，配置完成后即可实现数据自动同步。同时还支持数据过滤及过期等功能，极大提高了开发效率及数据的可靠性。

图 4-215 原理图



### 4.16.1.2 开启和使用内存加速

本章节主要介绍如何开启内存加速功能，主要流程如下：

**步骤1：选取GeminiDB实例**

**步骤2：创建映射**

**步骤3：使用示例**

#### 注意事项

- 内存加速开启后，不允许在MySQL实例执行RESET MASTER、FLUSH LOGS等删除binlog的命令。
- 目前仅支持MySQL到GeminiDB Redis接口Hash类型的转换。
- 新规则的Redis键前缀+键分隔符不能是已有规则的Redis键前缀+键分隔符的子前缀，反之亦然。例如新规则的前缀为 "pre1:"，键分隔符为 ","，老规则前缀为 "pre1"，分隔符为 ":"，这种情况不允许创建新规则。
- 暂不支持ENUM、SET、JSON三种数据类型的同步。
- 由于是轻量级增量数据同步，目前仅支持单表查询，暂不支持联合查询。
- 目前仅GeminiDB Redis实例收费，该功能没有其他额外费用。
- 如果您删除了RDS实例，不会同步删除搭建DBcache的GeminiDB Redis实例，如不再使用，请及时删除GeminiDB Redis实例，避免产生额外费用。
- 您购买RDS实例时，如果“内存加速”参数，选择了“立即购买”，则会自动购买GeminiDB实例搭建DBcache，您可以跳过创建GeminiDB实例步骤，直接从[创建映射](#)开始使用。该功能目前处于公测阶段，您可以在管理控制台右上角，选择“[工单 > 新建工单](#)”联系客服进行咨询。

#### 步骤 1：选取 GeminiDB 实例

**步骤1** 登录[云数据库RDS控制台](#)。

**步骤2** 单击管理控制台左上角的 ，选择区域和项目。

**步骤3** 在“实例管理”页面，单击目标实例名称，进入“概览”页面。

**步骤4** 在左侧导航栏选择“内存加速”，进入“内存加速”页面，单击“创建GeminiDB实例”或者“选择已有GeminiDB实例”。

- 单击“创建GeminiDB实例”，并执行**步骤5**。
- 单击“选择已有GeminiDB实例”选择已有的GeminiDB Redis实例。

 **说明**

选择“选择已有GeminiDB实例”时，当前仅支持主备版类型实例，并且Region、VPC，子网、安全组等参数需要与当前RDS实例一致。

**步骤5** 参见**表4-83**选择并填写实例相关信息后，单击“提交”，完成实例创建。

**表 4-83 基本信息**

| 参数           | 描述                                                                                                                                                                                                                 |
|--------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| GeminiDB实例规格 | 实例的CPU和内存，详细规格见 <b>表4-84</b> 。                                                                                                                                                                                     |
| 数据库端口        | 数据库的访问端口号。<br>GeminiDB Redis支持自定义端口号，取值范围为：1024~65535，其中，如下端口号为禁用端口：2180、2887、3887、6377、6378、6380、8018、8079、8091、8479、8484、8999、12017、12333、50069。<br>当您不指定端口时，创建实例的访问端口默认为6379。<br><b>说明</b><br>暂时不支持实例创建成功后修改端口。 |
| 实例名称         | 设置实例名称时，需要满足如下规则。 <ul style="list-style-type: none"><li>• 实例名称允许和已有名称重复。</li><li>• 实例名称长度在4个到64个字节之间，必须以字母或中文字开头，区分大小写，可以包含字母、数字、中划线、下划线或中文（一个中文字符占用3个字节），不能包含其他特殊字符。</li></ul>                                    |
| 数据库密码        | 用户设置的数据库密码。 <ul style="list-style-type: none"><li>• 长度为8~32个字符。</li><li>• 必须是大写字母、小写字母、数字、特殊字符中至少两种的组合，其中可输入~!@#%^*-_+=?特殊字符。</li><li>• 系统会进行弱密码校验，安全起见，请输入高强度密码。请妥善保管您的密码，因为系统将无法获取您的密码信息。</li></ul>              |
| 确认密码         | 必须和数据库密码一致。                                                                                                                                                                                                        |

 **说明**

Region、可用区、VPC，子网等参数默认与当前RDS实例一致。

表 4-84 GeminiDB 实例规格

| 实例存储容量<br>(GB) | 节点数量 | vCPU<br>(个) | QPS基<br>准 | 单节点实例的最大<br>连接数 | 实例DB数 |
|----------------|------|-------------|-----------|-----------------|-------|
| 16             | 2    | 1           | 10,000    | 10,000          | 1,000 |
| 24             | 2    | 2           | 20,000    | 10,000          | 1,000 |
| 32             | 2    | 2           | 20,000    | 10,000          | 1,000 |
| 48             | 2    | 4           | 40,000    | 20,000          | 1,000 |
| 64             | 2    | 4           | 40,000    | 20,000          | 1,000 |
| 96             | 2    | 8           | 80,000    | 20,000          | 1,000 |
| 128            | 2    | 16          | 160,000   | 20,000          | 1,000 |

----结束

步骤 2：创建映射

- 步骤1 登录[云数据库RDS控制台](#)。
- 步骤2 单击管理控制台左上角的，选择区域和项目。
- 步骤3 在“实例管理”页面，单击目标实例名称，进入“概览”页面。
- 步骤4 在左侧导航栏选择“内存加速”，在“映射规则”区域单击“创建映射规则”。

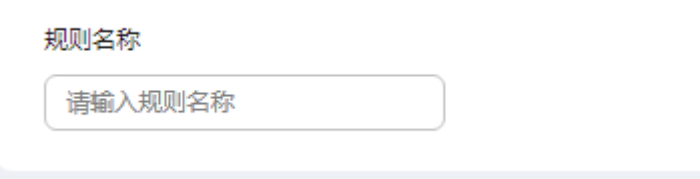
图 4-216 映射规则



- 步骤5 在“创建映射规则”页面，填写并选择相关信息。
1. 配置规则名称。

规则名称：设置映射规则名称。规则名称不能包含#和中文，长度不超过256个字符，且规则名称在同一个GeminiDB实例中不可重复。

图 4-217 规则名称



2. 源实例配置。
- 数据库名：选择需要加速实例的数据库。

- 表名：选择需要加速实例中的表。

图 4-218 源实例配置

源实例配置

源实例名称

数据库名

表名

db1

students

3. 加速实例端配置。
- Redis键前缀：可选。默认的前缀格式为“数据库名:数据表名:字段名1:字段名2...”，长度不超过1024个字符。如果用户自定义了Redis键前缀，那么以自定义前缀为准。

- 值存储类型：缓存的数据类型，当前仅支持Hash。

- DB号（0-999）：加速实例中保存缓存数据的DB号，默认为0。

- TTL（单位：秒，默认值为30天。）：加速实例中缓存数据的过期时间，默认30天（2592000秒），输入-1为不过期。

- 键分隔符：Redis键前缀与键（Key）、以及键（Key）各字段之间的分隔符，长度为1个字符。

图 4-219 加速实例端配置

加速实例端配置

加速实例名称

Redis键前缀

值存储类型

DB号（0-999）

TTL（单位：秒，默认为30天）

键分隔符

0

2,592,000

:

4. 单击“键（Key）设置”，选择加速实例的键（Key）字段，单击“确定”。

说明

如果加速实例键（Key）由多个源实例字段组成，那么需要具备唯一性（这些字段在MySQL实例中要能成功创建出唯一索引）；可以单击“上移”和“下移”调整各字段在键（Key）中的先后顺序。

图 4-220 键（Key）设置



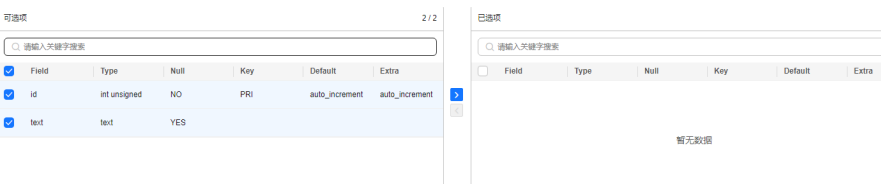
设置完成后，会在界面中显示键（Key）的构成方式，如下图所示：

图 4-221 键（Key）



5. 配置加速实例的域-值。  
将源实例中需要的字段，右移到加速实例的字段中。

图 4-222 配置加速实例字段



6. 配置信息填写完成后，单击“提交”，完成映射规则创建。
- 结束

步骤 3：使用示例

1. 在源实例MySQL中创建一个db1数据库，并在db1中创建一个students表，具体的SQL 如下：



```
mysql> CREATE DATABASE db1;
Query OK, 1 row affected (0.00 sec)

mysql> CREATE TABLE db1.students(
 sid INT UNSIGNED PRIMARY KEY AUTO_INCREMENT NOT NULL,
 sname VARCHAR(20),
 sclass INT,
 sgender VARCHAR(10),
 sbirthday DATE
);
Query OK, 0 rows affected (0.00 sec)

mysql> DESC db1.students;
+-----+-----+-----+-----+-----+-----+
| Field | Type | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
sid	int unsigned	NO	PRI	NULL	auto_increment
sname	varchar(20)	YES		NULL	
sclass	int	YES		NULL	
sgender	varchar(10)	YES		NULL	
sbirthday	date	YES		NULL	
+-----+-----+-----+-----+-----+-----+
5 rows in set (0.00 sec)
```

2.
- 建表后，在“内存加速”界面，创建一条映射规则，将students表中的每一行转换为一个Redis Hash。该Hash的键（Key）由”数据库名:数据表名:sid:<sid的值>”构成，域-值由选中的“sname，sclass，sgender，sbirthday”构成。

图 4-223 配置映射规则

规则名称

students-to-hash

源实例配置

源实例名称

数据库名

表名

加速实例配置

加速实例名称

Redis键前缀

键存储类型

DB号 (0-999)

TTL (单位: 秒, 默认为30天)

键分隔符

键 (Key) 设置

Hash db1:students:sid:<sid>

值 (value) 设置

可选项

已选项

3.
- 映射规则创建成功后，可以查看映射规则及映射信息。

图 4-224 映射信息



4. 在MySQL实例侧对students表插入一条新数据：  
mysql> INSERT INTO db1.students (sname, sclass, sgender, sbirthday) VALUES ('zhangsan', 1, 'male', '2015-05-20');  
Query OK, 1 row affected (0.01 sec)
- ```
mysql> SELECT * FROM db1.students;
+-----+-----+-----+-----+-----+
| sid | sname | sclass | sgender | sbirthday |
+-----+-----+-----+-----+-----+
| 1 | zhangsan | 1 | male | 2015-05-20 |
+-----+-----+-----+-----+-----+
1 row in set (0.00 sec)
```
5. 由于映射规则的存在，会把该数据自动同步到GeminiDB实例中。此时在GeminiDB实例中执行命令即可查询到该数据：
127.0.0.1:6379> KEYS *
1) "db1:students:sid:1"
- ```
127.0.0.1:6379> HGETALL db1:students:sid:1
1) "sbirthday"
2) "2015-05-20"
3) "sclass"
4) "1"
5) "sgender"
6) "male"
7) "sname"
8) "zhangsan"
```
6. 在MySQL实例侧对students表再次插入一条新数据：  
mysql> INSERT INTO db1.students (sname, sclass, sgender, sbirthday) VALUES ('lisi', 10, 'male', '2015-05-22');  
Query OK, 1 row affected (0.00 sec)
- ```
mysql> SELECT * FROM db1.students;
+-----+-----+-----+-----+-----+
| sid | sname | sclass | sgender | sbirthday |
+-----+-----+-----+-----+-----+
| 1 | zhangsan | 1 | male | 2015-05-20 |
| 2 | lisi | 10 | male | 2015-05-22 |
+-----+-----+-----+-----+-----+
2 rows in set (0.00 sec)
```
7. 新数据会被同步到GeminiDB实例中：
127.0.0.1:6379> KEYS *
1) "db1:students:sid:1"
2) "db1:students:sid:2"
- ```
127.0.0.1:6379> HGETALL db1:students:sid:2
1) "sbirthday"
2) "2015-05-22"
3) "sclass"
4) "10"
5) "sgender"
6) "male"
7) "sname"
```

## 8. 在 MySQL实例侧对students表执行更新数据操作：

```
mysql> UPDATE db1.students SET sclass=12, sname='wangwu' WHERE sid = 1;
Query OK, 1 row affected (0.00 sec)
Rows matched: 1 Changed: 1 Warnings: 0
```

```
mysql> SELECT * FROM db1.students;
+-----+-----+-----+-----+-----+
| sid | sname | sclass | sgender | sbirthday |
+-----+-----+-----+-----+-----+
| 1 | wangwu | 12 | male | 2015-05-20 |
| 2 | lisi | 10 | male | 2015-05-22 |
+-----+-----+-----+-----+-----+
2 rows in set (0.00 sec)
```

## 9. 在GeminiDB实例中查看数据被更新：

```
127.0.0.1:6379> KEYS *
1) "db1:students:sid:1"
2) "db1:students:sid:2"

127.0.0.1:6379> HGETALL db1:students:sid:1
1) "sbirthday"
2) "2015-05-20"
3) "sclass"
4) "12"
5) "sgender"
6) "male"
7) "sname"
8) "wangwu"
```

## 10. 在MySQL实例侧对students表执行删除数据操作：

```
mysql> DELETE FROM db1.students WHERE sid = 1;
Query OK, 1 row affected (0.00 sec)
```

```
mysql> SELECT * FROM db1.students;
+-----+-----+-----+-----+-----+
| sid | sname | sclass | sgender | sbirthday |
+-----+-----+-----+-----+-----+
| 2 | lisi | 10 | male | 2015-05-22 |
+-----+-----+-----+-----+-----+
1 row in set (0.00 sec)
```

## 11. 在 GeminiDB实例中查看数据被删除：

```
127.0.0.1:6379> KEYS *
1) "db1:students:sid:2"
```

### 4.16.1.3 修改和删除内存加速规则

创建内存加速映射后，可以实现MySQL实例到GeminiDB实例的数据自动同步。本章节主要介绍如何修改和删除内存加速规则。

#### 注意事项

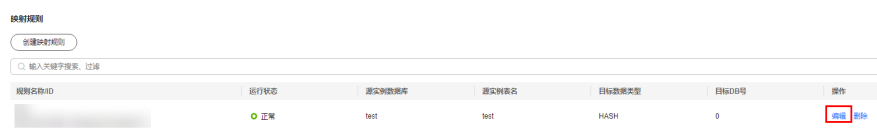
- 目前仅支持MySQL到GeminiDB Redis接口Hash类型的转换。
- 如果修改内存加速规则中MySQL实例的表名后，则需要重新配置内存加速规则。
- 暂不支持ENUM、SET、JSON三种数据类型的同步。
- 如果对加速规则中键（Key）字段中的一个或多个字段执行改名、删除等操作时，会使加速规则失效。

#### 修改映射规则

**步骤1** 登录[云数据库RDS控制台](#)。

- 步骤2 单击管理控制台左上角的，选择区域和项目。
- 步骤3 在“实例管理”页面，单击目标实例名称，进入“概览”页面。
- 步骤4 在左侧导航栏选择“内存加速”，在“映射规则”页面单击目标规则操作列的“编辑”。

图 4-225 映射规则



- 步骤5 编辑加速实例的域-值后，单击“提交”，完成映射规则修改。

图 4-226 编辑映射规则



----结束

删除映射规则

- 步骤1 登录[云数据库RDS控制台](#)。
- 步骤2 单击管理控制台左上角的，选择区域和项目。
- 步骤3 在“实例管理”页面，单击目标实例名称，进入“概览”页面。
- 步骤4 在左侧导航栏选择“内存加速”，在“映射规则”页面单击目标规则操作列的“删除”，即可完成映射规则删除。

----结束

4.16.1.4 查看和解除映射关系

用户可以在内存加速管理页面中查看映射列表，并可以解除映射关系。

使用须知

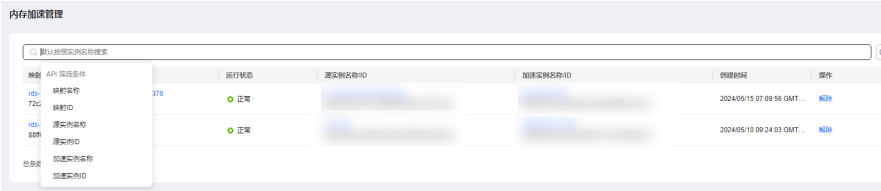
- 映射关系解除后，业务应用将无法从加速实例获取源库最新数据，请谨慎操作。
- 解除映射后，映射规则也会被清除。
- 源实例或加速实例运行状态不是正常时，不能解除映射。

查看映射列表

- 步骤1 登录[云数据库GeminiDB控制台](#)。

**步骤2** 在左侧导航栏选择“内存加速管理”，可以在“内存加速管理”页面查看全部映射关系，也可以在通过搜索关键字（如映射名称、映射ID等），找到具体的映射。

图 4-227 映射列表

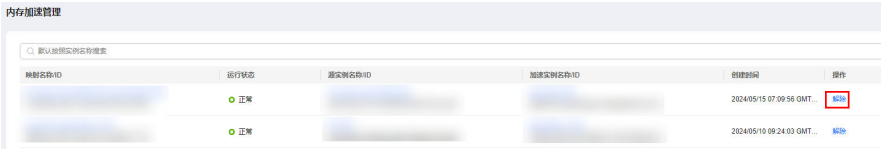


----结束

解除映射

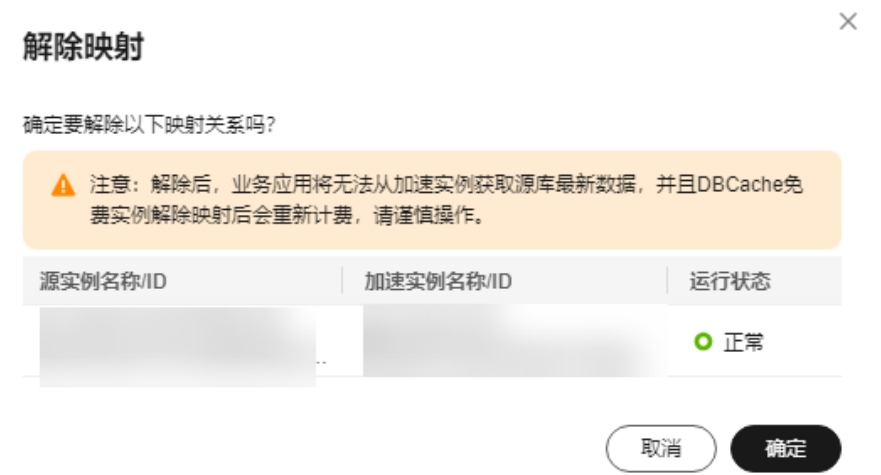
- 步骤1** 登录[云数据库GeminiDB控制台](#)。
- 步骤2** 在左侧导航栏选择“内存加速管理”，单击目标映射操作列的“解除”。

图 4-228 内存加速管理



**步骤3** 单击确认框的“确定”，即可完成映射关系解除。

图 4-229 解除映射



----结束

4.16.2 通过 GeminiDB Redis 实现 TaurusDB 内存加速

### 4.16.2.1 内存加速概述

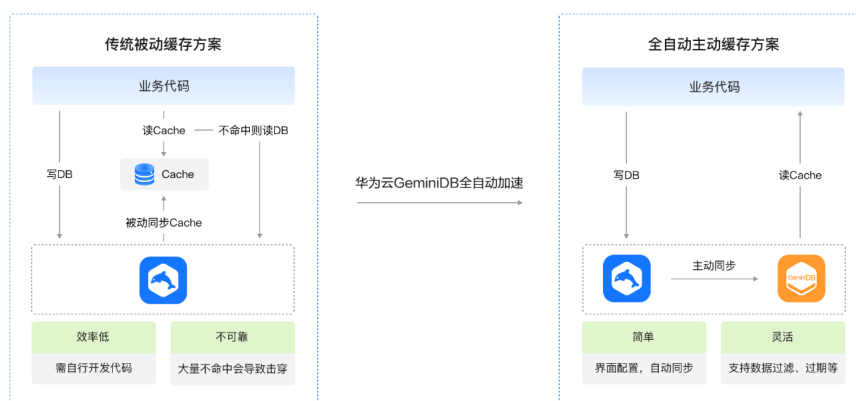
内存加速是GeminiDB Redis为了优化“传统被动缓存方案”而推出的功能，它可以让用户通过界面配置规则的形式，自动缓存TaurusDB的数据，加速TaurusDB的访问。

#### 说明

- 通过内存加速功能时所创建的GeminiDB Redis实例需单独收费，具体收费规则为：
  - 如果创建的GeminiDB Redis实例规格为16GB、24GB、32GB、48GB或64GB，则实例限期免费3个月，到期后按正常计费规则收费。
  - 其他规格的实例将直接按计费规则收费，计费详情请参见[计费说明](#)。
- 1个IAM账号在1个区域最多创建3个限期3个月免费试用的GeminiDB Redis实例（实例规格为16GB、24GB、32GB、48GB或64GB）。
- 支持创建的GeminiDB Redis实例规格变更，但是变更后GeminiDB Redis实例开始收费。
- 解除映射后，需要及时删除GeminiDB Redis实例，否则会对GeminiDB Redis实例收费。

如下图图4-230所示，“传统被动缓存方案”需要用户自行开发代码把TaurusDB中的数据写入到缓存中，存在效率低、不可靠的缺点。而采用云数据内存加速的“全自动主动缓存方案”，解决“缓存击穿”问题，防止交易雪崩。支持界面可视化配置，配置完成后即可实现数据自动同步。同时还支持数据过滤及过期等功能，极大提高了开发效率及数据的可靠性。

图 4-230 内存加速实现原理



### 4.16.2.2 开启和使用内存加速

本章节主要介绍如何开启内存加速功能，主要流程如下：

**步骤1：选取GeminiDB实例**

**步骤2：创建映射**

**步骤3：使用示例**

#### 注意事项

- 目前该功能处于公测阶段，您可以在管理控制台右上角，选择“[工单 > 新建工单](#)”联系客服进行咨询。
- 内存加速开启后，不允许在TaurusDB实例执行RESET MASTER、FLUSH LOGS等删除binlog的命令。

- 新规则的Redis键前缀+键分隔符不能是已有规则的Redis键前缀+键分隔符的子前缀，反之亦然。例如新规则的前缀为 "pre1:"，键分隔符为 ":"，老规则前缀为 "pre1"，分隔符为 "."，这种情况不允许创建新规则。
- 暂不支持ENUM、SET、JSON三种数据类型的同步。
- 由于是轻量级增量数据同步，目前仅支持单表查询，暂不支持联合查询。
- 目前仅GeminiDB Redis实例收费，该功能没有其他额外费用。
- 您购买TaurusDB实例时，如果“内存加速”参数，选择了“立即购买”，则会自动购买GeminiDB实例搭建DBcache，您可以跳过创建GeminiDB实例步骤，直接从[创建映射](#)开始使用。

## 步骤 1：选取 GeminiDB 实例

- 步骤1** 登录[云数据库TaurusDB控制台](#)。
- 步骤2** 单击管理控制台左上角的，选择区域和项目。
- 步骤3** 在“实例管理”页面，单击目标实例名称，进入“实例概览”页面。
- 步骤4** 在左侧导航栏选择“内存加速”，单击“创建GeminiDB实例”或者“选择已有GeminiDB实例”。
- 单击“创建GeminiDB实例”，并执行[步骤5](#)。
  - 单击“选择已有GeminiDB实例”选择已有的GeminiDB Redis实例。  
注意：选择“选择已有GeminiDB实例”时，当前仅支持主备版类型实例。
- 步骤5** 参见[表4-85](#)选择并填写实例相关信息后，单击“提交”，完成实例创建。

### 说明

Region、可用区、VPC，子网等参数默认与当前TaurusDB实例一致。

表 4-85 基本信息

| 参数           | 描述                                                                                                                                                                                                                 |
|--------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| GeminiDB实例规格 | 实例的CPU和内存，详细规格见 <a href="#">表4-86</a> 。                                                                                                                                                                            |
| 数据库端口        | 数据库的访问端口号。<br>GeminiDB Redis支持自定义端口号，取值范围为：1024~65535，其中，如下端口号为禁用端口：2180、2887、3887、6377、6378、6380、8018、8079、8091、8479、8484、8999、12017、12333、50069。<br>当您不指定端口时，创建实例的访问端口默认为6379。<br><b>说明</b><br>暂时不支持实例创建成功后修改端口。 |

| 参数    | 描述                                                                                                                                                                                              |
|-------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 实例名称  | 设置实例名称时，需要满足如下规则。 <ul style="list-style-type: none"><li>实例名称允许和已有名称重复。</li><li>实例名称长度在4个到64个字节之间，必须以字母或中文字开头，区分大小写，可以包含字母、数字、中划线、下划线或中文（一个中文字符占用3个字节），不能包含其他特殊字符。</li></ul>                     |
| 数据库密码 | 用户设置的数据库密码。 <ul style="list-style-type: none"><li>长度为8～32个字符。</li><li>必须是大写字母、小写字母、数字、特殊字符中至少两种的组合，其中可输入~!@#%^*-_+=?特殊字符。</li><li>系统会进行弱密码校验，安全起见，请输入高强度密码。请妥善管理您的密码，因为系统将无法获取您的密码信息。</li></ul> |
| 确认密码  | 必须和数据库密码一致。                                                                                                                                                                                     |

表 4-86 GeminiDB 实例规格

| 实例存储容量<br>( GB ) | 节点数量 | vCPU<br>( 个 ) | QPS基<br>准 | 单节点实例的最大<br>连接数 | 实例DB数 |
|------------------|------|---------------|-----------|-----------------|-------|
| 16               | 2    | 2             | 10,000    | 10,000          | 1,000 |
| 24               | 2    | 2             | 40,000    | 10,000          | 1,000 |
| 32               | 2    | 2             | 40,000    | 10,000          | 1,000 |
| 48               | 2    | 4             | 80,000    | 20,000          | 1,000 |
| 64               | 2    | 4             | 80,000    | 20,000          | 1,000 |
| 96               | 2    | 8             | 160,000   | 20,000          | 1,000 |
| 128              | 2    | 16            | 320,000   | 20,000          | 1,000 |

----结束

步骤 2：创建映射

- 步骤1 登录[云数据库TaurusDB控制台](#)。
- 步骤2 单击管理控制台左上角的，选择区域和项目。
- 步骤3 在“实例管理”页面，单击目标实例名称，进入“实例概览”页面。
- 步骤4 在左侧导航栏选择“内存加速”，在“映射规则”区域单击“创建映射规则”。



图 4-231 映射规则

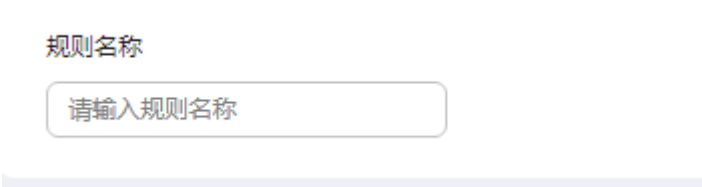


**步骤5** 在“创建映射规则”页面，填写并选择相关信息。

1. 配置规则名称。

规则名称：设置映射规则名称。规则名称不能包含#和中文，长度不超过256个字符，且规则名称在同一个GeminiDB实例中不可重复。

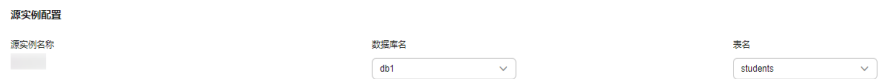
图 4-232 规则名称



2. 源实例配置。

- 数据库名：选择需要加速实例的数据库。
- 表名：选择需要加速实例中的表。

图 4-233 源实例配置



3. 加速实例端配置。

- Redis键前缀：可选。默认的前缀格式为“数据库名:数据表名:字段名1:字段名2...”，长度不超过1024个字符。如果用户自定义了Redis键前缀，那么以自定义前缀为准。
- 值存储类型：缓存的数据类型，当前仅支持Hash。
- DB号（0-999）：加速实例中保存缓存数据的DB号，默认为0。
- TTL（单位：秒，默认值为30天。）：加速实例中缓存数据的过期时间，默认30天（2592000秒），输入-1为不过期。
- 键分隔符：Redis键前缀与键（Key）、以及键（Key）各字段之间的分隔符，长度为1个字符。

图 4-234 加速实例端配置



4. 单击“键（Key）设置”，选择加速实例的键（Key）字段，单击“确定”。

 说明

如果加速实例键（Key）由多个源实例字段组成，那么需要具备唯一性（这些字段在TaurusDB实例中要能成功创建出唯一索引）；可以单击“上移”和“下移”调整各字段在键（Key）中的先后顺序。

图 4-235 键（Key）设置



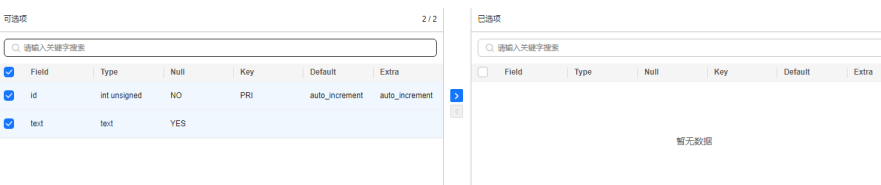
设置完成后，会在界面中显示键（Key）的构成方式，如下图所示：

图 4-236 键（Key）



5. 配置加速实例的域-值。  
将源实例中需要的字段，右移到加速实例的字段中。

图 4-237 配置加速实例字段



6. 配置信息填写完成后，单击“提交”，完成映射规则创建。
- 结束

步骤 3：使用示例

1. 在源实例TaurusDB中创建一个db1数据库，并在db1中创建一个students表，具体的 SQL 如下：
- ```
mysql> CREATE DATABASE db1;
Query OK, 1 row affected (0.00 sec)

mysql> CREATE TABLE db1.students(
  sid INT UNSIGNED PRIMARY KEY AUTO_INCREMENT NOT NULL,
  sname VARCHAR(20),
  sclass INT,
  sgender VARCHAR(10),
  sbirthday DATE
);
Query OK, 0 rows affected (0.00 sec)

mysql> DESC db1.students;
+-----+-----+-----+-----+-----+-----+
| Field | Type      | Null | Key | Default | Extra      |
+-----+-----+-----+-----+-----+-----+
| sid    | int unsigned | NO   | PRI | NULL    | auto_increment |
| sname  | varchar(20) | YES  |     | NULL    |              |
| sclass | int         | YES  |     | NULL    |              |
| sgender | varchar(10) | YES  |     | NULL    |              |
| sbirthday | date      | YES  |     | NULL    |              |
+-----+-----+-----+-----+-----+-----+
5 rows in set (0.00 sec)
```
2. 建表后，在“内存加速”界面，创建一条映射规则，将students表中的每一行转换为一个Redis Hash。该Hash的键（Key）由”数据库名:数据表名:sid:<sid的值>”构成，域-值由选中的“sname, sclass, sgender, sbirthday”构成。

图 4-238 配置映射规则

规则名称

students-to-hash

源实例配置

源实例名称

数据库名

表名

加速实例配置

加速实例名称

Redis键前缀

键存储类型

DB号 (0-999)

TTL (单位: 秒, 默认为365天)

键分隔符

键 (Key) 设置

Hash db1:students:sid:<sid>

值 (value) 设置

可选域

已选域

3. 映射规则创建成功后，可以查看映射规则及映射信息。

图 4-239 映射信息



4. 在TaurusDB实例侧对students表插入一条新数据：
mysql> INSERT INTO db1.students (sname, sclass, sgender, sbirthday) VALUES ('zhangsan', 1, 'male', '2015-05-20');
Query OK, 1 row affected (0.01 sec)
- ```
mysql> SELECT * FROM db1.students;
+-----+-----+-----+-----+-----+
| sid | sname | sclass | sgender | sbirthday |
+-----+-----+-----+-----+-----+
| 1 | zhangsan | 1 | male | 2015-05-20 |
+-----+-----+-----+-----+-----+
1 row in set (0.00 sec)
```
5. 由于映射规则的存在，会把该数据自动同步到GeminiDB实例中。此时在GeminiDB实例中执行命令即可查询到该数据：  
127.0.0.1:6379> KEYS \*  
1) "db1:students:sid:1"
- ```
127.0.0.1:6379> HGETALL db1:students:sid:1
1) "sbirthday"
2) "2015-05-20"
3) "sclass"
4) "1"
5) "sgender"
6) "male"
7) "sname"
8) "zhangsan"
```
6. 在TaurusDB实例侧对students表再次插入一条新数据：
mysql> INSERT INTO db1.students (sname, sclass, sgender, sbirthday) VALUES ('lisi', 10, 'male', '2015-05-22');
Query OK, 1 row affected (0.00 sec)
- ```
mysql> SELECT * FROM db1.students;
+-----+-----+-----+-----+-----+
| sid | sname | sclass | sgender | sbirthday |
+-----+-----+-----+-----+-----+
| 1 | zhangsan | 1 | male | 2015-05-20 |
| 2 | lisi | 10 | male | 2015-05-22 |
+-----+-----+-----+-----+-----+
2 rows in set (0.00 sec)
```
7. 新数据会被同步到GeminiDB实例中：  
127.0.0.1:6379> KEYS \*  
1) "db1:students:sid:1"  
2) "db1:students:sid:2"
- ```
127.0.0.1:6379> HGETALL db1:students:sid:2
1) "sbirthday"
2) "2015-05-22"
3) "sclass"
4) "10"
5) "sgender"
6) "male"
7) "sname"
```

8. 在 TaurusDB实例侧对students表执行更新数据操作:

```
mysql> UPDATE db1.students SET sclass=12, sname='wangwu' WHERE sid = 1;  
Query OK, 1 row affected (0.00 sec)  
Rows matched: 1 Changed: 1 Warnings: 0
```

```
mysql> SELECT * FROM db1.students;  
+-----+-----+-----+-----+-----+  
| sid | sname | sclass | sgender | sbirthday |  
+-----+-----+-----+-----+-----+  
| 1 | wangwu | 12 | male | 2015-05-20 |  
| 2 | lisi | 10 | male | 2015-05-22 |  
+-----+-----+-----+-----+-----+  
2 rows in set (0.00 sec)
```

9. 在 GeminiDB实例中查看数据被更新:

```
127.0.0.1:6379> KEYS *  
1) "db1:students:sid:1"  
2) "db1:students:sid:2"  
  
127.0.0.1:6379> HGETALL db1:students:sid:1  
1) "sbirthday"  
2) "2015-05-20"  
3) "sclass"  
4) "12"  
5) "sgender"  
6) "male"  
7) "sname"  
8) "wangwu"
```

10. 在 TaurusDB实例侧对students表执行删除数据操作:

```
mysql> DELETE FROM db1.students WHERE sid = 1;  
Query OK, 1 row affected (0.00 sec)
```

```
mysql> SELECT * FROM db1.students;  
+-----+-----+-----+-----+-----+  
| sid | sname | sclass | sgender | sbirthday |  
+-----+-----+-----+-----+-----+  
| 2 | lisi | 10 | male | 2015-05-22 |  
+-----+-----+-----+-----+-----+  
1 row in set (0.00 sec)
```

11. 在 GeminiDB实例中查看数据被删除:

```
127.0.0.1:6379> KEYS *  
1) "db1:students:sid:2"
```

4.16.2.3 修改和删除内存加速规则

创建内存加速映射后, 可以实现TaurusDB实例到GeminiDB实例的数据自动同步。本章节主要介绍如何修改和删除内存加速规则。

注意事项

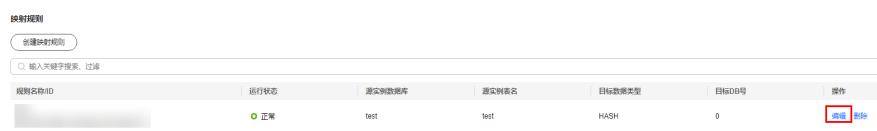
- 目前仅支持TaurusDB到GeminiDB Redis接口Hash类型的转换。
- 如果修改内存加速规则中TaurusDB实例的表名后, 则需要重新配置内存加速规则。
- 暂不支持ENUM、SET、JSON三种数据类型的同步。
- 如果对加速规则中键 (Key) 字段中的一个或多个字段执行改名、删除等操作时, 会使加速规则失效。

修改映射规则

步骤1 登录[云数据库TaurusDB控制台](#)。

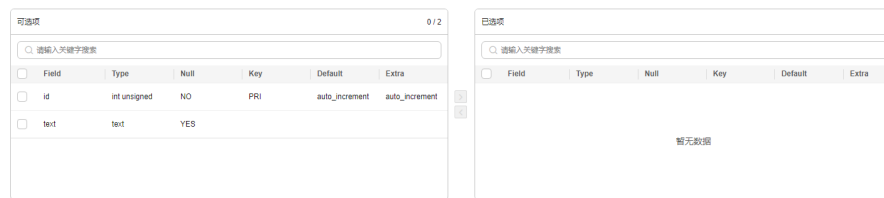
- 步骤2 单击管理控制台左上角的，选择区域和项目。
- 步骤3 在“实例管理”页面，单击目标实例名称，进入“实例概览”页面。
- 步骤4 在左侧导航栏选择“内存加速”，在“映射规则”页面单击目标规则操作列的“编辑”。

图 4-240 映射规则



- 步骤5 编辑加速实例的域-值后，单击“提交”，完成映射规则修改。

图 4-241 编辑映射规则



----结束

删除映射规则

- 步骤1 登录[云数据库TaurusDB控制台](#)。
- 步骤2 单击管理控制台左上角的，选择区域和项目。
- 步骤3 在“实例管理”页面，选择目标实例名称。
- 步骤4 在左侧导航栏选择“内存加速”，在“映射规则”页面单击目标规则操作列的“删除”，即可完成映射规则删除。

----结束

4.16.2.4 查看和解除映射关系

用户可以在内存加速管理页面中查看映射列表，并可以解除映射关系。

使用须知

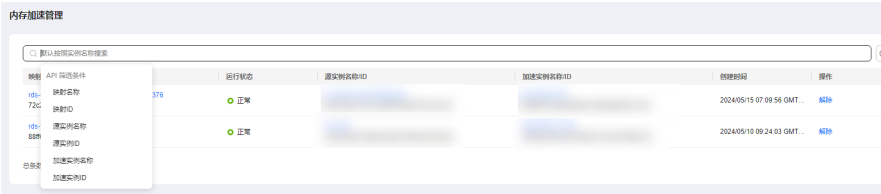
- 映射关系解除后，业务应用将无法从加速实例获取源库最新数据，请谨慎操作。
- 解除映射后，映射规则也会被清除。
- 源实例或加速实例运行状态不是正常时，不能解除映射。

