

ALIBABA CLOUD

# 阿里云

云数据库 Redis 版  
性能白皮书

文档版本：20201028

 阿里云

## 法律声明

阿里云提醒您在阅读或使用本文档之前仔细阅读、充分理解本法律声明各条款的内容。如果您阅读或使用本文档，您的阅读或使用行为将被视为对本声明全部内容的认可。

1. 您应当通过阿里云网站或阿里云提供的其他授权通道下载、获取本文档，且仅能用于自身的合法合规的业务活动。本文档的内容视为阿里云的保密信息，您应当严格遵守保密义务；未经阿里云事先书面同意，您不得向任何第三方披露本手册内容或提供给任何第三方使用。
2. 未经阿里云事先书面许可，任何单位、公司或个人不得擅自摘抄、翻译、复制本文档内容的部分或全部，不得以任何方式或途径进行传播和宣传。
3. 由于产品版本升级、调整或其他原因，本文档内容有可能变更。阿里云保留在没有任何通知或者提示下对本文档的内容进行修改的权利，并在阿里云授权通道中不时发布更新后的用户文档。您应当实时关注用户文档的版本变更并通过阿里云授权渠道下载、获取最新版的用户文档。
4. 本文档仅作为用户使用阿里云产品及服务的参考性指引，阿里云以产品及服务的“现状”、“有缺陷”和“当前功能”的状态提供本文档。阿里云在现有技术的基础上尽最大努力提供相应的介绍及操作指引，但阿里云在此明确声明对本文档内容的准确性、完整性、适用性、可靠性等不作任何明示或暗示的保证。任何单位、公司或个人因为下载、使用或信赖本文档而发生任何差错或经济损失的，阿里云不承担任何法律责任。在任何情况下，阿里云均不对任何间接性、后果性、惩戒性、偶然性、特殊性或刑罚性的损害，包括用户使用或信赖本文档而遭受的利润损失，承担责任（即使阿里云已被告知该等损失的可能性）。
5. 阿里云网站上所有内容，包括但不限于著作、产品、图片、档案、资讯、资料、网站架构、网站画面的安排、网页设计，均由阿里云和/或其关联公司依法拥有其知识产权，包括但不限于商标权、专利权、著作权、商业秘密等。非经阿里云和/或其关联公司书面同意，任何人不得擅自使用、修改、复制、公开传播、改变、散布、发行或公开发表阿里云网站、产品程序或内容。此外，未经阿里云事先书面同意，任何人不得为了任何营销、广告、促销或其他目的使用、公布或复制阿里云的名称（包括但不限于单独为或以组合形式包含“阿里云”、“Aliyun”、“万网”等阿里云和/或其关联公司品牌，上述品牌的附属标志及图案或任何类似公司名称、商号、商标、产品或服务名称、域名、图案标示、标志、标识或通过特定描述使第三方能够识别阿里云和/或其关联公司）。
6. 如若发现本文档存在任何错误，请与阿里云取得直接联系。

# 通用约定

格式	说明	样例
 危险	该类警示信息将导致系统重大变更甚至故障，或者导致人身伤害等结果。	 危险 重置操作将丢失用户配置数据。
 警告	该类警示信息可能会导致系统重大变更甚至故障，或者导致人身伤害等结果。	 警告 重启操作将导致业务中断，恢复业务时间约十分钟。
 注意	用于警示信息、补充说明等，是用户必须了解的内容。	 注意 权重设置为0，该服务器不会再接受新请求。
 说明	用于补充说明、最佳实践、窍门等，不是用户必须了解的内容。	 说明 您也可以通过按Ctrl+A选中全部文件。
>	多级菜单递进。	单击设置>网络>设置网络类型。
<b>粗体</b>	表示按键、菜单、页面名称等UI元素。	在结果确认页面，单击确定。
Courier字体	命令或代码。	执行 <code>cd /d C:/window</code> 命令，进入Windows系统文件夹。
斜体	表示参数、变量。	<code>bae log list --instanceid</code> <i>Instance_ID</i>
[ ] 或者 [a b]	表示可选项，至多选择一个。	<code>ipconfig [-all -t]</code>
{ } 或者 {a b}	表示必选项，至多选择一个。	<code>switch {active stand}</code>

