Cloud Redis Store

Best Practices

Product Introduction

Contents

Best Practices
PHP Connection Example

Last updated：2017-05-05 17:58:38

Preparations before running:

Use the client phpredis and download it at https://github.com/phpredis/phpredis

Sample Codes:

```
<?php
 /**Input your Redis instance's private IP, port number, instance ID and password for the following field.
 $host = "192.168.0.2";
 $port = 6379;
 $instanceid = "c532952f-55dc-4c22-a941-63057e560788";
 $pwd = "1234567q";

 $redis = new Redis();
 //Connect to Redis
 if ($redis->connect($host, $port) == false) {
   die($redis->getLastError());
 }
 //Authentication
 if ($redis->auth($instanceid . ":" . $pwd) == false) {
   die($redis->getLastError());
 }

 /**Then start to work with the Redis instance by referring to https://github.com/phpredis/phpredis */

 //Set the key
 if ($redis->set("redis", "tencent") == false) {
   die($redis->getLastError());
 }
 echo "set key redis suc, value is:tencent\n";

 //Get the key
 $value = $redis->get("redis");
 echo "get key redis is:".$value."\n";
?>
```

Execution results:

JAVA Connection Example

Last updated：2017-05-05 14:31:29

Preparations before running:

Use the client Jedis and download it at https://github.com/xetorthio/jedis/wiki/Getting-started

Sample codes:

```
import redis.clients.jedis.Jedis;

public class HelloRedis {

  public static void main(String[] args) {
      try {
         /**Input your Redis instance's private IP, port number, instance ID and password for the following
         String host = "192.168.0.195";
         int port = 6379;
         String instanceid = "84ffd722-b506-4934-9025-645bb2a0997b";
         String password = "1234567q";
         //Connect to Redis
         Jedis jedis = new Jedis(host, port);
         //Authentication
         jedis.auth(instanceid + ":" + password);

         /**Then start to work with the Redis instance by referring to:https://github.com/xetorthio/jedis */
         //Set the key
         jedis.set("redis", "tencent");
         System.out.println("set key redis suc, value is: tencent");
         //Get the key
         String value = jedis.get("redis");
         System.out.println("get key redis is: " + value);

         //Close to exit
         jedis.quit();
         jedis.close();
      } catch (Exception e) {
         e.printStackTrace();
      }
   }
}
```

Execution results:

NodeJS Connection Example

Last updated：2017-05-05 14:31:47

Preparations before running:

Install node-redis by executing the following command:
npm install hiredis redis

Sample Codes:

```
var redis = require("redis");

/**Input your Redis instance's private IP, port number, instance ID and password for the following fields*
var host = "192.168.0.2",
port = "6379",
instanceid = "c532952f-55dc-4c22-a941-63057e560788",
pwd = "1234567q";
//Connect to Redis
var client  = redis.createClient(port, host, {detect_buffers: true});
// Redis connection error
client.on("error", function(error) {
    console.log(error);
});
//Authentication
client.auth(instanceid + ":" + pwd);

/**Then start to work with the Redis instance */
//Set the key
client.set("redis", "tencent", function(err, reply){
    if (err) {
        console.log(err);
            return;
    }
    console.log("set key redis " + reply.toString() + ", value is tencent");
});

//Get the key
client.get("redis", function (err, reply) {
    if (err) {
        console.log(err);
        return;
    }
    console.log("get key redis is:" + reply.toString());
    //Close the client when the program ends
```

```
    client.end();
});
```

Execution results:

Python Connection Example

Last updated：2018-07-13 15:54:50

Preparations before running:

Download and install redis-py at
https://github.com/andymccurdy/redis-py?spm=5176.730001.3.11.WvETSA

Sample Codes:

```
#!/usr/bin/env python
#-*- coding: utf-8 -*-
import redis

#Replace with the host and port of the connected instance
host = '192.168.0.195'
port = 6379

#Replace with instance ID and instance password
user='username'
pwd='password'

#Specify AUTH information using parameter password upon connection. It is a combination of user and
r = redis.StrictRedis(host=host, port=port, password=user+':'+pwd)

#Once the connection is established, you can perform database operations. For more information, pleas
r.set('name', 'python_test');
print r.get('name')
```