查看映射列表

- 步骤1 登录[云数据库GeminiDB控制台](#)。

步骤2 在左侧导航栏选择“内存加速管理”，可以在“内存加速管理”页面查看全部映射关系，也可以在通过搜索关键字（如映射名称、映射ID等），找到具体的映射。

图 4-242 映射列表

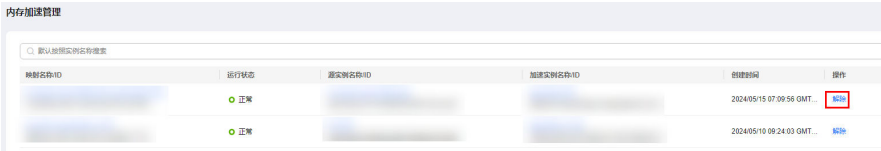


----结束

解除映射

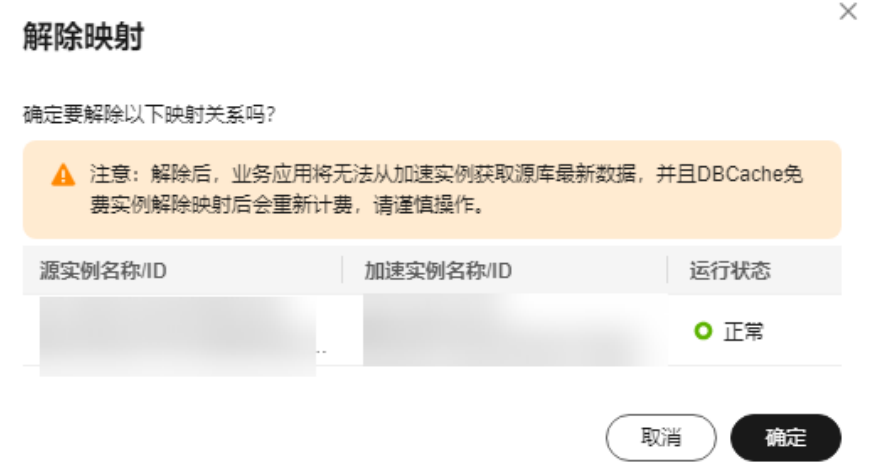
- 步骤1** 登录[云数据库GeminiDB控制台](#)。
- 步骤2** 在左侧导航栏选择“内存加速管理”，单击目标映射操作列的“解除”。

图 4-243 内存加速管理



步骤3 单击确认框的“确定”，即可完成映射关系解除。

图 4-244 解除映射



----结束

5 开发参考

5.1 开发运维规范

本章节结合华为云数据库开发运维经验，从开发和运维角度为您提供GeminiDB Redis产品使用规范。根据这些建议，您可以有效评估以及改善业务系统稳定性。

开发规范

在开发业务程序时需重点关注以下开发规范，避免因使用方式不当影响业务稳定。

表 5-1 GeminiDB Redis 开发规范

编号	开发规范	说明
1	业务程序需具备合理的自动重连机制。	在规格变更、补丁升级、HA倒换、网络链路抖动或丢包等场景中，业务程序与数据库实例的连接可能出现短暂中断。业务程序应当支持自动重连。 说明 尽量避免使用Lettuce客户端，因为该客户端在请求多次超时后，不再发起自动重连。建议使用更稳定的Jedis客户端替换Lettuce客户端。
2	业务程序需使用连接池并配置足够连接数。	为避免并发激增时程序获取不到连接，建议业务程序使用连接池，并配置合理的连接池参数。客户端的连接池推荐配置请参考客户端连接池参数配置。
3	针对部分重要操作，业务程序需要具备合理的命令重试机制。	在连接中断或请求超时等场景，业务程序请求可能会短时内执行失败。因此，针对部分重要操作需要设计业务容错机制，使用合理的命令重试间隔和次数，确保重要数据写入或修改成功。

编号	开发规范	说明
4	使用正确的高可用连接地址，避免单点故障场景影响业务。	业务程序内网连接数据库时，应使用负载均衡地址，实现高可用。应避免将业务程序直连到某个单独的计算节点。 说明 针对公网访问数据库的场景，同样应避免将业务程序直连到某个单独的计算节点，请参考通过弹性负载均衡绑定弹性公网IP连接实例使用外置ELB进行配置。
5	使用连接池，避免使用单连接以及大量短连接。	由于单连接有高可用风险，而短连接性能差且会消耗大量数据库CPU和网络资源，容易造成瓶颈，因此建议业务程序使用主流SDK的连接池连接GeminiDB Redis实例。
6	当实例中Key数量大于1000个时，禁止执行Keys命令	Keys命令是典型的高危命令，会一次性获取整个实例中的全部数据内容并返回给客户端。当实例中Key数量较多时，执行Keys命令会引起请求阻塞或实例状态异常。
7	避免元素Value过大，避免单Key包含的元素数量过多。	建议String类型Key的Value小于10KB，Hash/List/Zset/Set等类型Key中包含元素数小于5000，且单个元素Value小于1KB。 说明 与社区Redis一样，GeminiDB Redis对大Key存储并不做严格约束，因此需要用户根据规范合理开发业务程序。
8	避免单个命令中一次性操作过多元素或产生过大的网络收发包。	- 建议MSET/MGET等批量操作Key命令中同时操作的Key数量小于1000。 - 建议HMSET、HGETALL、LRANGE、ZADD、ZRange等同时操作Key中多元素的命令同时操作元素数量小于1000。 - 由于社区Redis语法中ZREMRANGEBYSCORE命令不提供limit参数，无法控制单次删除的元素数量，建议使用ZRANGEBYSCORE (带limit) +ZREM的先查后删方式替代。 说明 与社区Redis一样，GeminiDB Redis对大Key访问并不做严格约束，因此需要用户根据规范合理开发业务程序。

编号	开发规范	说明
9	合理打散Key分布，避免出现热点Key或热点hashtag瓶颈。	对单个Key或具有同hashtag的一组Key的高频访问依赖会产生热Key问题，容易引发计算资源倾斜，请求排队，响应变慢等瓶颈问题。热Key的产生往往来自业务设计对Key的拆分程度不足，需要业务拆分优化。 说明 反面案例： - 对全局超大排行榜高频集中访问。 - 全实例中仅保存几个Hash Key，每个Key用于存储一整张表格的信息量。 - 商品库存业务中，全实例Key仅使用少量hashtag分桶，导致热点hashtag查询排队。
10	单次Pipeline中打包命令数量建议小于100条。	使用Pipeline时应遵循“单次少量，分多次”原则。 根据使用经验，并非Pipeline打包数量越大性能越好，不合理的Pipeline用法将可能导致实例CPU、带宽等资源瓶颈，甚至引起请求阻塞。 说明 与社区Redis相比，GeminiDB Redis对Pipeline的使用并不做严格约束，因此需要用户根据规范合理开发业务程序。
11	避免在Lua脚本中引入高耗时代码。	使用Lua脚本时应遵循轻量化原则，避免引入执行命令次数过多、长时间sleep、大的循环语句等不合理用法。
12	避免在事务中打包过多命令。	使用事务时应遵循轻量化原则，避免单次事务中打包过多命令或复杂度过高的命令。执行的事务中包含命令过多会引起请求阻塞或实例状态异常。
13	避免在不同数据类型下使用相同的Key名称。	社区Redis强制禁止在不同数据类型下使用相同的Key名称。虽然GeminiDB Redis对此不做强制约束，但开发时应当避免此类用法，保持程序清晰和易维护。

编号	开发规范	说明
14	谨慎使用批量删除命令	若针对一个包含大量元素的LIST/ZSET对象使用批量删除命令（例如LREM/LTRIM/ZREMRANGEBYSCORE等），要避免在单条命令中删除过多的数据（比如删除数万或者数十万元素）。过多的数据删除可能导致该条删除命令耗时过长，影响其它正常命令，或者导致系统内存OOM进程重启，实例异常；极端场景下，造成服务进程反复拉起失败。
15	scan返回的cursor使用64位无符号类型进行解析	和开源Redis保持一致，GeminiDB使用64位无符号类型对cursor进行编码，因此，对返回的cursor用户需要使用64位无符号类型进行解析。否则可能会解析错误，错误解析结果输入到GeminiDB可能返回“invalid cursor”的报错。
16	使用redis inline时，命令长度需要小于64KB	和开源Redis行为保持一致，为避免OOM等风险，超过64KB输入服务端会返回错误“ERR Protocol error: too big inline request”。
17	尽量减少subscribe族、阻塞命令（brpop等）命令的使用，避免服务端连接资源消耗殆尽，无法接受新的连接	proxy到shard通信会维护一套共享连接池，普通命令会复用这些连接。对于阻塞命令（brpop等）、subscribe族命令使用，proxy需要创建到shard的独享连接，造成单个shard承载的连接数显著增加，最终可能会突破单进程的连接数限制，无法建立新的连接。因此，我们建议在单个proxy上使用阻塞命令或subscribe族命令的客户端连接小于1000，避免出现上述情况。

运维规范

在日常运维工作中需关注以下运维规范，预防潜在风险并掌握关键应急方案。

表 5-2 GeminiDB Redis 运维规范

编号	运维规范	说明
1	确保您的华为云账号绑定的电话号码、电子邮箱等联系方式有效，以便及时收到服务相关通知信息。	华为云服务会在变更、升级、故障通知等必要场景通过网站、邮箱、短信、站内信等方式向您送达通知信息，请确保账号绑定的联系方式有效。

编号	运维规范	说明
2	订阅重要告警。	- 订阅例如大Key访问、存储容量高、连接数使用率高、CPU使用率高等告警，从而在数据库实例出现风险时提早发现并及时处理。 - 推荐告警配置请参考设置告警规则。
3	使用负载均衡地址访问时，需要配置访问控制管理，而不是安全组。	负载均衡地址不支持安全组，创建完成后请配置IP访问控制，如未配置白名单，所有与VPC互通的IP地址均可访问实例。具体操作请参考 给GeminiDB Redis实例设置负载均衡内网访问控制 。
4	建议预留50%以上存储空间，避免实例进入只读。	GeminiDB采用rocksdb作为存储引擎，数据会定期compact以优化读性能。compact期间，会产生临时文件造成存储空间短暂上涨。与业务模型相关，最坏情况可能会上涨2倍，磁盘存储相对便宜，建议预留50%空间。
5	配置实例自动扩容。	GeminiDB Redis支持自动扩容，可以有效避免数据量突增问题。建议参见 自动扩容实例磁盘 开启自动扩容。
6	保持负载处于健康水位。	- 如业务数据量长期高于80%，建议及时扩容。 - 如业务流量超过实例QPS基准，或CPU利用率长期高于80%，建议及时升级规格或增加节点，避免过载影响业务访问。 - 在业务流量、连接数等指标突增导致实例算力资源超载的紧急场景，应通过水平添加节点快速提升集群算力。垂直升配CPU规格为滚动模式，耗时较长，不建议用于紧急场景。
7	重命名高危命令。	对于高危命令（如flushall、keys），建议您禁用或重命名，以增强实例的安全性。具体操作请参考文档 命令重命名 。
8	慎用账号管理中的flushall操作。	- 具有读写权限的子账号在执行flushall时，会清空实例所有的数据。 - 请禁用flushall操作，确认DB后可执行flushdb操作（慎用）。

编号	运维规范	说明
9	定期进行大key在线诊断。	大Key访问是Redis使用中的常见问题。建议在控制台定期查看实例的大Key诊断报告。具体操作请参考 大Key诊断 。
10	割接结束后执行dbsize。	dbsize可以保证最终一致性，例如，在割接场景，无过期key，数据导入完成后的的数分钟执行，可得到准确值，确保数据导入前后的一致。

5.2 命令兼容列表

GeminiDB（Redis接口）提供集群版和主备版，同时兼容社区Redis 5.0及以下版本（100%）、Redis 6.2(包含6.2.x)、Redis 7.0命令，应用上云无需修改代码。

本章节详细介绍GeminiDB（Redis 接口）Redis 5.0、Redis 6.2(包含6.2.x)、Redis 7.0命令兼容性，供DBA和开发参考。

GeminiDB 基础命令

针对社区Redis提供的全部命令类型，GeminiDB 5.0、6.2(包含6.2.x)和7.0兼容情况如下：

表 5-3 GeminiDB Redis 接口命令兼容度

Redis 命令分类	命令详情	GeminiDB Redis 5.0兼容情况	GeminiDB Redis 6.2(包含6.2.x)兼容情况	GeminiDB Redis 7.0兼容情况
String	String命令列表	100%兼容	100%兼容	100%兼容
Hash	Hash命令列表	100%兼容	100%兼容	100%兼容
List	List命令列表	100%兼容	100%兼容	100%兼容
Sorted set	Sorted set命令列表	100%兼容	100%兼容	100%兼容

Redis命令分类	命令详情	GeminiDB Redis 5.0兼容情况	GeminiDB Redis 6.2(包含 6.2.x)兼容情况	GeminiDB Redis 7.0兼容情况
Set	Set命令列表	100%兼容	100%兼容	100%兼容
Bitmap	Bitmap命令列表	100%兼容	100%兼容	100%兼容
Stream	Stream命令列表	100%兼容	暂不支持xgroup子命令create consumer	暂不支持7.0特性
Geo	Geo命令列表	100%兼容	100%兼容	100%兼容
Hyperloglog	Hyperloglog命令列表	100%兼容	100%兼容	100%兼容
发布订阅 (Pub/ Sub)	发布订阅命令列表	100%兼容	100%兼容	100%兼容
Lua脚本 (Scripting)	Lua脚本命令列表	100%兼容	100%兼容	暂不支持7.0特性
事务 (Transactions)	事务命令列表	100%兼容	100%兼容	100%兼容
其他通用Key操作	其他通用Key操作	暂不支持2个命令： SWAP DB、 MOVE	暂不支持2个命令： SWAP DB、 MOVE	100%兼容

GeminiDB Redis兼容支持部分Cluster命令族的命令，具体为CLUSTER INFO、CLUSTER KEYSLOT、CLUSTER MYID、CLUSTER NODES、CLUSTER SLOTS、CLUSTER SHARDS、CLUSTER HELP。

GeminiDB 高级命令

- ExHash类型
 - 使用场景：GeminiDB支持为Hash Key的每个Field设置过期时间，适用于构建频控、购物车等业务。
 - 命令列表：请参考[ExHash命令列表](#)。
 - 使用方式：请参考[广告频控业务ExHash方案](#)。
- 布隆过滤器（BloomFilter）类型
 - 功能介绍：布隆过滤器（BloomFilter）可用于检索某个元素是否存在于某个海量集合中，适用于Web拦截器、防缓存穿透等场景。
 - 命令列表：请参考[布隆过滤器（Bloom Filter）介绍](#)。
 - 使用方式：请参考[布隆过滤器（Bloom Filter）命令说明](#)。

GeminiDB 运维管理命令

GeminiDB在社区Redis的INFO/CLIENT/SLOWLOG/MONITOR/CONFIG等运维管理命令能力基础上，提供全面服务化且高度易用的DBA运维能力。

表 5-4 GeminiDB Redis 接口运维功能介绍

GeminiDB运维功能	GeminiDB运维功能简介	社区Redis同类命令/能力
监控指标	提供命令级别QPS/平均时延/P99时延指标。提供丰富节点级指标以及实例级汇聚指标。	INFO
实例会话	展示客户端IP信息以及Top会话统计，支持一键批量Kill会话。	CLIENT
参数配置	支持一键查询与修改内核参数。是社区Redis CONFIG命令的能力加强版。	CONFIG
慢日志	展示慢日志及诊断信息。	SLOWLOG
用户权限管理	支持账号管理功能。	ACL (Access Control List)
审计日志	审计高危命令操作。	MONITOR
实时大Key诊断	实时观测大Key信息，不影响业务。	不支持
实时热Key诊断	实时观测热Key情况，不影响业务。	不支持

GeminiDB运维功能	GeminiDB运维功能简介	社区Redis同类命令/能力
实时前缀Key分析	实时观测前缀Key分布，不影响业务。	不支持
高危命令重命名	支持修改命令别名以及查看重命名记录。	通过手动修改配置文件实现
异常Key熔断	一键屏蔽指定Key，拒绝被业务访问。用于快速恢复。	不支持

其它运维命令

为便于客户管理实例，GeminiDB Redis提供了一组运维命令，用于在客户特定场景业务下使用。

注意

以下运维命令执行前，建议您在管理控制台右上角，选择“[工单 > 新建工单](#)”，联系客服进行咨询，避免错误操作导致业务受影响。

- rocksconf 命令：
语法：ROCKSCONF SETOPTION/ SETDBOPTION *item value*
用途：
 - 根据业务的不同，可以使用该命令调整存储引擎RocksDB的参数，以适配业务需求。
 - item*是要调整的参数，*value*是需要设置的值, 具体可调整的值请查阅RocksDB 资料。
- compact 命令
语法：COMPACT *type partition_id*
用途：
 - 触发一次compact任务, 可以有效减少RocksDB墓碑，提高读性能和减少空间放大。
 - type*表示需要compact哪个数据类型，可选值：string, hash, zset, set, list。
 - partition_id*表示需要对哪个数据分片执行compact命令, 可以参考[通过负载均衡地址连接实例（推荐）](#)使用“info partitions”命令获取实例的分片信息。如下[图1 info partitions](#)，“179b5efc712a506c”就是一个合法的partition_id

图 5-1 info partitions

```
127.0.0.1:8635> info partitions
# Partitions
 0 | 179b5efc712a506c
 1 | 2e0f4b2bfdd185d5
 2 | 34dbac8a70931ae1
 3 | 3f9a19937025e813
 4 | 4de6802ed5750903
 5 | 799d54fd3f336e87
 6 | 9670d213ced6281d
 7 | cc9c568bcd3d948
 8 | d8d240b825327f85
 9 | dd73b20598cc3886
10 | dee0fe0aaa1d2f58
11 | f3602d53c8af4527
```

其它注意事项

- GeminiDB Redis暂不支持RESP3协议：RESP（Redis Serialization Protocol）是Redis服务端与客户端之间通信的协议，主流客户端Jedis等默认使用RESP2协议，使用客户端默认模式可正常访问GeminiDB Redis。
- GeminiDB Redis 7.0暂不支持function功能。

5.3 通过程序代码连接实例示例

5.3.1 通过 Jedis 连接实例

本章节介绍使用Jedis（JAVA客户端）访问GeminiDB Redis实例的方法。

由于GeminiDB Redis的“Proxy集群”架构提供了统一的负载均衡地址，并提供高可用能力，因此推荐使用JedisPool单机模式轻松接入。

此外，GeminiDB Redis对JedisSentinelPool、JedisCluster两种模式也提供支持。

前提条件

- 已成功创建GeminiDB Redis实例，且实例状态正常。创建GeminiDB Redis实例的方法请参见[购买GeminiDB Redis集群版实例](#)。
- 已创建弹性云服务器，创建弹性云服务器的方法，请参见《弹性云服务器快速入门》中“[购买弹性云服务器](#)”章节。
- 弹性云服务器上已经安装GCC等编译工具。
- 创建的弹性云服务器与GeminiDB Redis实例要保证区域、可用区、VPC和安全组一致。

pom 文件依赖

```
<dependency>
  <groupId>redis.clients</groupId>
  <artifactId>jedis</artifactId>
  <version>4.3.2</version>
</dependency>
<dependency>
```

```
<groupId>org.springframework.boot</groupId>
<artifactId>spring-boot-starter-data-redis</artifactId>
<version>2.3.6.RELEASE</version>
</dependency>
```

使用 JedisPool 访问（推荐）

样例代码

```
import org.apache.commons.pool2.impl.GenericObjectPoolConfig;
import redis.clients.jedis.Jedis;
import redis.clients.jedis.JedisPool;

public class JedisPoolTests {

    private static void testPool() {
        // 认证用的用户名和密码直接写到代码中有很大的安全风险，建议在配置文件或者环境变量中存放(密码应密文存放、使用时解密)，确保安全；
        // 本示例以用户名和密码保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量(环境变量名称请根据自身情况进行设置)EXAMPLE_USERNAME_ENV和EXAMPLE_PASSWORD_ENV。
        String pwd = System.getenv("EXAMPLE_PASSWORD_ENV");
        JedisPool pool = new JedisPool(new GenericObjectPoolConfig(), "172.xx.xx.xx", 6379,
            2000, pwd);
        Jedis jedis = pool.getResource();
        try {
            System.out.println(jedis.hgetAll("676296"));
            System.out.println(jedis.set("key1", "value1"));
        } finally {
            jedis.close();
        }
        pool.destroy();
    }

    public static void main(String[] args) {
        testPool();
    }
}
```

- 上述代码中“172.xx.xx.xx”为待连接GeminiDB Redis实例的负载均衡地址。
您可以单击实例名称，进入“基本信息”页面，在网络信息区域获取“负载均衡地址”。

图 5-2 查看负载均衡地址



- 上述代码中的“6379”为待连接GeminiDB Redis实例的端口，具体请以实际端口为准。获取端口的方法请参见[查看GeminiDB Redis IP地址和端口](#)。
- GeminiDB Redis支持与限制的命令，详情请参见[开发运维规范](#)。
- 由于GeminiDB Redis和开源Redis集群计算hash算法不同，因此GeminiDB Redis的部分命令需要给对应的key打上hashtag，否则会出现非预期的行为。hashtag使用建议参考[开发运维规范](#)。

使用 JedisCluster 访问

样例代码

```
import org.apache.commons.pool2.impl.GenericObjectPoolConfig;
import redis.clients.jedis.HostAndPort;
import redis.clients.jedis.JedisCluster;

public class ClusterTests {

    private static void testCluster() {
        String pwd = "a";
        JedisCluster cluster = new JedisCluster(new HostAndPort("172.xx.xx.xx", 6379),
            200, 2000, 5, pwd, new GenericObjectPoolConfig());
        System.out.println(cluster.hgetAll("676296"));
        System.out.println(cluster.set("key1", "value1"));
    }

    public static void main(String[] args) {
        testCluster();
    }
}
```

- 上述代码中“172.xx.xx.xx”为待连接GeminiDB Redis实例的负载均衡地址。您可以单击实例名称，进入“基本信息”页面，在网络信息区域获取“负载均衡地址”。

图 5-3 查看负载均衡地址



- 上述代码中的“6379”为待连接GeminiDB Redis实例的端口，具体请以实际端口为准。获取端口的方法请参见[查看GeminiDB Redis IP地址和端口](#)。
- GeminiDB Redis支持与限制的命令，详情请参见[开发运维规范](#)。
- 由于GeminiDB Redis和开源Redis集群计算hash算法不同，因此GeminiDB Redis的部分命令需要给对应的key打上hashtag，否则会出现非预期的行为。hashtag使用建议参考[开发运维规范](#)。

使用 JedisSentinelPool 访问

样例代码

```
import org.apache.commons.pool2.impl.GenericObjectPoolConfig;
import redis.clients.jedis.Jedis;
import redis.clients.jedis.JedisSentinelPool;
import java.util.HashSet;
import java.util.Set;

public void SentinelTest {
    public static void main(String[] args) {
        GenericObjectPoolConfig<Jedis> config = new GenericObjectPoolConfig<Jedis>();
        Set<String> mySentinels = new HashSet<String>();
        mySentinels.add("172.xx.xx.xx:6379");
        JedisSentinelPool pool = new JedisSentinelPool(master-name, mySentinels, config, 1000, password, 0);
        Jedis jedis = pool.getResource();
        jedis.auth(password);
        jedis.set("foo", "bar");
        String s = jedis.get("foo");
        System.out.println(s);
        jedis.close();
        pool.close();
    }
}
```

- 上述代码中“172.xx.xx.xx”为待连接GeminiDB Redis实例的负载均衡地址。您可以单击实例名称，进入“基本信息”页面，在网络信息区域获取“负载均衡地址”。

图 5-4 查看负载均衡地址



- 上述代码中的“6379”为待连接GeminiDB Redis实例的端口，具体请以实际端口为准。获取端口的方法请参见[查看GeminiDB Redis IP地址和端口](#)。
- 上述代码中的master-name只能是“mymaster”。
- GeminiDB Redis支持与限制的命令，详情请参见[开发运维规范](#)。
- 由于GeminiDB Redis和开源Redis集群计算hash算法不同，因此GeminiDB Redis的部分命令需要给对应的key打上hashtag，否则会出现非预期的行为。hashtag使用建议参考[开发运维规范](#)。

5.3.2 通过 Redisson 连接实例

本章节介绍使用Java的开源库Redisson访问GeminiDB Redis实例的方法，主要涉及单机模式和sentinel模式连接。

前提条件

- 已成功创建GeminiDB Redis实例，且实例状态正常。创建GeminiDB Redis实例的方法请参见[购买GeminiDB Redis集群版实例](#)。
- 已创建弹性云服务器，创建弹性云服务器的方法，请参见《弹性云服务器快速入门》中“[购买弹性云服务器](#)”章节。
- 弹性云服务器上已经安装GCC等编译工具。
- 创建的弹性云服务器与GeminiDB Redis实例要保证区域、可用区、VPC和安全组一致。

使用 SingleServer 模式连接实例

样例代码：

```
import org.redisson.Redisson;
import org.redisson.api.RedissonClient;
import org.redisson.config.Config;

public class SingleServerTests {

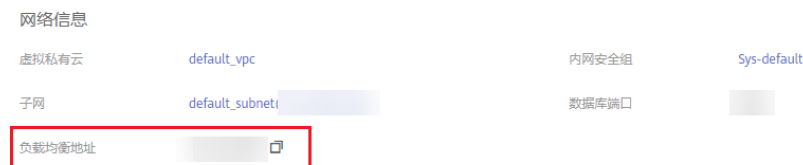
    private static void testSingleServer() {
        Config config = new Config();
        // 认证用的用户名和密码直接写到代码中有很大的安全风险，建议在配置文件或者环境变量中存放(密码应密文存放、使用时解密)，确保安全；
        // 本示例以用户名和密码保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量(环境变量名称请根据自身情况进行设置)EXAMPLE_USERNAME_ENV和EXAMPLE_PASSWORD_ENV。
        String password = System.getenv("EXAMPLE_PASSWORD_ENV");
        config.useSingleServer().setAddress("redis://172.xx.xx.xx:6379")
            .setPassword(password);
        RedissonClient redisson = Redisson.create(config);
    }
}
```

```
execute(redisson); // send requests to database
redisson.shutdown();
}

public static void main(String[] args) {
    testSingleServer();
}
}
```

- 上述代码中“172.xx.xx.xx”为待连接GeminiDB Redis实例的负载均衡地址。您可以单击实例名称，进入“基本信息”页面，在网络信息区域获取“负载均衡地址”。

图 5-5 查看负载均衡地址



- 上述代码中的“6379”为待连接GeminiDB Redis实例的端口，具体请以实际端口为准。获取端口的方法请参见[查看GeminiDB Redis IP地址和端口](#)。
- GeminiDB Redis支持与限制的命令，详情请参见[开发运维规范](#)。
- 由于GeminiDB Redis和开源Redis集群计算hash算法不同，因此GeminiDB Redis的部分命令需要给对应的key打上hashtag，否则会出现非预期的行为。hashtag使用建议参考[开发运维规范](#)。

使用 Sentinel 模式连接实例

样例代码：

```
import org.redisson.Redisson;
import org.redisson.api.RedissonClient;
import org.redisson.config.Config;
import static org.redisson.config.ReadMode.MASTER;

public class SingleServerTests {

    public static void testSentinel() {
        Config config = new Config();
        // 认证用的用户名和密码直接写到代码中有很大的安全风险，建议在配置文件或者环境变量中存放(密码应密文存放、使用时解密)，确保安全；
        // 本示例以用户名和密码保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量(环境变量名称请根据自身情况进行设置)EXAMPLE_USERNAME_ENV和EXAMPLE_PASSWORD_ENV。
        String password = System.getenv("EXAMPLE_PASSWORD_ENV");
        config.useSentinelServers()
            .setMasterName(master_name)
            .setCheckSentinelList(false)
            .setReadMode(MASTER)
            .setPassword(password)
            .addSentinelAddress("redis://172.xx.xx.xx:6379");
        RedissonClient redisson = Redisson.create(config);
        execute(redisson); // send requests to database
        redisson.shutdown();
    }

    public static void main(String[] args) {
        testSentinel();
    }
}
```

- 上述代码中“172.xx.xx.xx”为待连接GeminiDB Redis实例的负载均衡地址。您可以单击实例名称，进入“基本信息”页面，在网络信息区域获取“负载均衡地址”。

图 5-6 查看负载均衡地址



- 上述代码中的“6379”为待连接GeminiDB Redis实例的端口，具体请以实际端口为准。获取端口的方法请参见[查看GeminiDB Redis IP地址和端口](#)。
- GeminiDB Redis只是实现了Sentinel模式的接入方式，不采用原生sentinel的可用性功能。所以样例代码中，master_name固定为“mymaster”，必须设置“CheckSentinelsList”为“false”，必须设置ReadMode为MASTER。
- GeminiDB Redis支持与限制的命令，详情请参见[开发运维规范](#)。
- 由于GeminiDB Redis和开源Redis集群计算hash算法不同，因此GeminiDB Redis的部分命令需要给对应的key打上hashtag，否则会出现非预期的行为。hashtag使用建议参考[开发运维规范](#)。

使用 ClusterServer 模式连接实例

样例代码：

```
import org.redisson.Redisson;
import org.redisson.api.RedissonClient;
import org.redisson.config.Config;

public class ClusterServerTests {

    private static void testClusterServer() {
        Config config = new Config();
        config.useClusterServers()
            .addNodeAddress("redis://172.xx.xx.xx:6379")
            .setPassword(password);

        RedissonClient redisson = Redisson.create(config);
        execute(redisson); // send requests to database
        redisson.shutdown();
    }

    public static void main(String[] args) {
        testClusterServer();
    }
}
```

- 上述代码中“172.xx.xx.xx”为待连接GeminiDB Redis实例的负载均衡地址。您可以单击实例名称，进入“基本信息”页面，在网络信息区域获取“负载均衡地址”。

图 5-7 查看负载均衡地址



- 上述代码中的“6379”为待连接GeminiDB Redis实例的端口，具体请以实际端口为准。获取端口的方法请参见[查看GeminiDB Redis IP地址和端口](#)。
- GeminiDB Redis支持与限制的命令，详情请参见[开发运维规范](#)。
- 由于GeminiDB Redis和开源Redis集群计算hash算法不同，因此GeminiDB Redis的部分命令需要给对应的key打上hashtag，否则会出现非预期的行为。hashtag使用建议参考[开发运维规范](#)。

5.3.3 通过 Hiredis 连接实例

本章节主要介绍使用Hiredis访问GeminiDB Redis实例的方法。

前提条件

- 已成功创建GeminiDB Redis实例，且实例状态正常。创建GeminiDB Redis实例的方法请参见[购买GeminiDB Redis集群版实例](#)。
- 已创建弹性云服务器，创建弹性云服务器的方法，请参见《弹性云服务器快速入门》中“[购买弹性云服务器](#)”章节。
- 弹性云服务器上已经安装GCC等编译工具。
- 创建的弹性云服务器与GeminiDB Redis实例要保证区域、可用区、VPC和安全组一致。

操作步骤

步骤1 获取GeminiDB Redis实例的负载均衡地址和端口。

- 负载均衡地址的查看方法请参见[查看负载均衡地址及端口](#)。
- 端口的获取方法请参见[查看实例节点端口](#)。

步骤2 登录弹性云服务器，具体操作请参见《弹性云服务器快速入门》中“[登录弹性云服务器](#)”。

步骤3 使用如下命令，下载并解压Hiredis。

```
wget https://github.com/redis/hiredis/archive/master.zip
```

步骤4 进入到解压目录后编译安装Hiredis。

```
make  
make install
```

步骤5 编写测试代码connRedisTst.cc。

📖 说明

关于Hiredis的详细用法，请参考[redis官网](#)的使用介绍。

代码如下：

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <hiredis.h>
int main(int argc, char **argv) {
    unsigned int j;
    redisContext *conn;
    redisReply *reply;
    if (argc < 3) {
        printf("Usage: example {instance_ip_address} 6379 {password}\n");
        exit(0);
    }
    const char *hostname = argv[1];
    const int port = atoi(argv[2]);
    const char *password = argv[3];
    struct timeval timeout = { 1, 500000 }; // 1.5 seconds
    conn = redisConnectWithTimeout(hostname, port, timeout);
    if (conn == NULL || conn->err) {
        if (conn) {
            printf("Connection error: %s\n", conn->errstr);
            redisFree(conn);
        } else {
            printf("Connection error: can't allocate redis context\n");
        }
        exit(1);
    }
    /* AUTH */
    reply = redisCommand(conn, "AUTH %s", password);
    printf("AUTH: %s\n", reply->str);
    freeReplyObject(reply);

    /* Set */
    reply = redisCommand(conn, "SET %s %s", "key", "hiredis test ok!");
    printf("SET: %s\n", reply->str);
    freeReplyObject(reply);

    /* Get */
    reply = redisCommand(conn, "GET key");
    printf("GET key: %s\n", reply->str);
    freeReplyObject(reply);

    /* Disconnects and frees the context */
    redisFree(conn);
    return 0;
}
```

步骤6 执行如下命令进行编译。

```
gcc connRedis.c -o connRedis -I /usr/local/include/hiredis -lhiredis
```

如果有报错，可查找hiredis.h文件路径，并修改编译命令。

编译完后得到一个可执行文件connRedis。

步骤7 执行如下命令，连接GeminiDB Redis实例。

```
export LD_LIBRARY_PATH=/usr/local/lib:$LD_LIBRARY_PATH
```

```
./connRedis <redis_ip_address> 6379 <password>
```

其中，以下信息需按照实际情况进行替换：

- <redis_ip_address>为**步骤1**中获取到的GeminiDB Redis实例的负载均衡地址。
- “6379”为GeminiDB Redis实例的端口。
- <password>为创建GeminiDB Redis实例时自定义的密码。

步骤8 返回如下回显信息，表示成功连接GeminiDB Redis实例。

```
AUTH: OK  
SET: OK  
GET key: Hello, hiredis test ok!
```

----结束

5.3.4 通过 NodeJs 连接实例

本章节主要介绍使用NodeJs访问GeminiDB Redis实例的方法。

前提条件

- 已成功创建GeminiDB Redis实例，且实例状态为“正常”。
- 已创建弹性云服务器，创建弹性云服务器的方法，请参见《弹性云服务器快速入门》中“[购买弹性云服务器](#)”章节。
- 如果弹性云服务器为Linux系统，该弹性云服务器上必须已经安装GCC等编译工具。
- 创建的弹性云服务器与GeminiDB Redis实例要保证区域、可用区、VPC和安全组一致。

操作步骤

步骤1 获取GeminiDB Redis实例的负载均衡地址和端口。

- 负载均衡地址的查看方法请参见[查看负载均衡地址及端口](#)。
- 端口信息的获取方法请参见[查看实例节点端口](#)。

步骤2 登录弹性云服务器，具体操作请参见《弹性云服务器快速入门》中“[登录弹性云服务器](#)”。

步骤3 安装NodeJs。

- **方法一：**使用如下命令安装NodeJs。

```
yum install nodejs
```

说明

以CentOS([redhat系列](#))为例，如果是Ubuntu([debian系列](#))，则需要使用其对应的安装命令。

- **方法二：**如果方法一安装不了，也可以通过如下方式进行安装。

```
wget https://nodejs.org/dist/v0.12.4/node-v0.12.4.tar.gz --no-check-certificate ;  
tar -xvf node-v0.12.4.tar.gz;  
cd node-v0.12.4;  
./configure;  
make;  
make install;
```

说明

以CentOS([redhat系列](#))为例，如果是Ubuntu([debian系列](#))，则需要使用其对应的安装命令。

步骤4 NodeJs安装完成后，可执行如下命令，查看其版本号，确认NodeJs已安装成功。

```
node -v
```

步骤5 安装JS包管理工具npm。

```
yum install npm
```

步骤6 安装NodeJs redis客户端ioredis。

```
npm install ioredis
```

步骤7 编辑连接GeminiDB Redis实例的示例脚本。

- 使用NodeJs单机SDK连接GeminiDB Redis集群

```
var Redis = require('ioredis');
// 认证用的用户名和密码直接写到代码中有很大的安全风险，建议在配置文件或者环境变量中存放(密码应
// 密文存放、使用时解密)，确保安全；
// 本示例以用户名和密码保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量(环境变量
// 名称请根据自身情况进行设置)EXAMPLE_USERNAME_ENV和EXAMPLE_PASSWORD_ENV。
var pwd = process.env.EXAMPLE_PASSWORD_ENV;
var redis = new Redis({
  port: 6379,          // 此处为步骤1中获取到的GeminiDB Redis实例的端口。
  host: '192.xx.xx.xx', // 此处为步骤1中获取到的GeminiDB Redis实例的负载均衡地址。
  family: 4,           // 此处填写4表示IPv4，6表示IPv6。
  password: pwd,
  db: 0
});
redis.set('key', 'Nodejs tst ok!');
redis.get('key', function (err, result) {
  console.log(result);
});
```

- 使用NodeJs集群SDK连接GeminiDB Redis集群

```
const Redis = require("ioredis");
// 认证用的用户名和密码直接写到代码中有很大的安全风险，建议在配置文件或者环境变量中存放(密码应
// 密文存放、使用时解密)，确保安全；
// 本示例以用户名和密码保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量(环境变量
// 名称请根据自身情况进行设置)EXAMPLE_USERNAME_ENV和EXAMPLE_PASSWORD_ENV。
var pwd = process.env.EXAMPLE_PASSWORD_ENV;
const cluster = new Redis.Cluster([
  {
    port: 6379,          // 此处为步骤1中获取到的GeminiDB Redis实例的端口。
    host: '192.xx.xx.xx', // 此处为步骤1中获取到的GeminiDB Redis实例的负载均衡地址。
    family: 4,           // 此处填写4表示IPv4，6表示IPv6。
    password: pwd,
    db: 0
  },
]);
cluster.set("foo", "nodejs is ok!");
cluster.get("foo", (err, res) => {
  console.log(res);
});
```

步骤8 运行示例脚本，确认结果正常。

```
node ioredisdemo.js
```

----结束

5.3.5 通过 PHP 连接实例

本章节主要介绍使用PHP访问GeminiDB Redis实例的方法。

前提条件

- 已成功创建GeminiDB Redis实例，且实例状态为“正常”。
- 已创建弹性云服务器，创建弹性云服务器的方法，请参见《弹性云服务器快速入门》中“[购买弹性云服务器](#)”章节。
- 弹性云服务器上已经安装GCC等编译工具。
- 创建的弹性云服务器与GeminiDB Redis实例要保证区域、可用区、VPC和安全组一致。

操作步骤

步骤1 获取GeminiDB Redis实例的负载均衡地址和端口。

- 负载均衡地址的查看方法请参见[查看负载均衡地址及端口](#)。
- 端口信息的获取方法请参见[查看实例节点端口](#)。

步骤2 登录弹性云服务器，具体操作请参见《弹性云服务器快速入门》中“[登录弹性云服务器](#)”。

步骤3 安装PHP开发包与命令行工具。

执行如下命令，使用yum方式直接安装。

```
yum install php-devel php-common php-cli
```

说明

以CentOS([redhat系列](#))为例，如果是Ubuntu([debian系列](#))，则需要使用其对应的安装命令。

步骤4 安装完后可查看版本号，确认成功安装。