# 目录

1.社区版（集群版-双副本）	05
1.1. 测试环境	05
1.2. 测试工具	05
1.3. 测试命令与指标	06
1.4. 测试结果	06
2.企业版（持久内存型）	08
2.1. 测试环境	08
2.2. 测试工具	08
2.3. 测试方法	08
2.4. 测试结果	09
3.企业版（容量存储型）	12
3.1. 测试环境	12
3.2. 测试工具	12
3.3. 测试方法	12
3.4. 测试结果	14

# 1. 社区版（集群版-双副本）

## 1.1. 测试环境

本章节介绍云数据库Redis集群版-双副本的性能测试环境。

### 产品架构

云数据库Redis集群版-双副本实例由配置服务器、代理服务器和分片服务器组成：

- 配置服务器（Config Server）是采用双副本高可用架构的集群管理工具，用于存储集群配置信息及分区策略；
- 代理服务器（Proxy Server）为单节点架构，集群版结构中会有多个proxy，系统自动对所有proxy进行负载均衡及故障转移；
- 分片服务器（Shard Server）同样采用双副本高可用架构，主节点故障之后，系统会自动进行主备切换保证服务高可用。

Redis集群版本实例提供一个访问域名供客户端访问，您可以通过该域名进行正常的Redis访问及数据操作，代理服务器、分片服务器和配置服务器均不支持单独的直接访问。

客户端通过负载均衡（SLB）连接代理服务器，通过代理服务器对分片服务器进行访问，其架构图如下。

□

### 地域与可用区

所有测试均在华东1（杭州）地域的可用区E完成。

### ECS配置

测试环境中的ECS配置根据作为测试对象的Redis版本不同而有所区别。下文以128G集群版和64G集群版的测试为例。

测试128G集群版时的ECS配置：

- 4 vCPU，8G内存的ECS 10台；
- 12 vCPU，48G内存的ECS 4台；
- 网络类型：VPC；
- 操作系统：Cent OS 6.0，64位。

测试64G集群版时的ECS配置：

- 4 vCPU，8G内存的ECS 10台；
- 网络类型：VPC；
- 操作系统：Cent OS 6.0，64位。

### Redis实例配置

- Redis实例的规格根据测试对象决定。
- 本白皮书中使用Redis 2.8版本的实例进行基准测试，4.0版本测试结果与其相似。

## 1.2. 测试工具

本节介绍测试Redis集群版使用的工具。

## memtier\_benchmark简介

memtier\_benchmark是Redis Labs推出的命令行工具，可用于在键值存储数据库中生成数据负载并进行压力测试。

### 下载与安装

详细步骤请参见[memtier-benchmark使用方法](#)。

## 1.3. 测试命令与指标

本节介绍测试云数据库Redis集群版使用的命令和结果指标。

### 测试命令

```
./memtier_benchmark -s r-*****.redis.rds.aliyuncs.com -p 6379 -a <password> -c 20 -d 32 --threads=10 --ratio=1:1 --test-time=1800 --select-db=10
```

 说明 命令说明请参见[memtier\\_benchmark常用选项说明](#)。

### 测试指标

QPS

Queries Per Second，即数据库每秒处理的请求数。

## 1.4. 测试结果

本节介绍云数据库Redis版的性能测试结果。

### 集群版-双副本规格

规格	节点数 (个)	连接数上 限 (个)	内网带宽 上限 (MB)	CPU 处理 能力	QPS参考 值	说明
4 GB集群版	2	20000	96	2核	160000	高性能集群实例
8 GB集群版	2	20000	96	2核	160000	高性能集群实例
16 GB集群版	8	80000	384	8核	640000	高性能集群实例
32 GB集群版	8	80000	384	8核	640000	高性能集群实例
64 GB集群版	8	80000	384	8核	640000	高性能集群实例
128 GB集群版	16	160000	768	16核	1280000	高性能集群实例
256 GB集群版	16	160000	768	16核	1280000	高性能集群实例
512 GB集群版	32	320000	1536	32核	2560000	高性能集群实例

规格	节点数 (个)	连接数上 限 (个)	内网带宽 上限 (MB)	CPU 处理 能力	QPS参考 值	说明
1 TB集群版	64	640000	2150	64核	5120000	高性能集群实例
2 TB 集群版	128	1280000	4300	128核	10240000	高性能集群实例