Execution results:

C Connection Example

Last updated：2017-05-05 18:01:15

Preparations before running:

Download and install hiredis at
https://github.com/redis/hiredis

Sample codes:

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

#include <hiredis.h>

int main(int argc, char **argv) {
    unsigned int j;
    redisContext *c;
    redisReply *reply;

if (argc < 4) {
     printf("Usage: 192.168.0.195 6379 instance_id password\n");
     exit(0);
    }
    const char *hostname = argv[1];
    const int port = atoi(argv[2]);
    const char *instance_id = argv[3];
    const char *password    = argv[4];


    struct timeval timeout = { 1, 500000 }; // 1.5 seconds
    c = redisConnectWithTimeout(hostname, port, timeout);
    if (c == NULL || c->err) {
        if (c) {
            printf("Connection error: %s\n", c->errstr);
            redisFree(c);
        } else {
            printf("Connection error: can't allocate redis context\n");
        }
        exit(1);
    }

    /* AUTH */
```

```
    reply = redisCommand(c, "AUTH %s:%s", instance_id, password);
    printf("AUTH: %s\n", reply->str);
    freeReplyObject(reply);

    /* PING server */
    reply = redisCommand(c,"PING");
    printf("PING: %s\n", reply->str);
    freeReplyObject(reply);

    /* Set a key */
    reply = redisCommand(c,"SET %s %s", "name", "credis_test");
    printf("SET: %s\n", reply->str);
    freeReplyObject(reply);

    /* Try a GET  */
    reply = redisCommand(c,"GET name");
    printf("GET name: %s\n", reply->str);
    freeReplyObject(reply);

    /* Disconnects and frees the context */
    redisFree(c);

    return 0;
}
```

Execution results:

Go Connection Example

Last updated：2017-05-05 14:32:35

Preparations before running:

Use the client Go-redis and download it at https://github.com/alphazero/Go-Redis

Sample codes:

```
package main

import(

  "fmt"

  "redis"

  "log"

)



func main() {



  const host=192.168.0.195
  const port=6379
  const instanceId="84ffd722-b506-4934-9025-645bb2a0997b"
  const pass="1234567q"
  // Connect to the Redis server 192.168.0.195:6379 and authorize the use of instanceId password
  spec := redis.DefaultSpec().Host(host).Port(port).Password(instanceId+":"+pass);
  client, err := redis.NewSynchClientWithSpec(spec)



  if err != nil { // Whether an error occurs with the connection

    log.Println("error on connect redis server")

    return

  }
```

```
    newvalue :=[]byte("QcloudV5!");
    err=client.Set("name",newvalue);
    if err != nil { // Incorrect set value
      log.Println(err)
      return
    }

    value, err := client.Get("name") // Value
    if err != nil {
      log.Println(err)
      return
    }
    fmt.Println("name value is:",fmt.Sprintf("%s", value)) //Output
}
```

Execution results:

.Net Connection Example

Last updated：2017-05-05 14:33:25

Preparations before running:

Download and install ServiceStack.Redis at

https://github.com/ServiceStack/ServiceStack.Redis

Sample codes:

Without a connection pool:

```
using System.Collections.Generic;
using System.Linq;
using System.Text;
using ServiceStack.Redis;
using System;

namespace ConsoleApplication1
{
    class Program
    {
        static void Main(string[] args)
        {
            string host = "10.66.82.46";//Address for instance access to host
            int port = 6379;// Port information
            string instanceId = "bd87dadc-84f1-44f1-86dd-021dc4acde96";//Instance ID
            string pass = "1234567q";//Password

            RedisClient redisClient = new RedisClient(host, port, instanceId + ":" + pass);
            string key = "name";
            string value = "QcloudV5!";
            redisClient.Set(key, value); //Set value
            System.Console.WriteLine("set key:[" + key + "]value:[" + value + "]");
            string getValue = System.Text.Encoding.Default.GetString(redisClient.Get(key)); //Read value
            System.Console.WriteLine("value:" + getValue);
            System.Console.Read();
        }
    }
}
```

With ServiceStack 4.0 connection pools