```
php --version
```

步骤5 安装Redis的PHP客户端。

1. 下载phpredis源文件。

```
wget http://pecl.php.net/get/redis-4.1.0RC3.tgz
```

说明

以上是当前最新版本，也可以通过[PHP官网](#)下载其他版本的phpredis客户端。

2. 解压phpredis源文件包。

```
tar -zxvf redis-4.1.0RC3.tgz
```

```
cd redis-4.1.0RC3
```

3. 编译前先执行扩展命令。

```
phpize
```

4. 配置php-config文件。

```
./configure --with-php-config=/usr/bin/php-config
```

说明

不同的操作系统，安装PHP的方式不同，该配置文件位置可能不同。建议在配置前，先确认该文件的目录，命令如下：`find / -name php.ini`。

5. 编译和安装phpredis客户端。

```
make && make install
```

6. 安装完后在php.ini文件中增加extension配置项，用于增加redis模块的引用配置。
使用如下命令，查找到php.ini文件：

```
vim /usr/local/php/etc/php.ini
```

在php.ini文件中，增加如下配置项：

```
extension = "/usr/lib64/php/modules/redis.so"
```

说明

php.ini和redis.so两个文件的目录可能不同，可通过如下命令先查找确认。

```
find / -name php.ini
```

```
find / -name redis.so
```

7. 保存退出后确认扩展生效。

```
php -m |grep redis
```

如果以上命令返回了redis，表示php redis客户端环境搭建好了。

步骤6 使用phpredis客户端连接GeminiDB Redis实例。

1. 编写测试代码redis.php。

- 使用PHP单机SDK连接GeminiDB Redis集群

```
<?php
// 认证用的用户名和密码直接写到代码中有很大的安全风险，建议在配置文件或者环境变量中存
放(密码应密文存放、使用时解密)，确保安全；
// 本示例以用户名和密码保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量
(环境变量名称请根据自身情况进行设置)EXAMPLE_USERNAME_ENV和
EXAMPLE_PASSWORD_ENV。
$pwd = getenv('EXAMPLE_PASSWORD_ENV');
$redis_host = "192.xx.xx.xx"; //此处为步骤1获取到的负载均衡地址
$redis_port = 6379;
$user_pwd = $pwd;
$redis = new Redis();
if ($redis->connect($redis_host, $redis_port) == false) {
    die($redis->getLastError());
}
if ($redis->auth($user_pwd) == false) {
    die($redis->getLastError());
}
if ($redis->set("key", "php test ok!") == false) {
    die($redis->getLastError());
}
$value = $redis->get("key");
echo $value;
$redis->close();
?>
```

- 使用PHP集群SDK连接GeminiDB Redis集群

```
<?php
$redis_host = "192.xx.xx.xx"; //此处为步骤1获取到的负载均衡地址
$redis_port = 6379;
$user_pwd = "pwd";
// Connect with read/write timeout as well as specify that phpreid should use
// persistent connections to each node.
$redis = new RedisCluster(NULL, Array("$redis_host:$redis_port"), 1.5, 1.5, true,
$user_pwd);
if ($redis->set("key", "php test ok!") == false) {
    die($redis->getLastError());
}
$value = $redis->get("key");
echo $value;
$redis->close();
?>
```

2. 执行redis.php，确认结果正常。

----结束

5.3.6 通过 Python 连接实例

本章节主要介绍使用Python访问GeminiDB Redis实例的方法。

前提条件

- 已成功创建GeminiDB Redis实例，且实例状态为“正常”。
- 已创建弹性云服务器，创建弹性云服务器的方法，请参见《弹性云服务器快速入门》中“[购买弹性云服务器](#)”章节。
- 弹性云服务器上已经安装GCC等编译工具。
- 创建的弹性云服务器与GeminiDB Redis实例要保证区域、可用区、VPC和安全组一致。

操作步骤

步骤1 获取GeminiDB Redis实例的负载均衡地址和端口。

- 负载均衡地址的查看方法请参见[查看负载均衡地址及端口](#)。
- 端口信息的获取方法请参见[查看实例节点端口](#)。

步骤2 登录弹性云服务器，具体操作请参见《弹性云服务器快速入门》中“[登录弹性云服务器](#)”。

步骤3 安装Python和Redis的Python客户端Redis-py。

1. 如果系统没有自带Python，可以使用yum方式安装。

```
yum install python
```

2. 下载并解压redis-py。

```
wget https://github.com/andymccurdy/redis-py/archive/master.zip
```

3. 进入到解压目录后安装Redis的Python客户端Redis-py。

```
python setup.py install
```

4. 安装后执行python命令，返回如下信息说明成功安装Redis-py：

```
Python 2.6.6 (r266:84292, Aug 18 2016, 15:13:37)
[GCC 4.4.7 20120313 (Red Hat 4.4.7-17)] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> import redis
>>>
```

步骤4 使用Redis-py客户端连接GeminiDB Redis实例。

以下步骤以命令行模式进行示例（也可以将命令写入python脚本中再执行）：

- **使用单机方式连接GeminiDB Redis集群**

- a. 执行python命令，进入命令行模式。

返回如下信息说明已进入命令行模式：

```
Python 2.6.6 (r266:84292, Aug 18 2016, 15:13:37)
[GCC 4.4.7 20120313 (Red Hat 4.4.7-17)] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> import redis
>>>
```

- b. 在命令行中执行以下命令，确认结果正常。

```
>>> r = redis.StrictRedis(host='192.xx.xx.xx', port=6379, password='pwd');
>>> r.set('key', 'Python tst ok!')
True
```

```
>>> r.get('key')
'Python tst ok!'
```

说明

如下信息请按照实际值进行修改后，再执行上述命令。

- host和port为[步骤1](#)中获取到的GeminiDB Redis实例的负载均衡地址和端口。
- password为GeminiDB Redis实例的密码。

• 使用集群方式连接GeminiDB Redis集群

使用前需要设置：config set CompatibleMode ClusterClient

Python 3.7.4 (default, Jan 30 2021, 09:00:44)

[GCC 7.3.0] on linux

Type "help", "copyright", "credits" or "license" for more information.

```
>>> from redis.cluster import RedisCluster as Redis
```

```
>>> rc = Redis(host='127.0.0.1', port=6379, password='a')
```

```
>>> rc.set('key', 'Python test ok!')
```

```
True
```

```
>>> rc.get('key')
```

```
b'Python test ok!'
```

----结束

5.3.7 通过 Go 语言连接实例

本章节主要介绍使用Go语言访问GeminiDB Redis实例的方法。

前提条件

- 已成功创建GeminiDB Redis实例，且实例状态为“正常”。
- 已创建弹性云服务器，创建弹性云服务器的方法，请参见《弹性云服务器快速入门》中“[购买弹性云服务器](#)”章节。
- 弹性云服务器上已经安装GCC等编译工具。
- 创建的弹性云服务器与GeminiDB Redis实例要保证区域、可用区、VPC和安全组一致。

操作步骤

步骤1 获取GeminiDB Redis实例的负载均衡地址和端口信息。

- 负载均衡地址的查看方法请参见[查看负载均衡地址及端口](#)。
- 端口信息的获取方法请参见[查看实例节点端口](#)。

步骤2 登录弹性云服务器，具体操作请参见《弹性云服务器快速入门》中“[登录弹性云服务器](#)”。

步骤3 使用Golang客户端连接GeminiDB Redis实例，本文以使用go-redis SDK为例。

- go-redis的下载地址为：<https://github.com/go-redis/redis>。
- 使用单机方式连接GeminiDB Redis集群

```
package main
import (
    "fmt"
    "github.com/go-redis/redis"
    "os"
)
func main() {
    // 认证用的用户名和密码直接写到代码中有很大的安全风险，建议在配置文件或者环境变量中存放(密码
```

```
应密文存放、使用时解密)，确保安全；
// 本示例以用户名和密码保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量(环境变量名称请根据自身情况进行设置)EXAMPLE_USERNAME_ENV和EXAMPLE_PASSWORD_ENV。
password = os.Getenv("EXAMPLE_PASSWORD_ENV")
client := redis.NewClient(&redis.Options{
    Addr: "xx.xx.xx.xx:6379", //步骤1获取到的负载均衡地址
    Password: password,
    DB: 0, // 使用默认的0号数据库
})
pong, err := client.Ping().Result()
fmt.Println(pong, err)
err = client.Set("key1", "value1", 0).Err()
if err != nil {
    panic(err)
}
val, err := client.Get("key1").Result()
if err != nil {
    panic(err)
}
fmt.Println("key1", val)
}
```

预期输出为：

```
PONG
key1 value1
```

说明

- 使用go-redis连接GeminiDB Redis时，需要使用普通模式，不能使用集群模式，如上示例代码所示。
 - 上述实例代码中的GeminiDB Redis地址和密码，请根据实际情况填写。
- 使用集群方式连接GeminiDB Redis集群

```
package main
import (
    "fmt"
    "github.com/go-redis/redis"
)
func main() {
    client := redis.NewClusterClient(&redis.ClusterOptions{
        Addrs: []string{ // 步骤1获取到的负载均衡地址
            "xx.xx.xx.xx:6379",
        },
        Password: "xx", // 密码
    })
    pong, err := client.Ping().Result()
    fmt.Println(pong, err)

    err = client.Set("key1", "value1", 0).Err()
    if err != nil {
        panic(err)
    }
    val, err := client.Get("key1").Result()
    if err != nil {
        panic(err)
    }
    fmt.Println("key1", val)
}
```

----结束

5.3.8 通过 C#语言连接实例

本章节主要介绍使用C# 语言访问GeminiDB Redis实例的方法。

前提条件

- 已成功创建GeminiDB Redis实例，且实例状态为“正常”。
- 已创建弹性云服务器，创建弹性云服务器的方法，请参见《弹性云服务器快速入门》中“[购买弹性云服务器](#)”章节。
- 弹性云服务器上已经安装GCC等编译工具。
- 创建的弹性云服务器与GeminiDB Redis实例要保证区域、可用区、VPC和安全组一致。

操作步骤

步骤1 获取GeminiDB Redis实例的负载均衡地址和端口信息。

- 负载均衡地址的查看方法请参见查看[查看负载均衡地址及端口](#)。
- 端口信息的获取方法请参见查看[查看实例节点端口](#)。
- 各实例地址的查看方法请参见查看[查看内网IP地址或弹性公网IP](#)。

步骤2 登录弹性云服务器，具体操作请参见《弹性云服务器快速入门》中“[登录弹性云服务器](#)”。

步骤3 安装.Net。Windows主机前往进入[.NET官网](#)下载，Linux主机需要安装 .NET Core 的密钥和存储库，再安装.Net运行时和SDK：

```
sudo rpm -Uvh https://packages.microsoft.com/config/centos/8/packages-microsoft-prod.rpm
sudo yum install dotnet-sdk-7.0
sudo yum install dotnet-runtime-7.0
```

运行

```
dotnet --version
```

您将会看到您的.Net版本信息。

步骤4 使用StackExchange.Redis客户端连接GeminiDB Redis实例。

- 创建项目

在命令行窗口中运行以下命令创建一个新的C#控制台应用程序；或者在Visual Studio中创建一个新的C#控制台应用程序。

```
dotnet new console -o restdemo
```

- 安装Redis的C#客户端 StackExchange.Redis包。Visual Studio中您可以从NuGet包管理器中安装StackExchange.Redis。或者dotnet项目所在的命令行窗口，运行以下命令：

```
dotnet add package StackExchange.Redis
```

- 使用单机方式连接GeminiDB Redis

```
using System;
using StackExchange.Redis;
namespace restdemo
{
    class Program
    {
        static void Main(string[] args)
        {
            // 创建一个连接到Redis服务器的ConnectionMultiplexer对象
            string redisConnectionString = "192.xx.xx.xx:6379"; // 步骤1获取到的负载均衡地址
            ConfigurationOptions options = ConfigurationOptions.Parse(redisConnectionString);
            // 认证用的用户名和密码直接写到代码中有很大的安全风险，建议在配置文件或者环境变量中存放
            // (密码应加密存放、使用时解密)，确保安全；
            // 本示例以用户名和密码保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量
            // (环境变量名称请根据自身情况进行设置)EXAMPLE_USERNAME_ENV和EXAMPLE_PASSWORD_ENV。
            var redis = ConnectionMultiplexer.Connect(redisConnectionString, options);
            IDatabase db = redis.GetDatabase();
            db.KeySet().ForEachAsync(key => Console.WriteLine(key));
        }
    }
}
```



```
string password = Environment.GetEnvironmentVariable("EXAMPLE_PASSWORD_ENV");
options.Password = password;
ConnectionMultiplexer redis = ConnectionMultiplexer.Connect(options);

// 获取Redis数据库对象
IDatabase redisDb = redis.GetDatabase();

// 设置一个键值对
string key = "mykey";
string value = "myvalue";
redisDb.StringSet(key, value);
string valueGet = redisDb.StringGet(key);
Console.WriteLine($"键 {key} 对应的值为: {valueGet}");
}
```

预期输出

键 mykey 对应的值为: myvalue

- 使用集群方式连接GeminiDB Redis集群

```
using System;
using StackExchange.Redis;

namespace redisdemo
{
    class Program
    {
        static void Main(string[] args)
        {
            ConfigurationOptions options = new ConfigurationOptions();
            options.EndPoints.Add("192.xx.xx.xx:6379"); // 步骤1获得的实例集群节点1的地址和端口号
            options.EndPoints.Add("192.xx.xx.xx:6379"); // 步骤1获得的实例集群节点2的地址和端口号

            options.Password = "your_password"; // 设置密码
            ConnectionMultiplexer redis = ConnectionMultiplexer.Connect(options);
            // 获取Redis数据库对象
            IDatabase redisDb = redis.GetDatabase();
            // 设置一个键值对
            string key = "mykey";
            string value = "myvalue";
            redisDb.StringSet(key, value);
            string valueGet = redisDb.StringGet(key);
            Console.WriteLine($"键 {key} 对应的值为: {valueGet}");
        }
    }
}
```

预期输出

键 mykey 对应的值为: myvalue

----结束

5.3.9 使用 Sentinel 兼容模式连接实例

GeminiDB Redis采用自研的高可用服务HA组件，无需依赖Sentinel（哨兵）。但为了减少用户的代码改动，提高实例的兼容性，GeminiDB Redis兼容Redis Sentinel访问方式。开启Sentinel兼容模式，您可以像连接开源Redis Sentinel一样连接GeminiDB Redis实例。

前提条件

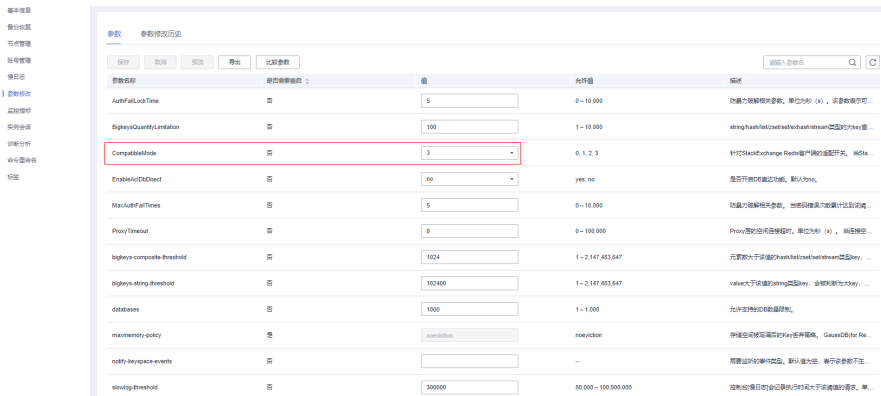
- 已成功创建GeminiDB Redis集群版或主备版实例，且实例状态为“正常”。
- 已创建弹性云服务器，创建弹性云服务器的方法，请参见《弹性云服务器快速入门》中“[购买弹性云服务器](#)”章节。

- 弹性云服务器上已经安装GCC等编译工具。
- 创建的弹性云服务器与GeminiDB Redis实例要保证区域、可用区、VPC和安全组一致。
- 使用Sentinel模式连接实例，必须要先开启Sentinel兼容模式。

开启 Sentinel 兼容模式

- 步骤1** 登录[云数据库GeminiDB控制台](#)。
- 步骤2** 在“实例管理”页面，选择指定的实例，单击实例名称，进入“基本信息”页面。
- 步骤3** 在左侧导航树，单击“参数修改”。
- 步骤4** 检查修改参数“CompatibleMode”值，单击“保存”。
- 如果是集群版实例，修改参数“CompatibleMode”值为3。
 - 如果是主备版实例，修改参数“CompatibleMode”值为2。

图 5-8 设置参数



----结束

Sentinel 模式连接实例

本示例以Java为例，通过开源库Redisson和Jedis访问GeminiDB Redis实例。

Redisson 代码示例

```
import org.redisson.Redisson;
import org.redisson.api.RedissonClient;
import org.redisson.config.Config;
import static org.redisson.config.ReadMode.MASTER;

public class SingleServerTests {

    public static void testSentinel() {
        Config config = new Config();
        // 认证用的用户名和密码直接写到代码中有很大的安全风险，建议在配置文件或者环境变量中存放(密码应密文存放、使用时解密)，确保安全；
        // 本示例以用户名和密码保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量(环境变量名称请根据自身情况进行设置)EXAMPLE_USERNAME_ENV和EXAMPLE_PASSWORD_ENV。
        String password = System.getenv("EXAMPLE_PASSWORD_ENV");
        config.useSentinelServers()
            .setMasterName("mymaster")
    }
}
```

```
.setCheckSentinelsList(false)
.setReadMode(ReadMode.MASTER)
.setPassword(password)
.addSentinelAddress("redis://172.xx.xx.xx:6379");
RedissonClient redisson = Redisson.create(config);
execute(redisson); // send requests to database
redisson.shutdown();
}

public static void main(String[] args) {
    testSentinel();
}
}
```

Jedis 代码示例

```
import java.util.HashSet;
import java.util.Set;
import org.apache.commons.pool2.impl.GenericObjectPoolConfig;
import redis.clients.jedis.Jedis;
import redis.clients.jedis.JedisSentinelPool;
public class TestJedisSentinel {
    public static void main(String[] args) {
        Set<String> sentinels = new HashSet<>();
        sentinels.add("192.xx.xx.xx:6379");
        GenericObjectPoolConfig<Jedis> poolConfig = new GenericObjectPoolConfig<>();
        poolConfig.setMaxIdle(100);
        poolConfig.setMaxWaitMillis(10000);
        poolConfig.setTestOnBorrow(true);
        int connectionTimeout = 5000;
        int soTimeout = 5000;
        int database = 0;
        // 认证用的用户名和密码直接写到代码中有很大的安全风险，建议在配置文件或者环境变量中存放(密码应密文存放、使用时解密)，确保安全；
        // 本示例以用户名和密码保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量(环境变量名称请根据自身情况进行设置)EXAMPLE_USERNAME_ENV和EXAMPLE_PASSWORD_ENV。
        String password = System.getenv("EXAMPLE_PASSWORD_ENV");
        try (JedisSentinelPool jspool = new JedisSentinelPool("mymaster", sentinels, poolConfig,
            connectionTimeout, soTimeout, password, database)) {
            Jedis jedis = jspool.getResource();
            jedis.mset("testkey", "AAA", "b", "BBB");
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

- MasterName：为固定字符串“mymaster”。
- CheckSentinelsList：固定为“false”。
- ReadMode：使用ReadMode.MASTER。
- Password：实例对应的密码。
- SentinelAddress：为GeminiDB Redis实例的负载均衡地址，具体请以实际IP地址和Port为准。

您可以通过单击实例名称，进入“基本信息”页面，在连接信息区域获取“负载均衡地址”。

图 5-9 查看负载均衡地址



GeminiDB Redis只是实现了Sentinel模式的接入方式，不采用原生Sentinel的可用性功能。所以样例代码中，“master_name”固定为“mymaster”，必须设置“CheckSentinelsList”为“false”，必须设置“ReadMode”为“MASTER”。

5.4 Lua 脚本编写规范

Lua是一种脚本语言，目的是为了嵌入应用程序中，为应用程序提供灵活的扩展和定制功能。GeminiDB Redis使用的是Lua5.1.5版本，与开源Redis5.0使用的Lua版本是一致的。

说明

使用Lua脚本时，需要经过谨慎的校验，否则可能出现死循环、业务超时等情况，甚至会导致业务不可用。

与开源 Redis Lua 的区别

1. EVAL/EVALSHA命令

命令格式：

EVAL script numkeys key [key ...] arg [arg ...]

EVALSHA sha1 numkeys key [key ...] arg [arg ...]

上述命令的语法与操作与开源Redis一致。用户需自己保证将脚本中使用到的Redis key显式地通过key参数传入，而不是直接在脚本中编码。

若使用集群版实例，如果带有多个key参数，则要求所有的key参数必须拥有相同的hashtag。

如果不遵循上述约束，则在Lua中执行涉及这些key的Redis操作时，可能会返回错误信息，甚至可能导致数据的一致性被破坏。

2. SCRIPT命令

SCRIPT命令包含了一组管理Lua脚本的子命令，具体可以通过SCRIPT HELP命令查询具体的操作。

SCRIPT大部分命令都与开源Redis兼容，其中需要特别说明的命令如下：

– SCRIPT KILL

GeminiDB Redis是多线程执行的环境，允许同时执行多个Lua脚本，执行SCRIPT KILL，会终止所有正在运行的Lua脚本。

为了方便使用，GeminiDB Redis扩展了SCRIPT KILL命令，用户可以通过‘SCRIPT KILL SHA1’来终止指定哈希值的脚本。若同一时间存在多个节点在执行哈希值相同的脚本，那么这些脚本都会被终止。

另外，由于用户无法设置Lua超时时间（config set lua-time-limit），因此在任意时刻执行SCRIPT KILL都能直接终止脚本，而不是等待脚本超时后才终止。

- **SCRIPT DEBUG**
目前GeminiDB Redis不支持DEBUG功能，所以该命令执行无效。
- **SCRIPT GET**
GeminiDB Redis新增命令, 可用于查询通过 "SCRIPT LOAD" 保存到数据库中的脚本内容。
语法为：SCRIPT GET SHA1。

3. **Lua脚本中执行Redis命令**

与开源Redis一致，GeminiDB Redis的Lua环境中也提供了一个全局的“redis”表，用于提供各类和Redis Server交互的函数。
如表5-5为GeminiDB Redis目前支持和不支持的操作列表。

表 5-5 函数列表

支持的操作	不支持的操作
• redis.call() • redis.pcall() • redis.sha1hex() • redis.error_reply() • redis.status_reply()	• redis.log() • redis.LOG_DEBUG • redis.LOG_VERBOSE • redis.LOG_NOTICE • redis.LOG_WARNING • redis.replicate_commands() • redis.set_repl() • redis.REPL_NONE • redis.REPL_AOF • redis.REPL_SLAVE • redis.REPL_REPLICA • redis.REPL_ALL • redis.breakpoint() • redis.debug()

4. **Lua执行环境限制**

开源Redis对Lua脚本的执行有一定的限制，比如限制脚本操作全局变量，限制随机函数的结果，限定能够使用的系统库和第三方库等。
GeminiDB Redis也继承了绝大多数的限制，但是针对如下情况，GeminiDB Redis与开源Redis存在差异：

- **Write Dirty**
开源Redis规定，如果某个脚本已经执行了写操作，那么就不能被SCRIPT KILL停止执行，必须使用SHUTDOWN NOSAVE来直接关闭Redis Server。
GeminiDB Redis不支持执行SHUTDOWN命令，因此这条限制不会被执行，用户仍然可以通过SCRIPT KILL来停止脚本的执行。
- **Random Dirty**
由于主从复制的原因，开源Redis规定，若脚本执行了带有随机性质的命令（Time, randomkey），则不允许再执行写语义的命令。

例如，如下Lua脚本：

```
local t = redis.call("time")
return redis.call("set", "time", t[1]);
```

当该脚本的执行传递到从节点时，Time命令获取到的时间一定晚于主节点，因此从节点执行的Set命令的值就会和主节点产生冲突。开源Redis引入了replicate_commands来允许用户决定这种场景下的行为模式。

对于GeminiDB Redis来说，由于没有主从的概念，数据在逻辑上只有一份，因此也就不存在该限制。

Lua 脚本中禁止执行的命令

Hash: hscan

List: blpop、brpop、brpoplpush、blmpop、blmove

Set: sscan

Sorted Set: bzpopmax、bzpopmin、zscan、bzmpop

Stream: xread、xreadgroup

Generic: rename、renamenx、restore、scan、client、command、config、dbsize、flushall、flushdb、info、keys

Lua: eval、evalsha、script

Pub/Sub: psubscribe、publish、punsubscribe、subscribe、unsubscribe

Transactions: discard、exec、multi、unwatch、watch

5.5 使用紧凑型编码实现小对象存储

GeminiDB采用RocksDB作为底层存储引擎。由于RocksDB仅支持键值（KV）存储模型，系统需要将各类数据结构转换为KV格式进行存储，这一过程涉及数据编码。在传统编码方案中，存储包含N个field-value对的hash对象需要N+1个KV对，这导致执行HGETALL命令时需要执行N+1次读取操作。在部分业务场景中，如广告RTA，hash类型数据特征为单个元素字节少且元素数量不多，可使用GeminiDB hash紧凑型编码方案，进一步提升读性能。紧凑型编码方案的核心在于将N个field-value对与1个元数据对编码到单个KV对中，从而减少读取次数，使HGETALL/HMGET命令的性能得到大幅提升。目前，紧凑型编码方案仅支持hash数据类型的小对象存储优化。

应用场景

由于紧凑型编码将所有field-value对存储在同一个KV对中，因此不适用于单个hash对象过大的场景。该编码方式最适合以下特征的Hash数据：

- 单个hash对象大小≤1KB。
- 单个hash对象field-value对数量在10-50之间。

使用须知

- 紧凑型编码仅支持hash类型。
- 紧凑型编码不适用于大对象，大对象场景使用紧凑型编码可能导致hset/hmset命令时延上升。

- 不支持编码降级，开关仅对新增对象生效，不会自动对已有对象进行改造。

配置与使用说明

- 启用/禁用紧凑型编码
参考[通过负载均衡地址连接实例（推荐）](#)通过执行“CONFIG SET”命令动态调整“small-hashobj-encode-size”参数，控制是否启用紧凑型编码。

参数	small-hashobj-encode-size
单位	Byte
默认值	0（禁用紧凑型编码，使用传统编码）
推荐值	1024
参数说明	0：禁用紧凑型编码，所有 hash 对象均采用传统编码存储。 1024：启用紧凑型编码，仅当 Hash 对象大小 $\leq 1024\text{B}$ 时生效；若对象大小超过该阈值，仍使用传统编码。

- 查看对象的编码格式
执行以下命令可查看指定 Key 的存储格式：

```
OBJECT STORAGEFORMAT <key>
```

```
127.0.0.1:6379> object storageformat hkey  
"normal"
```

```
127.0.0.1:6379> object storageformat hkey1  
"small-object"
```

- 手动转换对象的编码格式
若需将紧凑型编码的 Key 切换回传统编码，可执行：

```
OBJECT REFORMAT <key> NORMAL
```

注意：不支持将传统编码转换为小对象编码。

说明

命令无差别：无论对象采用传统编码或紧凑型编码，所有操作命令（如 HGET、HSET 等）均保持兼容，无需额外适配。

5.6 keyspace 通知功能

所有支持订阅与发布功能的客户端都可以在不做任何修改的情况下，直接使用键空间通知功能。

使用须知

- 因为开启键空间通知功能需要消耗一些CPU，所以在默认配置下，该功能处于关闭状态。
- 不建议在高压场景下使用键空间通知功能，不仅会影响实例性能，而且部分事件通知可能被丢弃。

与开源 Redis 的区别

1. 配置方法
- 直接使用CONFIG SET命令来开启或关闭键空间通知功能。
- Config set notify-keyspace-events Ex
- 当notify-keyspace-events选项的参数为空字符串或参数不含K、E时，表示键空间通知功能关闭。

说明

• 双引号表示为空字符串。

GeminiDB Redis控制台客户端不支持通过将notify-keyspace-events选项的参数设置为空字符串来关闭键空间功能。

- 当notify-keyspace-events选项的参数不是空字符串且配置正确时（具体请参见表5-6），表示键空间通知功能开启。

2. 支持的通知类型
- 表 5-6 支持的通知类型
- | 字符 | 发送的通知 | GeminiDB Redis是否支持 |
|----|-----------------------------------|--------------------|
| K | 键空间通知。所有通知以__keyspace@<db>__ 为前缀。 | 支持 |
| E | 键事件通知。所有通知以__keyevent@<db>__ 为前缀。 | 支持 |
| g | 与DEL、EXPIRE、RENAME等类型无关的通用命令的通知。 | 支持 |
| \$ | 字符串命令的通知。 | 支持 |
| l | 列表命令的通知。 | 支持 |
| s | 集合命令的通知。 | 支持 |
| h | 哈希命令的通知。 | 支持 |
| z | 有序集合命令的通知。 | 支持 |
| x | EXPIRED过期通知。 | 支持 |
| e | 驱逐(evict)事件通知 | 不涉及 |
| A | 参数g\$lshzxe的别名。 | 支持 |
- 如表5-6，GeminiDB Redis当前版本不涉及evict事件。

输入的参数中至少要有一个K或者E， 否则的话，不管其余的参数是什么，都不会有任何通知被分发。

举个例子，如果只想订阅键空间中和列表相关的通知，那么参数就应该设为Kl，诸如此类。

将参数设为字符串 "AKE" 表示发送所有类型的通知。
- 文档版本 01 (2025-11-13)

版权所有 © 华为云计算技术有限公司

418

5.7 ExHash 命令列表

ExHash是一种可为field设置过期时间和版本的Hash类型数据结构，提高了Hash数据结构的灵活性，简化了很多场景下的业务开发工作。

主要特征

- field支持单独设置expire和version。
- field支持高效灵活的主动、被动过期淘汰（expire）策略。
- 语法和原生Redis Hash数据类型类似。

命令列表

表 5-7 ExHash 命令介绍

命令	语法	说明
EXHSET	EXHSET key field value [EX time] [EXAT time] [PX time] [PXAT time] [NX XX] [VER ABS GT version] [KEEPTTL]	向Key指定的ExHash中插入一个field。如果ExHash不存在则自动创建一个，如果field已经存在则覆盖其值。
EXHGET	EXHGET key field	获取key指定的ExHash中一个field的值，如果ExHash不存在或者field不存在，则返回nil。
EXHPTTL	EXHPTTL key field	查看key指定的ExHash中一个field的剩余过期时间，结果精确到毫秒。
EXHTTL	EXHTTL key field	查看key指定的ExHash中一个field的过期时间，结果精确到秒。
EXHVER	EXHVER key field	查看key指定的ExHash中一个field的当前版本号。
EXHINCRBY	EXHINCRBY key field num [EX time] [EXAT time] [PX time] [PXAT time] [VER ABS GT version] [MIN minval] [MAX maxval] [KEEPTTL]	将key指定的ExHash中一个field的value增加num，num为一个整数。如果ExHash不存在则自动新创建一个，如果指定的field不存在，则在加之前插入该field并将其值设置为0。 说明 为Key的field设置了超时时间后，再次执行该命令时如果没有设置超时时间，该field将被设置为永不过期。

命令	语法	说明
EXHINCRBYFLOAT	EXHINCRBYFLOAT key field num [EX time] [EXAT time] [PX time] [PXAT time] [VER ABS GT version] [MIN minval] [MAX maxval] [KEEPTTL]	将key指定的ExHash中一个field的value增加num，num为一个浮点数。如果ExHash不存在则自动新创建一个，如果指定的field不存在，则在加之前插入该field并将其值设置为0。 说明 为Key的field设置了超时时间后，再次执行该命令时如果没有设置超时时间，该field将被设置为永不过期。
EXHMGGET	EXHMGGET key field [field ...]	同时获取key指定的ExHash多个field的值，如果ExHash不存在或者field不存在，则返回nil。
EXHLEN	EXHLEN key [NOEXP]	获取key指定的ExHash中field个数，该命令不会触发对过期field的淘汰，也不会将其过滤掉，所以结果中可能包含已经过期但还未被删除的field。如果只想返回当前没有过期的field个数，可以在命令中设置NOEXP选项。
EXHGETALL	EXHGETALL key	获取key指定的ExHash中所有field及其value。
EXHDEL	EXHDEL key field [field ...]	删除key指定的ExHash中的一个field，如果ExHash不存在或者field不存在则返回0，成功删除返回1。
DEL	DEL <key> [key ...]	使用原生Redis的DEL命令可以删除一条或多条ExHash数据。
EXISTS	EXISTS <key> [key ...]	使用原生Redis的EXISTS命令可以查看一条或多条ExHash数据是否存在。

复杂命令、选项详细介绍

- EXHSET

表 5-8 EXHSET 命令介绍

类别	说明
语法	EXHSET key field value [EX time] [EXAT time] [PX time] [PXAT time] [NX XX] [VER GT ABS version] [KEEPTTL]

类别	说明
命令描述	向Key指定的ExHash中插入一个field。如果ExHash不存在则自动创建一个，如果field已经存在则覆盖其值。 为Key的field设置了超时时间后，再次执行该命令时如果没有设置超时时间，该field将被设置为永不过期。
选项	Key: ExHash的key，用于指定作为命令调用对象的ExHash。
	field: ExHash中的一个元素，一个ExHash key可以有多个field。
	value: field对应的值，一个field只能有一个value。
	EX: 指定field的相对过期时间，单位为秒，为0表示马上过期，不传此参数表示不过期。
	EXAT: 指定field的绝对过期时间，单位为秒，为0表示马上过期，不传此参数表示不过期。
	PX: 指定field的相对过期时间，单位为毫秒，为0表示马上过期，不传此参数表示不过期。
	PXAT: 指定field的绝对过期时间，单位为毫秒，为0表示马上过期，不传此参数表示不过期。
	NX: 只在field不存在时插入。
	XX: 只在field存在时插入。
	VER: 版本号。如果field存在，和当前版本号做比较：如果相等，继续操作，且版本号加1。如果不相等，返回异常。如果field不存在或者field当前版本为0，忽略传入的版本号并继续操作，成功后版本号变为1。
	GT: 比当前更大的版本号（ Greater Than ），设置的版本号如果比当前的版本号小，则返回失败。
	ABS: 绝对版本号，不论field是否存在，可以在插入field时设置为本参数所指定的版本号。
返回值	KEEPTTL: 在不指定EX、EXAT、PX或PXAT选项时，使用KEEPTTL选项会保留field当前的过期设置。
	新建field并成功为它设置值：1。
	field已经存在，成功覆盖旧值：0。
	指定了XX且field不存在：-1。
	指定了NX且field已经存在：-1。
	指定了VER且版本和当前版本不匹配："ERR update version is stale"。
	其它情况返回相应的异常信息。

- 示例

field过期示例:

```
127.0.0.1:6579> EXHSET k1 f1 v1 ex 10
(integer) 1
127.0.0.1:6579> EXHGET k1 f1
"v1"
127.0.0.1:6579> EXHSET k1 f2 v2 ex 10
(integer) 1
127.0.0.1:6579> EXHGET k1 f1
(nil)
127.0.0.1:6579> EXHGETALL k1
127.0.0.1:6579> EXHGETALL k1
(empty array)
```

field支持version示例:

```
127.0.0.1:6579> EXHSET k1 f1 v1
(integer) 1
127.0.0.1:6579> EXHVER k1 f1
(integer) 1
127.0.0.1:6579> EXHSET k1 f1 v1 ver 2
(error) ERR update version is stale
127.0.0.1:6579> EXHSET k1 f1 v1 ver 1
(integer) 0
127.0.0.1:6579> EXHVER k1 f1
(integer) 2
127.0.0.1:6579> EXHSET k1 f1 v1
(integer) 0
127.0.0.1:6579> EXHVER k1 f1
(integer) 3
127.0.0.1:6579> EXHSET k1 f1 v1 GT 3
(error) ERR update version is stale
127.0.0.1:6579> EXHSET k1 f1 v1 GT 2
(error) ERR update version is stale
127.0.0.1:6579> EXHSET k1 f1 v1 GT 4
(integer) 0
127.0.0.1:6579> EXHVER k1 f1
(integer) 4
127.0.0.1:6579> EXHSET k1 f1 v1 abs 2
(integer) 0
127.0.0.1:6579> EXHVER k1 f1
(integer) 2
```

• EXHINCRBY

表 5-9 EXHINCRBY 命令介绍

类别	说明
语法	EXHINCRBY key field num [EX time] [EXAT time] [PX time] [PXAT time] [VER GT ABS version] [MIN minval] [MAX maxval] [KEEPTTL]
命令描述	将Key指定的ExHash中一个field的value增加num，num为一个整数。如果ExHash不存在则自动新创建一个，如果指定的field不存在，则在加之前插入该field并将其值设置为0。 为Key的field设置了超时时间后，再次执行该命令时如果没有设置超时时间，该field将被设置为永不过期。
选项	Key：ExHash的key，用于指定作为命令调用对象的ExHash。
	field：ExHash中的一个元素，一个ExHash key可以有多个field。

类别	说明
	num：需要为field的value增加的整数值。
	EX：指定field的相对过期时间，单位为秒，为0表示马上过期，不传此参数表示不过期。
	EXAT：指定field的绝对过期时间，单位为秒，为0表示马上过期，不传此参数表示不过期。
	PX：指定field的相对过期时间，单位为毫秒，为0表示马上过期，不传此参数表示不过期。
	PXAT：指定field的绝对过期时间，单位为毫秒，为0表示马上过期，不传此参数表示不过期。
	VER：版本号。如果field存在，和当前版本号做比较：如果相等，继续操作，且版本号加1。如果不相等，返回异常。如果field不存在或者field当前版本为0，忽略传入的版本号并继续操作，成功后版本号变为1。
	GT：比当前更大的版本号（Greater Than），设置的版本号如果比当前的版本号小，则返回失败。
	ABS：绝对版本号，不论field是否存在，可以在插入field时设置为本参数所指定的版本号。
	KEEPTTL：在不指定EX、EXAT、PX或PXAT选项时，使用KEEPTTL选项会保留field当前的过期设置。
	MIN：value的最小值，小于该值则提示异常。
	MAX：value的最大值，大于该值则提示异常。
返回值	成功：与num相加后value的值。
	其它情况返回异常。

- 示例

MIN，MAX选项使用示例：

```
127.0.0.1:6579> EXHINCRBY k1 f1 5 min 6
(error) ERR increment or decrement would overflow
127.0.0.1:6579> EXHINCRBY k1 f1 5 min 4
(integer) 5
127.0.0.1:6579> EXHINCRBY k1 f1 5 max 9
(error) ERR increment or decrement would overflow
127.0.0.1:6579> EXHINCRBY k1 f1 3 max 9
(integer) 8
```