#### ② 说明

- 集群版-双副本规格中的节点数为主节点数量。
- 其它版本的规格信息请参见[云数据库Redis版规格性能](#)。

## 测试结果

下图展示了云数据库Redis集群版-双副本部分规格的基准测试结果。

□

## 2. 企业版（持久内存型）

### 2.1. 测试环境

本文介绍Redis企业版（持久内存型）性能测试所使用的具体环境。

#### 产品概述

Redis企业版（持久内存型）基于Intel 傲腾™数据中心级持久内存，为您提供大容量、兼容Redis的内存数据库产品。单实例成本对比Redis社区版最高可降低30%，且数据持久化不依赖传统磁盘，保证每个操作持久化的同时提供近乎Redis社区版的吞吐和延时，极大提升业务数据可靠性。更多详情请参见[持久内存型](#)。

#### 测试环境

测试环境信息	说明
地域和可用区	所有测试均在华北3（张家口）地域的可用区A中完成。
Redis实例架构	标准版（双副本）架构，详情请参见 <a href="#">标准版-双副本</a> 。
部署压测工具的机器	<a href="#">云服务器ECS</a> 实例，规格为ecs.g6e.8xlarge，详情请参见 <a href="#">实例规格族</a> 。
持久存储型实例规格	由于测试结果受规格影响较小，本次测试以tair.scm.standard.16m.64d规格为例，规格详情请参见 <a href="#">持久内存型-标准版</a> 。

### 2.2. 测试工具

Redis企业版（持久内存型）使用YCSB压测工具进行性能测试，本文为您介绍测试工具及其使用方法。

采用开源社区的YCSB压测工具进行压测。YCSB是一款Java编写的支持多种数据库的性能测试工具，具体安装和使用方法请参见[YCSB](#)。

由于开源版本仅支持测试Hash结构的数据，下载YCSB工具后，需要将 `YCSB/redis/src/main/java/site/yccb/db/RedisClient.java` 文件替换为 `RedisClient.java` 文件，用于测试String结构的数据。

### 2.3. 测试方法

本文介绍Redis企业版（持久内存型）性能测试的具体方法。

#### 工作负载

测试的总数据量8 GB，数据分布方法为zipfian，具体测试场景如下：

- Load：100%的string set操作（写操作）。
- Workload C：100%的string get操作（读操作）。
- Workload A：50%的string set操作（更新操作）与50%的string get操作。

关于Workload的详细介绍，请参见[Core Workloads](#)。

#### 测试命令

```

#加载数据
workload=a
./bin/ycsb load redis -s -P workloads/workload${workload} -p "redis.host=${server_ip}" -p "redis.port=${port}"
-p "recordcount=${recordcount}" -p "operationcount=${operationcount}" -p "redis.timeout=30000" -p "redis.
command_group=${command_group}" -p "fieldcount=${fieldcount}" -p "fieldlength=${fieldlength}" -threads $
{threads} -p "redis.password=***:*****"

#运行Workload C
workload=c
./bin/ycsb run redis -s -P workloads/workload${workload} -p "redis.host=${server_ip}" -p "redis.port=${port}"
-p "recordcount=${recordcount}" -p "operationcount=${operationcount}" -p "redis.timeout=30000" -p "redis.c
ommand_group=${command_group}" -p "fieldcount=${fieldcount}" -p "fieldlength=${fieldlength}" -threads ${t
hreads} -p "redis.password=***:*****"

#运行Workload A
workload=a
./bin/ycsb run redis -s -P workloads/workload${workload} -p "redis.host=${server_ip}" -p "redis.port=${port}"
-p "recordcount=${recordcount}" -p "operationcount=${operationcount}" -p "redis.timeout=30000" -p "redis.c
ommand_group=${command_group}" -p "fieldcount=${fieldcount}" -p "fieldlength=${fieldlength}" -threads ${t
hreads} -p "redis.password=***:*****"

```

## 参数说明

参数	说明
server_ip	Redis实例的IP地址。
port	Redis实例的服务端口。
recordcount	数据装载阶段准备的数据量。
operationcount	执行操作的数据量。
command_group	测试类型，配置为String。
fieldcount	字段个数，配置为1。
fieldlength	值长度，根据测试需求配置。
threads	YCSB线程数，根据实例规格配置。

