非关系型的数据库的一个类型：键值(Key-Value)存储数据库

Redis是用C语言开发的一个开源的高性能键值对（key-value）数据库。它通过提供多种键值数据类型来适应不同场景下的存储需求。

目前为止Redis支持的键值数据类型如下：

字符串类型

散列类型

列表类型

集合类型

有序集合类型。

应用场景

缓存（数据查询、短连接、新闻内容、商品内容等等）。（最多使用）

聊天室的在线好友列表。

任务队列。（秒杀、抢购、12306等等）

应用排行榜。

网站访问统计。

数据过期处理（可以精确到毫秒）

分布式集群架构中的session分离。

安装到CentOS和使用

1、安装gcc环境

```
yum install gcc-c++
```

2、将redis-3.0.0.tar.gz的压缩文件移到CentOS去

方法：（rz　　ftps　　ftp）

3、解压文件

```
tar -xvf jdk.tar -C /redis
```

4、进入解压文件夹/redis/redis-3.0.0对其进行编译

```
make
```

5、安装相应的服务组件

make PREFIX=/redis install

安装完成后会在/redis/bin目录下产生相应的服务组件

redis-benchmark　　----性能测试工具

redis-check-aof　　----AOF文件修复工具

redis-check-dump　　----RDB文件检查工具（快照持久化文件）

redis-cli　　----命令行客户端

redis-server　　----redis服务器启动命令

6、进入/redis/redis-3.0.7修改redis启动配置文件

vi redis.conf

```
################################## GENERAL  #####################################

# By default Redis does not run as a daemon. Use 'yes' if you need it.
# Note that Redis will write a pid file in /var/run/redis.pid when daemonized.
daemonize yes
```

修改后台起动模式

7、配置文件拷贝到 /redis/bin

cp redis.conf　/redis/bin

8、启动服务在bin目录　　停止服务：./redis-cli shutdown

./redis-server　redis.conf

9、连接客户端　redis-cli -h ip地址 -p 端口

默认：在bin目录输入　　./redis-cli

使用：

10、set/get，使用set和get可以向redis设置数据、获取数据

set name liujinrong

get name

11、del，删除指定key的内容　　del name

12、keys * ，查看当前库中所有的key值

Redis默认端口6379，通过当前服务进行查看

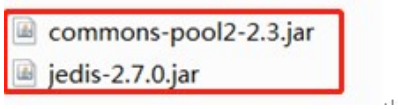
ps -ef | grep -i redis

Jedis：

Redis不仅是使用命令来操作，现在基本上主流的语言都有客户端支持，比如java、C、C#、C++、php、Node.js、Go等。在官方网站里列一些Java的客户端，有Jedis、Redisson、Jredis、JDBC-Redis、等其中官方推荐使用Jedis和Redisson。 在企业中用的最多的就是Jedis

Java连接Redis

1、导入Jar包



2、改CentOS的端口号

/sbin/iptables -I INPUT -p tcp --dport 6379 -j ACCEPT　改端口号

/etc/rc.d/init.d/iptables save　保存

/etc/init.d/iptables status　查看

3、写代码

3.1、单实例连接

```
@Test
public void testJedisSingle(){
        //1 设置ip地址和端口
        Jedis jedis = new Jedis("192.168.137.128", 6379);
        //2 设置数据
        jedis.set("name", "itheima");
        //3 获得数据
        String name = jedis.get("name");
        System.out.println(name);
        //4 释放资源
        jedis.close();
}
```

3.2、连接池连接

```
@Test
public void testJedisPool(){
        //1 获得连接池配置对象，设置配置项
        JedisPoolConfig config = new JedisPoolConfig();
        // 1.1 最大连接数
        config.setMaxTotal(30);
        // 1.2　最大空闲连接数
        config.setMaxIdle(10);
        //2 获得连接池
        JedisPool jedisPool = new JedisPool(config, "192.168.137.128", 6379);
        //3 获得核心对象
        Jedis jedis = null;
```

```
        try {
                jedis = jedisPool.getResource();
                //4 设置数据
                jedis.set("name", "itcast");
                //5 获得数据
                String name = jedis.get("name");
                System.out.println(name);
        } catch (Exception e) {
        } finally{
                if(jedis != null){
                        jedis.close();
                }
                // 释放pool资源
                if(jedisPool != null){
                        jedisPool.close();
                }
        }
}
```

其中1、2步可用以下代码 GenericObjectPoolConfig与JedisPoolConfig是继承关系

```
        //设置连接配置信息
        GenericObjectPoolConfig config = new GenericObjectPoolConfig();
        config.setMaxTotal(20);
        config.setMaxIdle(10);
        //获取连接池对象
        JedisPool jedisPool = new JedisPool(config , "192.168.137.128", 6379);
        //获取连接对象
        Jedis resource = jedisPool.getResource();
```

Redis的数据结构（蓝色关键字为JAVA中通过对象可以使用的方法）

字符串（String）

方法：

```
jedis.set("name", "itheima");  存
String name = jedis.get("name");  取
```

数值增减

incr key：将指定的key的value原子性的递增1.

decr key：将指定的key的value原子性的递减1

incrby key increment：将指定的key的value原子性增加increment

decrby key decrement：将指定的key的value原子性减少decrement

append key value：拼凑字符串。如果该key存在，则在原有的value后追加该值；如果该key不存在，则重新创建一个key/value

判断

hexists key field：判断指定的key中的filed是否存在

hlen key：获取key所包含的field的数量

获取所有

hkeys key ：获得所有的key

hvals key：获得所有的value

哈希（**hash**）

hset key field value：为指定的key设定field/value对（键值对）。

hmset key field value [field2 value2 …]：设置key中的多个filed/value

取值

hget key field：返回指定的key中的field的值

hmget key fileds：获取key中的多个filed的值

hgetall key：获取key中的所有filed-vaule

添加/删除元素

sadd key values[value1、value2…]：向set中添加数据，如果该key的值已有则不会重复添加

srem key members[member1、member2…]：删除set中指定的成员

获得集合中的元素

smembers key：获取set中所有的成员