ExHash 命令使用示例

JAVA(Jedis)

```
package nosql.cloud.huawei.jedis;

import redis.clients.jedis.*;
import redis.clients.jedis.util.SafeEncoder;
```

```
import java.util.ArrayList;

public class Main{
    public static void main(String[] args) throws InterruptedException {
        // 初始化Jedis资源池配置
        JedisPoolConfig jedisPoolConfig = new JedisPoolConfig();
        // 设置资源池中的最大连接数
        jedisPoolConfig.setMaxTotal(10);
        // 设置资源池允许的最大空闲连接数
        jedisPoolConfig.setMaxIdle(10);
        // 设置资源池确保的最少空闲连接数
        jedisPoolConfig.setMinIdle(2);

        // 根据配置初始化Jedis资源池
        // note: 若版本不支持user ACL机制, user必须为null
        JedisPool jedisPool = new JedisPool(jedisPoolConfig, "127.0.0.1", 6379, null, "*****");

        // 从资源池里获取连接
        try (Jedis jedis = jedisPool.getResource()) {
            // example for: EXHSET key field value [EX time] [EXAT time] [PX time] [PXAT time] [NX | XX] [VER
            ABS | GT version] [KEEPTTL]
            jedis.sendCommand(() -> SafeEncoder.encode("exhset"), "key", "field1", "value1");
            jedis.sendCommand(() -> SafeEncoder.encode("exhset"), "key", "field2", "value2", "EX", "5");

            // example for: EXHGET key field
            byte[] byteArray = (byte[]) jedis.sendCommand(() -> SafeEncoder.encode("exhget"), "key", "field1");
            System.out.println(new String(byteArray));
            byteArray = (byte[]) jedis.sendCommand(() -> SafeEncoder.encode("exhget"), "key", "field2");
            System.out.println(new String(byteArray));

            // example for: EXHGETALL key
            ArrayList<byte[]> byteArrayList = (ArrayList<byte[]>) jedis.sendCommand(() ->
            SafeEncoder.encode("exhgetall"), "key");
            for (byte[] ba : byteArrayList) {
                System.out.print(new String(ba));
                System.out.print(" ");
            }
            System.out.println();

            // sleep for 5 seconds
            Thread.sleep(5000);

            // exhgetall after sleeping
            byteArrayList = (ArrayList<byte[]>) jedis.sendCommand(() -> SafeEncoder.encode("exhgetall"),
            "key");
            for (byte[] ba : byteArrayList) {
                System.out.print(new String(ba));
                System.out.print(" ");
            }
        }

        // 关闭资源池
        jedisPool.close();
    }
}
```

ExHash命令最佳实践请参考[通过ExHash实现广告频控业务方案概述](#)。

5.8 大 Bitmap 初始化

Bitmap，即位图类型，开源Redis直接使用STRING类型表达，因此可能会产生超大的STRING数据，进而在某些场景下出现大KEY的性能问题。GeminiDB Redis的Bitmap类型采用的是特殊编码的格式，内部采用分片算法，可以规避产生一个超大的STRING数据，并且可以支持更高效的随机位数的插入和删除操作。

但是，实际应用场景中，我们可能会从其它地方获取一个超大的Bitmap数据，而这些数据通常会用STRING类型来表达。对于一个超大的Bitmap数据，例如 64 MB，如果直接使用SET命令插入GeminiDB Redis，会执行较长时间，并且对其它正常访问产生干扰，造成时延抖动。因此我们提供了一套平滑的插入方案，其原理是，对于超大的初始数据，我们先将其拆分为较小的字串（例如1MB），然后首次插入仍然采用SET命令，然后通过一个GETBIT的只读命令将其转化为Bitmap类型，后续的字串，通过APPEND命令进行插入即可。

注意事项

- 目前内核版本5.0.5.00版本及以上版本支持该功能，您可以在管理控制台右上角，选择“[工单 > 新建工单](#)”提工单联系客服咨询实例版本是否支持该功能。如需使用该功能，您可以参考[升级内核小版本](#)升级内核版本。
- 由于APPEND命令对顺序有要求，因此整个流程要避免出现APPEND乱序（并发APPEND的场景）。
- 可以使用PIPELINE模式加速，PIPELINE本身也是保证执行顺序的，因此不会有乱序的问题。
- 拆分的粒度可以根据实际情况选择，拆得越细，产生的时延毛刺就越小，但是初始化时间就越长，通常建议256KB-1MB左右的值。

代码参考

C++示例

```
#include <string>
#include <vector>

#include "hiredis/hiredis.h"
constexpr std::size_t kBitmapSubSize = 1024 * 1024; // 1 MB
void SmoothInitBitmap(std::string bitmap) {
    // Split bitmap
    std::vector<std::string> sub_bitmaps;
    std::size_t pos = 0;
    while (pos < bitmap.size()) {
        sub_bitmaps.emplace_back(bitmap.substr(pos, kBitmapSubSize));
        pos += kBitmapSubSize;
    }
    std::string key = "BITMAP_KEY";
    // Connect to redis
    redisContext* redis = redisConnect("127.0.0.1", 6666);
    redisReply* reply = nullptr;
    // First part use 'SET' command
    reply = (redisReply*)redisCommand(redis, "SET %b %b", key.data(), key.size(), sub_bitmaps[0].data(),
    sub_bitmaps[0].size());
    freeReplyObject(reply);
    // Use 'GETBIT' to transform to bitmap format
    reply = (redisReply*)redisCommand(redis, "GETBIT %b 0", key.data(), key.size());
    freeReplyObject(reply);
    // Use 'APPEND' for remaining bitmap data
    for (auto i = 1u; i < sub_bitmaps.size(); ++i) {
        reply = (redisReply*)redisCommand(redis, "APPEND %b %b", key.data(), key.size(), sub_bitmaps[i].data(),
        sub_bitmaps[i].size());
        freeReplyObject(reply);
    }
}

int main() {
    std::string bitmap
    ="123457890abcdef123457890abcdef123457890abcdef123457890abcdef123457890abcdef123456";
    SmoothInitBitmap(bitmap);
}
```

JAVA(Jedis)示例

```
package nosql.cloud.huawei.jedis;

import redis.clients.jedis.Jedis;

import java.nio.ByteBuffer;
import java.util.BitSet;

public class BitMapOperation {
    private Jedis jedis;

    public BitMapOperation(Jedis jedis) {
        this.jedis = jedis;
    }

    /**
     * SetBit operation especially for big bitmap
     *
     * @param key      key
     * @param value    value
     * @param groupLength groupLength (Unit: byte)
     */
    public void setBitGrouped(byte[] key, BitSet value, int groupLength) {
        if (value.isEmpty()) {
            jedis.set(key, new byte[0]);
            return;
        }

        byte[] byteArray = disposeBitMap(value);

        // round count
        int round = byteArray.length % groupLength == 0 ? byteArray.length / groupLength : byteArray.length / groupLength + 1;
        // last round length
        int lastPacketLength = byteArray.length % groupLength == 0 ? groupLength : byteArray.length % groupLength;

        if (round == 1) {
            // if only one round
            byte[] lastPacketByte = new byte[lastPacketLength];
            System.arraycopy(byteArray, 0, lastPacketByte, 0, lastPacketLength);
            // set and getBit
            setAndGetBit(key, lastPacketByte);
            return;
        }

        byte[] packetByte = new byte[groupLength];
        byte[] lastPacketByte = new byte[lastPacketLength];
        for (int i = 0; i < round; i++) {
            if (i == 0) {
                // first set
                System.arraycopy(byteArray, i * groupLength, packetByte, 0, groupLength);
                // set and getBit
                setAndGetBit(key, packetByte);
            } else if (i != round - 1) {
                // regular append
                System.arraycopy(byteArray, i * groupLength, packetByte, 0, groupLength);
                jedis.append(key, packetByte);
            } else {
                // last append
                System.arraycopy(byteArray, i * groupLength, lastPacketByte, 0, lastPacketLength);
                jedis.append(key, lastPacketByte);
            }
        }
    }

    private byte[] disposeBitMap(BitSet bitSet) {
        // get words and count the number of word(Long)
        long[] words = bitSet.toLongArray();
    }
}
```



```
int n = words.length;
if (n == 0)
    return new byte[0];
for (int i = 0; i < n; i++) {
    // reverse
    words[i] = reverseLong(words[i]);
}
return longToBytes(words);
}

public static byte[] longToBytes(long[] longArray) {
    ByteBuffer buffer = ByteBuffer.allocate(longArray.length * 8);
    for (long value : longArray) {
        buffer.putLong(value);
    }
    return buffer.array();
}

public void setAndGetBit(byte[] key, byte[] value) {
    jedis.set(key, value);
    jedis.getbit(key, 0);
}

public static long reverseLong(long n) {
    n = (n >> 32) | (n << 32);
    n = ((n & 0xFFFF0000FFFF0000L) >>> 16) | ((n & 0x0000FFFF0000FFFFL) << 16);
    n = ((n & 0xFF00FF00FF00FF00L) >>> 8) | ((n & 0x00FF00FF00FF00FFL) << 8);
    n = ((n & 0xF0F0F0F0F0F0F0F0L) >>> 4) | ((n & 0x0F0F0F0F0F0F0F0FL) << 4);
    n = ((n & 0xCCCCCCCCCCCCCCCCCL) >>> 2) | ((n & 0x3333333333333333L) << 2);
    n = ((n & 0xAAAAAAAAAAAAAAAAAL) >>> 1) | ((n & 0x5555555555555555L) << 1);
    return n;
}
}
```

Python 示例

```
import redis
import random
import string
from bitmap import BitMap # pip install bitmap
# 参数
max_bytes = 1024 * 1024 * 64 # 构造一个64MB的bitmap
max_bits = max_bytes * 8 # 一个byte可以存储8个bit, 对应大概5亿多元素
# 这个方案不需要python内置的bitmap类型
# index_list 存储了所有要设置为1的下标
index_list = []
for i in range(1000000):
    index_list.append(random.randint(0, max_bits - 1))
# 使用bytearray构造位图
byte_array = bytearray(max_bytes)
for i in index_list:
    index = i // 8
    offset = i % 8
    byte_array[index] |= (1 << (7 - offset))
# 转化成bytes类型, 用于后续的redis操作
bitmap_str = bytes(byte_array)
# 连接到redis
r = redis.Redis(host='127.0.0.1', port=6379)
r.execute_command("auth a")
key = "BITMAP_KEY"
# 分割参数
bitmap_pos = 0
bitmap_sub_size = 256 * 1024 # 调整分片大小
step = bitmap_sub_size - 1
# 处理第一部分
first_part = bitmap_str[bitmap_pos : bitmap_pos + step]
r.execute_command("SET", key, first_part)
r.execute_command("GETBIT", key, 0) # 使用getbit进行bitmap编码优化
# 处理剩余的部分
```

```
bitmap_pos += step
while bitmap_pos < len(bitmap_str) :
    rest_part = bitmap_str[bitmap_pos : bitmap_pos + step]
    r.execute_command("APPEND", key, rest_part)
    bitmap_pos += step
# 下面是测试验证的代码，注意会比较耗时，因为需要执行100w次getbit进行验证
# 注意，最后一个bitcount命令是O(n)命令，会产生百毫秒的毛刺，请勿随意在生产环境使用
# 构造一个python内置的bitmap类型进行数据验证(可选)
bm = BitMap(max_bits)
for i in index_list:
    bm.set(i)
print('BitMap.count(): ' + str(bm.count()))
# 调用redis命令校验是否设置正确
success = True
for i in index_list:
    if r.execute_command("GETBIT", key, i) != 1:
        print('GETBIT check error, pos is' + str(i))
        success = False

if success:
    print('GETBIT check success')

print("Bitcount: " + str(r.execute_command("BITCOUNT", key)))
```

5.9 大 Bitmap 分页查询

针对大位图（Bitmap）或指定区间[start, end]子位图的获取场景，传统实现方案存在两大瓶颈：

- **低效数据拼接问题**：若客户端通过循环调用GETBIT逐位获取数据并拼接，会产生高频次网络请求与计算开销，存在显著的性能瓶颈。
- **大Key访问风险**：若直接使用GET获取完整位图，数据库层需传输超大二进制数据块，易引发网络阻塞及服务端延迟波动，影响系统稳定性。

为此，数据库侧提供增强型位图操作命令RANGEBITARRAY，其核心优势为：

- **分块高效查询**：支持通过范围参数直接获取位图指定区间的二进制数据块，客户端可通过多次调用完成大数据集的分段获取与组装。
- **规避大Key风险**：通过按需分块传输机制，避免单次操作触及大Key的性能临界点，有效降低服务端延迟抖动。
- **端到端性能优化**：减少网络往返次数与服务端资源消耗，实现查询吞吐量与响应稳定性的双重提升。

建议在海量位图数据读取场景优先使用此命令。

表 5-10 RANGEBITARRAY

类别	说明
语法	RANGEBITARRAY <i>key start end</i>
时间复杂度	O(C)，C表示[start, end]区间长度。
命令描述	获取Bitmap指定区间中所有bit值（0、1）组成的字符串。
选项	● Key：Key名称（Bitmap数据结构）。 ● start：起始的偏移量（包含该值）。 ● end：结束的偏移量（包含该值）。

类别	说明
返回值	- 执行成功：返回指定区间中所有bit值（0、1）组成的字符串。 - 若key不存在：返回空字符串。 - 其他情况返回相应的异常信息（如“不支持老编码”和“Key必须为Bitmap类型”）。
示例	提前执行如下命令预置数据： <pre>SETBIT foo 2 1 SETBIT foo 3 1 SETBIT foo 5 1</pre> - 命令示例：<pre>RANGEBITARRAY foo 0 5</pre> - 返回示例：<pre>"001101"</pre>

注意事项

- 您可以在管理控制台右上角，选择“[工单 > 新建工单](#)”联系客服咨询实例版本是否支持该功能。如需使用该功能，您可以参考[升级内核小版本](#)升级内核版本。
- Key必须是Bitmap类型，STRING类型需先用GETBIT命令进行转换。
- start和end的语义与GETRANGE命令相同。
- 客户端如果需要多次执行RANGEBITARRAY并进行拼接，可以使用PIPELINE模式加速。
- 范围查询的粒度可以根据实际情况选择，粒度越细，产生的时延毛刺就越小；粒度越粗，产生的时延毛刺就越大，通常建议32KB的范围查询粒度，再进行拼接。

代码参考

Java（jedis）示例

```
import redis.clients.jedis.Jedis;
import redis.clients.jedis.JedisPool;
import redis.clients.jedis.util.SafeEncoder;
import redis.clients.jedis.Protocol;
import java.nio.charset.StandardCharsets;

public class BitmapRangePager {

    private final JedisPool jedisPool;
    private static final int PAGE_SIZE_BYTES = 32 * 1024; // 32KB分页（基于字节单位）

    public BitmapRangePager(JedisPool jedisPool) {
        this.jedisPool = jedisPool;
    }

    public String getFullBitmap(String key) {
        try (Jedis jedis = jedisPool.getResource()) {
            // 获取Bitmap总字节长度
            long totalBytes = jedis.strlen(key);
            if (totalBytes == 0) return "";

            StringBuilder result = new StringBuilder();

            // 基于字节的分页循环（更直观）
```

```
for (long byteOffset = 0; byteOffset < totalBytes; byteOffset += PAGE_SIZE_BYTES) {
    // 计算当前页的字节数（最后一页可能不足32KB）
    int currentPageBytes = (int) Math.min(PAGE_SIZE_BYTES, totalBytes - byteOffset);

    // 转换为bit区间（闭区间）
    long bitStart = byteOffset * 8;
    long bitEnd = (byteOffset + currentPageBytes) * 8 - 1; // 包含end位

    // 执行命令并拼接结果
    String pageBits = executeRangeBitArray(jedis, key, bitStart, bitEnd);
    result.append(pageBits);
}
return result.toString();
}
}

private String executeRangeBitArray(Jedis jedis, String key, long start, long end) {
    // 发送自定义命令RANGEBITARRAY key start end
    Object response = jedis.sendCommand(
        new Protocol.Command("RANGEBITARRAY"),
        SafeEncoder.encode(key),
        Protocol.toByteArray(start),
        Protocol.toByteArray(end)
    );

    // 处理响应（确保二进制转字符串）
    if (response instanceof byte[]) {
        return new String((byte[]) response, StandardCharsets.US_ASCII);
    } else {
        return response.toString();
    }
}
}
```

Python 示例

```
import redis
import math

class BitmapRangePager:
    def __init__(self, redis_client):
        self.redis = redis_client
        self.PAGE_SIZE_BYTES = 32 * 1024 # 32KB 分页（基于字节单位）

    def get_full_bitmap(self, key):
        """
        获取完整的 Bitmap 字符串
        :param key: Redis 键名
        :return: 由 '0'/'1' 组成的完整字符串
        """
        # 获取 Bitmap 总字节数
        total_bytes = self.redis.strlen(key)
        if total_bytes == 0:
            return ""

        result = []
        # 基于字节的分页循环
        for byte_offset in range(0, total_bytes, self.PAGE_SIZE_BYTES):
            # 计算当前页的字节数（最后一页可能不足 32KB）
            current_page_bytes = min(self.PAGE_SIZE_BYTES, total_bytes - byte_offset)

            # 转换为 bit 区间闭区间 [start, end]
            bit_start = byte_offset * 8
            bit_end = (byte_offset + current_page_bytes) * 8 - 1

            # 执行命令并收集结果
            page_bits = self.execute_range_bitarray(key, bit_start, bit_end)
            result.append(page_bits)
```

```
        return ''.join(result)

    def execute_range_bitarray(self, key, start, end):
        """
        执行自定义命令 RANGEBITARRAY key start end
        :return: 由 '0'/'1' 组成的字符串
        """
        # 发送自定义命令（需服务端支持）
        response = self.redis.execute_command(
            "RANGEBITARRAY", # 注意实际命令名可能需要调整
            key,
            start,
            end
        )
        # 处理响应（redis-py 返回 bytes 类型）
        return response.decode('ascii') # 假设响应是 ASCII 字符串

# 使用示例
if __name__ == "__main__":
    # 创建 Redis 客户端
    r = redis.Redis(host='localhost', port=6379, decode_responses=False) # 保持原始 bytes 响应

    pager = BitmapRangePager(r)
    full_bitmap = pager.get_full_bitmap("my_large_bitmap")

    # 输出示例（仅打印前100字符）
    print(f"Bitmap (first 100 chars): {full_bitmap[:100]}")
    print(f"Total length: {len(full_bitmap)} bits")
```

5.10 客户端连接池参数配置

合理的连接池参数设置能够有效提升客户端对Redis的使用效率，不合理的配置（如连接数配置过小）可能引起应用程序无法获取连接，影响生产业务。本文以Redis客户端Jedis的连接池JedisPool为例，介绍JedisPool的使用和参数进行说明，为业务开发人员提供最佳配置的参考。

使用方法

以Jedis 4.3.1版本为例，下面是Maven依赖配置：

```
<dependency>
  <groupId>redis.clients</groupId>
  <artifactId>jedis</artifactId>
  <version>4.3.1</version>
  <scope>compile</scope>
</dependency>
```

Jedis使用Apache Commons-pool2对连接池进行管理，在定义JedisPool时需注意其关键参数GenericObjectPoolConfig（连接池）。该参数的使用示例如下：

```
GenericObjectPoolConfig<Jedis> config = new GenericObjectPoolConfig<>();
config.setMaxTotal(100);
config.setMaxIdle(50);
config.setMinIdle(5);
config.setTestWhileIdle(true);
....
```

Jedis的初始化方法如下：

```
JedisPool pool = new JedisPool(config, host, port, timeout, password); // 创建连接池
try (Jedis jedis = pool.getResource()) { // 获取连接，执行完自动释放连接
    // 利用Jedis客户端执行命令
} catch (Exception e) {
    e.printStackTrace();
}
pool.close(); // 关闭连接池
```

参数说明

Jedis连接就是连接池中JedisPool管理的资源，JedisPool保证资源在一个可控范围内，并且保障线程安全。使用合理的GenericObjectPoolConfig配置能够提升Redis的服务性能，降低资源开销。下将对一些重要参数进行说明，并提供设置建议。

表 5-11 Jedis 常用参数配置说明

Jedis参数	参数说明	默认值	建议值
maxTotal	当前资源池可并发的最大连接数，单位：个。 Redis连接数要根据具体的业务量进行设置，连接数太大浪费资源，过小无法获取连接，影响业务。	8	客户端节点数*maxTotal不能超过Redis的最大连接数。 假设一个连接的QPS大约是1s/1ms = 1000，而业务期望的单个Redis的QPS是50000，那么理论上需要的资源池大小（即MaxTotal）是50000 / 1000 = 50。
maxIdle	资源池中最大的空闲连接数，单位：个。 达到后资源池会开始回收空闲连接，直到空闲连接数达到minIdle个数。主要避免空连接占用，资源浪费。	8	maxIdle实际上才是业务需要的最大连接数，maxTotal是为了给出余量，所以maxIdle不要设置得过小，否则会有new Jedis（新连接）开销。
minIdle	资源池中保持最小的空闲可用连接数，单位：个 这部分连接数不会被回收，可防止流量增量时，连接创建不及时。	0	10~20
maxWaitMillis	资源池连接用尽后，调用者的最大等待时间，单位：毫秒。	-1	建议设置一个合理的超时时间，避免出现当连接池用尽后，应用阻塞不响应的情况。
testWhileIdle	是否在空闲资源监测时通过PING命令监测连接有效性，无效连接将被销毁。	false	true
testOnBorrow	每次向资源池获取连接时是否做连接有效性检测（发送PING请求），无效连接会被释放。	false	建议使用默认值。设为true相当于在每个命令执行完前先发一个PING命令，对高并发请求应用的性能有影响。对于业务可用性要求比较高的场景，可以设为true，可以保证连接有效。
testOnReturn	每次向连接池归还连接时是否做连接有效性检测（发送PING请求），无效连接会被释放。	false	建议使用默认值。设为true相当于在每个命令执行完后再发一个PING命令，对高并发请求应用的性能有影响。

Jedis参数	参数说明	默认值	建议值
timeout	Jedis的socket timeout值，单位：毫秒。	2000	200~1000

5.11 通过 openAPI 实现自定义周期备份

游戏业务经常选用GeminiDB Redis存储玩家数据，兼顾低时延和数据持久化特性。本文介绍通过openAPI编写手动备份脚本，实现自定义备份周期（例如每2小时或4小时备份一次）。

GeminiDB Redis备份时仅需在存储层打秒级快照并上传OBS保存，因此速度极快，且对业务影响小（优于自建开源Redis的全量dump方案）。

注意事项

- 触发备份时，可能会产生轻微的瞬时抖动。
- 备份上传时，需占用一定CPU和带宽资源。
- 手动备份默认永久保存，控制台支持批量删除不需要的备份。

操作步骤

步骤1 pom文件导入样例，详情请参考[SDK官方文档](#)：

```
<dependency>
  <groupId>com.huaweicloud.sdk</groupId>
  <artifactId>huaweicloud-sdk-gaussdbfornosql</artifactId>
  <version>3.1.92</version>
</dependency>
```

认证信息配置auth：

```
// 配置认证信息
// 可通过环境变量等方式配置认证信息。
ICredential auth = new BasicCredentials()
    .withAk(System.getenv("HUAWEICLOUD_SDK_AK"))
    .withSk(System.getenv("HUAWEICLOUD_SDK_SK"));
```

withAk、withSk是使用账号下的AK、SK，获取方式可以参考[获取AK/SK](#)。

步骤2 获取GeminiDB服务客户端：

```
// 创建服务客户端
VpcClient client = VpcClient.newBuilder()
    .withCredential(auth)
    .withRegion(VpcRegion.valueOf("cn-north-4"))
    .build();
```

此示例是以Vpc服务为参考，需要将VpcClient 修改为：GaussDBforNoSQLClient。

- withCredential：认证信息，传入第1步中构造的信息auth即可。
- withRegion：将VpcRegion替换成GaussDBforNoSQLRegion即可，Region信息可以在[APIE](#)上查看获取。



步骤3 构造请求：

```
// 创建请求
ListVpcsRequest request = new ListVpcsRequest();
```

将ListVpcsRequest修改为CreateBackRequest，其中CreateBackRequest需要设置如下两个值，具体方法请参考[创建手动备份](#)：

- request.setInstanceld("instancelid"); // 需要创建备份的实例ID。
- request.setBody(backupRequestBody); // 备份信息：备份名称和备份描述。

步骤4 调用API：

```
ListVpcsResponse response = client.listVpcs(request);
System.out.println(response.toString());
```

使用[步骤2](#)获取的服务客户端调用创建备份接口：

```
CreateBackResponse backResponse = client.createBack(request);
```

步骤5 加入定时触发逻辑（也可以自己选择其他类似方式实现）。

需用户编写一个定时任务方法，将[步骤1](#)至[步骤4](#)的代码放入该方法中。在方法上使用@Scheduled(cron = "0 0/1 * * * ?")进行配置调度时间策略，如下为具体参考配置示例：

Cron表达式：[秒] [分] [小时] [日] [月] [周] [年]。

- *表示所有值。在分钟里表示每一分钟触发，在小时、日期、月份等里面表示每一小时、每一日、每一月。
- ? 表示不指定值。表示不关心当前位置设置的值，比如不关心是周几，则周的位置填写?。
- -表示区间。小时设置为10-12，表示10点、11点、12点均会触发。
- ,表示多个值。小时设置成10,12，表示10点和12点会触发。
- /表示递增触发。5/15表示从第5秒开始，每隔15秒触发。
- L表示最后，只能出现在日和周两个域。L前加数据，表示在该数据的最后一个触发，如6L表示在最后一个星期五触发。
- W表示离指定日期最近的工作日触发。15W表示离该月15号最近的工作日触发。
- #表示每月的第几个周几。6#3表示该月的第三个周五触发。

配置方式举例：

- "0 15 10 ? * *"表示每天上午10:15触发。
- "0 15 10 * * ? 2024"表示2024年的每天上午10:15触发。
- "0 0/5 14 * * ?"表示在每天下午2点到下午2:55期间的每5分钟触发。
- "0 0/5 14,18 * * ?"表示在每天下午2点到下午2:55期间和下午6点到下午6:55期间的每5分钟触发。

- "0 0 */6 * * *"表示每6个小时触发。

----结束

5.12 通过并行 SCAN 提升全库扫描速度

当实例中的key数量较多时，传统的SCAN命令扫描耗时久，开源Redis较难解决该痛点问题。GeminiDB Redis基于分布式架构，支持用户同时对多个数据分区进行并行扫描，达到并行提速效果。

注意事项

- 该方案仅适用于GeminiDB Redis集群版实例。
- 当为SCAN命令指定了PARTITION参数时，返回的cursor在继续扫描时必须指定同一个partition，不能临时更换partition，否则会出现扫描数据不符合预期或者命令报错等情况。

操作步骤

步骤1 获取GeminiDB Redis实例的全部分区信息，用于后续的并行扫描。

数据分区：GeminiDB Redis集群实例底层存在许多数据分区，分布在不同节点上，每个分区名称为一个16字符长度的ID。一个实例底层的数据分区名称和总数量在实例创建时就固定，不会随实例变更而发生改变。

获取数据分区列表：通过执行INFO ROUTE命令可以获取GeminiDB Redis实例的全部数据分区。如下样例中，查询实例共有4个数据分区：efb06d5c7a4ecb31, c7a36e9eee0103c1, 6fd3dfdbcca37686, 7f7666870a88501b。

```
127.0.0.1:6379>info route
# Route
server: 127.0.0.1:16379 //展示第一个节点上的数据分区
  efb06d5c7a4ecb31 //数据分区
  c7a36e9eee0103c1 //数据分区
server: 127.0.0.1:26379 //展示第二个节点上的数据分区
  6fd3dfdbcca37686 //数据分区
  7f7666870a88501b //数据分区
```

步骤2 启动多个SCAN任务，分别扫描不同数据分区，实现并行提速。

在开源语法基础上，GeminiDB Redis的SCAN命令额外支持了新参数“PARTITION”，支持扫描指定数据分区。因此，用户可基于该特性灵活编写并行扫描脚本，同时对多个数据分区进行SCAN，实现高达数倍的扫描性能提升。

- 标准SCAN命令语法请参考[SCAN](#)。
- GeminiDB Redis新增可选的“PARTITION”参数语法参考：
SCAN cursor [MATCH pattern] [COUNT count] [TYPE type] [PARTITION partition_index]

其中，MATCH/COUNT/TYPE参数语法与开源Redis一致；

- PARTITION参数：指定扫描某个数据分片，当SCAN命令返回的cursor游标为0时，说明该数据分区已扫描完成。
- partition_index: 是全部数据分片ID的字典序号，从0开始。例如：当实例中共有4个数据分区，则全部分区的partition_index为[0,3]；当实例中有240个数据分区，则全部分区的partition_index为[0,239]。例如：

```
127.0.0.1:6379> scan 0 count 2 partition 1
1) "1125900712148994"
```

```
2) 1) "memtier-1"  
2) "memtier-12"
```

----结束

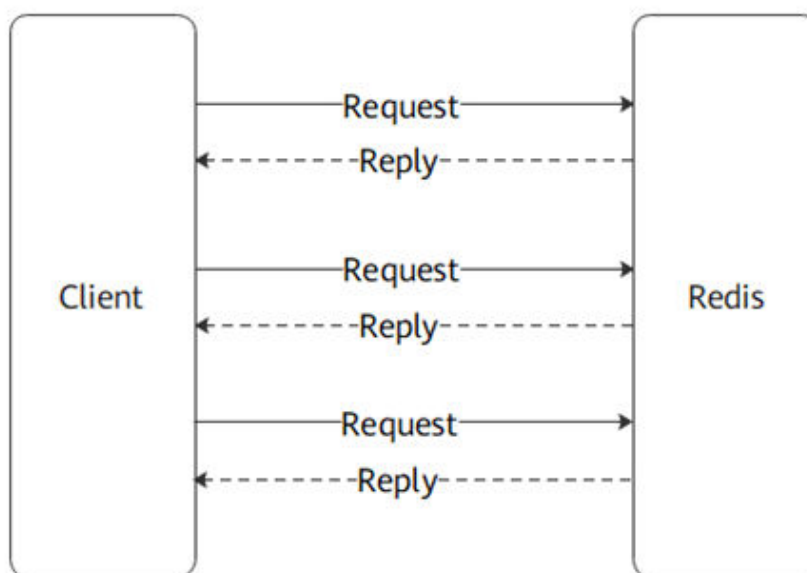
5.13 使用 Pipeline 访问 GeminiDB Redis

本文介绍了使用 Pipeline 访问 GeminiDB Redis 的原理以及注意事项。

Pipeline 简介

Redis 是一个请求/响应模型的服务，通常执行一个命令的流程为：

图 5-10 Redis 命令执行流程

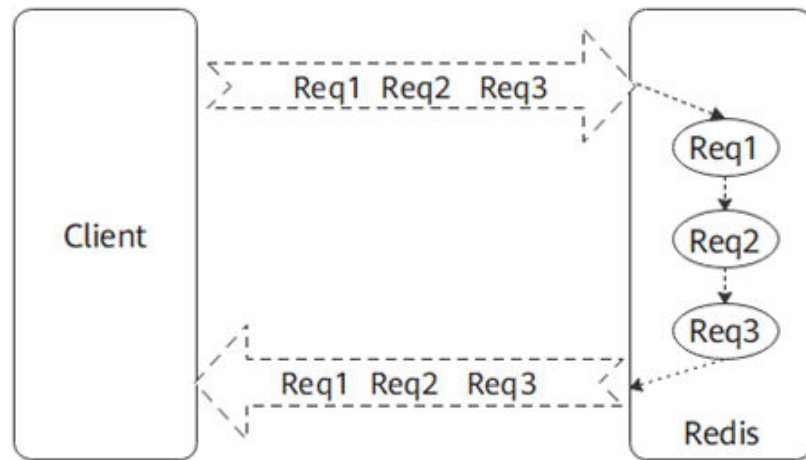


1. 客户端发送命令到 Redis 服务器。
2. Redis 服务器收到命令，排队等待处理。
3. Redis 服务器执行命令。
4. Redis 服务器返回结果给客户端。

上述4个步骤中，步骤1和4是IO操作，速度慢并且受网络状况影响，很容易成为瓶颈点。

为了减少网络开销，充分发挥 Redis 高性能的优势，可以使用 Pipeline 流水线执行命令。Pipeline 机制通过将一组 Redis 命令进行组装，一次发送多个命令，并在执行完成后一次性返回结果集，可以减少网络传输带来的开销，流程如下图所示：

图 5-11 使用 Pipeline 访问 Redis



上图中的 Pipeline 将3条命令封装成一组，只需要进行一次网络IO就完成了3条命令的执行。

除了网络层面的开销，使用 Pipeline 还可以减少客户端/服务器的 read()/write() 调用，提高程序的执行效率。

Pipeline 大小选择及注意事项

虽然使用 Pipeline 可以节省网络IO开销，但 Pipeline 并不是越大越好；使用 Pipeline 对程序性能的提升是有上限的，随着 Pipeline 逐渐增加，提高 Pipeline 的收益逐渐降低；如果一次组装的 Pipeline 数据量过大，一方面会导致客户端等待时间增加，另一方面，如果大的 Pipeline 导致 socket buffer 写满，可能会带来网络阻塞，反而引起性能下降；

根据经验，30 - 100 大小的 Pipeline 就已经可以充分发挥数据库的性能，具体最佳值与实际业务有关，建议以实际测试为准。

其它注意事项：

- Pipeline 不保证原子性，服务端在处理批量命令时，解析出多个单命令并按顺序执行，各个命令相互独立，服务端有可能在该过程中执行其他客户端的命令。某个命令执行失败不会影响其他命令的执行，如需原子性，需要使用事务或lua实现。
- 单次 Pipeline 的大小需要适当，大Pipeline可能会有OOM风险，也可能造成网络阻塞；因为Redis 必须在处理完所有命令前，先缓存起所有命令的处理结果。这样会有内存的消耗，过大Pipeline造成内存上涨甚至OOM；对于单个命令数据长度较大的场景，需要适当地减小 Pipeline。
- 对于时延敏感的场景，不建议使用大 Pipeline，需要结合业务场景选择合适的 Pipeline 大小。

5.14 GeminiDB Redis 事务处理

事务介绍

事务将应用程序的多个读、写操作捆绑在一起成为一个逻辑操作单元，是一个执行的整体，整个事务要么成功，要么失败。在一个连接中，当客户端执行multi命令后，redis开始将后续收到的命令缓存在队列中，当客户端发送exec命令时，redis按照顺序依次执行队列中的所有命令。如果有一个命令执行失败则事务回滚，所有命令要么全部成功，要么全部失败。

表 5-12 相关命令

命令	含义
WATCH	用于监视一个或多个key，如果事务执行之前该key被改动，则事务被打断。
UNWATCH	用于取消watch命令对所有key的监视。
MULTI	用于标识一个事务块的开始。
EXEC	用于执行事务块内的所有命令。
DISCARD	用于取消事务块，放弃执行事务块内的所有命令。

注意

- 使用proxy集群时，为保证事务执行的原子性，需确保事务中的所有key拥有相同的hashtag。若不使用hashtag，事务会被拆分成普通命令执行，则无法保证事务的原子性。
- 由于整个事务的执行是原子性的，要么全部成功要么全部失败，因此，在编写事务时，需要注意命令的合法性。
- 由于事务中的命令是按照顺序执行的，所以，在编写事务时，需要注意命令的先后顺序。
- 使用事务时应遵循轻量化原则，避免单次事务中打包过多命令或复杂度过高的命令，执行的事务中包含命令过多会引起请求阻塞或实例状态异常。

代码示例

Client在事务中同时修改key1和key2，key1和key2要么同时修改成功，要么同时修改失败。

```
package nosql.cloud.huawei.jedis;
import java.util.List;
import redis.clients.jedis.Jedis;
import redis.clients.jedis.Transaction;

public class TranscationTest {
    private static final String host = "127.0.0.1";
    private static final int port = 6379;
```

```
private static final String pwd = "password";

private static Jedis jedis;
static {
    jedis = new Jedis(host, port);
    String authString = jedis.auth(password);
    if (!authString.equals("OK")) {
        jedis.close();
        jedis = null;
    }
}

public static void main(String[] args) {
    if (jedis == null) {
        return;
    }

    String str_key1 = "{str}key1";
    String str_key2 = "{str}key2";
    jedis.set(str_key1, "0");
    jedis.set(str_key2, "0");
    jedis.watch(str_key1);
    // 启动事务处理
    Transaction tx = jedis.multi();
    tx.set(str_key1, "500");
    tx.get(str_key1);
    tx.set(str_key2, "1000");
    tx.get(str_key2);
    List<Object> result = tx.exec();
    if (result.isEmpty()) {
        System.out.println("Error: 事务执行中断");
    } else {
        System.out.println("Succ: 事务执行成功");
    }
    System.out.println("str_key1: {}, str_key2: {}", jedis.get(str_key1), jedis.get(str_key2));
    jedis.close();
}
```

5.15 GeminiDB Redis 客户端重试机制

Redis客户端设置重试机制可以在网络不稳定或服务器临时故障的情况下，尽量保持应用程序的高可用性和稳定性。

GeminiDB Redis可能会遇到如下一些临时性故障场景：

原因	说明
触发高可用机制的场景	GeminiDB Redis支持自动监测各个节点的健康状态，在节点宕机后自动触发主备倒换或集群数据分片接管。通常，触发高可用机制可能有以下场景： • 某个节点的GeminiDB进程发生重启（例如OOM/底层故障等）。 • 规格变更/节点扩缩容过程中主动剔除/加入节点。 在这些场景下，客户端可能遇到秒级连接闪断或命令超时等故障。
网络波动	客户端与GeminiDB服务端之间的网络链路通常较为复杂，偶发性的网络抖动、数据丢包重传是难以避免的。一旦触发网络波动，可能会导致客户端发起的请求出现超时。

原因	说明
服务端过载	在数据库服务出现高负载、慢查询等场景，客户端发起的请求可能无法立即响应，从而出现超时。

客户端设置重试机制时，需注意以下原则：

原则	说明
设置合适的重试次数和间隔	应根据实际业务情况，设置合适的重试次数和间隔。重试次数过多可能会延长故障恢复时间，间隔过短可能会对服务器造成额外压力。对于高负载场景，建议使每次重试的间隔时间指数增长，避免在服务器负载高时大量请求同时重试，导致雪崩效应。
避免对非幂等命令执行重试	当客户端检测到命令执行超时后，存在服务端已经执行完命令但是在回包阶段出现异常的场景，此时执行重试可能会导致命令被重复执行。因此，通常推荐只针对幂等操作进行重试（例如SET操作，多次执行得到的结果不变），而对于非幂等操作，则需业务确认是否能容忍重复数据（例如INCR操作，多次操作可能导致计数器额外增加）。
客户端记录日志	建议重试时打印客户端日志，例如连接的IP和端口，报错的命令和涉及的key等，方便问题定位和排查。