## 2.4. 测试结果

本文介绍Redis企业版（持久内存型）在不同测试场景下的测试结果。

## 工作负载

测试的总数据量8 GB，数据分布方法为zipfian，具体测试场景如下：

- Load: 100%的string set操作（写操作）。
- Workload C: 100%的string get操作（读操作）。
- Workload A: 50%的string set操作（更新操作）与50%的string get操作。

关于Workload的详细介绍，请参见[Core Workloads](#)。

## 测试指标

测试指标	说明
QPS	每秒处理的读写操作数，单位为次/秒。
INSERT Average Latency	写操作平均延迟，单位为微秒（us）。
INSERT 99th Percentile Latency	处理速度最快的99%写操作中，最长的延迟时间，单位为微秒。例如该指标的值为500微秒，表示99%的请求可以在500微秒内得到处理。
READ AverageLatency	读操作平均延迟，单位为微秒。
READ 99thPercentileLatency	处理速度最快的99%读操作中，最长的延迟时间，单位为微秒。
UPDATE AverageLatency	更新操作平均延迟，单位为微秒。
UPDATE 99thPercentileLatency	处理速度最快的99%更新操作中，最长延迟时间，单位为微秒。

## Load场景测试结果

Value长度	QPS（次/秒）	INSERT Average Latency（微秒）	INSERT 99th Percentile Latency（微秒）
128字节	134478	473	687
256字节	126139	504	828
1024字节	99775	638	1051
2048字节	77130	826	1157
4096字节	60646	1050	1534

## Workload C场景测试结果

Value长度	QPS（次/秒）	READ AverageLatency（微秒）	READ 99thPercentileLatency（微秒）
128字节	170699	362	546
256字节	163829	380	565

Value长度	QPS (次/秒)	READ AverageLatency (微秒)	READ 99thPercentileLatency (微秒)
1024字节	161491	386	569
2048字节	130189	487	729
4096字节	115433	548	808

## Workload A场景测试结果

Value长度	QPS (次/秒)	READ AverageLatency (微秒)	READ 99thPercentile Latency (微秒)	UPDATE AverageLatency (微秒)	UPDATE 99thPercentile Latency (微秒)
128字节	141120	451	616	450	618
256字节	137551	463	617	461	618
1024字节	124165	516	724	508	725
2048字节	92652	695	881	678	871
4096字节	78994	819	1042	791	1024

## 3. 企业版（容量存储型）

### 3.1. 测试环境

本文介绍Redis企业版（容量存储型）性能测试所使用的具体环境。

#### 产品概述

Redis企业版（容量存储型）基于云盘ESSD研发，兼容Redis核心数据结构与接口，可提供大容量、低成本、强持久化的数据库服务。容量存储型在降低成本和提升数据可靠性的同时，也解决了原生Redis因fork而预留部分内存的问题。适用于兼容Redis、需要大容量且较高访问性能的温冷数据存储场景。更多详情，请参见[容量存储型](#)。

#### 测试环境

测试环境信息	说明
地域和可用区	所有测试均在华北1（杭州）地域的可用区I中完成。
Redis实例架构	标准版（双副本）架构，详情请参见 <a href="#">标准版-双副本</a> 。
部署压测工具的机器	<a href="#">云服务器ECS</a> 实例，规格为cs.g6e.13xlarge（U内存），规格详情请参见 <a href="#">实例规格族</a> 。
容量存储型实例规格	<ul style="list-style-type: none"><li>tair.essd.standard.xlarge</li><li>tair.essd.standard.2xlarge</li><li>tair.essd.standard.4xlarge</li><li>tair.essd.standard.8xlarge</li><li>tair.essd.standard.13xlarge</li></ul> 规格详情请参见 <a href="#">容量存储型-标准版</a> 。

### 3.2. 测试工具

Redis企业版（容量存储型）使用YCSB压测工具进行性能测试，本文为您介绍测试工具及其使用方法。

采用开源社区的YCSB压测工具进行压测。YCSB是一款Java编写的支持多种数据库的性能测试工具，具体安装和使用方法请参见[YCSB](#)。

本测试中，对YCSB的相关内容做了一定的修改，使其支持导入long类型的recordcount参数、支持测试Redis的String相关命令，修改后的完整源代码请参见[YCSB源码](#)。

