**Redis**

برای نصب redis بر روی ویندور 10 طیق این [لینک](https://www.youtube.com/watch?v=_nFwPTHOMIY) پیش روید.

**Redis چیست؟**

Redis یک سرور داده-ساختار باز و متن باز است که به عنوان یک پایگاه داده key-vlaue در حافظه عمل می کند**.**

1. **ویژگی هایRedis**
   * Redis یک سرور داده-ساختار است که داده ها را در حافظه نگه می دارد.
   * این به معنای عملکرد سریع تر نسبت به پایگاه داده های معمولی است که داده ها را در دیسک ذخیره می کنند.
   * Redis یک پایگاه داده key-vlaue است که انواع داده مختلفی را پشتیبانی می کند مانند رشته ها، لیست ها، مجموعه ها و غیره.
2. **استفاده از Redis**
   * Redis به طور گسترده ای برای کش کردن داده ها، مدیریت جلسه، شمارنده ها و سایر موارد استفاده می شود.
   * همچنین می تواند به عنوان یک صف پیام رسان عمل کند.
3. **کاربردهای Redis**
   * کش کردن داده ها
   * مدیریت جلسه
   * شمارنده ها و محدودیت های سرعت
   * صف پیام رسان
   * پردازش داده های سریع

این [لینک](https://www.youtube.com/watch?v=8sHCdz_tOjk&list=PL4cUxeGkcC9h3V2eqhi8rRdIDJshP-b4P) میتونه خیلی مفید باشه.

**کامندهای ساده redis**

کامندهای redis به شرح زیر است:

**1)set**

* **توابعset(name, value, ex=None, px=None, nx=False, xx=False, get=False)**
* کاربرد: برای ذخیره یک مقدار جدید در ردیس استفاده میشه.
* پارامترها:
  + name کلید مربوط به مقدار مورد نظر
  + value مقدار مورد نظر برای ذخیره
  + exزمان انقضای مقدار بر حسب ثانیه (اختیاری)
  + pxزمان انقضای مقدار بر حسب میلی ثانیه (اختیاری)
  + Nx اگر True باشه، مقدار فقط در صورتی ذخیره میشه که کلید وجود نداشته باشه (اختیاری)
  + xx اگر True باشه، مقدار فقط در صورتی ذخیره میشه که کلید قبلا وجود داشته باشه (اختیاری)
  + get اگر True باشه، مقدار قبلی کلید قبل از ذخیره بازگردونده میشه (اختیاری)

**2)get**

* تابع get(name, default=None)
* کاربرد: برای بازخوانی مقدار یک کلید در ردیس استفاده میشه.
* پارامترها:
  + name کلید مورد نظر برای بازخوانی مقدار
  + default مقدار پیشفرض اگر کلید وجود نداشته باشه (اختیاری)

از این [لینک](https://redis.io/docs/latest/commands/) می توانید انواع کامندهای معروف redis را مشاهده کنید.

**تمرین اول:**

فرض کنید سرویس فروشگاهی ای داریم و برای هر کدام از وندور های قرار است یک صورت حساب ایجاد کنیم که فرایند پیچیده ای دارد و ما نیاز داریم این مقدار را یک روز در کش نگهداری کنیم تا در منابع صرفه جویی کنیم. فرض کنید دیتا ها به این شکل هستند:

**Ali => 1000**

**Alex => 2000**

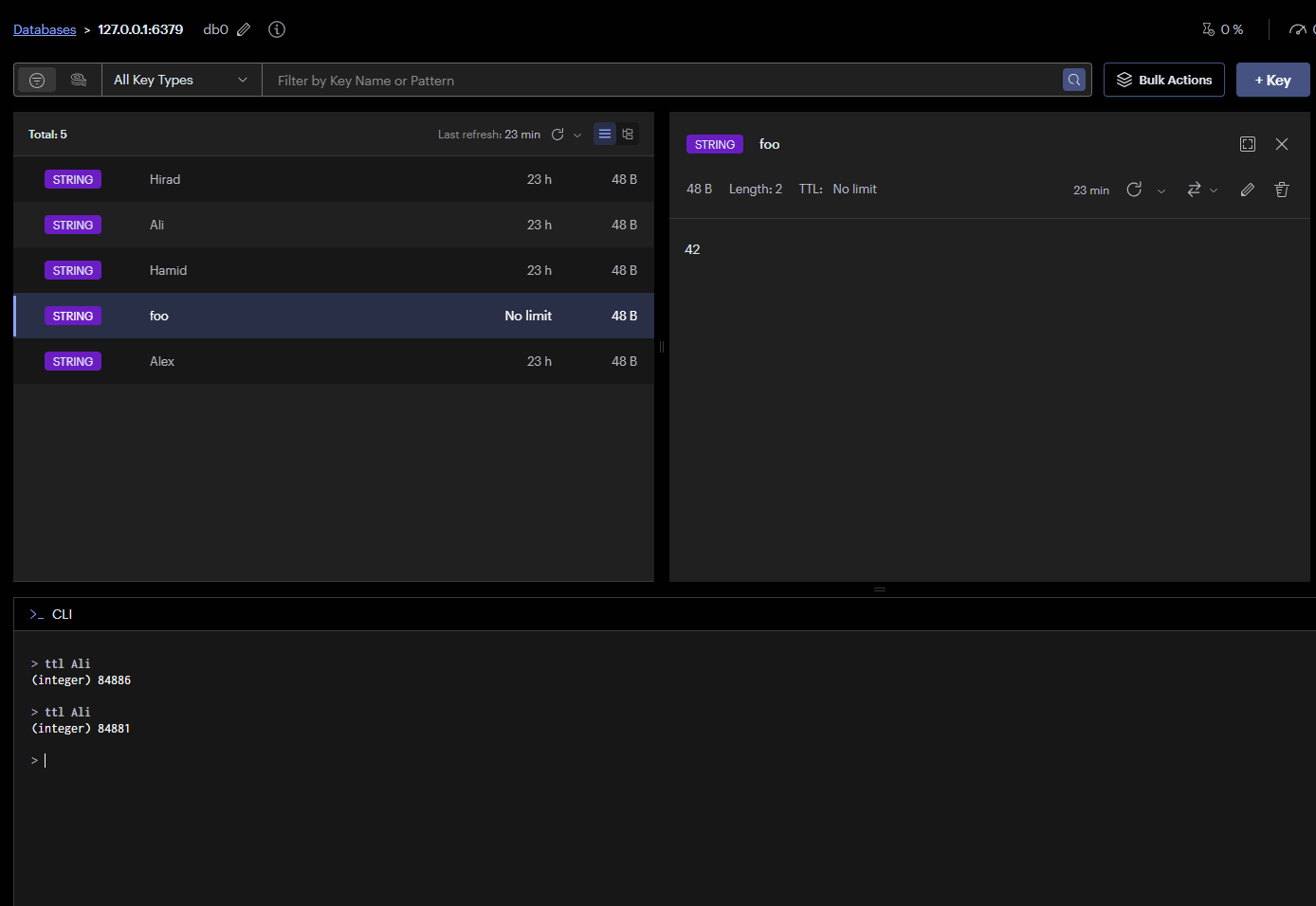
**Hamid => 1356**

**Hirad => 10180**

این دیتا های را با ttl برابر با یک روز روی redis بریزید و بعد سعی کنید مقدار های ست شده را بخوانید.

پاسخ: در فایل task1.py کد موجود است.





**نکات جالب درباره redis :**

معماری redis کلاینت سرور هست و هم چنین redis یک single thread هست در مقایسه با Memcached که multi thread هست. هم چنین redis TCP را ساپورت میکند ولی UDP را ساپورت نمی کند ولی Memecached هر دو را ساپورت می کند. برای اطلاعات بیشتر [لینک](https://scalegrid.io/blog/redis-vs-memcached-2021-comparison/) مطالعه کنید.

**پیاده سازی لاک با redis :**

هدف لاک این است که اطمینان حاصل شود که در بین چند اینستنس کد که ممکن است سعی کنند کار مشابهی را انجام دهند، فقط یکی از آنها آن کار را در واقع انجام دهد حداقل فقط یکی در یک زمان. این کار ممکن است نوشتن برخی از داده ها در یک سیستم ذخیره سازی مشترک، انجام برخی محاسبات، فراخوانی برخی از API های خارجی یا موارد مشابه باشد.از طرفی باید این نکته را در نظر داشت که ردیس یک دیتابیس core single است و در هر لحظه فقط میتواند یک کامند رو اجرا کند. ما میخواهیم از این ویژگی ردیس برای پیاده سازی تمرین زیر استفاده کنیم.

**تمرین دوم:**

فرض کنید یک سرویس داریم که اینستنسهای مختلفی از آن در حال اجراست. یک درخواست مشابه به همه ی پادها ارسال میشود که دیتابیس رو آپدیت کنند، حال میخواهیم با استفاده از ردیس کاری کنیم که اگر پادها درخواست های مشابه ایی همزمان دریافت کردند فقط یکی از پاد کار را انجام دهد. با جستوجو وب راه حل مناسب رو پیدا کنید.