以下提供部分语言SDK的代码示例，仅供参考。

Jedis（Java 客户端）

对于JedisPool模式，Jedis版本在4.0.0以上支持重试，以4.0.0为例，该示例展示如何通过设置MAX_ATTEMPTS参数来设置合适的重试次数。

```
package nosql.cloud.huawei.jedis;

import redis.clients.jedis.DefaultJedisClientConfig;
import redis.clients.jedis.HostAndPort;
import redis.clients.jedis.JedisClientConfig;
import redis.clients.jedis.UnifiedJedis;
import redis.clients.jedis.providers.PooledConnectionProvider;
import java.time.Duration;

// UnifiedJedis API supported in Jedis >= 4.0.0
public class UnifiedJedisDemo {
    private static final int MAX_ATTEMPTS = 5;
    private static final Duration MAX_TOTAL_RETRIES_DURATION = Duration.ofSeconds(15);
    public static void main(String[] args) {
        // Basic connection config
        JedisClientConfig jedisClientConfig = DefaultJedisClientConfig.builder().password("xxx").build();
        // Implement retry
        PooledConnectionProvider provider = new
            PooledConnectionProvider(HostAndPort.from("{ip}:{port}"), jedisClientConfig);
        // MAX_ATTEMPTS 设置合适的重试次数
        UnifiedJedis jedis = new UnifiedJedis(provider, MAX_ATTEMPTS, MAX_TOTAL_RETRIES_DURATION);
        try {
            System.out.println("set key: " + jedis.set("key", "value"));
        } catch (Exception e) {
            // Signifies reaching either the maximum number of failures,
            MAX_ATTEMPTS, or the maximum query time, MAX_TOTAL_RETRIES_DURATION
        }
    }
}
```

```
        e.printStackTrace();
    }
}
```

Redisson (Java 客户端)

以下是一个简单的示例，展示如何通过设置RETRY_ATTEMPTS、RETRY_INTERVAL参数来设置合适的重试次数和间隔。

```
package nosql.cloud.huawei.jedis;
import org.redisson.Redisson;
import org.redisson.api.RBucket;
import org.redisson.api.RedissonClient;
import org.redisson.config.Config;

public class RedissonDemo {
    private static final int TIME_OUT = 3000;
    private static final int RETRY_ATTEMPTS = 5;
    private static final int RETRY_INTERVAL = 1500;

    public static void main(String[] args) {
        Config config = new Config();
        config.useSingleServer()
            .setPassword("xxx")
            .setTimeout(TIME_OUT)
            .setRetryAttempts(RETRY_ATTEMPTS)
            .setRetryInterval(RETRY_INTERVAL) // 设置合适的重试次数和间隔
            .setAddress("redis://{ip}:{port}");
        RedissonClient redissonClient = Redisson.create(config);
        RBucket<String> bucket = redissonClient.getBucket("key");
        bucket.set("value");
    }
}
```

Go-redis (Go 客户端)

以下是一个简单的示例，展示如何通过设置MaxRetries、MinRetryBackoff、MaxRetryBackoff参数来设置合适的重试次数和间隔。

```
package main

import (
    "context"
    "fmt"
    "time"
    "github.com/redis/go-redis/v9"
)

var ctx = context.Background()

func main() {
    client := redis.NewClient(&redis.Options{
        Addr:     "localhost:6379",
        Password: "", // no password set
        DB:       0, // use default DB
        MaxRetries: 3, // set max retry times
        MinRetryBackoff: time.Duration(1) * time.Second, // set retry interval
        MaxRetryBackoff: time.Duration(2) * time.Second, // set retry interval
    })

    // Execute command
    err := client.Set(ctx, "key", "value", 0).Err()
    if err != nil {
        panic(err)
    }
}
```

```
// Test
pong, err := client.Ping(ctx).Result()
if err != nil {
    fmt.Println("Failed:", err)
    return
}
fmt.Println("Success:", pong)
}
```

Redis-py (Python 客户端)

以下是一个简单的示例，展示如何通过设置Retry函数的参数来设置合适的重试次数。

```
import redis
from redis.retry import Retry
from redis.exceptions import ConnectionError
from redis.backoff import ExponentialBackoff
from redis.client import Redis
from redis.exceptions import (
    BusyLoadingError,
    ConnectionError,
    TimeoutError
)

# Run 3 retries with exponential backoff strategy
retry_strategy = Retry(ExponentialBackoff(), 3)

# Redis client with retries
client = redis.Redis(
    host = 'localhost',
    port = 6379,
    retry = retry_strategy,
    # Retry on custom errors
    retry_on_error = [BusyLoadingError, ConnectionError, TimeoutError],
    # Retry on timeout
    retry_on_timeout = True
)

try:
    client.ping()
    print("Connected to Redis!")
except ConnectionError:
    print("Failed to connect to Redis after retries.")

try:
    client.set('key', 'value')
    print("Set key and value success!")
except ConnectionError:
    print("Failed to set key after retries.")
```

Hiredis (C 客户端)

Hiredis 是一个低级别的C语言库，不提供内置的自动重试机制，需要手动编写逻辑。

以下是一个简单的示例，展示如何通过一个循环和延迟来实现建立连接阶段的自动重试，对于执行命令的重试，也是类似的方案。

```
#include <hiredis/hiredis.h>
#include <stdio.h>
#include <unistd.h>

redisContext* connect_with_retry(const char *hostname, int port, int max_retries, int retry_interval) {
    redisContext *c = NULL;
    int attempt = 0;
    // 通过一个循环和延迟来实现建立连接阶段的自动重试
    while (attempt < max_retries) {
```



```
c = redisConnect(hostname, port);
if (c != NULL && c->err == 0) {
    printf("Connection success!\n");
    return c;
}

if (c != NULL) {
    printf("Connection error: %s\n", c->errstr);
    redisFree(c);
} else {
    printf("Connection failed\n");
}

printf("Retrying in %d seconds...\n", retry_interval);
sleep(retry_interval);
attempt++;
}

return NULL;
}

int main() {
    const char* hostname = "127.0.0.1";
    int port = 6379;
    int max_retries = 5;
    int retry_interval = 2;

    redisContext *c = connect_with_retry(hostname, port, max_retries, retry_interval);
    if (c == NULL) {
        printf("Failed to connect to Redis after %d attempts\n", max_retries);
        return 1;
    }

    redisFree(c);
    return 0;
}
```

5.16 GeminiDB Redis 消息发布与订阅

华为云数据库GeminiDB Redis 100%兼容开源Redis的消息发布（Publish）与订阅（Subscribe）功能，可参考此文档对自身业务进行适配。

Pub/Sub 介绍

SUBSCRIBE、**UNSUBSCRIBE** 和 **PUBLISH** 实现了[发布/订阅消息传递范式](#)。在该范式中，消息发送者（发布者）并不直接将消息发送给特定的接收者（订阅者），而是将消息发布到一个主题（channel）上。所有对该主题感兴趣的订阅者都能够接收到消息。这种模式的优势在于它的解耦性，发布者和订阅者之间不需要相互了解，降低了系统的耦合度。

使用场景

Pub/Sub模型在许多场景都可以发挥重要作用，以下罗列了一些常见的使用场景：

- 实时聊天应用

在即时通讯应用中，用户之间的消息需要快速传递。通过Pub/Sub功能，用户可以订阅各自的聊天频道，发布消息后，所有订阅该频道的用户立即收到消息。这种方式可以确保消息的实时性和高效性。

- 实时通知系统

在电商网站或社交媒体平台中，用户需要实时接收订单状态更新、评论、点赞等通知。利用 GeminiDB Redis的Pub/Sub能力，系统可以在状态变更时立即发布通知，所有相关用户都会及时收到更新。

- 监控和日志系统

在微服务架构中，服务之间的状态监控和日志收集可以使用Pub/Sub模型。服务可以将状态信息或日志消息发布到特定频道，监控服务或日志收集服务可以订阅这些频道，从而实现实时监控和数据收集。

- 游戏实时消息传递

在多人在线游戏中，玩家之间的互动需要快速同步。Pub/Sub可以用于玩家之间的消息传递、游戏事件的通知等，确保所有玩家在同一时间收到游戏状态更新。

- 数据流处理

在数据流应用中，实时数据处理和分析是关键。通过Pub/Sub，数据生产者可以发布数据流，消费者可以订阅这些流以进行实时处理和分析。

基本用法

例如，要订阅频道"channel11"和"ch:00"，客户端可以使用以下命令：

```
SUBSCRIBE channel11 ch:00
```

已订阅的客户端将收到其他客户端发送到这些频道的消息，并按照消息的发送顺序进行接收。

高级用法

Pub/Sub实现支持模式匹配。客户端可以订阅名称符合特定模式的频道，特定模式通过通配符实现，例如：

```
PSUBSCRIBE news.*
```

订阅者将会收到所有发送到news.art.figurative, news.music.jazz等等频道的消息。

⚠ 注意

- **消息丢失**：Pub/Sub 机制不保证消息的持久性，因此在网络故障或订阅者未连接时，可能会丢失消息。
- **性能考虑**：在高并发环境下，发布/订阅的性能可能受到限制。需要根据具体场景进行性能测试和优化。
- 如果同时使用SUBSCRIBE和PSUBSCRIBE，可能收到重复消息，请仔细确认是否符合业务逻辑。

Java 示例代码（Jedis 版）

消息发布者

```
import redis.clients.jedis.Jedis;
public class GeminiDBPubClient {
    private Jedis jedis;

    public GeminiDBPubClient(String ip, int port, String password){
        jedis = new Jedis(ip, port);
    }
}
```

```
// The instance password for GeminiDB.
String authString = jedis.auth(password);
if (!authString.equals("OK"))
{
    System.err.println("AUTH Failed: " + authString);
    return;
}
}

public void pub(String channel, String message){
    System.out.println(" >>> Publish > Channel: " + channel + " > Sent Message: " + message);
    jedis.publish(channel, message);
}

public void close(String channel){
    System.out.println(" >>> Publish End > Channel:" + channel + " > Message:quit");
    // The message publisher has finished sending, sending a "quit" message.
    jedis.publish(channel, "quit");
}
}
```

消息订阅者

```
import redis.clients.jedis.Jedis;
import redis.clients.jedis.JedisPubSub;
public class GeminiDBSubClient extends Thread {
    private Jedis jedis;

    private String channel;

    private JedisPubSub listener;

    public GeminiDBSubClient(String ip, int port, String password){
        jedis = new Jedis(host,port);
        // The instance password for GeminiDB.
        String authString = jedis.auth(password); //password
        if (!authString.equals("OK"))
        {
            System.err.println("AUTH Failed: " + authString);
            return;
        }
    }

    public void setChannelAndListener(JedisPubSub listener, String channel){
        this.listener=listener;
        this.channel=channel;
    }

    private void subscribe(){
        if(listener==null || channel==null){
            System.err.println("Error:SubClient> listener or channel is null");
        }
        System.out.println(" >>> Subscribe > Channel:" + channel);
        // The receiver will block the process while listening for subscribed messages until it receives a "quit"
        message (passive mode) or actively cancels the subscription.
        jedis.subscribe(listener, channel);
    }

    public void unsubscribe(String channel){
        System.out.println(" >>> Unsubscribe > Channel:" + channel);
        listener.unsubscribe(channel);
    }

    @Override
    public void run(){
        try {
            System.out.println("-----Subscribe Start-----");
            subscribe();
            System.out.println("-----Subscribe End-----");
        }
    }
}
```

```
    } catch(Exception e){  
        e.printStackTrace();  
    }  
}  
}
```

消息监听者

```
import redis.clients.jedis.JedisPubSub;  
public class GeminiDBListener extends JedisPubSub {  
    @Override  
    public void onMessage(String channel, String message) {  
        System.out.println(" <<< Subscribe < Channel:" + channel + " > Receive Message:" + message );  
        // When the received message is "quit," unsubscribe (passive mode).  
        if(message.equalsIgnoreCase("quit")){  
            this.unsubscribe(channel);  
        }  
    }  
    @Override  
    public void onPMessage(String pattern, String channel, String message) {  
        // TODO Auto-generated method stub  
    }  
    @Override  
    public void onSubscribe(String channel, int subscribedChannels) {  
        // TODO Auto-generated method stub  
    }  
    @Override  
    public void onUnsubscribe(String channel, int subscribedChannels) {  
        // TODO Auto-generated method stub  
    }  
    @Override  
    public void onPUnsubscribe(String pattern, int subscribedChannels) {  
        // TODO Auto-generated method stub  
    }  
    @Override  
    public void onPSubscribe(String pattern, int subscribedChannels) {  
        // TODO Auto-generated method stub  
    }  
}
```

6 最佳实践

6.1 通过账号直达功能实现 GeminiDB Redis 实例的多租户管理

云数据库GeminiDB Redis持续完善企业级增强特性，其中很经典的企业级特性是多租户能力，支持添加只读账号、读写账号，且具备强大的数据库级鉴权能力，即可以约束每个账号可访问的数据库（DB）范围，避免误操作其他租户数据。该特性可以帮助企业在共享Redis实例的情况下，保护不同租户的数据安全，为企业的开发和管理提供便利。

应用场景

多租户是数据库用户的常用功能。例如，企业中有两个业务部门A和B，都需要使用Redis来存储各自的数据，如果不使用多租户权限功能，那么A和B的数据就会混在一起，这样就会存在数据泄露和误操作的风险。使用了多租户管理功能后，就可以将A和B的数据分别存储在不同的Redis实例或DB中，并且对这些实例或DB进行权限控制，从而保障数据的安全性和可靠性。

在数据库领域，多租户技术往往有一些标准属性：比如读写权限控制、跨DB鉴权隔离等。而GeminiDB Redis实例就具备完善的多租户管理技术，实现了读写权限控制和数据库（DB）隔离这两大特性的完美融合。

方案优势

关于权限控制，开源Redis虽然在新版本有权限控制列表（Access Control List，简称ACL），但只能设置为只读、读写权限，每个账号还是可以看到所有的DB，这个设计跟数据库多租户的原理背道而驰。例如，业务开发小王应该用DB1，但有一天不小心清空了小张的DB0，导致发生生产事故。而GeminiDB Redis的权限隔离就可以解决此问题，如小王被设置为只有DB1的权限而没有DB2的权限，那么即使误操作也不会对DB0的数据产生影响。

此外，开源Redis的多租户功能只有单机才可以使用，一旦业务量增加需要集群，多DB功能反而就不可用了，只剩一个DB0。GeminiDB Redis基于自身的集群架构做了多DB增强，支持DB 1000+，同时可创建200+个ACL子账号，满足多种业务场景的需要。

表 6-1 开源 Redis 和 GeminiDB Redis 所具备的权限管理能力比较

Redis产品	是否支持账户读写权限控制	是否支持账户权限隔离	多DB是否支持集群	可支持DB数量
开源Redis	支持	支持	不支持	默认16
GeminiDB Redis	支持	支持	支持	默认1000

方案详情

GeminiDB Redis实例的租户管理功能，需要用户在管理控制台的账号管理页面建立账号，并为每个账号设置DB的只读/读写权限，操作非常直观方便。具体操作步骤请参考[账号管理](#)。

账号建立好之后，用户可以通过auth USER PWD 或 auth USER:PWD的方式鉴权，并且通过SELECT DB访问到该账号所拥有权限的数据库。具体的使用步骤请参考文档[开启账号直达DB](#)。

6.2 通过 FastLoad 极速导入实现广告投放业务 RTA

应用场景

广告投放是企业宣传营销不可或缺的一部分。尤其是在新媒体发展白热化的当下，不仅广告渠道多样化，投放模式也更细节化和个性化。

随着客户广告投放产出比意识的加强，以短视频平台为例，在投放目标选择上，广告主通常需要通过配置年龄、性别、学历等规则，才能将广告投放给满足标签的受众。广告投放中这一灵活性不足的限制，常常会让广告主难以抉择，导致投放效果不佳。广告主企业往往每年需花费数亿甚至数十亿广告费，却依然难以准确触达目标用户，造成大量资金浪费。那该如何解决“让广告主对每一条广告请求，有投递或者拒绝的自主权”这一问题，广告RTA应运而生！

RTA（Realtime API），是一种用于满足广告主实时、个性化的投放需求的技术手段。

业务挑战

广告主的RTA系统，是从核心的画像数据库读取数据并进行投放决策的，数据越新，投放效果越好。因此，大数据平台生成的最新数据，需要及时写入画像数据库。综合来看，广告RTA业务面临高并发、超低时延、超大数据量等实际特性需求。因此，对核心画像数据库有如下诉求：

- **海量数据快速导入，确保决策精准性**
需要定期将成百GB甚至数TB全量画像数据导入画像数据库；全量数据导入越快，模型越精准，广告投放效果越好。
- **承载高并发访问**
RTA系统要承接大量的实时竞价请求。以电商、金融客户的RTA系统为例，日常数据库QPS在几十万到数百万之间。
- **保持稳定的低时延**

媒体侧要求广告主在40-100ms内返回决策结果，数据库需要在个位数毫秒内执行完请求。

- **降低业务成本**

为了追求极致的性能体验，RTA业务通常使用开源自建Redis，然而TB级别数据存储成本非常昂贵，成本也是广告主选型的重要考虑因素。

在广告RTA中，通常选用以下数据库作为画像数据库：

- MySQL：难以满足数十万至百万QPS并发和低时延的要求。
- MongoDB/Hbase：可以存储TB级数据，成本便宜，但无法满足稳定低时延诉求，超时率高，容易导致停投，影响商业利益。
- 内存数据库：能提供高并发、低时延极致性能，如开源自建Redis，是业界选用比较多的方案。但存在着稳定性差，数据丢失等风险。对于TB级用户画像数据，存在导入速度慢和成本高的痛点。

而云数据库GeminiDB Redis接口完全具备稳定低时延、高性价比、FastLoad离线数据极速导入等核心能力。

方案总览

云数据库GeminiDB Redis接口在广告RTA场景上，不仅解决了开源自建Redis存储成本高、离线数据导入慢等瓶颈问题，还具备稳定低时延、存储成本低、FastLoad极速导入三大核心能力，拥有丰富的线上广告、推荐类业务的实践案例。

方案优势

- **FastLoad极速数据导入，效率提升5-10倍**

传统数据库只能通过标准协议逐条写入数据，先经过计算层复杂结算，再写入存储层。因此，大数据平台定期导入的数百GB乃至数TB的画像数据，通常需要数小时或者数天，且对在线业务影响比较大。

GeminiDB Redis提供的FastLoad企业级特性，依托RTA业务场景大数据平台的高并发处理能力和自身存储引擎的数据编排能力，将海量数据通过专属高速持久化通道直接传入存储引擎，数据导入速度提升5-10倍，并降低对在线业务的影响。

- **提供百万级并发和亚毫秒级延迟，无惧业务洪峰**

云数据库GeminiDB Redis采用存储计算分离架构，通过分布式高性能存储池实现三副本、强一致的数据存储，所有节点高效读/写访问，支持算力水平和垂直扩展，能够轻松应对业务规模和数据量的爆炸式增长。

通过采用多线程架构和高性能存储池，配合内存数据结构和访问算法的深度优化，GeminiDB Redis能够实现亚毫秒级的数据请求响应。这种超低时延的性能，对需要实时数据处理和分析的应用场景，如在线游戏、金融科技、广告系统和实时推荐系统，提供了强大的数据支持。因此，GeminiDB Redis成为处理大规模实时交互和高频交易等场景的理想选择。

根据现网的案例经验，在**百万+QPS**流量下，GeminiDB可稳定保持**平均时延1ms，p99时延2ms**。

- **高效数据压缩存储，效率与成本并行**

GeminiDB Redis使用“逻辑数据+块数据”双重压缩机制，在不影响性能的前提下，大幅度降低数据的存储占用。同时，采用存储计算分离架构，将算力和数据

存储解耦，支持独立弹性扩展。可以使企业以更低的成本存储更多的数据，极大地优化资源利用效率，降低整体的使用成本。

根据现网案例经验，GeminiDB Redis的数据压缩比通常为4:1，即实际12TB数据，在GeminiDB Redis中仅占用3TB左右的存储空间。

6.3 通过 PITR 实现游戏回档

应用场景

在实际业务场景中，客户数据库难免会出现数据损毁、数据丢失、数据误删除等故障场景。为保障业务的正常运行，通常需要将数据库恢复到故障发生前的某一个正常时刻。传统数据库采取周期性备份策略，即在系统故障时对数据进行恢复。因其数据恢复耗时较长，可恢复时间颗粒度较大，导致客户业务受损严重。

方案总览

PITR (Point-in-Time Recovery)，是指数据库的“时间点恢复”功能。它是一种数据库恢复技术，通常用于恢复误删除的数据或者误操作导致损坏的数据，将其恢复到一个指定时间点的数据状态。

以游戏场景为例，在游戏运行期间，有玩家利用游戏漏洞复制装备、货币，使游戏公平性遭到破坏。传统数据库备份频率一般是一天全备一次，备份间隔即一整天，不仅恢复时间长、时间粒度大，甚至无法恢复到想要时间点等。而GeminiDB Redis接口新增的PITR特性能够让游戏数据快速回档，可根据客户自定的备份粒度，最低支持5分钟粒度，自行选择需要恢复的时间点，实现数据的快速恢复。

方案优势

GeminiDB Redis接口的PITR技术执行数据快照业务无感，通常可在5分钟以内恢复到指定时间点，尤其是在业务异常时可以快速回退，降低损失，有效解决传统备份方案时间长、可恢复时间粒度大等痛点问题。因此，GeminiDB Redis接口在游戏、金融等行业有着广泛应用。

- **备份任务无感，业务更平稳**

GeminiDB Redis接口的PITR功能不涉及数据的复制，备份任务业务无感知，不影响数据访问，让客户业务更加平稳。

GeminiDB Redis接口快照原理是通过记录文件系统的状态来实现的，是瞬时生成，而不是通过复制文件本身来实现的。快照存储当前时刻的底层数据的元数据信息，比如数据块信息、寻址信息等，形成快照。因此，当打数据快照时，业务可以继续运行，而不会受到任何影响。

- **支持分钟级快速恢复，恢复时长与数据大小无关**

PITR数据快照文件可以在本地保存，不用上传到冷存储介质，因此，不涉及数据的复制搬迁，还可支持随时数据恢复。

PITR恢复，数据恢复时长与数据大小无关，能快速恢复数百GB数据，通常可在5分钟以内恢复数据，保证客户业务可靠性。除此以外，PITR还可多次前后恢复，恢复到指定时间点后，既可向前，也可向后，让客户使用更省心。

- **比开源Redis数据备份性能更优**

开源Redis使用多进程写时复制机制来实现快照的持久化。在持久化过程中，调用fork()产生一个子进程，fork()会阻塞Redis长达数百毫秒，对业务产生抖动。fork()的写时复制技术（COW）会造成内存过度使用，如果fork()期间产生大量的写操作，会导致内存严重浪费甚至OOM，通常内存利用率不足50%。而 GeminiDB Redis接口的PITR特性不涉及数据的复制搬迁，因此对业务基本无影响，且具有快照速度快、数据稳定、安全等特点。

方案详情

如何开启GeminiDB Redis接口PITR功能并执行恢复到指定时间点，请参考[通过PITR回档恢复到当前实例](#)。

6.4 通过 ExHash 实现广告频控业务

ExHash类型是一种支持Field过期的新型数据类型，它在原先的Hash类型基础上进行了扩展：在支持Hash类型的通用功能以外，ExHash类型还支持为Field设置过期时间和版本，增强了数据结构的灵活性，从而简化了很多复杂场景下的业务开发工作。

本文以两种常见的场景（频控场景&购物车场景）为例，通过使用GeminiDB Redis接口中的ExHash类命令来实现复杂的业务，简化开发难度。

ExHash 命令介绍

ExHash命令详细介绍请参考[ExHash命令列表](#)。

应用场景

- 频控场景
- 频控指的是对用户在一定时间内（例如一天、一周、一个月）进行某种操作的次数进行限制，可以控制特定广告或信息在一定时间内在特定平台上的展示次数，以避免过度曝光和广告疲劳，同时优化广告效果和用户体验；对于广告来说，也可以提高广告的效果和转化率。此外，频控还可以避免恶意行为，如刷流量、刷评论、刷点赞等。
- 频控的3个要素包含用户ID、广告ID、触发次数；以用户ID为Key，广告ID为Field，指定时间内的触发次数为Value，恰好构成频控的三要素。先配置好各个广告的指定频控策略，如下图所示，根据如下的三种方式来实现频控：

图 6-1 频控 Hash 方案

Key	key TTL	Field	Value	Key	key TTL	Field	Value	Key	key TTL	Field	Value	field TTL
User_1	one day	AD_1	1	User_1	one week	AD_1	1#one day	User_1	one week	AD_1	1	one day
		AD_2	2			AD_2	2#8 hours			AD_2	2	8 hours
		AD_3	1			AD_3	1#one week			AD_3	1	one week
User_2	...	...	...	User_2	...	...	...	User_2	...	...	...	...
...	...	...	...	...	...	...	...	...	...	...	...	...

Hash方案一Hash方案二ExHash方案

- 最左边通过Hash类型来实现，使用expire命令设置User_1的过期时间为一天，每进行一次推送，使用 hincrby 命令来增加并记录指定广告的推送次数。在每次推送广告前，可以使用 hget 命令获取前一天的推送次数，然后判断是否需要继续进行推送。一天后该用户的数据自动过期无需手动清理，这

样便可以简单地实现频控。但这个方案的缺点在于对于每个用户（即每个Key）只能设置一个过期时间，无法做到例如8小时3次这样指定时间段内的灵活的频控策略。

- 为了做到对每个广告都配置指定时间段内的灵活频控，如中间图所示可以通过将时间戳拼接在Value里的方式用Hash类型来实现，但这种方案无疑是增加了业务侧开发的工作量。
 - 如最右图所示，支持给Field设置过期时间的ExHash类型可以很好地解决Hash类型面对频控场景的缺点。由于Field支持过期时间设置，那么该场景下，平台可以给每个广告都配置不同时间段内的频次要求，假设此时给AD_2配置的频控策略为8小时内2次，那么如图所示在下次再准备给User_1推送AD_2广告前，先通过exhget User_1 AD_2命令获取到了该值已经是2时，便可以判断出此时根据平台频控策略，不应该再给User_1推送AD_2广告了。而当8小时一过，User_1的AD_2这个Field过期后，exhget无法再获取到这个Field的信息，则可以继续给User_1推送AD_2广告了。
- 购物车场景

购物节期间，很多用户的购物车里都填满了各种想要清空的宝贝，以购物车场景为例介绍该场景的几种不同Redis类型的实现，并比较这几种实现方案的优缺点。

a. 基于String实现购物车功能

如图图6-2所示，基于String可以轻松地完成各个用户的购物车功能，该方案需要将用户ID与商品ID进行拼接作为Key，例如User_1#Earphones_1，Key对应的Value为购物车中用户准备购买的数量，其中可能有部分商品为限时特购，所以有过期时间，为Key对应的过期时间。

图 6-2 String 方案

Key	Value	TTL
User_1#Earphones_1	1	null
User_1#Keyboard_1	2	time_1
User_1#Charger_2	1	time_2
User_2#...	...	...
...	...	...

String

■ 涉及命令如下：

```
incrby User_N#Product_N [Number] # 增加商品数量
set User_N#Product_N [Number] # 设置商品数量
expire User_N#Product_N Time_N # 设置指定用户购物车中指定物品的过期时间
get User_N#Product_N # 获取商品数量
scan 0 match User_N* # 查找所有User_N下的所有商品
del User_N#Product_N # 删除指定用户购物车中的指定商品
```

■ 该方案会存在如下问题：

- 额外拼接增加编、解码开发工作量。
 - 某个用户获取自己的购物车清单时还需要通过scan命令前缀匹配扫描所有Key，并通过get命令去获取对应的值。
 - 想要直接获取清单长度时，仍然需要遍历整个前缀Key的数目，方法复杂。
 - 存在大量重复的用户名前缀，浪费存储空间。
- b. 基于Hash实现购物车功能

可以根据如图3所示的Hash类型来实现购物车的管理，用户ID作为Key，商品ID作为Field，Value为购物车中对应商品的数量。其中对于部分限时特购的商品，其过期时间通过拼接的方式放到Field对应的Value里。

图 6-3 Hash 方案

Key	Field	Value
User_1	Earphones_1	1
	Keyboard_1	2
	Charger_2	1#time_1
User_2	...	...
...	...	...

Hash

- 涉及命令如下：
hset User_N Product_N [Number#Time_N] # 设置指定用户购物车中指定商品的数量和过期时间
hincrby User_N Product_N [Number] # 增加指定用户购物车中的指定商品数量
hget User_N Product_N # 获取指定用户购物车中指定商品的信息
hgetall User_N # 获取指定用户的所有商品信息
hlen User_N # 获取指定用户购物车中的总商品数量
hdel User_N Product_N # 删除指定用户购物车中的指定商品
- 该方案相对于String类型的方案有了不少优化：
 - 获取某个用户购物车中的所有商品清单仅需要一个hgetall命令即可。
 - 获取某个用户的清单长度时直接hlen获取即可。
 - 不存在大量重复的用户名前缀问题。

然而该方案仍存在一个明显的缺点，即对于部分限时特购的商品处理起来复杂：对于User_1的Keyboard_1商品，如果要再加一个数量，不能直接使用hincrby，而是需要先hget获取Keyboard_1商品的值并解码，再加上指定的数量再编码后hset对应的值。

c. 基于ExHash实现购物车功能

根据如图4所示的ExHash类型来实现购物车的管理，同Hash类型一样，用户ID作为Key，商品ID作为Field，Value为购物车中对应商品的数量。其中对于部分限时特购的商品，由于ExHash类型可以为Field设置过期时间，其过期时间可通过hset命令直接设置。

图 6-4 ExHash 方案

Key	Field	Value	TTL
User_1	Earphones_1	1	null
	Keyboard_1	2	null
	Charger_2	1	time_1
User_2	...	...	...
...	...	...	...

ExHash

- 涉及命令如下：
exhset User_N Product_N ex Time_N # 设置指定用户购物车中指定商品的数量和过期时间
exhincrby User_N Product_N [Number] keepttl # 增加指定用户购物车中的指定商品数量,保留原先过期时间
exhget User_N Product_N # 获取指定用户购物车中指定商品的信息
exhgetall User_N # 获取指定用户的所有商品信息
exhlen User_N # 获取指定用户购物车中的总商品数量
exhdel User_N Product_N # 删除指定用户购物车中的指定商品
del User_N # 清空指定用户的购物车
- 该方案相对于Hash类型的优化主要体现在可以直接为各Field设置过期时间，使业务侧使用起来简单又高效。可以看到ExHash类型相关的命令和Hash类型是类似的，使用起来学习成本很低，业务侧改造成本相对也比较低。

广告频控业务代码示例

```
import redis
import datetime
import os
def get_cur_time():
    return "[" + datetime.datetime.utcnow().strftime('%Y-%m-%d %H:%M:%S.%f')[:-3] + "]"
def get_redis():
    """
    该方法用于连接redis实例。
    * host: redis实例连接地址。
    * port: redis实例的端口号，默认为6379。
    * password: redis实例的密码。
    """
    # 认证用的用户名和密码直接写到代码中有很大的安全风险，建议在配置文件或者环境变量中存放(密码应密文存放、使用时解密)，确保安全
    # 本示例以用户名和密码保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量(环境变量名称请根据自身情况进行设置)EXAMPLE_USERNAME_ENV和EXAMPLE_PASSWORD_ENV
    password = os.getenv('EXAMPLE_PASSWORD_ENV')
```

```
return redis.Redis(host=****, port=6379, password=password)
"""全局的频控策略,广告1策略为3秒内最多2次, 广告2策略为5秒内最多5次"""
frequency_strategie = {"ad_1": [2, 3], "ad_2": [5, 5]}
def push_ad_to_user(userId: str, adId: str):
    """ 该方法用于推送指定广告给指定用户。
    * userId: 用户ID
    * adId: 广告ID
    """
    # 没有设置对该广告的频控策略, 则直接投放
    if adId not in frequency_strategie:
        print("no need control frequency, push ", adId, "to", userId)
        return True
    # 根据userId和广告ID获取该广告在某个用户上的投放次数
    # 命令用法: EXHGET Key Field
    cnt = get_redis().execute_command("EXHGET " + userId + " " + adId)
    # 该用户没有这个广告的投放记录, 则直接投放
    if cnt == None:
        # 命令用法: EXHINCRBY Key Field num [EX time]
        # 使用说明: EXHINCRBY 用户ID 广告ID 投放次数(1) EX 该广告的过期时间
        cmd = "EXHINCRBY " + userId + " " + adId + " 1 EX " + str(frequency_strategie[adId][1])
        cur_cnt = get_redis().execute_command(cmd)
        print(get_cur_time(), "push", adId, "to", userId, "first time during", str(frequency_strategie[adId][1]),
              "seconds")
        return True
    # redis-py返回的结果是bytes类型, 转换成str后转int类型
    cnt = int(cnt.decode("utf-8"))
    if cnt < frequency_strategie[adId][0]:
        # 命令用法: EXHINCRBY Key Field num KEEPTTL 保持Field原先的过期时间
        cmd = "EXHINCRBY " + userId + " " + adId + " 1 KEEPTTL"
        cur_cnt = get_redis().execute_command(cmd)
        print(get_cur_time(), "push", adId, "to", userId, "current cnt:", cur_cnt)
        return True
    print(get_cur_time(), "Control frequency, can't push", adId, "to", userId, ", max cnt:",
          frequency_strategie[adId][0])
    return False
if __name__ == "__main__":
    for i in range(3):
        push_ad_to_user("usr_1", "ad_1")
    for i in range(6):
        push_ad_to_user("usr_1", "ad_2")
    for i in range(3):
        push_ad_to_user("usr_1", "ad_1")
    for i in range(12):
        push_ad_to_user("usr_1", "ad_2")
```

脚本运行输出:

其中由于python脚本本身运行较慢, 广告2的过期时间设置得只有5s, 所以当第一次投放广告2的时间2023-12-15 07:09:51.349的5s之后, 也就是2023-12-15 07:09:56.530时间点再次给这个用户推送广告2时就可以推送成功了。

```
[2023-12-15 07:09:50.086] push ad_1 to usr_1 first time during 3 seconds
[2023-12-15 07:09:50.503] push ad_1 to usr_1 current cnt: 2
[2023-12-15 07:09:50.794] Control frequency, can't push ad_1 to usr_1 , max cnt: 2
[2023-12-15 07:09:51.349] push ad_2 to usr_1 first time during 5 seconds
[2023-12-15 07:09:51.745] push ad_2 to usr_1 current cnt: 2
[2023-12-15 07:09:52.128] push ad_2 to usr_1 current cnt: 3
[2023-12-15 07:09:52.889] push ad_2 to usr_1 current cnt: 4
[2023-12-15 07:09:53.417] push ad_2 to usr_1 current cnt: 5
[2023-12-15 07:09:53.632] Control frequency, can't push ad_2 to usr_1 , max cnt: 5
[2023-12-15 07:09:54.120] push ad_1 to usr_1 first time during 3 seconds
[2023-12-15 07:09:54.769] push ad_1 to usr_1 current cnt: 2
[2023-12-15 07:09:54.915] Control frequency, can't push ad_1 to usr_1 , max cnt: 2
[2023-12-15 07:09:55.211] Control frequency, can't push ad_2 to usr_1 , max cnt: 5
[2023-12-15 07:09:55.402] Control frequency, can't push ad_2 to usr_1 , max cnt: 5
[2023-12-15 07:09:55.601] Control frequency, can't push ad_2 to usr_1 , max cnt: 5
[2023-12-15 07:09:55.888] Control frequency, can't push ad_2 to usr_1 , max cnt: 5
[2023-12-15 07:09:56.087] Control frequency, can't push ad_2 to usr_1 , max cnt: 5
[2023-12-15 07:09:56.530] push ad_2 to usr_1 first time during 5 seconds
```

```
[2023-12-15 07:09:57.133] push ad_2 to usr_1 current cnt: 2
[2023-12-15 07:09:57.648] push ad_2 to usr_1 current cnt: 3
[2023-12-15 07:09:58.107] push ad_2 to usr_1 current cnt: 4
[2023-12-15 07:09:58.623] push ad_2 to usr_1 current cnt: 5
[2023-12-15 07:09:58.865] Control frequency, can't push ad_2 to usr_1 , max cnt: 5
[2023-12-15 07:09:59.096] Control frequency, can't push ad_2 to usr_1 , max cnt: 5
```

本文介绍了GeminiDB Redis接口的ExHash类型的特性、使用方法及应用场景。为客户提供了一种语法与原生Redis Hash类型类似、和Hash类型的使用相互隔离、支持给Field单独设置过期时间和版本的ExHash类型作为各种复杂场景的解决方案。未来，GeminiDB Redis接口将持续致力于开发更多好用的企业级特性，帮助客户轻松运维，高效开发。

6.5 GeminiDB Redis 在 IM 场景中的应用

背景

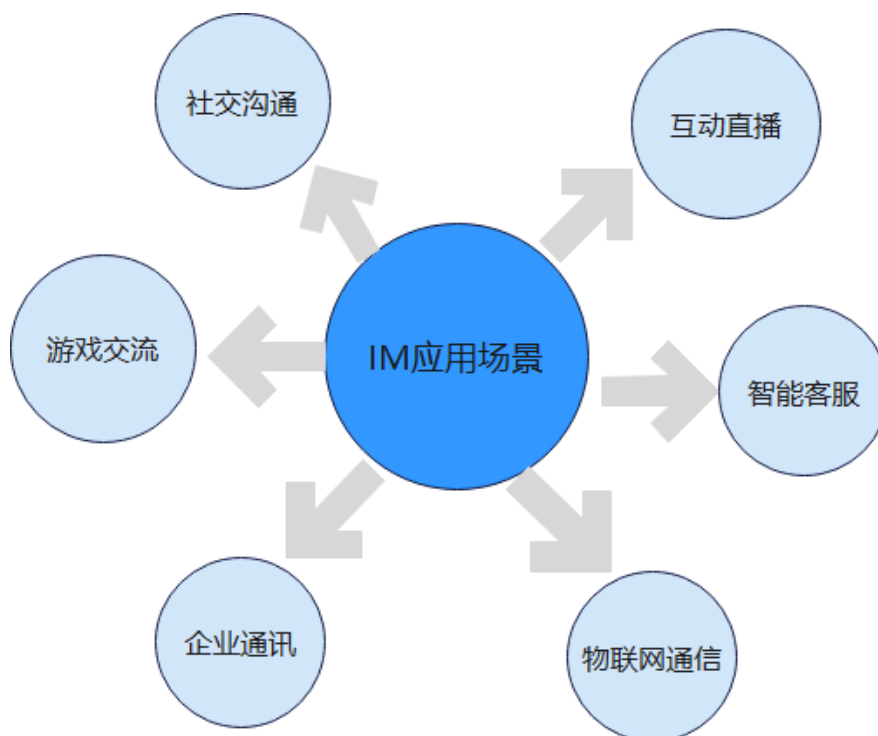
即时通讯（Instant Messaging，简称IM）是一个实时通信系统，允许两人或多人使用网络实时地传递文字消息、文件、语音与视频。微信、QQ等IM类产品在这个高度信息化的互联网时代已成为生活必备品，IM系统中最核心的部分是消息系统，消息系统中最核心的功能是消息的同步、存储和检索。