### 3.3. 测试方法

本文介绍Redis企业版（容量存储型）性能测试的具体方法。

#### 工作负载

- Load：100%的string set操作（写操作）。
- Uniform-Read：采用Workload A，100%均匀随机string get操作（读操作），主要测试严苛条件下的读

性能。

- Zipfian-Read: 采用Workload C, 数据分布方法为zipfian, 测试大部分读请求访问小部分数据的性能, 符合大部分的读场景。
- Uniform-50%Read-50%Update: 采用Workload A, 50%的string set操作（更新操作）与50%的string get操作, 主要测试随机更新的性能。

关于Workload的详细介绍, 请参见[Core Workloads](#)。

## 测试场景

测试主要针对下述两种场景进行:

- 内存大于数据场景: 绝大部分数据可以在内存中访问到, 此场景下内存与数据的比例约为7:1。
- 数据大于内存场景: 只有部分数据缓存在内存, 绝大多数访问需要读写硬盘, 此场景下内存与数据的比例约为1:4。

## 测试命令

下述脚本以数据大于内存场景为例。

```
#!/bin/bash

ip=192.168.0.23
port=3100
timeout=30000
command_group=string
recordcount=64000000
run_operationcount=20000000
fieldcount=1
fieldlength=100
threads=32
load_sleep_time=600
run_sleep_time=60

echo "##### $command_group #####"
#####
#Load
./bin/ycsb load redis -s -P workloads/workloada -p "redis.host=${ip}" -p "redis.port=${port}" -p "recordcount=${recordcount}" -p "operationcount=${recordcount}" -p "redis.timeout=${timeout}" -p "redis.command_group=${command_group}" -p "fieldcount=${fieldcount}" -p "fieldlength=${fieldlength}" -threads ${threads}
sleep ${load_sleep_time}

#Uniform-Read
./bin/ycsb run redis -s -P workloads/workloadc -p "redis.host=${ip}" -p "redis.port=${port}" -p "recordcount=${recordcount}" -p "operationcount=${run_operationcount}" -p "redis.timeout=${timeout}" -p "redis.command_group=${command_group}" -p "fieldcount=${fieldcount}" -p "fieldlength=${fieldlength}" -p "requestdistributi
```

```

on=uniform" -threads ${threads}
sleep ${run_sleep_time}

#Zipfian-Read
./bin/ycsb run redis -s -P workloads/workloadc -p "redis.host=${ip}" -p "redis.port=${port}" -p "recordcount=${recordcount}" -p "operationcount=${run_operationcount}" -p "redis.timeout=${timeout}" -p "redis.command_group=${command_group}" -p "fieldcount=${fieldcount}" -p "fieldlength=${fieldlength}" -p "requestdistribution=zipfian" -threads ${threads}
sleep ${run_sleep_time}

#Uniform-50%Read-50%Update
./bin/ycsb run redis -s -P workloads/workloada -p "redis.host=${ip}" -p "redis.port=${port}" -p "recordcount=${recordcount}" -p "operationcount=${run_operationcount}" -p "redis.timeout=${timeout}" -p "redis.command_group=${command_group}" -p "fieldcount=${fieldcount}" -p "fieldlength=${fieldlength}" -p "requestdistribution=uniform" -threads ${threads}

```

## 参数说明

参数	说明
ip	Redis实例的IP地址。
port	Redis实例的服务端口。
timeout	测试命令的超时时间。
command_group	测试类型，配置为String。
recordcount	数据装载阶段准备的数据量。
run_operationcount	Run阶段操作的数据量。本测试中： <ul style="list-style-type: none"> <li>内存大于数据场景下，配置和recordcount参数相同的值。</li> <li>数据大于内存场景下，配置的值recordcount参数值除以32。</li> </ul>
fieldcount	字段个数，配置为1。
fieldlength	值长度，配置为100。
threads	YCSB线程数，根据实例规格配置。