پاسخ:

راهش استفاده از distributed locker هست. می تونیم از یک کلید redis به عنوان lock استفاده کنیم که هر پاد باید قبل از اینکه هر پاد عملیات آپدیت دیتابیس را انجام دهد، باید این قفل را بگیرد. اگر قفل در دست یکی دیگر از پادها باشد، آن پاد باید عملیات آپدیت را نمی تونه انجام بده. دو لینک زیر اطلاعت خوبی درباره distributed lock میدن:

<https://reintech.io/blog/implementing-distributed-lock-redis>

<https://www.youtube.com/watch?v=2hIxBJa23M0>

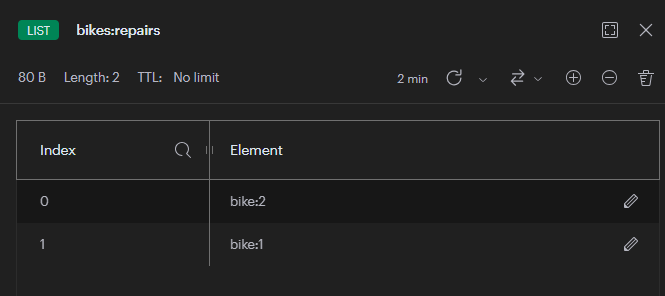
**ساختمان داده های مختلف در redis**

داده ساختارهای مختلفی در redis داریم که تو تمرین یک از string که ساده ترینش هست استفاده کردیم.

مثلا می تونیم از داده ساختار list استفاده کنیم که به صورت زیر است:



و خروجی نیز به صورت زیر است:

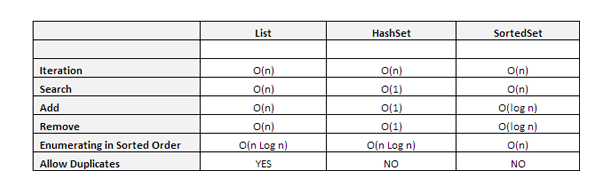


انواع داده ساختارهای دیگری هم موجود است:

Sort,sortset,hashes,…

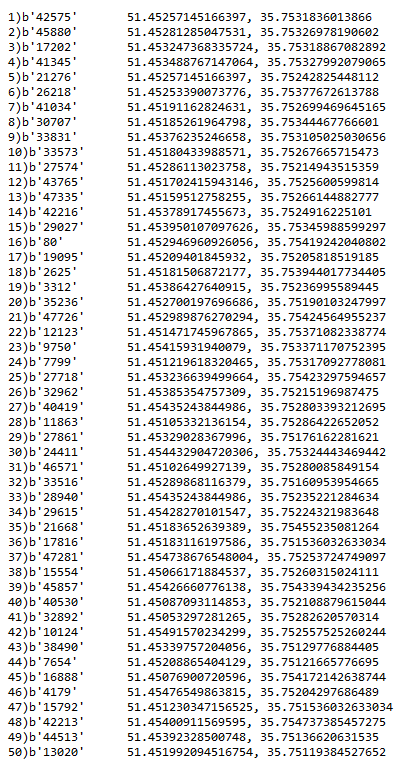
چندتا از داده ساختارها کدش در فایل datastructure\_redis.py موجود است.

مثلا کاربرد sorted set به این صورت هست که ما در یک مسابقه که ممکن هست میلیون ها کاربر داشته باشد و هر دفعه که میخوایم امتیازها رو اپدیت کنیم ممکن هست کلی طول بکشه علاوه بر این ممکن هست بخوایم امتیازات سورت شده داشته باشیم که میدانیم بهترین اردر الگوریتم های سورت برابر با o(nlogn) است ولی تو sorted set هر وقت که المان جدیدی اضافه می کنیم خودش در پوزیشن درست با اردر زمانی o(logn) قرار میگیره یا مثلا می خوایم کاربرانی که امتیازاشون بین دو محدوده خاص هست را داشته باشیم جدول زیر زیر به خوبی پیچیدگی زمانی هر کدام از عملیات را توضیح داده است.