```
using System.Collections.Generic;
using System.Linq;
using System.Text;
using ServiceStack.Redis;
using System;

namespace ConsoleApplication2
{
    class Program
    {
        static void Main(string[] args)
        {
            string[] testReadWriteHosts = new[] {
            "redis://:fb92bf2e0abf11e5:1234561178a1A@127.0.0.1:6379"/*redis: //: instance ID: password@ac
            };
            RedisConfig.VerifyMasterConnections = false;//Required to be set
            PooledRedisClientManager redisPoolManager = new PooledRedisClientManager(10/*Number c
            10/*Timeout of connection pool*/, testReadWriteHosts);
            for (int i = 0; i < 100; i++)
           {
                IRedisClient redisClient = redisPoolManager.GetClient();//Get the connection
                RedisNativeClient redisNativeClient = (RedisNativeClient)redisClient;
                redisNativeClient.Client = null;/
                try
               {
                  string key = "test1111";
                  string value = "test1111";
                  redisClient.Set(key, value);
                  redisClient.Dispose();//
               }
               catch (Exception e)
               {
                   System.Console.WriteLine(e.Message);
               }
           }
           System.Console.Read();
        }
    }
}
```

With ServiceStack 3.0 connection pools

```
using System.Collections.Generic;
using System.Linq;
```

```
using System.Text;
using ServiceStack.Redis;
using System;

namespace ConsoleApplication3
{
  class Program
  {
    static void Main(string[] args)
    {
          string[] testReadWriteHosts = new[] {
             "fb92bf2e0abf11e5:1234561178a1A@127.0.0.1:6379" /*instance ID: password@access address: p
             };
             PooledRedisClientManager redisPoolManager = new PooledRedisClientManager(10/*Number
             for (int i = 0; i < 100; i++)
            {
             IRedisClient redisClient = redisPoolManager.GetClient();//Get the connection
             try
            {
               string key = "test1111";
               string value = "test1111";
               redisClient.Set(key, value);
               redisClient.Dispose();//
            }
             catch (Exception e)
           {
                System.Console.WriteLine(e.Message);
           }
         }
         System.Console.Read();
     }
   }
}
```

Execution results:

Ranking by Go

Last updated：2017-05-05 14:33:44

```
package main

import (
    "fmt"
    "math/rand"
    "time"
    "strconv"
    "strings"
    "github.com/garyburd/redigo/redis"
)

func checkErr(err error) {
    if err != nil {
        panic(err.Error())
    }
}

func randomName(length int) string {
    rand.Seed(time.Now().UnixNano())
    rn := make([]string, length)
    for start := 0; start < length; start++ {
        ch := rand.Intn(3)
        switch ch {
        case 0:
            rn = append(rn, strconv.Itoa(rand.Intn(10)))
        case 1:
            rn = append(rn, string(rand.Intn(26) + 65))
        default:
            rn = append(rn, string(rand.Intn(26) + 97))
        }
    }
    return strings.Join(rn, "")
}

func main() {
    const TOTAL_SIZE = 10000
    redisServer := "localhost:6379"
    client, err := redis.Dial("tcp", redisServer)
    checkErr(err)
    defer client.Close()
```