- 消息同步：将消息完整地、快速地从发送方发送至接收方。消息同步系统最重要的衡量指标是消息传递的实时性、完整性、顺序性以及支撑的消息规模。
- 消息存储：即消息的持久化，传统消息系统通常支持消息在接收端的本地存储，数据基本不具备可靠性。现代消息系统支持消息在云端存储，从而实现消息漫游：账号可在任意客户端登录查看所有历史消息。
- 消息检索：消息一般是文本，所以支持全文检索也是必备的能力之一。传统消息系统通常来说基于本地存储的消息数据来构建索引，支持消息的本地检索。而现代消息系统支持消息的在线存储以及存储过程中构建索引，提供全面的消息检索功能。

使用场景

IM系统可用于聊天，游戏、智能客服等诸多行业。不同行业对IM系统的成本、性能、可靠性、时延等指标的需求是不同的，架构设计需要进行平衡。

图 6-5 IM 应用场景

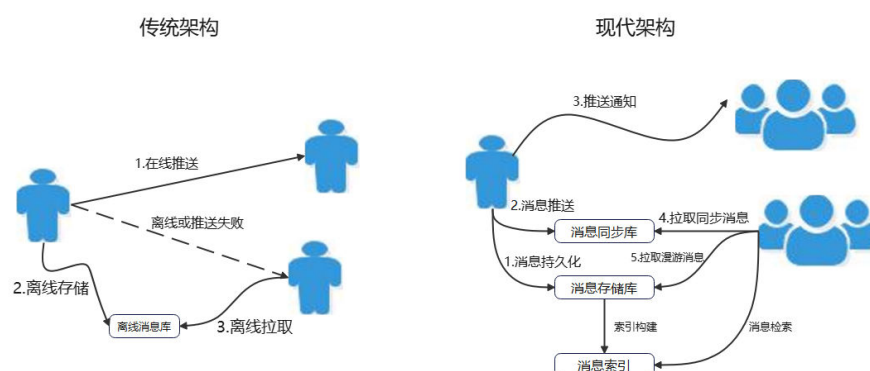


IM 系统架构

首先介绍IM系统架构设计所涉及的一些基本概念。

- 传统架构和现代架构对比

图 6-6 传统架构和现代架构对比



传统架构：

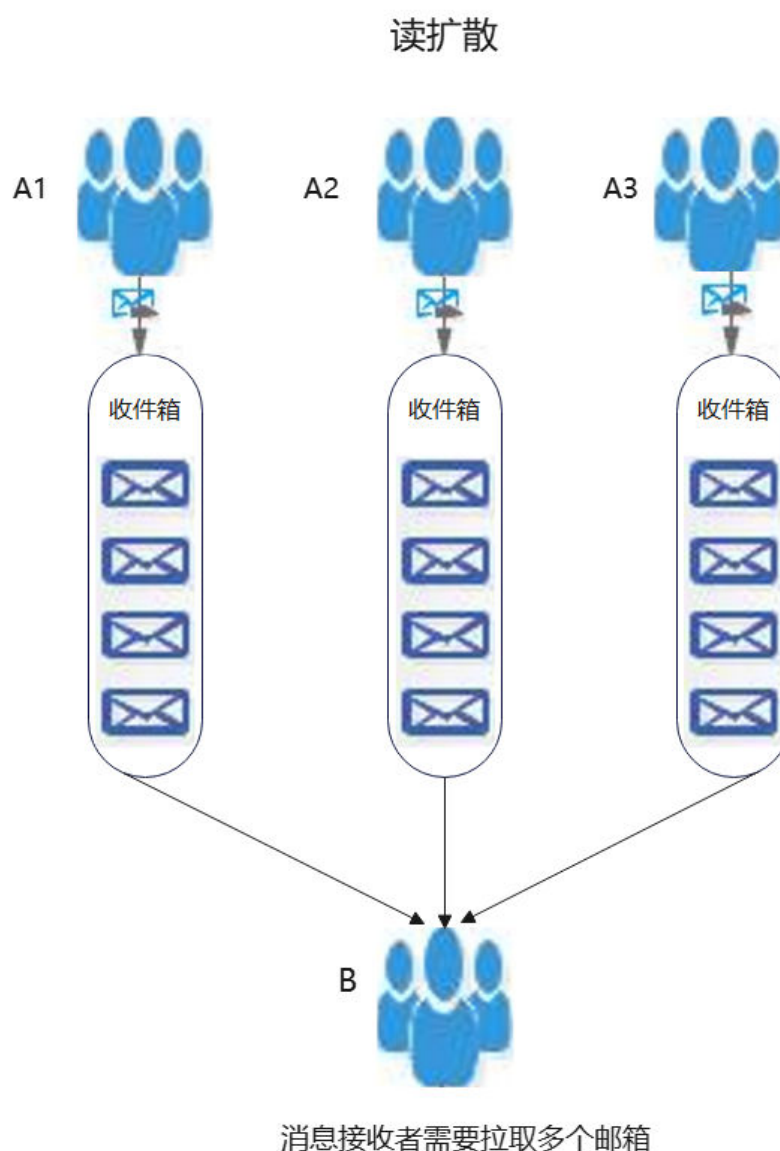
- 先同步后存储。
- 在线消息同步和离线消息缓存。
- 服务端不会对消息进行持久化，无法支持消息漫游。

现代架构：

- 先存储后同步。
- 划分消息存储库与消息同步库。消息存储库用于全量保存所有会话消息，主要用于支持消息漫游。消息同步库，主要用于接收方的多端同步。

- 提供消息全文检索能力。
- **读扩散和写扩散对比**
针对用户消息较多的场景，如何保证消息传递的可靠性、一致性并且有效地降低服务器或者客户端的压力。其中，采用何种读写模型对IM系统至关重要，这里介绍两种模型：读扩散和写扩散。

图 6-7 读扩散



如上图所示，用户B与每个聊天的人(A1,A2,A3)都有一个信箱（一种数据结构的抽象，用于存储消息），B在查看聊天信息时需读取所有有新消息的信箱。IM系统里的读扩散通常是每两个相关联的人就有一个信箱。

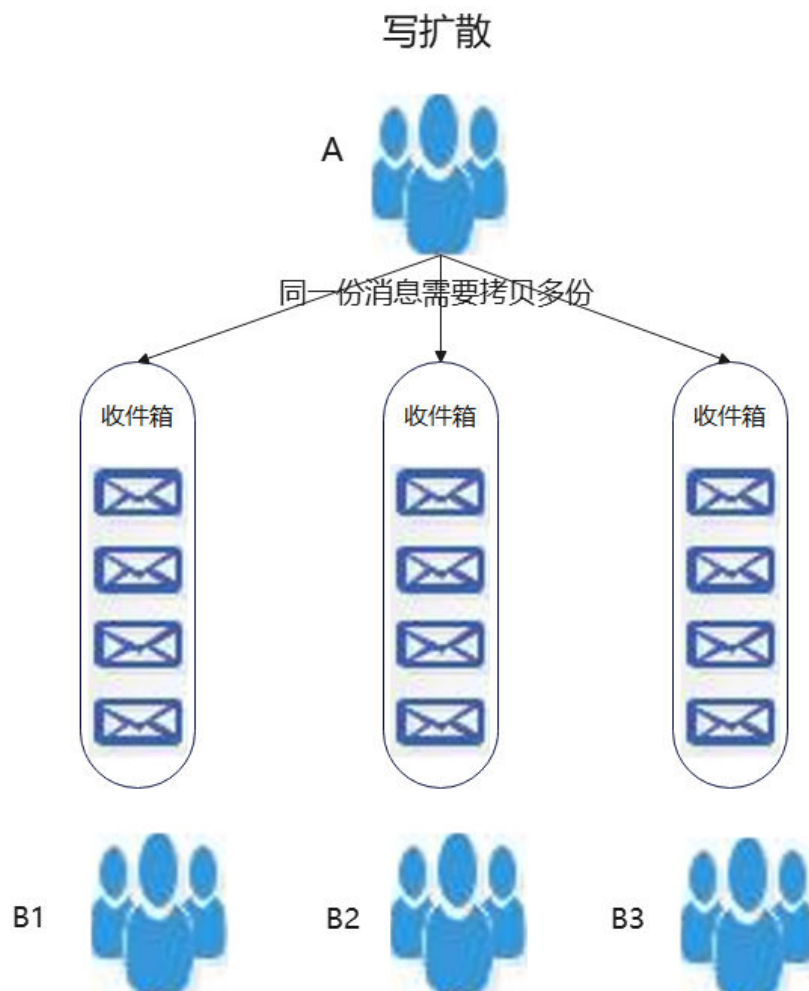
读扩散优点：

- 写操作(发消息)轻量，不管是单聊还是群聊，只需要往相应的信箱写一次即可。
- 每一个信箱天然就是两个人的聊天记录，可以方便查看和搜索聊天记录。

读扩散缺点：

- 读操作(读消息)很重，存在读放大效应。

图 6-8 写扩散



如上图，在写扩散中，用户(B1,B2,B3)都只从自己的信箱里读取消息，但写(发消息)的时候，对于单聊跟群聊处理如下：

- 单聊：往自己的信箱跟对方的信箱都写一份消息；同时，如果需要查看两个人的聊天历史记录的话还需要再写一份。
- 群聊：发信息时需要针对所有群成员的信箱都写一份消息。群聊使用的是写扩散模型，而写扩散很消耗资源，因此群聊有人数上限(目前是500)。

写扩散优点：

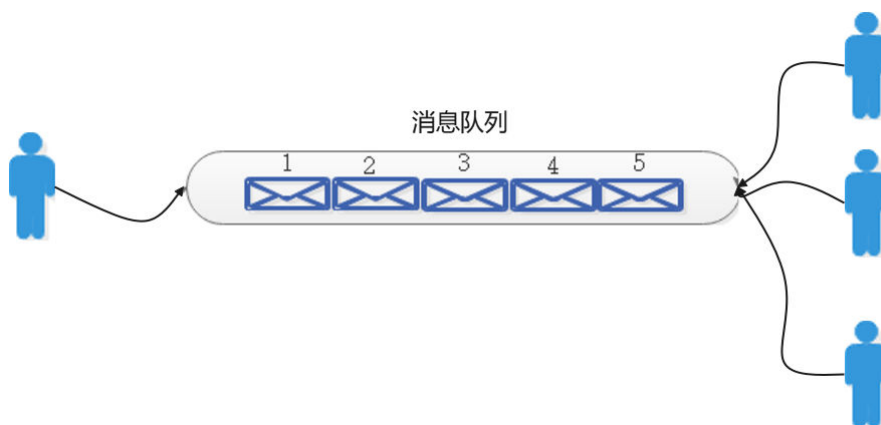
- 读操作很轻量，只需要读取自己的邮箱。
- 可以很方便实现消息的多终端同步。

写扩散缺点：

- 写操作很重，尤其是对于群聊来说。

- 推模式、拉模式和推拉结合模式对比

图 6-9 推模式、拉模式与 推拉结合模式的对比

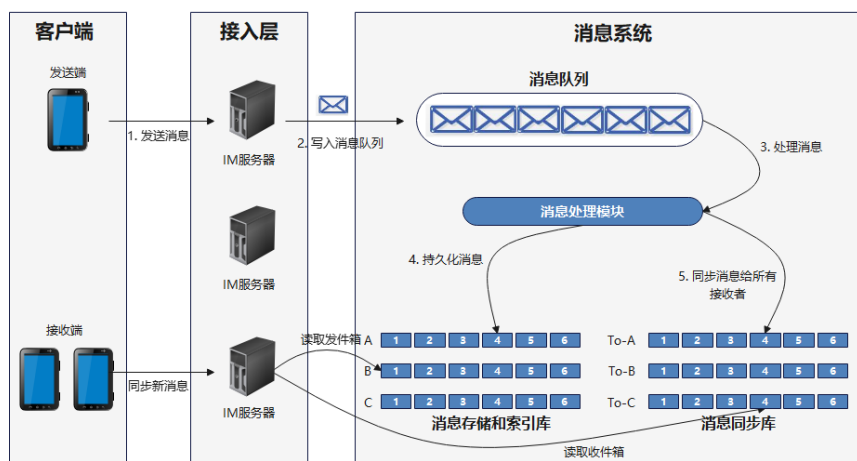


在IM系统中，消息的获取通常有三种模式：

- 推模式(Push)：新消息到达时由服务器主动推送给所有客户端；需要客户端和服务端建立长连接，实时性很高，对客户端来说只需要接收处理消息即可；缺点是服务端不知道客户端处理消息的能力，可能会导致数据积压。
- 拉模式(Pull)：由前端主动发起拉取消息的请求，为了保证消息的实时性，一般采用推模式，拉模式一般用于获取历史消息；因客户端拉取新消息的时间间隔不好预设，太短可能会导致大量的连接拉取不到数据，太长导致数据接收不及时。
- 推拉结合模式：兼顾Push和Pull两种模式的优点。新消息来临时服务器会先推送一个有新消息到达的通知给前端，前端接收到通知后就向服务器拉取消息。

IM 技术挑战

图 6-10 IM 系统总体架构图



Client双方通信会经过Server转发来完成消息传递。其核心为消息存储库和消息同步库。这两种库对存储层的性能有极高的要求。

- 支撑海量数据存储：对于消息存储库来说，如果需要消息永久存储，则随着时间的积累，数据规模会越来越大，需存储库支持容量无限扩展以应对日益增长的消息数据。

- 低存储成本：消息数据具有明显的冷热特征，大部分查询集中在热数据，冷数据需要一个低成本的存储方式，否则随着时间的积累，数据量不断膨胀，存储成本会不断上升。
- 数据生命周期管理：不管是对于消息数据的存储还是同步，数据都需要定义生命周期。存储库是用于在线存储消息数据本身，通常需要设定一个较长周期的保存时间。而同步库是用于写扩散模式的在线或离线推送，通常设定一个较短的保存时间。
- 极高的写入吞吐：绝大部分IM类场景，通常是采用写扩散模型，写扩散要求底层存储具备极高的写入吞吐能力，从而应对消息洪峰。
- 低延迟的读：消息系统通常应用于在线场景，具备较高的实时性，读取延迟应尽可能低。

GeminiDB Redis 在 IM 场景中的优势

IM系统的核心是存储层，其性能差异将直接影响IM系统的用户体验。目前存储层可选择的产品有很多，如HBase、开源Redis等等。选择何种数据库，需根据业务规模、成本、性能等指标来进行综合选择。这里介绍一种NoSQL数据库：GeminiDB Redis，在性能和规模上，可以满足IM系统对存储层的严格要求：海量数据存储、低存储成本、生命周期管理、写入吞吐大、读取时延低。

GeminiDB Redis是华为云数据库团队自主研发且兼容Redis5.0协议的云原生数据库，采用计算存储分离架构。存储侧使用自研的存储系统，容量无限扩展、强一致、高可靠。计算侧基于 LSM 存储引擎实现，通过将大量的随机写转换为顺序写，从而极大地提升了数据写入性能，同时也通过读缓存、bloom filter 等极大优化了读取性能。下图是GeminiDB Redis在IM场景的优势介绍。

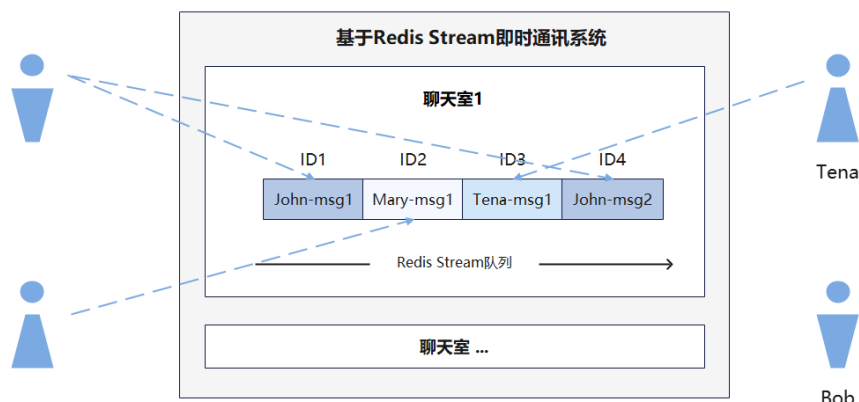
图 6-11 GeminiDB Redis 优势



基于GeminiDB Redis的IM应用案例

下图是基于GeminiDB Redis的IM系统模型图，使用stream作为基本数据结构。redis stream不仅可以作为消息存储容器，还实现了生产者、消费者等基本模型，具有IM系统的基本功能，如消息订阅，分发、增加消费者等，用户可基于GeminiDB Redis快速构建一套IM系统。创建一个群聊时，在redis中对应地为该群聊创建一个stream队列。在发送消息时，每个用户都将消息按照时间顺序添加到stream队列中，保证了消息的有序性。stream是一个持久化的队列，可保证信息不丢失。

图 6-12 基于 Redis Stream 及时通讯系统



GeminiDB Redis通过一系列技术创新实现了读写性能水平扩展，秒级扩容，低成本以及自动备份等功能，可作为IM系统的存储层，其优异的读写性能和高级特性将会极大助力IM应用。同时，GeminiDB Redis 在开源Redis的基础之上，较好平衡了性能和成本，能够广泛应用在智慧医疗、流量削峰、计数器等领域。

6.6 GeminiDB Redis 基于 lua 实现分布式锁

在分布式系统中，为了保证数据的一致性和防止并发冲突，经常需要使用分布式锁来确保同一时间只有一个进程或线程能够执行特定的代码段。

本文将介绍如何利用lua实现分布式锁。

Redis 分布式锁原理

Redis分布式锁的基本原理是利用Redis的原子命令来创建一个锁。最简单的实现方式是使用SETNX命令，这个命令只有在键不存在时才会设置键的值，如果键已经存在，命令不会执行任何操作。这样，第一个获取锁的进程会成功设置键的值，而后续尝试获取锁的进程都会失败，直到锁被释放。

为了防止锁永远不被释放（例如，持有锁的进程崩溃），通常会给锁设置一个过期时间，这可以通过EXPIRE命令来实现。Redis 2.6.12版本之后，SET命令增加了EX和NX选项，可以在设置键的同时设置过期时间，这个操作是原子的。

- 加锁

可以通过下面的命令实现加锁

```
SET resource_name my_random_value NX PX 30000
```

NX参数会检查key的存在性，当key不存在，即没有人持有锁，才会加锁成功；

PX参数用于设置锁过期时间，单位毫秒，这个参数必须设置，以确保及时锁的持有者异常退出了，锁过期即自动解锁，以确保死锁问题不会发生。

- 解锁

解锁的动作为复杂一些，解锁的时候需要检查你是否是锁的持有者，检查通过才能解锁，为了串行化地执行这个解锁的过程，需要依赖lua脚本来完成

lua脚本：

```
if redis.call("get",KEYS[1]) == ARGV[1] then
    return redis.call("del",KEYS[1])
else
    return 0
end
```

这个lua脚本需要搭配EVAL命令来执行，示例如下：

```
EVAL 'if redis.call("get",KEYS[1]) == ARGV[1] then return redis.call("del",KEYS[1]) else return 0 end' 1
resource_name my_random_value
```

通过这个脚本，就能保证只有解锁的持有者才能解锁成功。

- 方案分析

上述的方案简单易用，但功能有点不足：

- 锁过期释放，业务没执行完；
- 锁不可重入；
- 没有通知机制，需要轮询抢锁，CPU耗费较多；

所以在生产应用场景下，还是比较推荐直接使用成熟的Redis分布式锁类库来实现分布式锁，以平衡功能及性能；

下面以redisson为例，介绍一下Redis分布式锁类库的用法。

通过 Redisson 实现分布式锁

Redisson是一个基于Redis的Java客户端，提供了分布式锁的功能。分布式锁是一种用于在分布式系统中同步访问共享资源的机制。Redisson通过Redis的原子操作来实现分布式锁，确保只有一个客户端能够同时访问某个资源。

Redisson的分布式锁主要有以下几个特点：

- 高效性：利用Redis的高性能和内存存储特性，分布式锁操作非常快速。
- 简单易用：提供了丰富的API，使开发者可以轻松地在Java应用中使用分布式锁。
- 可靠性：Redisson的分布式锁具有高可靠性，即使在网络分区或节点宕机的情况下，也能保证锁的正确性。

使用示例

```
import org.redisson.Redisson;
import org.redisson.api.RLock;
import org.redisson.api.RedissonClient;
import org.redisson.config.Config;

public class LockExamples {
    public static void main(String[] args) {
        // 创建Redisson客户端
        Config config = new Config();
        config.useSingleServer().setAddress("redis://127.0.0.1:7200");
        RedissonClient redisson = Redisson.create(config);

        // 获取分布式锁
        RLock lock = redisson.getLock("myLock");

        try {
            // 加锁
            lock.lock();
            System.out.println("Lock acquired, executing critical section...");

            // 执行需要加锁的代码
            // ...
            System.out.println("Critical section executed, releasing lock...");
        }
    }
}
```

```
} catch (Exception e) {  
    e.printStackTrace();  
} finally {  
    // 释放锁  
    lock.unlock();  
}  
  
// 关闭Redisson客户端  
redisson.shutdown();  
}
```

更多的分布式锁实现推荐

鉴于Redisson是Java生态的，其实分布式锁在各语言下都有了成熟的实现，这里直接引用[redis官网](#)的推荐：

- [Redlock-rb](#) (Ruby implementation). There is also a [fork of Redlock-rb](#) that adds a gem for easy distribution.
- [RedisQueuedLocks](#) (Ruby implementation).
- [Redlock-py](#) (Python implementation).
- [Pottery](#) (Python implementation).
- [Aioredlock](#) (Asyncio Python implementation).
- [RedisMutex](#) (PHP implementation with both [Redis extension](#) and [Predis library](#) clients support).
- [Redlock-php](#) (PHP implementation).
- [cheprasov/php-redis-lock](#) (PHP library for locks).
- [rtckit/react-redlock](#) (Async PHP implementation).
- [Redsync](#) (Go implementation).
- [Redisson](#) (Java implementation).
- [Redis::DistLock](#) (Perl implementation).
- [Redlock-cpp](#) (C++ implementation).
- [Redis-plus-plus](#) (C++ implementation).
- [Redlock-cs](#) (C#/.NET implementation).
- [RedLock.net](#) (C#/.NET implementation). Includes async and lock extension support.
- [ScarletLock](#) (C# .NET implementation with configurable datastore).
- [Redlock4Net](#) (C# .NET implementation).
- [node-redlock](#) (NodeJS implementation). Includes support for lock extension.
- [simple-redis-mutex](#) (Node.js implementation) Available as an [NPM package](#).
- [Deno DLM](#) (Deno implementation)
- [Rslock](#) (Rust implementation). Includes async and lock extension support.

6.7 GeminiDB Redis 指标告警配置建议

通过在云监控服务界面设置告警规则，用户可自定义监控目标与通知策略，及时了解实例的运行状况，从而起到预警作用。具体操作参见[设置告警规则](#)。

本章节介绍了设置GeminiDB Redis 指标告警规则的配置建议。

表 6-2 GeminiDB Redis 指标告警配置建议

指标ID	指标名称	指标维度	最佳实践 阈值	最佳 实践 告警 级别	告警后的处理建议
redis688_qps_send_total	实例级发送总流量	实例级	连续3个周期，原始值 >=875000000byte/s/s	重要	建议遵循开发规范中的使用建议，详情参考 开发运维规范 。
redis689_qps_receive_total	实例级接收总流量	实例级	连续3个周期，原始值 >=875000000byte/s/s	重要	建议遵循开发规范中的使用建议，详情参考 开发运维规范 。
redis803_cluster_all_avg_usec	实例的平均时延	实例级	连续5个周期 原始值 >=50000us	重要	- 建议检查实例是否存在CPU、内存、连接数等的性能瓶颈，如果有的话参考相关指标建议解决性能瓶颈问题。 - 建议在业务无法优化的场景下，升配实例规格或变更到磁盘性能更优的规格，详情参考变更实例的CPU和内存规格和添加节点。
redis804_cluster_all_p99_usec	实例的P99时延	实例级	连续5个周期，原始值 >=150000us	重要	- 建议检查实例是否存在CPU、内存、连接数等的性能瓶颈，如果有的话参考相关指标建议解决性能瓶颈问题。 - 建议在业务无法优化的场景下，升配实例规格或变更到磁盘性能更优的规格，详情参考变更实例的CPU和内存规格和添加节点。
redis816_cluster_disk_usage	实例的存储容量利用率	实例级	连续3个周期，原始值 >=80%	重要	建议根据业务情况评估增加存储容量，详情参考 手动扩容实例磁盘 。

指标ID	指标名称	指标维度	最佳实践 阈值	最佳 实践 告警 级别	告警后的处理建议
nosql001_cpu_usage	CPU利用率	节点级	连续3个周期 原始值<=85%	重要	建议评估业务情况考虑升配规格或者增加节点来降低CPU负载，详情参考 变更实例的CPU和内存规格 和 添加节点 。
nosql002_mem_usage	内存利用率	节点级	连续3个周期 原始值<=85%	重要	建议评估业务情况考虑升配内存规格，详情参考 变更实例的CPU和内存规格 。
redis669_connection_usage	连接数使用率	节点级	连续3个周期 原始值<=80%	重要	建议遵循开发规范中的使用建议，详情参考 开发运维规范 。
gemini004_bytes_in	网络输入吞吐量	节点级	连续3个周期 原始值<=87500000bytes/s	重要	建议遵循开发规范中的使用建议，详情参考 开发运维规范 。
gemini003_bytes_out	网络输出吞吐量	节点级	连续3个周期 原始值<=87500000bytes/s	重要	建议遵循开发规范中的使用建议，详情参考 开发运维规范 。

6.8 GeminiDB Redis 在商品相关性分析的应用

Redis提供了丰富的数据结构（如集合、有序集合、哈希表等），本文以电商场景为背景，介绍了如何利用Redis的数据结构来构建网购商品相关性分析的程序。您可以使用云数据库GeminiDB Redis接口尝试下面的案例。

使用场景

可以根据同时出现在用户的购物车中的某些商品来分析商品相关性。这种分析对于电商行业至关重要，因为能够帮助商家洞察用户的购买习惯。例如：

- 在商品详情页，为用户推荐与当前商品经常一起购买的其他商品。
- 在购物车页面，当用户添加某件商品后，推荐与之搭配的其他商品。
- 在货架陈列时，将经常一起购买的商品摆放在一起，方便用户选购。

数据结构设计

- **商品浏览记录**

记录用户浏览商品的行为，可以使用 Redis 的"SET"数据结构来存储用户浏览过的商品ID。

- Key: user:<user_id>:viewed
- Value: 商品ID的集合。

示例：

用户ID为123的用户浏览了商品ID为1001，1002，1003的商品。

```
user:123:viewed -> {1001, 1002, 1003}
```

- **2.2 商品购买记录**

记录用户购买商品的行为，同样可以使用"SET"数据结构。

- Key: user:<user_id>:purchased
- Value: 商品ID的集合。

示例：

用户ID为123的用户购买了商品ID为1001，1005的商品。

```
user:123:purchased -> {1001, 1005}
```

- **2.3 商品相关性计数**

记录商品之间的相关性，可以使用 Redis 的"ZSET"（有序集合）来存储商品之间的共现次数。

- Key: product:<product_id>:related
- Value: 与该商品相关的其他商品ID及其共现次数。

示例：

与商品1001相关的其他商品ID及其共现次数。

```
product:1001:related -> {1002:5, 1003:3, 1005:2}
```

数据收集

- **用户浏览行为**

每当用户浏览一个商品时，将商品ID添加到用户的浏览记录集合中。C++代码示例如下：

```
void recordView(int user_id, int product_id) {  
    std::cout << "User ID " << user_id << " has viewed Product ID " << product_id << std::endl;  
    redis.sadd("user:" + std::to_string(user_id) + ":viewed", std::to_string(product_id));  
}
```

- **用户购买行为**

每当用户购买一个商品时，将商品ID添加到用户的购买记录集合中。C++代码示例如下：

```
void recordPurchase(int user_id, int product_id) {  
    std::cout << "User ID " << user_id << " has purchased Product ID " << product_id << std::endl;  
    redis.sadd("user:" + std::to_string(user_id) + ":purchased", std::to_string(product_id));  
}
```

查询接口

查询相关商品：根据商品ID，查询与该商品最相关的其他商品。C++代码示例如下：

```
std::vector<std::pair<std::string, double>> getRelatedProducts(int product_id) {  
    std::vector<std::pair<std::string, double>> result;
```

```
redis.zrevrange("product:" + std::to_string(product_id) + ":related", 0, -1, std::back_inserter(result));  
return result;  
}
```

C++完整示例代码

以下是完整的示例代码，使用 C++的redis客户端[redis++](#)实现。

```
#include <iostream>  
#include <iterator>  
#include <set>  
#include <vector>  
#include <string>  
#include <utility>  
  
#include "sw/redis+/redis++.h"  
  
using namespace sw::redis;  
  
auto redis = Redis("tcp://127.0.0.1:6379");  
  
void recordView(int user_id, int product_id) {  
    std::cout << "User ID " << user_id << " has viewed Product ID " << product_id << std::endl;  
    redis.sadd("user:" + std::to_string(user_id) + ":viewed", std::to_string(product_id));  
}  
  
void recordPurchase(int user_id, int product_id) {  
    std::cout << "User ID " << user_id << " has purchased Product ID " << product_id << std::endl;  
    redis.sadd("user:" + std::to_string(user_id) + ":purchased", std::to_string(product_id));  
}  
  
void updateRelatedProducts(int user_id) {  
    std::set<std::string> viewed_products;  
    redis.smembers("user:" + std::to_string(user_id) + ":viewed", std::inserter(viewed_products,  
viewed_products.end()));  
  
    std::set<std::string> purchased_products;  
    redis.smembers("user:" + std::to_string(user_id) + ":purchased", std::inserter(purchased_products,  
purchased_products.end()));  
  
    for (const auto& product_id : viewed_products) {  
        for (const auto& related_product_id : viewed_products) {  
            if (product_id != related_product_id) {  
                // 浏览的权重为1  
                redis.zincrby("product:" + product_id + ":related", 1, related_product_id);  
            }  
        }  
    }  
  
    for (const auto& product_id : purchased_products) {  
        for (const auto& related_product_id : purchased_products) {  
            if (product_id != related_product_id) {  
                // 购买行为权重更高  
                redis.zincrby("product:" + product_id + ":related", 2, related_product_id);  
            }  
        }  
    }  
}  
  
std::vector<std::pair<std::string, double>> getRelatedProducts(int product_id) {  
    std::vector<std::pair<std::string, double>> result;  
    redis.zrevrange("product:" + std::to_string(product_id) + ":related", 0, -1, std::back_inserter(result));  
    return result;  
}  
  
int main() {  
    int user_id = 123;  
  
    // 模拟用户行为
```

```
recordView(user_id, 1001);
recordView(user_id, 1002);
recordView(user_id, 1003);
recordView(user_id, 1004);
recordView(user_id, 1005);
recordPurchase(user_id, 1001);
recordPurchase(user_id, 1005);

// 根据用户行为更新商品相关性信息
updateRelatedProducts(user_id);

// 查询与商品1001相关商品
int product_id = 1001;
auto related_products = getRelatedProducts(product_id);
std::cout << "Product ID " << product_id << "'s related product is: " << std::endl;
for (const auto& [product_id, score] : related_products) {
    std::cout << "Product ID: " << product_id << ", Score: " << score << std::endl;
}

return 0;
}
```

运行结果如下：

```
User ID 123 has viewed Product ID 1001
User ID 123 has viewed Product ID 1002
User ID 123 has viewed Product ID 1003
User ID 123 has viewed Product ID 1004
User ID 123 has viewed Product ID 1005
User ID 123 has purchased Product ID 1001
User ID 123 has purchased Product ID 1005
Product ID 1001's related product is:
    Product ID: 1005, Score: 3
    Product ID: 1004, Score: 1
    Product ID: 1003, Score: 1
    Product ID: 1002, Score: 1
```

6.9 GeminiDB Redis 在线课堂的应用

使用Redis可以轻松构建一个在线课堂应用，实现诸如用户认证、课程信息管理、实时消息推送、在线互动等功能。以下是一个基于Redis的在线课堂应用的基本架构和实现思路，包括数据结构设计、功能模块实现。GeminiDB Redis 100%兼容Redis协议，您可以使用GeminiDB Redis进行尝试搭建。

数据结构设计

- 用户信息
存储用户的基本信息和认证数据。
 - Key: user:<user_id>
 - Value: 用户信息（使用 Redis 的 "HASH" 数据结构）。

示例：

```
user:123 -> {name: "ZhangSan", email: "zhangsan@example.com", password_hash: "hashed_password"}
```

- 课程信息

存储课程的基本信息和元数据。

- Key: course:<course_id>
- Value: 课程信息（使用Redis的 "HASH" 数据结构）。

示例：

```
course:101 -> {title: "Introduction to Redis", description: "Learn the basics of Redis.", instructor: "LiSi"}
```

- 用户课程关系

记录用户与课程的关联关系。

- Key: user:<user_id>:courses
- Value: 用户参加的课程ID集合（使用Redis的 "SET" 数据结构）。

示例：

```
user:123:courses -> {101, 102}
```

- Key: course:<course_id>:students
- Value: 参加该课程的用户ID集合（使用Redis的 "SET" 数据结构）。

示例：

```
course:101:students -> {123, 456}
```

- 实时消息

存储课程中的实时消息和互动内容。

- Key: course:<course_id>:messages
- Value: 课程中的消息列表（使用Redis的 "LIST" 数据结构）。

示例：

```
course:101:messages -> ["ZhangSan: Hello everyone!", "LiSi: Welcome to the class!"]
```

- 在线状态

记录用户的在线状态。

- Key: user:<user_id>:online
- Value: 在线状态（使用Redis的 "STRING" 数据结构，值为 "1" 表示在线，"0" 表示离线）。

示例：

```
user:123:online -> 1
```

功能模块实现

以下示例代码使用 C++ 的redis客户端[redis++](#)实现。

- 用户认证

使用 Redis 存储用户信息和会话信息。

```
void registerUser(int user_id, const std::string& name, const std::string& email, const std::string& password_hash) {
    redis.hset("user:" + std::to_string(user_id), "name", name);
    redis.hset("user:" + std::to_string(user_id), "email", email);
    redis.hset("user:" + std::to_string(user_id), "password_hash", password_hash);
}

bool loginUser(int user_id, const std::string& password_hash) {
    std::string stored_password_hash = redis.hget("user:" + std::to_string(user_id), "password_hash").value();
    return stored_password_hash == password_hash;
}
```

- 课程管理

添加和查询课程信息。

```
void addCourse(int course_id, const std::string& title, const std::string& description, const std::string&
instructor) {
    redis.hset("course:" + std::to_string(course_id), "title", title);
    redis.hset("course:" + std::to_string(course_id), "description", description);
    redis.hset("course:" + std::to_string(course_id), "instructor", instructor);
}

std::string getCourseTitle(int course_id) {
    return redis.hget("course:" + std::to_string(course_id), "title").value();
}
```

- 用户课程关系

管理用户与课程的关系。

```
void enrollUserInCourse(int user_id, int course_id) {
    redis.sadd("user:" + std::to_string(user_id) + ":courses", std::to_string(course_id));
    redis.sadd("course:" + std::to_string(course_id) + ":students", std::to_string(user_id));
}

bool isUserEnrolledInCourse(int user_id, int course_id) {
    return redis.sismember("user:" + std::to_string(user_id) + ":courses", std::to_string(course_id));
}
```

- 实时消息推送

支持课程中的实时消息。

```
void sendMessageToCourse(int course_id, const std::string& message) {
    redis.rpush("course:" + std::to_string(course_id) + ":messages", message);
}

std::vector<std::string> getCourseMessages(int course_id) {
    std::vector<std::string> output;
    redis.lrange("course:" + std::to_string(course_id) + ":messages", 0, -1, std::inserter(output, output.end()));
    return output;
}
```

- 在线状态

管理用户的在线状态。

```
void setOnlineStatus(int user_id, bool online) {
    redis.set("user:" + std::to_string(user_id) + ":online", online ? "1" : "0");
}

bool isUserOnline(int user_id) {
    std::string status = redis.get("user:" + std::to_string(user_id) + ":online").value();
    return status == "1";
}
```

6.10 使用 GeminiDB Redis 在 Web 应用中实现会话管理

在Web应用中使用Redis实现会话管理是一种高效且常见的方法，尤其适用于分布式系统。以下是基于GeminiDB Redis（100%兼容redis协议）实现会话管理的详细步骤和实现方法。

使用场景

- 分布式支持：在分布式系统中，传统的服务器端会话管理（如基于内存的 Session）无法跨多个服务器实例共享会话信息。GeminiDB Redis作为公共服务，可以轻松解决这一问题。
- 高性能：GeminiDB Redis是分布式可扩展架构，能够高效地处理高并发场景。
- 自动过期：GeminiDB Redis支持键的自动过期机制，可以方便地管理会话的生命周期。

GeminiDBRedis 进行会话管理的优势

- 高可用性：GeminiDB Redis的持久化能力确保会话数据不会因单点故障而丢失。
- 可扩展性：GeminiDB Redis支持集群模式，可以轻松扩展以应对高并发。
- 安全性：通过配置访问控制和加密功能，可以增强会话数据的安全性。

前提条件

已购买GeminiDB Redis实例，具体参见[购买GeminiDB Redis实例](#)。

步骤 1 创建 Spring Boot 项目

创建Spring Boot项目：使用 Spring Initializr 创建一个 Spring Boot 项目，并添加以下依赖：

```
<dependency>
  <groupId>org.springframework.session</groupId>
  <artifactId>spring-session-data-redis</artifactId>
</dependency>
<dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-data-redis</artifactId>
</dependency>
<dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-web</artifactId>
</dependency>
```



注意

spring-boot请使用3.0.x及以上版本，且Jdk版本>=17。

工程目录结构：

```
src
├── main
│   ├── java
│   │   ├── com
│   │   │   └── huawei
│   │   │       ├── sessionmanagement
│   │   │       │   ├── SessionConfig.java
│   │   │       │   ├── SessionController.java
│   │   │       └── SessionManagementApplication.java
│   └── resources
│       └── application.properties
```

步骤 2 配置 Redis 并启用 Redis 会话管理

在"application.properties"文件中配置 GeminiDB Redis 连接信息：

```
spring.data.redis.host=${ip}
spring.data.redis.port=${port}
spring.data.redis.password=${password}
server.servlet.session.timeout=30 # 会话超时时间（分钟）
```

在Spring Boot应用的主类或配置类中，使用"@EnableRedisHttpSession"注解启用Redis会话支持：

```
package com.huawei.sessionmanagement;
import org.springframework.session.data.redis.config.annotation.web.http.EnableRedisHttpSession;

@EnableRedisHttpSession
public class SessionConfig {
    // 该注解会自动配置 Spring Session 使用 Redis 存储会话信息
}
```