این [لینک](https://medium.com/@sandeep4.verma/building-real-time-leaderboard-with-redis-82c98aa47b9f#id_token=eyJhbGciOiJSUzI1NiIsImtpZCI6IjNkNTgwZjBhZjdhY2U2OThhMGNlZTdmMjMwYmNhNTk0ZGM2ZGJiNTUiLCJ0eXAiOiJKV1QifQ..Xe4twGobSpyN0nITQPeGIX-5OIiOa4x5WAm16RzaX9z2-NlaMdhtS186LopIK0iohg-idgh5vJm5PhADjAQDStxhpJ48CKXAaM620JALvdbdj_cQzCyLL4PpELHzqnM-so1QFhh_iW3bAu7MC2DwgOqzPHoe8Sd1CAMaBO91YuG01922qDadx7xayi0ypQ1WDGrZXbitr7791VfFt3WWumpnC9Uy5nbfuJ4Boy4gT2oi4xIvjnOZ3QvMXKGQ3YoWfGJY0bWcjsFQWENl8KlccRv6Adpe0232VTeA4_JJwl315NVlh-9JSZ-BbSUXBppUNcRaSuzVneeIndQYRSewNQ) خیلی خوب این موارد را توضیح داده است.

Geospatial هم خیلی برای داده های جغرافیایی مناسب هست مثلا می خوایم ادرس هایی در 5 کیلومتری اون محدوده آدرسی ما هست پیدا کنیم. در نوتبوک resturant\_geospatial.ipynb من با استفاده از flask و پایتون کد زدم که در ابتدا حدود 50000 مختصات رندوم با فواصل مختلف نسبت به تهران رو در redis اد میکنه و بعد میاد یک نقطه رندوم بهش میدم و بعد 50 تا از نزدیک ترین آدرس های به اون رستوران رو میده.



**lua scripts & Pipeline , transactions**

**برای مطالعه این [لینک](https://rafaeleyng.github.io/redis-pipelining-transactions-and-lua-scripts) مطالعه کنید**

**:pipeline**

Pipeline به این صورت هست که در ابتدا چندین دستور را در لیست قرار میدیم و به صورت یکجا ارسال میکنیم و بعد redis دستورات را به صورت متوالی اجرا میکنه و در نهایت همه رو با هم بر میگردونه. مزیت اصلی هم این هست که کاهش ping بین کلاینت و سرور redis رو به همراه داره. ولی باید حواسمون باشه که response دستور قبلی به دستور بعدی وابسته نباشه.

**transactions**

فرقی که با pipeline داره این هست که تمام دستورات به صورت یکپارچه اجرا میشن ولی در pipeline به صورت مجزا اجرا می شوند. اگر در حین اجرای transaction، خطایی رخ دهد، تمام تغییرات لغو می‌شوند و داده‌ها در وضعیت قبلی باقی می‌مانند.

**Lua scripts**

Lua Scripts در Redis یک قابلیت قدرتمند است که برای اجرای کدهای پیچیده‌تر در سمت سرور استفاده می‌شود و به توسعه‌دهندگان این امکان را می‌دهد که منطق پیچیده‌تری را در بستر Redis پیاده‌سازی کنند.

**تمرین سوم:**

فرض کنید میخواهیم ۱۰۰۰ کاربر جدید در ثانیه در ردیس بسازیم. برای این کار اگر به صورت عادی این کار را انجام دهیم به ازای هر ریکوئست ۱ میلی ثانیه زمان صرف می شود و در کل این کار ۱۰ثانیه طول خواهد کشید اما اگر این کاررا به یکباره و با transaction یا pipeline ها انجام دهیم زمان بسیار کمتری صرف خواهد شد. به این صورت که این بار لیست کامند ها را به ردیس میدهیم و ردیس کامند ها را به ترتیب انجام میدهد.فرض کنید ساختارکاربران به این شکل است و سعی کنید کد این برنامه را پیاده کنید.

User-12: hash (

Name: Ali

LastName: Alavi

)

من در نوتبوک task3.ipynb ابتدا بدون pipeline و سپس با استفاده از pipeline یوزرها را اد کردم اگه نگاه کنید بدون pipeline 0.3 ثانیه طول کشیده است و با pipeline 0.1 ثانیه طول کشیده. با transaction نیز اینکار انجام دادم و همین 0.1 ثانیه طول کشید.

**مطالعه:**

برای مطالعه ی این موضوع که چطور pipeline پرفورمنس رو بهتر میکند این [لینک](https://virgool.io/@mhrlife/%D9%BE%D8%A7%DB%8C%D9%BE-%D9%84%D8%A7%DB%8C%D9%86-pipeline-%D8%AF%D8%B1-%D8%B1%D8%AF%DB%8C%D8%B3-%D9%81%D8%B4%D8%A7%D8%B1-%DA%A9%D9%85%D8%AA%D8%B1-%D9%88-%D8%B3%D8%B1%D8%B9%D8%AA-%D8%A8%DB%8C%D8%B4%D8%AA%D8%B1-n4y0wuqqhvrw) رو مشاهده کنید.

برای مطالعه بیشتر نیز درباره امنیت از این [لینک](https://redis.io/topics/security) بخونید.