```
    key := "Game Rank"
    client.Do("DEL", key)
    playerList := make([]string, 0)
    for i := 0; i < TOTAL_SIZE; i++ {
        playerList = append(playerList, randomName(8))
    }
    fmt.Println("*Input all " + strconv.Itoa(TOTAL_SIZE) + " players*")
    r := rand.New(rand.NewSource(time.Now().UnixNano()))
    for i := 0; i < len(playerList); i++ {
        score := r.Intn(5000)
        member := playerList[i]
        if i < 10 {
            fmt.Println("player: " + member + ", score: " + strconv.Itoa(score))
        }
        client.Do("ZADD", key, score, member)
    }
    fmt.Println("*more player.....*")
    fmt.Println()
    fmt.Println("*" + key + "*")
    fmt.Println("*Top 100 players*")
    scoreList, err := redis.Strings(client.Do("ZREVRANGE", key, 0, 99, "WITHSCORES"))
    checkErr(err)
    loop := 0
    for i := 0; i < len(scoreList) - 1; i += 2 {
        player := scoreList[i]
        score := scoreList[i + 1]
        if loop < 10 {
            fmt.Println("player: " + player + ", score: " + score)
        }
        loop++
    }
    fmt.Println("*more player.....*")
    fmt.Println("*" + key + "*")
    selectPlayer := playerList[0]
    rank, err := redis.Int(client.Do("ZREVRANK", key, selectPlayer))
    checkErr(err)
    fmt.Println("The rank of player " + selectPlayer + " is " + strconv.Itoa(rank))
}
```

The result is as follows:

```
[root@VM_44_132_centos RankTop]# go run RankTop.go
*Input all 10000 players*
player: 8wLf5S1R, score: 1922
player: h6N396f5, score: 3753
player: B3tbrLQ1, score: 747
player: igS7Xh55, score: 2127
player: FCKgLYX1, score: 3606
player: 0j1a857v, score: 3985
player: G9U2Kvwm, score: 969
player: 2lSqr4s6, score: 1699
player: 4hyx7Ca9, score: 2079
player: Qp7175dg, score: 2484
*more player......*

*Game Rank*
*Top 100 players*
player: dhw5R4z1, score: 4999
player: HE86UaLf, score: 4999
player: Fiky9RsK, score: 4999
player: z714KSM5, score: 4998
player: pVrv6zxh, score: 4998
player: mkNU26jw, score: 4998
player: H9fnMl42, score: 4998
player: Tw12J5yk, score: 4996
player: C2cOtt98, score: 4996
player: 8w2v8jU5, score: 4996
*more player......*
*Game Rank*
The rank of player 8wLf5S1R is 6180
```

Getting Friend Relationship by Go

Last updated：2017-05-05 14:34:03

```
package main

import (
    "fmt"
    "github.com/garyburd/redigo/redis"
)

func checkErr(err error) {
    if err != nil {
        panic(err)
    }
}

func main() {
    redisServer := "localhost:6379"
    client, err := redis.Dial("tcp", redisServer)
    checkErr(err)
    defer client.Close()
    //my friends
    client.Do("SADD", "myfriends", "John")
    client.Do("SADD", "myfriends", "Emily")
    client.Do("SADD", "myfriends", "Ben")
    client.Do("SADD", "myfriends", "Steven")
    fmt.Println("my friends are: ")
    myList, err := redis.Strings(client.Do("SMEMBERS", "myfriends"))
    checkErr(err)
    for _, item := range myList {
        fmt.Print(item + " ")
    }
    fmt.Println()
    //your friends
    client.Do("SADD", "yourfriends", "Mark")
    client.Do("SADD", "yourfriends", "Tim")
    client.Do("SADD", "yourfriends", "Willim")
    client.Do("SADD", "yourfriends", "Ben")
    client.Do("SADD", "yourfriends", "Steven")
    fmt.Println("your friends are: ")
    youList, err := redis.Strings(client.Do("SMEMBERS", "yourfriends"))
    checkErr(err)
    for _, item := range youList {
```

```
        fmt.Print(item + " ")
    }
    fmt.Println()
    fmt.Println("our common friends are: ")
    commonList, err := redis.Strings(client.Do("SINTER", "myfriends", "yourfriends"))
    checkErr(err)
    for _, item := range commonList {
        fmt.Print(item + " ")
    }
    fmt.Println()
}
```

The result is as follows:

```
[root@VM_44_132_centos SocialNetwork]# go run SocialNetwork.go
my friends are:
Ben Steven Emily John
your friends are:
Ben Steven Tim Willim Mark
our common friends are:
Ben Steven
```