步骤 3 实现登录和登出逻辑

创建一个控制器来处理用户登录和登出逻辑：

```
package com.huawei.sessionmanagement;
import jakarta.servlet.http.HttpSession;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestParam;
import org.springframework.web.bind.annotation.RestController;

@RestController
public class SessionController {
    @PostMapping("/login")
    public String login(@RequestParam String username, HttpSession session) {
        session.setAttribute("username", username);
        return "Login successful";
    }

    @PostMapping("/logout")
    public String logout(HttpSession session) {
        session.invalidate();
        return "Logout successful";
    }
}
```

步骤 4 获取会话信息及自定义会话存储

可以通过"HttpSession"获取会话信息，在SessionController类中添加以下代码：

```
import org.springframework.web.bind.annotation.GetMapping;

@GetMapping("/getSessionInfo")
public String getSessionInfo(HttpSession session) {
    String username = (String) session.getAttribute("username");
    return "Username: " + username;
}
```

如果需要更细粒度的控制，可以直接使用 RedisTemplate 存储和管理会话数据：

```
import org.springframework.data.redis.core.RedisTemplate;
import org.springframework.stereotype.Service;

@Service
public class SessionService {

    private final RedisTemplate<String, Object> redisTemplate;
```



```
public SessionService(RedisTemplate<String, Object> redisTemplate) {
    this.redisTemplate = redisTemplate;
}

public void saveSession(String sessionId, Object sessionData) {
    redisTemplate.opsForValue().set(sessionId, sessionData, 30, TimeUnit.MINUTES);
}

public Object getSession(String sessionId) {
    return redisTemplate.opsForValue().get(sessionId);
}

public void deleteSession(String sessionId) {
    redisTemplate.delete(sessionId);
}
}
```

通过上述步骤，您可以在Web应用中高效地使用GeminiDB Redis实现会话管理，提升应用的性能和可扩展性。

6.11 GeminiDB Redis 连接检测机制和客户端重连重试建议

GeminiDB Redis 连接检测机制

在网络状况良好且客户端连接正常的前提下，GeminiDB Redis不会主动断开用户的连接。和开源Redis连接机制相同，服务端启用了 TCP Keepalive 机制，可以检测并识别由于网络中断、客户端崩溃或NAT超时等原因导致的异常连接，并及时清理这些无效连接，以防止服务端的连接资源被耗尽，从而保障服务的稳定性与可用性。

参数解释

表 6-3 参数说明

参数	说明
ClientTimeout	- 空闲超时参数，表示用于判断服务端检测到该连接空闲多久后，主动断开。 - 默认值为0，表示GeminiDB不会主动关闭空闲连接。 - 只作用于普通客户端连接，对Pub/Sub连接和阻塞命令（如brpop）连接不生效。
tcp-keepalive	表示GeminiDB向空闲连接发送TCP Keepalive探测包的频率，默认为120s。
TCP_KEEPINTVL	表示探测包发送的时间间隔（秒），默认为40s。
TCP_KEEPCNT	表示发送多少次探测包，默认为3次。

检测机制

- 当连接空闲时间超过tcp-keepalive设置的时间后，GeminiDB会开始发送Keepalive探测包，以确认连接是否仍然有效。

2. 如果对方没有响应，GeminiDB会每隔TCP_KEEPINTVL秒重试一次探测。
3. 如果连续TCP_KEEPCNT次探测均无响应，则判定该连接已失效，GeminiDB会主动断开该连接，以释放服务端的连接资源，保障整体服务的稳定性和可用性。

📖 说明

如果客户使用的GeminiDB Redis实例内核小版本低于5.0.6.4，其tcp-keepalive设置较为敏感，且不支持动态调整，建议升级至最新版本。新版本不仅优化了探活机制，降低了因网络波动导致连接误断的小概率问题，还增强了连接管理的灵活性与稳定性，有助于提升整体服务的可靠性。

客户端连接重连和命令重试使用介绍

在实际应用中，客户端与服务端之间的网络通信可能会受到各种因素的影响，如网络丢包、抖动异常或临时性故障等。为提升系统的健壮性和用户体验，建议客户端引入连接重连和命令重试机制，以降低网络抖动的影响。

连接重连

在网络出现丢包、抖动等情况，或是进行GeminiDB在线滚动节点规格变更、版本升级等操作时，可能会导致客户端与GeminiDB实例之间的连接中断。为了确保操作能够成功执行，客户端应当具备自动重连机制。推荐采用主流的Redis客户端，比如Jedis，在获取连接之前先进行连接检测，或者定期检查空闲连接的状态，以此来防止因连接问题而导致的操作超时或失败。这样可以有效提升系统的连接性和可靠性。

- 设置testonBorrow，在获取连接时进行有效性检测。

```
JedisPoolConfig poolConfig = new JedisPoolConfig();  
// 设置从连接池获取连接时进行有效性检查  
poolConfig.setTestOnBorrow(true);  
// 其他配置...  
  
// 创建连接池时使用配置  
JedisPool jedisPool = new JedisPool(poolConfig, "localhost", 6379);
```

当testOnBorrow设置为true时，每次从连接池获取连接时会进行连接有效性检测（即执行一次ping操作）。如果检测发现连接已失效，当前连接会被丢弃，并尝试获取另一个可用连接。这个设置可以确保每次获取到的连接都是有效的，但会带来一次额外的ping检测开销。在业务并发量较高的场景下，可以考虑启用空闲连接检测（如testWhileIdle），这样可以在连接未被使用时提前检测其有效性，减少每次获取连接时的检测开销，从而提升整体性能。

- 空闲连接检测，通过异步线程来探测连接是否正常，相比testonBorrow开销小一些。

```
JedisPoolConfig poolConfig = new JedisPoolConfig();  
poolConfig.setTestWhileIdle(true);  
// 空闲连接检测周期（30-60秒）  
poolConfig.setTimeBetweenEvictionRunsMillis(30000);  
// 每次检查的连接数（建议为负值，表示检查全部空闲连接）  
poolConfig.setNumTestsPerEvictionRun(-1);  
// 连接最小空闲时间（超过此值才开始检测）  
poolConfig.setMinEvictableIdleTimeMillis(60000); // 60秒  
JedisPool jedisPool = new JedisPool(poolConfig, "localhost", 6379);
```

表 6-4 参数说明

名称	说明	默认值	建议
testWhileIdle	是否在空闲资源监测时通过ping命令监测连接有效性，无效连接将被销毁。	false	true
timeBetweenEvictionRunsMillis	空闲资源的检测周期（单位为毫秒）	-1（不检测）	建议设置，周期自行选择，可设置30000。
minEvictableIdleTimeMillis	资源池中资源的最小空闲时间（单位为毫秒），达到此值后空闲资源将被移除。	1,800,000（即30分钟）	建议设置为默认值。
numTestsPerEvictionRun	做空闲资源检测时，每次检测资源的个数。	3	可根据自身应用连接数进行微调，建议设置为 -1，对所有连接做空闲监测。

命令重试

在Redis命令执行失败时，例如由于网络抖动导致的超时，对于像SET key value这类具有幂等性的操作，可以考虑进行重试，从而避免短暂网络问题带来的影响。而对于LPUSH、LPOP等非幂等操作，重试可能会导致数据被多次插入或删除，进而引发业务逻辑错误。因此，在使用重试策略时，需结合具体业务场景进行分析，判断是否适合重试，并合理设置重试次数与重试间隔，以保障操作的成功执行。

```
public <T> T executeWithRetry(RedisCommand<T> command, int maxAttempts) {
    JedisException lastException = null;

    for (int attempt = 1; attempt <= maxAttempts; attempt++) {
        try (Jedis jedis = jedisPool.getResource()) {
            return command.execute(jedis);
        } catch (JedisConnectionException e) {
            lastException = e;
            log.warn("Redis connection failed (attempt {}/{}): {}",
                attempt, maxAttempts, e.getMessage());

            // 可通过sleep或者指数退避等待策略，间隔一段时间再重试
            Thread.sleep(waitTime);
        }
    }
    throw new RedisOperationException("Max retry attempts reached", lastException);
}
```

7 性能白皮书

7.1 GeminiDB Redis 接口通用性能数据

7.1.1 性能测试方法

本章介绍GeminiDB Redis性能测试的方法，具体包括测试环境，测试工具，测试指标，测试模型，以及测试步骤。

测试环境

- 区域：华北-北京四
- 可用区：可用区1
- 弹性云服务器（Elastic Cloud Server，简称ECS）：规格选择c6.4xlarge.2，16U32GB，操作系统镜像使用CentOS 7.5 64位版本。
- 被测试实例的配置：每个实例均包含3个节点。
- 被测试实例的规格：覆盖以下规格类型，详见表7-1。

表 7-1 实例规格

编号	规格
cluster1	4U*3节点
cluster2	8U*3节点

测试工具

本次测试采用Redis Labs推出的多线程压测工具memtier_benchmark，具体使用方法请参见[memtier_benchmark](#)。下面就使用到的memtier_benchmark的部分功能进行简单介绍。

Usage: memtier_benchmark [options]

A memcache/redis NoSQL traffic generator and performance benchmarking tool.

Connection and General Options:	
-s, --server=ADDR	Server address (default: localhost)
-p, --port=PORT	Server port (default: 6379)
-a, --authenticate=PASSWORD	Authenticate to redis using PASSWORD
-o, --out-file=FILE	Name of output file (default: stdout)
Test Options:	
-n, --requests=NUMBER	Number of total requests per client (default: 10000)
-c, --clients=NUMBER	Number of clients per thread (default: 50)
-t, --threads=NUMBER	Number of threads (default: 4)
--ratio=RATIO	Set:Get ratio (default: 1:10)
--pipeline=NUMBER	Number of concurrent pipelined requests (default: 1)
--distinct-client-seed	Use a different random seed for each client
--randomize	Random seed based on timestamp (default is constant value)
Object Options:	
-d --data-size=SIZE	Object data size (default: 32)
-R --random-data	Indicate that data should be randomized
Key Options:	
--key-prefix=PREFIX	Prefix for keys (default: memtier-)
--key-minimum=NUMBER	Key ID minimum value (default: 0)
--key-maximum=NUMBER	Key ID maximum value (default: 10000000)

测试指标

表 7-2 测试指标

指标缩写	指标描述
QPS	每秒执行的读写操作数，单位为次/秒。
Avg Latency	读写操作的平均时延，单位为毫秒 (ms)。
P99 Latency	- 读写操作的P99时延。 99%的操作可以在这个时延内执行完成，即只有1%的操作时延会超过这个值。 - 单位为毫秒(ms)。

测试模型

- workload模型

表 7-3 workload 模型

workload模型编号	测试模型
100% Write	100%写操作（string set）。
100% Read	100%读操作（string get），采用均匀随机访问模型，按照严苛场景测试读性能。
50% Read+50% Write	50%读操作（string get）+ 50%写操作（string set）。

- 数据模型

表 7-4 数据模型

数据模型编号	数据模型
value length	随机生成value，长度为100字节。

测试场景

表 7-5 测试场景

测试场景	测试场景描述
总数据量小于内存	数据可以全部缓存进内存。
总数据量大于内存	部分数据可以缓存进内存，部分数据需要访问DFV存储池。

测试步骤

以4U*3节点数据库实例为例：

1.首先进行总数据量小于内存大小场景下的写入，读取，以及同时写入和读取操作，并记录各操作的QPS、Avg Latency、P99 Latency。各个workload模型的性能指标的方法如下所示：

- 测试模型：100% Write模型
使用30个线程，每个线程创建3个client连接，即总共建立的90个连接并发写入60,000,000次长度为100字节的数据，其中数据为各client采用不同seed在[1, 60,000,000]范围内随机生成。基于key的给定范围，本次写入总数据大小小于数据库集群的内存容量。

```
./memtier_benchmark -s ${ip} -a ${passwd} -p ${port} -c 3 -t 30 -n 1000000 --random-data --randomize --distinct-client-seed -d 100 --key-maximum=60000000 --key-minimum=1 --key-prefix= --ratio=1:0 --out-file=./output_filename
```
- 测试模型：100% Read模型
使用30个线程，每个线程创建3个client连接，即总共建立的90个连接并发均匀随机读取60,000,000次数据，读取key范围在[1, 60,000,000]内。

```
./memtier_benchmark -s ${ip} -a ${passwd} -p ${port} -c 3 -t 30 -n 1000000 --random-data --randomize --distinct-client-seed --key-maximum=60000000 --key-minimum=1 --key-prefix= --ratio=0:1 --out-file=./output_filename
```
- 测试模型：50% Read+50% Write模型
使用30个线程，每个线程创建3个client连接，即总共建立的90个连接并发写入和读取60,000,000次的数据，写入和读取key范围在[1, 60,000,000]内，同时写入和读取操作比例为1:1。基于key的给定范围，本次写入和读取总数据大小小于数据库集群的内存容量。

```
./memtier_benchmark -s ${ip} -a ${passwd} -p ${port} -c 3 -t 30 -n 1000000 --random-data --randomize --distinct-client-seed -d 100 --key-maximum=60000000 --key-minimum=1 --key-prefix= --ratio=1:1 --out-file=./output_filename
```

2. 在数据库中增加超过数据库集群内存容量的数据：使用30个线程，每个线程创建3个client连接，即总共建立的90个连接并发写入20,000,000次长度为100字节的数据，其中数据为各client采用不同seed在[60,000,001, 780,000,000]范围内随机生成，同时通过设置pipeline参数，增加数据写入效率。基于key的给定范围以及总共写入次数，本次写入总数据大小大于数据库集群的内存容量。

```
./memtier_benchmark -s ${ip} -a ${passwd} -p ${port} -c 3 -t 30 -n 20000000 --random-data --randomize --distinct-client-seed -d 100 --key-maximum=780000000 --key-minimum=60000001 --pipeline=100 --key-prefix= --ratio=1:0 --out-file=./output_filename
```

3. 数据库存储数据量大于数据库集群内存容量条件下，进行写入、读取、同时写入和读取操作，并记录各操作的QPS、Avg Latency、P99 Latency。各个workload模型的性能指标的方法如下所示：

- 100% Write模型

使用30个线程，每个线程创建3个client连接，即总共建立的90个连接并发写入500,000次长度为100字节的数据，其数据为各client随机生成，key范围在[1, 780,000,000]内。

```
./memtier_benchmark -s ${ip} -a ${passwd} -p ${port} -c 3 -t 30 -n 500000 --random-data --randomize --distinct-client-seed -d 100 --key-maximum=780000000 --key-minimum=1 --key-prefix= --ratio=1:0 --out-file=./output_filename
```

- 100% Read模型

使用30个线程，每个线程创建3个client连接，即总共建立的90个连接并发均匀随机读取500,000次数据，key范围在[1, 780,000,000]内。

```
./memtier_benchmark -s ${ip} -a ${passwd} -p ${port} -c 3 -t 30 -n 500000 --random-data --randomize --distinct-client-seed --key-maximum=780000000 --key-minimum=1 --key-prefix= --ratio=0:1 --out-file=./output_filename
```

- 50% Read+50% Write模型

使用30个线程，每个线程创建3个client连接，即总共建立的90个连接并发写入和读取500,000次的数据，其数据为各client随机生成，key范围在[1, 780,000,000]内。

```
./memtier_benchmark -s ${ip} -a ${passwd} -p ${port} -c 3 -t 30 -n 500000 --random-data --randomize --distinct-client-seed -d 100 --key-maximum=780000000 --key-minimum=1 --key-prefix= --ratio=1:1 --out-file=./output_filename
```

7.1.2 性能测试结果

本章介绍GeminiDB Redis性能测试结果，根据上述测试方法操作，展示在各种数据模型、测试场景、Workload模型组合下的性能指标。当前性能白皮书仅呈现中小规格并发能力下的数据库性能数据，如需更高的并发能力，可水平或垂直升级数据库规格。

- 总数据量小于内存场景下的测试数据请参见[表7-6](#)。
- 总数据量大于内存场景下的测试数据请参见[表7-7](#)。

表 7-6 总数据量小于内存场景测试数据

实例规格	测试模型	Workload模型	QPS（次/秒）	Average Latency（毫秒）	P99 Latency（毫秒）
4U*3节点	value_length=100字节 clients=90	100% Write	125590	0.66	1.85

实例规格	测试模型	Workload 模型	QPS (次/秒)	Average Latency (毫秒)	P99 Latency (毫秒)
	value_length=100字节 clients=105	100% Read	139741	0.62	1.51
	value_length=100字节 clients=90	50% Read +50% Write	125620	Read:0.56	Read:1.32
				Write:0.55	Write:1.30
8U*3节点	value_length=100字节 clients=128	100% Write	216392	0.62	1.92
	value_length=100字节 clients=128	100% Read	202970	0.62	1.89
	value_length=100字节 clients=128	50% Read +50% Write	212052	Read:0.63	Read:1.94
				Write:0.63	Write:1.92

表 7-7 总数据量大于内存场景测试数据

实例规格	测试模型	Workload 模型	QPS (次/秒)	Average Latency (毫秒)	P99 Latency (毫秒)
4U*3节点	value_length=100字节 clients=75	100% Write	123942	0.62	1.30
	value_length=100字节 clients=96	100% Read	125351	0.63	1.54
	value_length=100字节 clients=96	50% Read +50% Write	122485	Read:0.64	Read:1.65
				Write:0.64	Write:1.61
8U*3节点	value_length=100字节 clients=120	100% Write	196596	0.62	2.02
	value_length=100字节 clients=120	100% Read	187716	0.62	1.90

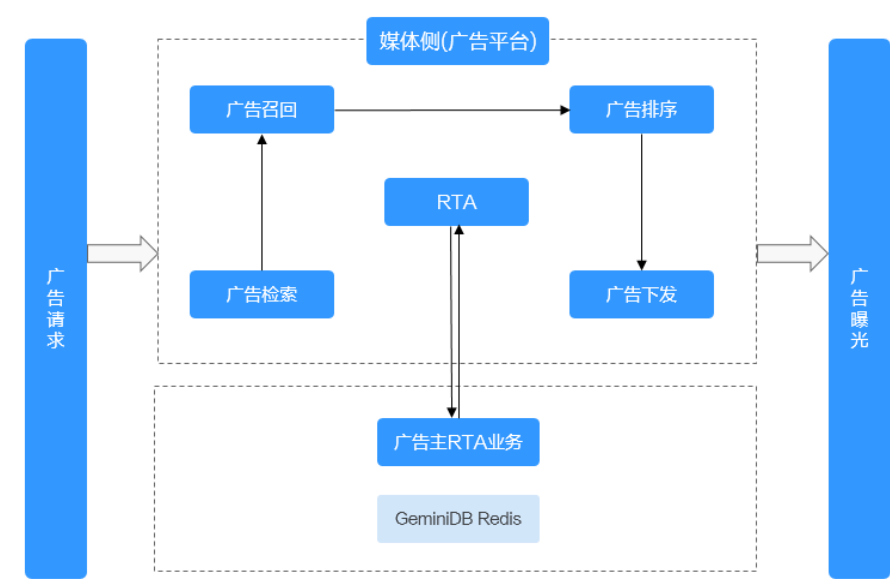
实例规格	测试模型	Workload 模型	QPS (次/秒)	Average Latency (毫秒)	P99 Latency (毫秒)
	value_length=100字节 clients=120	50% Read +50% Write	197097	Read:0.62 Write:0.62	Read:1.94 Write:1.94

注：clients是连接数，对应memtier命令中t和c字段的乘积。

7.2 GeminiDB Redis 接口广告 RTA 场景性能数据

RTA（RealTime API）系统是满足实时精细化投放的技术手段，属于核心业务。广告平台在进行广告检索阶段，通过RTA接口向广告主询问其广告是否参与竞价，广告平台在接收到该信息后再进行广告检索召回，一个完整的RTA广告流程如下图所示：

图 7-1 RTA 广告流程



7.2.1 性能测试方法

测试目的

广告RTA业务对广告主的技术要求较高，对于广告主来说，一方面需要满足媒体侧的快速响应要求，另一方面还要求数据存储成本可控。近年来，越来越多的RTA业务使用云数据库GeminiDB Redis作为KV特征库，性能与成本双赢。

本章节基于真实RTA业务做压力测试，评估GeminiDB Redis的数据压缩能力、QPS、带宽、时延等各项性能指标表现。

测试环境

本次测试使用的GeminiDB Redis集群规格和弹性云服务器（ Elastic Cloud Server，简称ECS ）规格如下：

- GeminiDB Redis规格

局点	上海一
可用区类型	可用区一/二/三混合部署
节点CPU规格	16 vCPUs
节点数量	20
实例总容量	2 TB

- ECS规格：

可用区类型	AZ1
规格	c7.4xlarge.2，3台
CPU	16vCPUs
内存	32GiB
操作系统	CentOS 8.2 64bit

测试工具

本次测试采用Redis Labs推出的多线程压测工具memtier_benchmark，具体使用方法请参见[memtier_benchmark](#)。

测试指标

本次模拟的广告业务场景（ RTA ）业务规模大致抽象为：1TB数据量、160w QPS、1.5Gbit/s带宽。

1. 数据样本

本次测试使用的数据样本主要分为以下三种：

类型	Key	Value
Hash	34位字符	10对field(10位)-value(20-80位)
String	68位字符	32位随机字符
String	19位字符	500 – 2000位随机字符

其中，需要存储在Redis中的Key总数约为**40亿条**。各类型数据占比约为2:7:1，高频访问的数据约占总体的50%。

2. 评估指标

对于上述测试模型及场景，记录各数据库操作的如下测试指标：

指标缩写	指标描述
QPS	每秒执行的请求数，单位为次/秒。
Avg Latency (ms)	请求的平均时延，代表GeminiDB Redis整体性能表现。
P99 Latency (ms)	请求的P99时延，是比较严格的时延指标，表示99%的请求执行时间都小于该值。
P9999 Latency(ms)	请求的P9999时延，是非常严格的时延指标，表示99.99%的请求执行时间小于该值，仅少量尾部请求超过该值。

测试步骤

1. 注入测试数据

测试前，生成并注入数据库测试数据。基于测试模型三种类型的分布，对三种数据类型进行如下配置：

a. hash类型

- key: 34位字符，使用字符串前缀+9位数字，数字由1亿-9亿连续，以控制数据总量和热数据分布。
- field-value共注入10对，其中field为10位字符，value为20-80位随机字符，注入测试数据时取均值50位。
- 构造并注入约8亿个key:

```
memtier_benchmark -s ${ip} -a ${passwd} -p ${port} -c 20 -t 20 -n 7500000 -d 32 --key-maximum=3800000000 --key-minimum=1000000000 --key-prefix='cefkjlrithuir123894873h4523blj4b2jkjh2iw13bnfdhsbnkfhsdjk' --key-pattern=P:P --ratio=1:0 --pipeline=100
```

b. string类型

- key: 68位字符，使用字符串前缀+10位数字，数字由10亿-38亿连续，以控制数据总量和热数据分布。
- value: 注入32位随机字符。
- 构造并注入约28亿个key:

```
memtier_benchmark -s ${ip} -a ${passwd} -p ${port} -c 20 -t 20 -n 2500000 --command='hset __key__ mendke398d __data__ mebnkejkehe __data__ fmebejdbnf __data__ j3i45u8923 __data__ j43245i908 __data__ jhiriu2349 __data__ 21021034ji __data__ jh23ui45j2 __data__ jiu5rj9234 __data__ j23io45u29 __data__' -d 50 --key-maximum=900000000 --key-minimum=100000000 --key-prefix='ewfdjkff43ksdh41fuihikucl' --command-key-pattern=P --pipeline=100
```

c. string类型

- key: 19位字符，使用字符串前缀+9位数据，数字由1亿到3亿连续，以控制数据总量和热数据分布。
- value: 500 – 2000位随机字符，注入测试数据时取均值1250位。

- 构造并注入约4亿个key:

```
memtier_benchmark -s ${ip} -a ${passwd} -p ${port} -c 20 -t 20 -n 520000 -d 1250 --key-maximum=300000000 --key-minimum=100000000 --key-prefix='miqjkfdjiu' --key-pattern=P:P --ratio=1:0 --pipeline=100
```

数据注入完成后，观察其key个数为3,809,940,889个key（约38亿）。观察GeminiDB Redis控制台中使用数据总量，计算GeminiDB Redis的数据压缩比。压缩后的存储容量约为155GB，即压缩比约为13.8%。

⚠ 注意

- 受memtier_benchmark数据平铺时数据生成影响，生成数据在40亿条左右，各类型间数据分布不受影响。
- memtier_benchmark工具构造随机字符串中连续字符较多，因此压缩比偏低。根据经验，实际生产数据压缩比一般在30%-50%左右，仍可以达到很好的压缩效果。

2. 压测命令

在三台ECS上对GeminiDB Redis实例执行多个压测任务，压测任务分别为：

- a. ECS1上，对类型一进行hgetall查询操作，通过key范围控制仅访问部分高频数据：

```
memtier_benchmark -s ${ip} -a ${passwd} -p ${port} -c 20 -t 30 --test-time 1200 --random-data --randomize --distinct-client-seed --command='hgetall __key__' --key-maximum=600000000 --key-minimum=200000000 --key-prefix='ewfdjkff43ksdh41fuihikucl' --out-file=./output_filename
```

- b. 对类型二进行get查询操作，通过key范围控制仅访问部分高频数据：

```
memtier_benchmark -s ${ip} -a ${passwd} -p ${port} -c 70 -t 30 --test-time 1200 --random-data --randomize --distinct-client-seed --key-maximum=2400000000 --key-minimum=1000000000 --key-prefix='cefkjlrithuin123894873h4523bhj4b2jkjh2iu13bnfdhsbnkfhdsdjkjh' --ratio=0:1 --out-file=./output_filename
```

- c. 对类型三进行get查询操作，通过key范围控制仅访问部分高频数据：

```
memtier_benchmark -s ${ip} -a ${passwd} -p ${port} -c 10 -t 30 --test-time 1200 --random-data --randomize --distinct-client-seed --key-maximum=300000000 --key-minimum=100000000 --key-prefix='miqjkfdjiu' --ratio=0:1 --out-file=./output_filename
```

其中，连接数(c、t两个参数乘积)通过调整各个压测实例的client数量及配置使整体达到160w QPS，同时读请求流量1.5Gb/s。保持该业务流量，评估GeminiDB Redis的性能表现。

7.2.2 性能测试结果

基于上述样本，预先注入1TB+数据并进行压力测试，测试结果如下：

- 数据压缩率：

写入1.1TB数据(约38亿条)，压缩后数据占用约为155GB，数据压缩比约为13.8%；

- 性能表现：

维持业务总QPS达到约160w，此时读请求总流量约为1.5Gb/s，实例CPU利用率在60%-70%。

平均时延约为0.7ms，P99长尾时延约为1.77ms。

本次测试结果表明，在大规模RTA场景，GeminiDB Redis有稳定的时延性能，同时基于数据压缩和支持计算/存储独立选配的特性，非常适合作为广告业务的KV数据库选型。

8 常见问题

8.1 高频常见问题

产品咨询

- [GeminiDB Redis和开源Redis、其他开源Redis云服务有什么区别？](#)
- [和开源Redis相比，GeminiDB Redis性能如何？](#)
- [GeminiDB Redis兼容Redis哪些版本，兼容哪些命令，客户端连接是否需要修改](#)
- [自建Redis是否可以搬迁至GeminiDB Redis，需要注意什么](#)
- [GeminiDB Redis实例总容量是总内存吗，内存和容量之间是什么联系](#)
- [购买GeminiDB Redis实例时，如何选择合适的节点规格和节点数量？](#)
- [购买x GB的GeminiDB Redis的实例，优选主备还是集群？](#)
- [GeminiDB Redis持久化机制是怎样的，会丢数据吗](#)
- [GeminiDB Redis的内存淘汰策略是什么](#)
- [GeminiDB Redis是否支持布隆过滤器等modules](#)

数据库连接

- [如何接入GeminiDB Redis](#)
- [如何使用GeminiDB Redis提供的多个节点IP地址](#)
- [GeminiDB Redis提供的ELB的实现方式是怎样的](#)
- [GeminiDB Redis实例购买成功后是否支持更换VPC](#)
- [内网如何访问GeminiDB Redis](#)
- [GeminiDB Redis自带的负载均衡地址是否能绑定公网IP？如何通过公网连接GeminiDB Redis实例？](#)
- [设置了安全组，还需要设置负载均衡内网访问控制吗？](#)
- [绑定了弹性公网IP但是连接不上数据库](#)
- [常见客户端报错及解决方法](#)

数据库使用

- [scan指定match参数，数据中确实存在匹配的key，为什么返回的是空](#)
- [业务侧原本做了数据分片，切换到GeminiDB Redis后如何处理这部分逻辑](#)
- [如何处理报错：CROSSSLOT Keys in request don't hash to the same slot](#)
- [GeminiDB Redis存储扩容需要多久，对业务有影响吗？](#)
- [GeminiDB Redis多个节点同时扩容需要多长时间，对业务影响如何？](#)
- [GeminiDB Redis规格变更包含的在线变更和离线变更有什么区别，通常需要多长时间，对业务有哪些影响？](#)
- [GeminiDB Redis版本补丁升级包含的在线升级和离线升级有什么区别，通常需要多长时间，对业务有哪些影响？](#)
- [GeminiDB Redis备份文件是否可以下载到本地，是否支持线下恢复数据](#)
- [GeminiDB Redis数据备份工作机制是怎样的，对业务有哪些影响？](#)
- [购买GeminiDB Redis 1U*2节点特惠型实例后，业务访问量比较少，但CPU占用率比较高，是什么原因？](#)
- [GeminiDB Redis监控面板上key数量下降又恢复至正常数量是什么原因？](#)
- [GeminiDB Redis节点CPU偶发冲高，可能是哪些原因](#)
- [GeminiDB Redis集群版实例中，哪些命令需要使用hashtag](#)
- [如何处理报错：CROSSSLOT Keys in request don't hash to the same slot](#)
- [如何处理报错“ERR unknown command sentinel”](#)

备份与恢复

[GeminiDB Redis实例能够保存多长时间的备份](#)

数据迁移

- [4.0、5.0以及6.2版本的自建Redis能迁移至GeminiDB Redis吗？](#)
- [自建Redis主备、集群实例如何迁移到GeminiDB Redis？](#)
- [为什么阿里云Redis、腾讯云Redis等云服务不能使用DRS进行数据迁移？](#)
- [自建主备Redis，迁移到GeminiDB Redis集群，需要考虑哪些因素？](#)
- [迁移完成后数据量变少了，100GB的数据迁移到GeminiDB Redis只有20-30GB，数据是不是没迁移完？](#)

8.2 产品咨询

8.2.1 GeminiDB Redis 和开源 Redis、其他开源 Redis 云服务有什么区别？

开源Redis是常规的缓存产品，纯内存架构。GeminiDB Redis在开源Redis上做了增强，是一款兼容Redis协议的弹性KV数据库，支持远超内存的容量和极致的性能。热数据在内存中，全量数据在存储池中。有以下特点：

- 稳定低延迟：

单点读写平均时延<1ms，P99<2ms；采用多线程架构，单实例QPS可支撑1w-1000w线性扩展。

- 高性价比：
综合降本30%：无需备节点，具备4:1超高数据压缩，存储扩容成本更低。
- 运维效率提升：
可实现2GB到百TB的无感存储扩容，支持极速数据回档（PITR）等能力。
- 具有更多的增强特性：
支持Hash子field过期能力、布隆过滤器、数据极速导入、内存加速等企业级特性。

GeminiDB Redis和自建开源KV数据库详细对比请参考[产品优势](#)。

8.2.2 和开源 Redis 相比，GeminiDB Redis 性能如何？

GeminiDB Redis采用多线程架构，QPS可随CPU数量线性扩展，可支持1w-1000w线性扩展；

单点访问通常平均时延<1ms，P99<2ms，和开源Redis时延接近。

详细性能数据可参考[性能测试报告](#)。

8.2.3 GeminiDB Redis 兼容 Redis 哪些版本，兼容哪些命令，客户端连接是否需要修改

GeminiDB Redis完全兼容Redis 6.2(包含6.2.x)协议及以下版本，如5.0、4.0、2.8，部分兼容7.0版本。

6.2及以下版本（如5.0、4.0、2.8）等都可以迁移到GeminiDB Redis，无需业务修改，且兼容任一Redis客户端接入。

GeminiDB Redis命令兼容详情请参考[命令兼容列表](#)。

8.2.4 自建 Redis 是否可以搬迁至 GeminiDB Redis，需要注意什么

可以。搬迁前请确定以下事项：

- 版本：如果源实例版本小于等于6.2(包含6.2.x)，可直接搬迁；若使用高版本命令，则需要评估后搬迁至6.2版本，您可以在管理控制台右上角，选择“[工单](#) > [新建工单](#)”，联系客服申请开通。
- 选择合适的规格：根据源端实例的QPS、数据量等信息，选择合适的规格。

8.2.5 什么是 GeminiDB Redis 实例可用性

实例可用性的计算公式：

实例可用性=（1-故障时间/服务总时间）×100%

其中，故障时间是指数据库实例购买完成后，运行期间累计发生故障的总时长。服务总时间指数据库实例购买完成后运行的总时长。

8.2.6 GeminiDB Redis 实例总容量是总内存吗，内存和容量之间是什么联系

不是。

在开源Redis中，全量数据存储在内存中，总容量由内存大小决定。

在GeminiDB Redis中，全量数据存储在高性能共享存储池中，热数据存储在内存中，通常用户仅需关注实例总容量和使用情况即可。如果QPS增大，CPU使用率会上涨，需要进行规格扩容。

8.2.7 购买 GeminiDB Redis 实例时，如何选择合适的节点规格和节点数量？

购买Redis实例通常需要关注2个主要指标：QPS和数据量大小。购买GeminiDB Redis实例时，也是依据这两个指标进行选择，可以采用“快速选择”和“自定义”购买实例：

- 使用“快速选择”购买实例，例如原先采用的是16GB集群，可以对标选择GeminiDB Redis16GB集群。若对应的规格QPS不满足业务需求，则需要选择更高规格。
- 使用“自定义”选择时，计算和存储需要单独选择，节点规格和节点数量决定实例的QPS，实例总容量决定数据存储容量上限。选择节点规格、节点数量、实例总容量后，可以从“实例规格预览”查看所选择实例的QPS、连接数等信息。

8.2.8 购买 x GB 的 GeminiDB Redis 的实例，优选主备还是集群？

优选集群。集群最小支持4GB，相比主备架构，可扩展性好，同规格QPS也会更高。集群有以下优点：

- GeminiDB Redis集群所有计算节点可读可写，节点资源利用率100%；而主备架构，备节点shard不能承担写操作，资源利用率低。
- GeminiDB Redis集群为Proxy集群，兼容单机、Cluster集群访问方式。

8.2.9 GeminiDB Redis 持久化机制是怎样的，会丢数据吗

开源Redis定期持久化，因此异常场景数据丢失风险概率比较大。GeminiDB Redis数据实时刷新至存储池，数据安全性更高。

和业界的其他NoSQL数据库一样，出于性能考虑，GeminiDB Redis的WAL(Write Ahead Log)日志写完操作系统的BUFFER会立即返回，再由操作系统刷新至存储池，因此异常断电情况可能存在少量数据丢失。

GeminiDB Redis在规格变更、版本升级、添加节点等日常正常运维操作，则不会存在数据丢失。同步写会严重降低写入性能，若数据可靠性要求更高，需要开启同步写，您可以在管理控制台右上角，选择“[工单 > 新建工单](#)”进行咨询。

8.2.10 GeminiDB Redis 的内存淘汰策略是什么

开源Redis key如果被内存淘汰，后续就读不到该数据。

GeminiDB Redis默认支持noeviction淘汰策略，即不淘汰用户key；数据全量在存储池中，如果内存的热数据被淘汰，还可以从存储池中读取到该数据，访问后重新被加载到内存中，不会删除用户key。

因此，GeminiDB Redis用户无需自行设置或修改maxmemory-policy参数，如果客户有数据可能不需要，建议通过添加过期时间来避免数据量膨胀。

8.2.11 GeminiDB Redis 是否支持布隆过滤器等 modules

布隆过滤器（BloomFilter）可用于检索某个元素是否存在于某个海量集合中，适用于Web拦截器、防缓存穿透等场景。

GeminiDB Redis 支持布隆过滤器特性。

除此之外，GeminiDB Redis还支持Hash子field过期、分片扫描、大数据极速导入（FastLoad）等功能。

8.3 计费相关

8.3.1 GeminiDB Redis 包年/包月和按需计费模式有什么区别

包年/包月的计费模式也称为包周期计费模式，是一种预付费方式，按订单的购买周期计费，适用于可预估资源使用周期的场景，价格比按需计费模式更优惠。对于长期使用用户，推荐该方式。

按需计费是后付费模式，按实际使用时长计费，这种购买方式比较灵活，可以即开即停。以自然小时为单位整点计费，不足一小时按使用时长计费。

具体请参见[计费模式概述](#)。

8.3.2 GeminiDB Redis 支持包年/包月和按需计费模式相互转换吗

包年/包月和按需计费模式支持相互转换。

- GeminiDB Redis包年/包月实例转按需实例请参见[包年/包月转按需](#)。
- GeminiDB Redis按需计费实例转包年/包月实例请参见[按需转包年/包月](#)。

8.4 数据库使用

8.4.1 scan 指定 match 参数，数据中确实存在匹配的 key，为什么返回的是空

问题描述

如下图所示，数据库中存在key为test的数据，用scan match的方式却没有返回这个数据。

```
139.9.177.148: 6379> scan 1 match tes*
1) "21"
2) (empty list or set)
139.9.177.148: 6379> get test
"abc"
```

```
139.9.177.148: 6379>scan 0 match tes*
1) "21"
2) (empty list or set)
139.9.177.148: 6379>
```

问题分析

MATCH选项让命令只返回和给定模式相匹配的元素，对元素的模式匹配工作是在命令从数据集中取出元素之后，向客户端返回元素之前的这段时间内进行的，如果取出的元素都和模式不匹配，则不会返回任何元素。

解决方案

多次scan，以返回的游标值是否为0作为全遍历结束的标记，每次scan时使用上次scan返回的游标值。

相关知识

- **SCAN命令的基本用法**

SCAN cursor [MATCH pattern] [COUNT count]

SCAN命令是一个基于游标的迭代器（cursor based iterator）。SCAN命令每次被调用之后，都会向用户返回一个新的游标，用户在下次迭代时需要使用这个新游标作为SCAN命令的游标参数，以此来延续之前的迭代过程。

当SCAN命令的游标参数被设置为0时，服务器将开始一次新的迭代，而当服务器向用户返回值为0的游标时，表示迭代已结束。

```
redis 127.0.0.1:6379> scan 0
1) "17"
2) 1) "key:12"
   2) "key:8"
   3) "key:4"
   4) "key:14"
   5) "key:16"
   6) "key:17"
   7) "key:15"
   8) "key:10"
   9) "key:3"
  10) "key:7"
  11) "key:1"
redis 127.0.0.1:6379> scan 17
1) "0"
2) 1) "key:5"
   2) "key:18"
   3) "key:0"
   4) "key:2"
   5) "key:19"
   6) "key:13"
   7) "key:6"
   8) "key:9"
   9) "key:11"
```