## 3.4. 测试结果

本文介绍Redis企业版（容量存储型）在不同测试场景下的测试结果。

### 测试指标

测试指标	说明
QPS	每秒处理的读写操作数，单位为次/秒。
Average Latency	读或写操作的平均延迟，单位为微秒（us）。
99th Percentile Latency	处理速度最快的99%的操作中，最长的延迟时间，单位为微秒。例如该指标的值为500微秒，表示99%的请求可以在500微秒内得到处理。

## 内存大于数据场景下测试结果

--

实例规格	YCSB配置	工作负载	QPS（次/秒）	Average Latency（微秒）	99th Percentile Latency（微秒）
tair.essd.standard.xlarge	recordcount=2000000 run_operationcount=2000000 threads=32	Load	36740	851	1595
		Uniform-Read	103890	294	907
		Zipfian-Read	106357	288	865
		Uniform-50%Read-50%Update	46610	Read: 530	Read: 1108
Update: 795	Update: 1684				
tair.essd.standard.2xlarge	recordcount=4000000 run_operationcount=4000000 threads=50	Load	54670	911	1528
		Uniform-Read	150796	314	995
		Zipfian-Read	151110	314	977
		Uniform-50%Read-50%Update	69137	Read: 537	Read: 948
Update: 878	Update: 1479				
air.essd.standard.4xlarge	recordcount=8000000 run_operationcount=8000000 threads=100	Load	90703	1099	1697
		Uniform-Read	285833	339	1196
		Zipfian-Read	288750	335	1162
		Uniform-50%Read-50%Update	110316	Read: 757	Read: 1114
Update: 1041	Update: 1536				

实例规格	YCSB配置	50%Update 工作负载	QPS (次/秒)	Average Latency (微秒)	99th Percentile Latency (微秒)
tair.essd.standard.8xlarge	recordcount=16000000 run_operationcount=160000000 threads=120	Load	117581	1011	1692
		Uniform-Read	477099	242	784
		Zipfian-Read	494550	234	727
		Uniform-50%Read-50%Update	196245	Read: 519	Read: 829
Update: 691	Update: 1096				
tair.essd.standard.13xlarge	recordcount=24000000 run_operationcount=240000000 threads=160	Load	126366	1249	2281
		Uniform-Read	673183	231	637
		Zipfian-Read	691383	230	652
		Uniform-50%Read-50%Update	197803	Read: 678	Read: 940
Update: 935	Update: 1925				

## 数据大于内存场景

实例规格	YCSB配置	工作负载	QPS (次/秒)	Average Latency (微秒)	99th Percentile Latency (微秒)
tair.essd.standard.xlarge	recordcount=64000000 run_operationcount=200000000 threads=32	Load	25561	1245	3497
		Uniform-Read	25727	1239	2042
		Zipfian-Read	47559	667	1217
		Uniform-50%Read-50%Update	19731	Read: 1576	Read: 6383
				Update: 1639	Update: 6487

实例规格	YCSB配置	工作负载	QPS (次/秒)	Average Latency (微秒)	99th Percentile Latency (微秒)
tair.essd.standard.2xlarge	recordcount=128000000 run_operationcount=40000000 threads=50	Load	42287	1179	3465
		Uniform-Read	35794	1394	1880
		Zipfian-Read	77759	637	1219
		Uniform-50%Read-50%Update	28656	Read: 1716	Read: 8863
Update: 1761	Update: 8951				
air.essd.standard.4xlarge	recordcount=256000000 run_operationcount=80000000 threads=100	Load	65923	1514	6615
		Uniform-Read	44753	2232	7903
		Zipfian-Read	120337	826	1382
		Uniform-50%Read-50%Update	38470	Read: 2577	Read: 8535
Update: 2617	Update: 8583				
tair.essd.standard.8xlarge	recordcount=512000000 run_operationcount=160000000 threads=120	Load	89231	1340	9575
		Uniform-Read	51175	2343	2955
		Zipfian-Read	131317	911	1573
		Uniform-50%Read-50%Update	38930	Read: 3063	Read: 8695
Update: 3097	Update: 8735				
tair.essd.standard.13xlarge	recordcount=768000000 run_operationcount=240000000 threads=160	Load	92163	1733	9879
		Uniform-Read	51267	3510	16623
		Zipfian-Read	138522	1152	2131
		Uniform-50%Read-50%Update	39584	Read: 4022	Read: 12159
Update: 4057	Update: 12239				