在上面这个例子中，第一次迭代使用0作为游标，表示开始一次新的迭代。

第二次迭代使用的是第一次迭代时返回的游标，即命令回复第一个元素的值：17。

从上面的示例可以看到，SCAN命令的回复是一个包含两个元素的数组，第一个数组元素是用于进行下一次迭代的新游标，而第二个数组元素则是一个数组，这个数组中包含了所有被迭代的元素。

在第二次调用SCAN命令时，命令返回了游标0，这表示迭代已经结束，整个数据集（collection）已经被完整遍历过了。

以0作为游标开始一次新的迭代，一直调用SCAN命令，直到命令返回游标0，我们称这个过程为一次完整遍历（full iteration）。

- **COUNT选项**

虽然增量式迭代命令不保证每次迭代所返回的元素数量，但我们可以使用COUNT选项，对命令的行为进行一定程度上的调整。

基本上，COUNT选项的作用就是让用户告知迭代命令，在每次迭代中应该从数据集里返回多少元素。COUNT参数的默认值为10。

并非每次迭代都要使用相同的COUNT值。

用户可以在每次迭代中按自己的需要随意改变COUNT值，只要记得将上次迭代返回的游标用到下次迭代里面。

- **MATCH选项**

MATCH选项让命令只返回和给定模式相匹配的元素，以下是一个使用MATCH选项进行迭代的示例。

```
redis 127.0.0.1:6379> mset foo a foobar b feelsgood c bar d (integer) 4
redis 127.0.0.1:6379> scan 0 match f*
1) "0"
2) 1) "foo"
   2) "feelsgood"
   3) "foobar"
```

需要注意的是，对元素的模式匹配工作是在命令从数据集中取出元素之后，向客户端返回元素之前的这段时间内进行的，所以如果被迭代的数据集中只有少量元素和模式相匹配，那么迭代命令或许会在多次执行中都不返回任何元素。

以下是这种情况的一个例子。

```
redis 127.0.0.1:6379> scan 0 MATCH *11*
1) "288"
2) 1) "key:911"
redis 127.0.0.1:6379> scan 288 MATCH *11*
1) "224"
2) (empty list or set)
redis 127.0.0.1:6379> scan 176 MATCH *11* COUNT 1000
1) "0"
2) 1) "key:611"
   2) "key:711"
   3) "key:118"
   4) "key:117"
   5) "key:311"
   6) "key:112"
   7) "key:111"
   8) "key:110"
   9) "key:113"
  10) "key:211"
  11) "key:411"
  12) "key:115"
  13) "key:116"
  14) "key:114"
  15) "key:119"
  16) "key:811"
  17) "key:511"
  18) "key:11"
```

以上的大部分迭代都不返回任何元素。在最后一次迭代，我们通过将COUNT选项的参数设置为1000，强制命令为本次迭代扫描更多元素，从而使得命令返回的元素也变多了。

8.4.2 业务侧原本做了数据分片，切换到 GeminiDB Redis 后如何处理这部分逻辑

充分考虑到业务后期的规模扩张，GeminiDB Redis采用存算分离架构，在计算层实现了动态数据分片管理，提供强劲的平滑扩缩容能力。

因此，接入GeminiDB Redis实例后，业务侧无需再做数据分片。

8.4.3 GeminiDB Redis 接口是否支持 keys 命令的模糊查询

支持。

使用keys命令进行模糊查询时，可能导致OOM，时延升高等现象，需要谨慎使用。

您可以在业务测试时使用keys命令，生产环境下的业务禁止使用keys命令，建议您使用scan搭配match来进行模糊匹配。

关于scan的使用请参见[scan相关知识](#)。

8.4.4 GeminiDB Redis 是否支持多 DB

GeminiDB Redis的多DB功能已于2022年3月底上线，此前创建的实例暂不支持该功能，也不支持通过升级开启该功能。

使用GeminiDB Redis多DB功能时，您需要关注以下约束与限制：

- DB数支持的范围为0~999。
- 不支持swapdb命令。
- dbsize结果非实时刷新，flushdb后dbsize返回的结果不会立刻降为0，会在一段时间后变为0。
- 不支持在Lua脚本中使用select和flushdb命令。
- 不支持在事务中使用select和flushdb命令。
- 暂不支持move命令。

8.4.5 对于 scan 类的操作，GeminiDB Redis 接口与开源 Redis 5.0 的返回值顺序为什么有差异

开源Redis没有规定如下情况的排序准则，故GeminiDB Redis接口的返回顺序可能和开源Redis不一样，但两者均满足开源文档描述行为。

- scan/hscan/sscan操作的返回值。
- 在zset类型的元素具有相同score时，执行zscan操作的返回值。

8.4.6 对于 scan 类的操作，返回的 cursor 为什么数字很大，与开源 Redis 不一样？

对于scan类的操作，返回的 cursor 为什么数字很大，与开源Redis不一样的原因：

- GeminiDB Redis 返回的 cursor 的编码规则与社区Redis不一样，所以长度会很大；使用方式和行为和开源Redis都一致，能正确遍历客户端所需要的数据。
- 实际上，当 key 的数量足够多的时候，社区 Redis 的cursor 也会增长到这么大，所以这个 cursor 是符合标准的；业务在使用scan类命令时，需要注意将返回的

cursor当成 64位无符号数进行解析，否则可能会导致解析失败或者错误解析，错误解析结果输入到数据库返回“invalid cursor”的报错。

8.4.7 针对某些不合法命令，GeminiDB Redis 接口与开源 Redis 5.0 的报错信息为什么有差异

GeminiDB Redis接口先进行命令语法检查，再进行key存在性校验，而开源Redis没有准则，处理不合法命令的结果存在随机性。

因此在某些不合法命令的报错信息上二者有时会存在差异。

8.4.8 如何处理报错：CROSSSLOT Keys in request don't hash to the same slot

报错场景

GeminiDB Redis执行多Key命令时，可能会出现“CROSSSLOT Keys in request don't hash to the same slot”报错。

报错可能的原因

Redis集群架构不支持跨Slot执行涉及多Key的命令，例如EVAL、BRPOPLPUSH等。

解决方法

- 通过改造Key名称，增加hashtag使其保证在同一个slot，该方案在使用过程中需避免数据倾斜，更多信息请参见[GeminiDB Redis集群版实例中，哪些命令需要使用hashtag](#)。
- 改造实例为主备架构，适用于无法使用Hashtag的业务场景。参见[兼容接口和版本](#)。

8.4.9 GeminiDB Redis 单次事务推荐包含的命令条数

执行单次事务时，推荐所包含命令不高于100条。

如果包含O(n)复杂度的命令，则应该慎重评估。

8.4.10 GeminiDB Redis 集群版实例中，哪些命令需要使用hashtag

Hashtag 简介及使用方法

相较于单机/主备单分片多key命令的使用，原生Cluster集群多key命令的使用需要遵循[hashtag机制](#)，将包含相同hashtag的key分配到相同的哈希槽上（hash slot），以确保多key命令操作的原子性和性能，否则会返回报错“CROSSSLOT Keys in request don't hash to the same slot”。Hashtag使用规则如下：

1. 基本格式

hashtag 用 {} 包裹：只有 {} 内的内容会参与哈希槽计算。

例如：{user:1000}.profile 和 {user:1000}.settings 的 hashtag 都是 user:1000，因此它们会被分配到同一个哈希槽。

2. 位置规则

{ } 可以出现在键的任何位置：

例如：foo{user:1000}bar 的 hashtag 仍然是 user:1000。

只有第一个 { } 有效：

如果键中有多个 { }，只有第一个 { } 的内容会被用作 hashtag。

例如：{user:1000}.{profile} 的 hashtag 是 user:1000，而 profile 会被忽略。

3. 使用场景

- 事务操作：在Redis集群中，事务（MULTI/EXEC）要求所有涉及的键必须在同一个节点上。通过 hashtag 可以确保这些键被分配到同一个哈希槽。
- Lua 脚本：Lua 脚本中的键也需要位于同一节点，hashtag 可以保证这一点。
- 多key操作：如String（mset、mget）、List（blpop、brpop、brpoplpush、rpoplpush）、Set（sdiff、sdiffstore、sinter、sinterstore、sintercard、sunion、sunionstore）、Zset（zinter、zinterstore、zintercard、zunion、zunionstore、zdiff、zdiffstore、zrangestore）、Key管理（del、exists、unlink、touch、rename、renamenx、sort）、Stream（xread、xreadgroup）、bitop等操作。

集群使用示例：

1. 字符串String: mset / mget

- 设置多个键（用户数据）

```
mset {user:1000}:name "Alice" {user:1000}:email "alice@example.com" {user:1000}:age 30
```

- 获取多个键

```
mget {user:1000}:name {user:1000}:email {user:1000}:age
```

2. 事务操作：MULTI/EXEC

- 开启事务

```
MULTI
SET {order:1234}:status "processing"
EXPIRE {order:1234}:status 3600
EXEC
```

3. Lua脚本：

- 脚本减少库存并记录日志

```
EVAL "redis.call('DECR', KEYS[1]); redis.call('SET', KEYS[2], 'updated');" 2 {product:100}:stock
{product:100}:log
```

GeminiDB Redis Proxy 集群支持的拆分命令

相较于原生Cluster集群，Proxy集群承担着命令转发、负载均衡和故障转移等职责，可以简化客户使用逻辑，无需过多关注分片管理，同时支持多DB等高级功能。Proxy集群兼容单机、哨兵、Cluster接入方式，推荐使用Proxy集群架构。

与原生Cluster集群相比，Proxy集群支持部分多key命令拆分，分配到不同的后端节点进行处理，完成后在Proxy上进行聚合再返回客户端，简化了多key命令使用逻辑。

GeminiDB Redis Proxy集群支持拆分命令如下：

- Key管理相关：del、exists、unlink、touch

- String: mget、mset
- Set: sdiff、sdiffstore、sinter、sinterstore、sintercard、sunion、sunionstore
- Zset: zinter、zinterstore、zintercard、zunion、zunionstore、zdiff、zdiffstore、zrangestore
- 支持对事务中多个命令的拆分；如果事务中包含了不支持拆分的多 key 命令，这些命令涉及的 key 需要添加 hashtag。

其他命令则不支持拆分，推荐在集群中使用hashtag，以保证多key命令操作的原子性和性能。其中Key管理及string多key命令，相比Set和Zset更为轻量级，在shard执行后，在Proxy简单聚合即返回客户端；而Set和Zset的多key命令，需要将每个key读入到Proxy再进行相关的逻辑操作，大key场景会有内存增加、访问变慢的风险，不推荐使用。

8.4.11 如何处理报错“ERR unknown command sentinel”

报错场景

GeminiDB Redis实例执行“SENTINEL”命令时，可能会出现“ERR unknown command sentinel”报错。

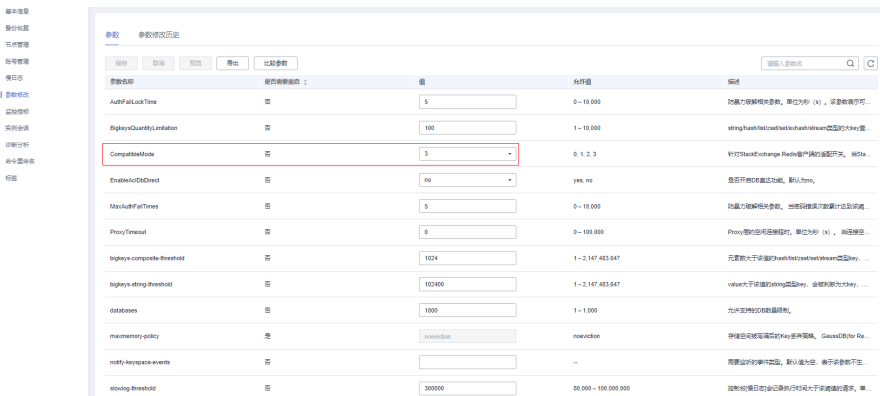
报错可能原因

GeminiDB Redis集群版实例参数“CompatibleMode”值不为3，或者主备版实例参数“CompatibleMode”值不为2，不支持Sentinel命令。

解决方法

- 步骤1** 登录[云数据库GeminiDB控制台](#)。
- 步骤2** 在“实例管理”页面，选择指定的实例，单击实例名称，进入“基本信息”页面。
- 步骤3** 在左侧导航树，单击“参数修改”。
- 步骤4** 修改参数“CompatibleMode”值，单击“保存”。
- 如果是集群版实例，修改参数“CompatibleMode”值为3。
 - 如果是主备版实例，修改参数“CompatibleMode”值为2。

图 8-1 设置参数



----结束

8.4.12 对于阻塞命令，GeminiDB Redis 接口（主备实例）与开源 Redis 的返回值为什么可能有差异

开源Redis没有规定“当有客户端执行了阻塞命令等待key写入的情况，并发对这些key执行写命令”的返回值。

GeminiDB Redis主备实例的接口返回值可能和开源Redis不一样，但两者均满足开源文档描述行为。

8.4.13 GeminiDB Redis 存储扩容需要多久，对业务有影响吗？

GeminiDB Redis存储扩容仅需秒级完成，扩容期间业务零感知。

磁盘扩容或者缩容具体操作请参考[容量变更](#)。

自动扩容具体操作请参考[自动扩容实例磁盘](#)。

8.4.14 GeminiDB Redis 多个节点同时扩容需要多长时间，对业务影响如何？

GeminiDB Redis支持多个节点同时扩容，原理是并行添加多个节点，通常可在5分钟内完成。

说明

由于采用的是共享存储，节点添加成功后，不涉及数据搬迁，但是会重新均衡slot，对业务影响是部分访问产生秒级慢时延或者秒级抖动，需要业务有重试机制。

8.4.15 GeminiDB Redis 规格变更包含的在线变更和离线变更有什么区别，通常需要多长时间，对业务有哪些影响？

- 在线变更：变更过程中，节点依次滚动变更，变更时长跟节点数正相关，每个节点约需5~10分钟。除此之外，主备和集群实例都包含3个内部管理节点，会同时进行变更。例如一个GeminiDB Redis实例有3个worker节点，包含3个内部管理节点，共计6个节点，在线变更时长约在30-60分钟。单个节点变更过程中，连接会断开，节点负责的slot分区会关闭，然后分配至正常节点进行接管以提供业务访问，对业务影响为数秒抖动，具体表现为连接断开、访问超时、访问的对应数据分区暂时不可见，需要业务有重连机制。建议在业务低峰期间变更，实例CPU和内存利用率保持低位，避免单节点变更时，其他节点负载过高，进程拉起失败等异常情况。
- 离线变更：离线变更时，全部节点将并行变更，变更过程中业务不可用，会导致业务中断约10-20分钟。通常适用于停服期、无业务访问等场景。请谨慎操作！

对于您的线上生产业务，建议使用在线变更。详细操作内容请参[变更实例的CPU和内存规格](#)。

8.4.16 GeminiDB Redis 版本补丁升级包含的在线升级和离线升级有什么区别，通常需要多长时间，对业务有哪些影响？

- 在线升级：和在线变更原理类似，实例节点依次滚动升级，执行时长和节点数正相关，每个节点约需2~5分钟。主备和集群实例，都包含3个内部管理节点，会同时进行版本升级。例如一个GeminiDB实例有3个worker节点，包含3个内部管理节点，共计6个节点，在线升级时长约在12-30分钟。单个节点升级过程中，对业务影响是秒级抖动，需要业务有重连机制。建议在业务低峰期间升级，实例CPU和内存利用率保持低位，避免单节点变更时，其他节点负载过高，进程拉起失败等异常情况。
- 离线升级：全部节点将并行升级，升级过程中业务不可用，会导致业务中断约10-20分钟。通常可用于停服期间，无业务访问等场景。请谨慎操作！

GeminiDB Redis版本补丁升级详情操作请参考[升级内核小版本](#)。

8.4.17 GeminiDB Redis 备份文件是否可以下载到本地，是否支持线下恢复数据

GeminiDB Redis的备份文件并非开源Redis的RDB文件，用户无法使用，因此，不支持下载到本地。

当实例数据受损时，可将备份数据备份恢复至新实例。

数据备份和数据恢复可参考[备份概述](#)和[恢复备份到新实例](#)。

8.4.18 GeminiDB Redis 数据备份工作机制是怎样的，对业务有哪些影响？

GeminiDB Redis执行数据备份时，仅需对存储层进行秒级快照，不影响计算节点，因此对业务无感；

备份上传时，会消耗少量的CPU和带宽资源，可能产生轻微的抖动。

GeminiDB Redis实例支持自动备份和手动备份两种方案，详细请参考[备份概述](#)。

8.4.19 购买 GeminiDB Redis 1U*2 节点特惠型实例后，业务访问量比较少，但 CPU 占用率比较高，是什么原因？

GeminiDB Redis会进行指标采集、监控上报等任务，1U*2节点特惠型实例由于规格较小，因此CPU使用占用比较明显。

建议1U*2节点特惠型规格用于测试环境，生产环境建议使用1U标准型、2U及以上规格。

实例规格详细具体请参见[实例规格说明](#)。

8.4.20 GeminiDB Redis 监控面板上 key 数量下降又恢复至正常数量是什么原因？

GeminiDB Redis的key数量是由后台异步扫描统计的，可以保证最终一致性。

当实例进程重启（由于节点重启、实例故障、规格变更、版本升级等导致）时，节点会重新开始统计key数量，此时显示的key数量会短暂下降，随着后台逐步统计会达到一个精确值。

8.4.21 GeminiDB Redis 节点 CPU 偶发冲高，可能是哪些原因

导致GeminiDB Redis节点CPU冲高的原因有很多，比如业务侧原因：业务流量突增、执行大Key操作、网络抖动，以及服务端后台任务如数据备份、垃圾回收等。

如果仅为偶发现象，不用过于关注。

若非QPS增高等业务侧原因造成，您可以在管理控制台右上角，选择“[工单 > 新建工单](#)”，联系客服进行咨询。

8.4.22 GeminiDB Redis 如何从 5.0 版本升级到 6.2 版本

GeminiDB Redis接口兼容社区Redis 7.0、Redis 6.2(包含6.2.x)、Redis 5.0及以下版本。

如果需要对已有实例进行版本升级，您可以在管理控制台右上角，选择“[工单 > 新建工单](#)”，联系客服开通升级白名单。

升级操作详情请参考 [升级内核小版本](#)。

8.4.23 GeminiDB Redis 什么时候进入只读

为保护GeminiDB Redis实例的正常运行，在存储空间即将被写满的时候，数据库会被设置成只读状态，该状态下只支持数据的读取，不支持写入和更新，可以通过扩容存储容量来解除该状态，重新获得写入和更新的能力。

表 8-1 实例设置只读状态说明

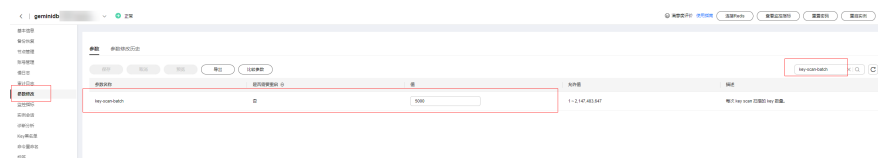
存储容量	说明
小于600GB	- 存储容量使用率达到97%时，实例状态被设置为只读。 - 存储容量使用率下降到85%，实例自动解除只读状态。
大于或等于600GB	- 存储容量剩余空间小于18GB，实例状态被设置为只读。 - 存储容量剩余空间大于或等于90GB，实例自动解除只读状态。

8.4.24 key 数量统计是什么机制？迁移完成后，GeminiDB 监控中 key 数量不准，还在持续增长，是什么原因？

- key数量统计机制：key数量统计并不是实时的，是异步扫描+多个节点累加得到的，如果key的增长速度大于扫描速度，如数据迁移、短时间数据大量灌入，这个值统计会滞后。key扫描速度由参数“key-scan-batch”决定，速率是恒定的，所以key数量增长的斜率是个稳定值。
- 如何加快key扫描统计：参考[修改当前实例的参数](#)在实例基本信息页，单击左侧导航栏的“参数修改”，搜索“key-scan-batch”并调大。key-scan-batch会消耗

CPU，调整的值越大消耗的CPU越多。迁移场景对比key数量可临时调大加快统计，迁移完成后建议恢复默认值避免消耗过多CPU，跟在线业务形成资源争抢。

图 8-2 key-scan-batch 参数修改



8.5 数据库连接

8.5.1 如何接入 GeminiDB Redis

目前GeminiDB Redis提供使用DAS连接、内网、公网、域名、负载均衡地址、程序代码等方式接入GeminiDB Redis，具体方法请参见[连接GeminiDB Redis实例方式介绍](#)。

- GeminiDB Redis支持通过[管理控制台的Web客户端](#)连接Redis实例。
- 支持通过[内网IP](#)、[内网域名](#)或[负载均衡地址](#)连接GeminiDB Redis实例。
- 支持通过[公网域名](#)、或[弹性公网IP](#)连接GeminiDB Redis实例。
- 通过不同代码的方式访问GeminiDB Redis实例的方法，详情请参见[通过程序代码连接实例示例](#)。

8.5.2 如何使用 GeminiDB Redis 提供的多个节点 IP 地址

GeminiDB Redis提供多个IP地址供客户连接，连接任何一个IP地址都可以访问整个集群，其目的是为了提供负载均衡和容灾能力。

您可以通过以下三种方法使用多个IP。

1. 业务侧实现连接池，实现负载均衡和故障检测处理。
2. 您可以在管理控制台右上角，选择“[工单 > 新建工单](#)”联系客服，为您配置ELB（Elastic Load Balance，弹性负载均衡）服务，提供唯一IP供客户使用。
3. 对多个proxy IP配置DNS域名，详情请参见[通过负载均衡地址连接实例（推荐）](#)章节中“通过内网域名连接实例”的方法。

8.5.3 GeminiDB Redis 提供的 ELB 的实现方式是怎样的

GeminiDB Redis采用的是独享型负载均衡器，采用弹性规格，带宽最高支持10Gbps。详细原理参考[独享型负载均衡器](#)。

8.5.4 如何创建和连接弹性云服务器

1. 创建弹性云服务器，请参见《[弹性云服务器用户指南](#)》。
 - 该弹性云服务器用于连接GeminiDB Redis的实例，需要与目标实例处于同一虚拟私有云和子网内。
 - 正确配置目标实例安全组，使得弹性云服务器处于目标实例所属安全组允许访问的范围内。

2. 连接弹性云服务器，请参见《弹性云服务器快速入门》中“登录弹性云服务器”的内容。

8.5.5 GeminiDB Redis 实例购买成功后是否支持更换 VPC

GeminiDB Redis实例创建完成后暂不支持直接通过控制台更换VPC。

您可以通过已有的全量备份恢复到新实例的方法切换到目标VPC。

具体操作请参考[恢复备份到新实例](#)。

8.5.6 绑定了弹性公网 IP 但是连接不上数据库

可能原因

安全组未放开对应端口限制，导致绑定弹性IP后连接不上。

处理步骤

步骤1 单击实例名称，进入基本信息页面。

步骤2 在“网络信息”页面，单击内网安全组。

图 8-3 网络信息



步骤3 选择“入方向规则”，单击“添加规则”，添加入方向规则具体配置信息如下图所示：

图 8-4 入方向规则



也可参考[配置安全组规则](#)文档进行配置。

----结束

8.5.7 内网如何访问 GeminiDB Redis

内网可以通过负载均衡(ELB)和直连节点访问实例，有以下区别：

- 通过负载均衡地址连接实例（推荐）：ELB底层是一个高可用集群，提供一个负载均衡地址给用户访问。会定期与后端节点探活，可有效避免后端节点单点故障。
- 通过节点连接实例：GeminiDB Redis节点包含代理进程，用户连接任意一个节点可访问整个集群。该访问方式可能存在单点故障，推用于测试场景。

通过内网连接GeminiDB Redis实例请参考[通过内网连接GeminiDB Redis](#)。

8.5.8 GeminiDB Redis 自带的负载均衡地址是否能绑定公网 IP？如何通过公网连接 GeminiDB Redis 实例？

GeminiDB Redis实例自带的负载均衡地址不支持绑定公网IP，可以通过以下2种方式进行公网访问：

- 通过节点绑定弹性公网IP：推荐在测试场景使用，生产业务如遇单点故障，会导致业务受损。
- 通过购买弹性负载均衡，绑定弹性公网IP的方式连接GeminiDB Redis实例，详细内容请参考[通过弹性负载均衡绑定弹性公网IP连接实例](#)。

8.5.9 设置了安全组，还需要设置负载均衡内网访问控制吗？

用户可以通过节点和负载均衡两种方式访问GeminiDB Redis实例，因此需要同时设置安全组和负载均衡内网访问控制，以保证实例的安全性：

- 安全组仅对节点生效。安全组是一个逻辑上的分组，为同一个虚拟私有云内具有相同安全保护需求，并相互信任的弹性云服务器和GeminiDB Redis实例提供访问策略。具体操作步骤请参考[给GeminiDB Redis实例节点配置安全组规则](#)。
- 安全组无法对负载均衡生效。在默认未配置的状态下，所有与所在VPC互通的IP地址均可通过负载均衡地址访问实例。因此，同时需要合理配置，具体配置操作请参考[给GeminiDB Redis实例设置负载均衡内网访问控制](#)。

8.5.10 如何处理客户端连接池报错 “Could not get a resource from the pool”

报错场景

客户端业务发现大量GeminiDB请求卡住，异常堆栈中出现如下报错信息：

- Jedis客户端：
`redis.clients.jedis.exceptions.JedisConnectionException: Could not get a resource from the pool`
- Lettuce客户端：
`redis.connection.lettuce.LettucePoolingConnectionProvider.getConnection`
- Go-redis客户端：
`redis: connection pool timeout`
You can get that error when there are no free connections in the pool for Options.PoolTimeout duration. If you are using redis.PubSub or redis.Conn, make sure to properly release PuSub/Conn resources by calling Close method when they are not needed any more.
You can also get that error when Redis processes commands too slowly and all connections in the pool are blocked for more than PoolTimeout duration.

但是通过查看实例的[监控指标](#)信息，发现数据库QPS、时延、连接数等信息一切正常，无慢请求信息。

报错可能原因

上述问题一般为客户端连接池配置问题。对于使用连接池的应用来说，连接池是有最大数量限制的。如果业务所使用QPS超过连接池的承受限制，或者没有及时释放导致连接泄露耗尽，线程就无法获取新的连接而影响业务。

解决方法

通过客户端排查过去2小时内QPS、流量指标是否异常激增，以及使用的Redis客户端（Jedis、Lettuce等）配置的连接池参数是否满足业务需求。

说明

Redis客户端使用配置请参考[开发运维规范](#)和[客户端连接池参数配置](#)。

8.5.11 常见客户端报错及解决方法

问题现象 1

- 问题现象：客户端打印网络超时10秒，连接失败。
CommonResponseAspect exception!Redis command timed out; nested exception is io.lettuce.core.RedisCommandTimeoutException: Command timed out after 10 second(s)
- 报错客户端：Lettuce。
- 可能原因：客户端网络带宽打满。
- 解决方法：排查业务侧资源，解决客户端侧资源瓶颈。

问题现象 2

- 问题现象：客户端偶尔打印连接不可用。
[redisClient=[addr=XXXX], channel=[id: 0x0a0d20bc, L:0.0.0.0/0.0.0.0:53192]] is not active!
- 报错客户端：Redisson。
- 可能原因：客户端重连机制不完善导致，当服务侧HA倒换后可能遇到该现象。
- 解决方法：重启客户端。

问题现象 3

- 问题现象：应用报错“Could not get a resource from the pool”，同时检查业务程序所在ECS有大量CLOSE_WAIT的TCP连接。
- 报错客户端：Jedis、Lettuce。
- 可能原因：通常由于客户端程序配置的连接池过小，在业务并发激增时与Redis建立的连接不够用。
- 解决方法：排查业务代码，设置足够的连接池配置。

问题现象 4

- 问题现象：客户程序连接池超时，报错信息如下：
“redis: connection pool timeout
You can get that error when there are no free connections in the pool for Options.PoolTimeout duration. If you are using redis.PubSub or redis.Conn, make sure to properly release PuSub/Conn resources by calling Close method when they are not needed any more.
You can also get that error when Redis processes commands too slowly and all connections in the pool are blocked for more than PoolTimeout duration.”
- 报错客户端：Go-redis。
- 可能原因：通常由于客户端程序配置的连接池过小，在业务并发激增时与Redis建立的连接不够用。
- 解决方法：排查业务代码，设置足够的连接池配置。

8.6 备份与恢复

8.6.1 GeminiDB Redis 实例能够保存多长时间的备份

GeminiDB Redis实例的自动备份有效期根据用户设置的备份天数而定。手动备份没有时间限制，用户可根据需要进行删除。

更多备份信息请参见[管理自动备份](#)和[管理手动备份](#)。

8.7 区域和可用区

8.7.1 不同的可用区是否影响内网互通

可用区是同一服务区内，电力和网络互相独立的地理区域，一般是一个独立的物理机房，这样可以保证可用区的独立性。

一个区域内有多个可用区，一个可用区发生故障后不会影响同一区域内的其它可用区。

默认情况下，同一个VPC下的不同可用区之间内网互通。

更多可用区信息请参见[区域和可用区](#)。

8.7.2 GeminiDB Redis 购买成功后是否支持更换区域

不支持。资源创建成功后不能更换地域。

说明

请就近选择靠近您业务的区域，可减少网络时延，提高访问速度。不同区域内的产品内网不互通，且购买后不能更换，请谨慎选择。

8.8 数据迁移

8.8.1 如何处理报错 ERR the worker queue is full, and the request cannot be executed?

报错含义

服务进程shard队列已满，无法接受新的请求。

可能发生原因

1. 业务流量激增，节点负载过高，导致shard队列被填满。
2. 由于业务中存在大key、热key等问题，使得某个shard队列中的命令排队等候，最终队列被完全占满。

整改建议

1. 若队列满是由于大key或热key引起，建议业务侧进行优化整改，可参考开发与运维规范进行调整。
2. 若实例整体负载较高，建议升级到更高规格，以提升命令处理效率。

说明

如果上述方法仍无法解决问题，您可以在管理控制台右上角，选择“[工单 > 新建工单](#)”，联系客服进行处理。

8.8.2 报错 ERR the request queue of io thread is full, and the request cannot be executed

迁移流量过大导致内部队列满，您可以在管理控制台右上角，选择“[工单 > 新建工单](#)”，联系客服进行处理。

8.8.3 报错 read error, please check source redis log or network

源端发送缓冲区太小，需修改源端redis参数配置，该参数client-output-buffer-limit可热生效。

8.8.4 报错 slaveping_thread.cc-ThreadMain-90: error: Ping master error

执行pika-port时指定的IP填写了 127.0.0.1, 需要配置为其它IP，比如eth0的IP。

8.8.5 同步状态正向迁移速度太慢

修改redis-shake参数source.rdb.parallel，可调整迁移并发度，默认为0，按db和分片数确定。

8.8.6 同步状态正向迁移速度太快，报错：ERR server reply timeout, some responses may lose, but requests have been executed

修改参数parallel，可调整全量过程传输RDB的并发度，默认为32。

8.8.7 4.0、5.0 以及 6.2 版本的自建 Redis 能迁移至 GeminiDB Redis 吗？

GeminiDB Redis同时兼容6.2(包含6.2.x)及其以下版本，如5.0、4.0、2.8。

8.8.8 自建 Redis 主备、集群实例如何迁移到 GeminiDB Redis？

在线迁移可以使用DRS服务进行迁移：

- 单机Redis迁移至GeminiDB Redis主备操作请参考[将Redis迁移到GeminiDB Redis](#)。
- 集群Redis迁移至GeminiDB Redis集群操作请参考[将Redis集群迁移到GeminiDB Redis](#)。

RDB文件迁移到GeminiDB Redis操作请参考[RDB文件恢复到GeminiDB Redis](#)。

8.8.9 为什么阿里云 Redis、腾讯云 Redis 等云服务不能使用 DRS 进行数据迁移？

通常有以下因素：

- 不支持PSync迁出协议：存在一些自研的Redis-like数据库没有兼容PSync协议。
- 架构限制：较多云厂商支持代理模式，即在用户与Redis服务之间增加Proxy组件。因为Proxy组件的存在，所以PSync协议无法支持。
- 安全限制：在原生Redis中PSync协议基本会触发fork，会导致内存膨胀与用户请求延迟增加，甚至可能会发生“out of memory”。
- 商业策略：较多用户使用RedisShake是为了下云或者换云，屏蔽了PSync协议。

通常可以使用对应云服务的数据迁移服务，可以完成迁移，迁移详情请参考[Redis数据迁移方案概览](#)。

8.8.10 自建主备 Redis，迁移到 GeminiDB Redis 集群，需要考虑哪些因素？

GeminiDB Redis集群为Proxy集群，支持以单机、主备的方式直接访问，无需修改。对于多key操作，Redis主备和集群用法不一样，若涉及多key命令，则需要业务修改。

自建Redis单机、主备架构所有数据都在同一个节点上。因此，对于多key操作，比如lua、rpoplpush、集合（sdiff、sunion）等操作，可以保证其操作的原子性。在自建Redis集群中，对key进行hash取模，决定key被哪一个分片（节点）所处理。因此，对跨分片的多个key进行操作难以保证原子性。为了保证集群中多key操作的原子性，Redis cluster采用hashtag来保证需要操作的多个key落在同一个节点。

Redis hashtag的使用方法为，为需要操作的多key加增相同的字符串。例如{aaa}list1、{aaa}list2，Redis服务在处理以上key时，识别到“{”时，仅根据“{”的字符串“aaa”计算hash值，因此可以确保其落在相同的节点上。

GeminiDB Redis集群中需要增加hashtag的命令请参考[GeminiDB Redis集群版实例中，哪些命令需要使用hashtag](#)。

8.8.11 迁移完成后数据量变少了，100GB 的数据迁移到 GeminiDB Redis 只有 20-30GB，数据是不是没迁移完？

GeminiDB Redis具有4:1超高数据压缩，现网多个用户数据迁移完都符合此规律。用户可以进行数据一致性校验，判断是否迁移完成：比如key的数量、key抽样校验等。

8.8.12 GeminiDB Redis 是否支持 sync 和 psync 协议？能否放开这些接口？

1. 开源Redis通过sync和psync接口实现主从、节点之间的数据迁移和同步。GeminiDB Redis虽然兼容Redis协议，但采用了不同的机制来实现节点间的数据同步，并不依赖于sync或psync。因此，GeminiDB Redis不支持与开源Redis相同语义的sync和psync接口，也无法使用第三方工具如redis-shake进行在线数据迁移和同步。所以这些接口目前无法开放。

2. DRS（数据复制服务）是GeminiDB配套的数据流转服务，专门用于实现实例间的数据同步。低于5.0.6.59的版本，存在已知的稳定性问题，比如在生成备份的过程中可能会导致在线业务暂时受阻等。在使用该功能时，请您在管理控制台右上角，选择“[工单 > 新建工单](#)”，联系客服进行咨询。

8.9 内存加速

8.9.1 开启内存加速，MySQL 数据更新后，会把数据全部缓存到 GeminiDB Redis 吗？

不是，需要用户通过界面操作：指定MySQL表空间（db）、表名（table）、字段（field）到GeminiDB Redis的转换规则。配置好后，符合规则的数据会自动同步到GeminiDB Redis。

8.9.2 开启内存加速，GeminiDB Redis 数据会不断增长，需要扩容吗？如何进行缓存数据管理？

GeminiDB Redis默认每条数据设置30天过期时间，用户也可以灵活调整；如果数据不断增长，需要及时扩容GeminiDB Redis存储空间。

8.9.3 客户已有业务实现“db”+“缓存”，推荐使用内存加速吗？哪种场景可以让客户使用内存加速方案？

客户已有使用“db”+“缓存”的业务，建议继续使用原方案，可以把缓存数据迁移到GeminiDB Redis继续使用。新业务推荐使用内存加速，以简化开发。

8.9.4 RDS MySQL 到 GeminiDB Redis 数据延迟大概是多少，会受哪些因素影响

可以做到实时同步，具体时延建议实测。时延可能受到以下因素影响：

- RDS MySQL到GeminiDB Redis本身的物理距离，通常建议同Region。
- RDS MySQL到GeminiDB Redis的实例规格，通常建议使用同CPU规格。

8.9.5 开启内存加速后，是否会对源端 MySQL 有影响

内存加速的主要工作原理是订阅MySQL的binlog，对源端MySQL几乎无影响。

8.9.6 开启内存加速的 GeminiDB Redis 短时间要处理大量的 binlog，会不会占用较多的资源，导致在线业务受损

如果源端MySQL存在较大的DDL操作，会消耗较多的GeminiDB Redis资源，用户可通过观察内存加速的每秒操作数（dbcache_ops_per_sec）。建议用户配置基础资源告警，详情请参考[设置告警规则](#)。

8.10 资源冻结/释放/删除/退订

GeminiDB Redis 资源为什么被释放了？

客户在华为云购买产品后，如果没有及时地进行续费或充值，将进入宽限期。如宽限期满仍未续费或充值，将进入保留期。在保留期内资源将停止服务。保留期满仍未续费或充值，存储在云服务中的数据将被删除、云服务资源将被释放。请参见[资源停止服务或逾期释放说明](#)。

GeminiDB Redis 资源为什么被冻结了？

资源冻结的类型有多种，最常见类型为欠费冻结。

单击了解[资源冻结的类型、冻结后对续费、退订的影响](#)。

实例被冻结了，还可以备份数据吗？

不支持，如果是欠费冻结，需要您先续费解冻GeminiDB Redis实例后才能备份数据。

怎样将资源解冻？

欠费冻结：用户可通过续费或充值来解冻资源，恢复GeminiDB Redis正常使用。欠费冻结的GeminiDB Redis允许续费、释放或删除；已经到期的包年/包月GeminiDB Redis不能发起退订，未到期的包年/包月GeminiDB Redis可以退订。

冻结、解冻、释放资源时对业务的影响

- 资源冻结时：
 - 资源将被限制访问和使用，会导致您的业务中断。例如GeminiDB Redis被冻结时，会使得用户无法再连接至数据库。
 - 包年/包月资源被冻结后，将被限制进行变更操作。
 - 资源被冻结后，可以手动进行退订/删除。
- 资源解冻时：资源将被解除限制，用户可以连接至数据库。
- 资源释放时：资源将被释放，实例将被删除，删除前将依据用户策略决定是否执行[回收站备份](#)。

怎样续费？

包年/包月方式购买的GeminiDB Redis期后，请在管理控制台[续费管理](#)页面进行续费操作。详细操作请参考[续费管理](#)。

资源被释放了能否恢复？/退订错了可以找回吗？

实例被删除，如果有回收站备份，可以通过[回收站备份](#)来恢复实例，反之则无法找回数据。

退订资源前请一定要仔细确认资源信息。如果退订错了建议重新购买使用。

怎样删除 GeminiDB Redis 实例？

- 按需实例，请参见[删除按需实例](#)。
- 包年/包月实例，请参见[如何退订包年/包月实例](#)。