## 常识：

|  |
| --- |
| 计算机数据是存在磁盘，磁盘的维度有两个指标：   1. 寻址（速度毫秒级） 2. 带宽 单位时间可以有多少字节流过去，多少数据量流过去。G/M 级别   内存   1. 寻址 纳秒   2-内存的带宽（内存数据往外走） 很大  （秒--> 毫秒--》微秒--》纳秒）【磁盘比内存寻址慢了10w倍】  I/O buffer： 成本问题  磁盘有磁道和扇区，一个扇区512Byte  有一个成本问题，如果访问一个硬盘，都是以一个扇区512字节来找，一个硬盘1T有很多个512Byte，要找到你存放数据的那个扇区，索引成本很大。  操作系统，无论你读多少，都是最少4K从磁盘上拿 （类比CPU缓存行的概念） |

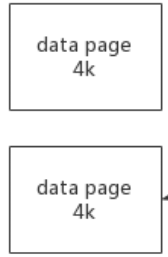
## **文件存数据**

最早的时候，数据存在文件里，比如data.txt ，Linux里有grep 、ark，你还可以用java写一个程序，基于IO流的文件查找。

随着文件的变大，读取速度变慢，为什么？因为文件变大，回收到硬盘读取的瓶颈，（I/O成为瓶颈）

## 数据库存储

数据库有data page，大小是4K。

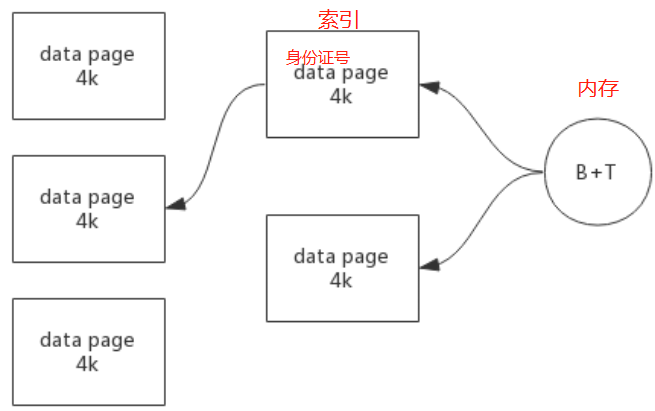


数据库的表，行记录存到磁盘的时候，用了很多4K 4K的小格子，这个4K刚好和磁盘的一次读取大小4K对应上，这个4K定义小的话是浪费，应为读取磁盘一次还是4K，定义大于4K比如8K 16K无所谓。

你有1w行数据，会散落在这一个个4K的小格子里，光有小格子的话，查找数据的成本、复杂度，还和直接读取文件一样，还是要将一个个4K都读到内存，挨个的去找数据。走的还是全量IO，跟直接读取文件的IO量是一样的，所以一定会很慢。所以数据库只是建表了，没有建索引的话，读取速度还是很慢。

提速的话用索引，索引用的也是4K存储模型。索引存储在硬盘

比如你在身份证号上建立索引，



关系型数据库，建表的时候，必须给出建表schema建表语句，给出一共有多少列，每个列的类型，类型：规定了字节宽度。只要给出了schema，每一行数据的宽度就定死了。

这样的话，假设一行有10个字段，你插入一行，你只给出第一个和第七个字段，其他字段都没有给，数据向data page放的时候，那些没有数据的字段都会用0去开辟，用空的东西去补充那些字节，进行占位。好处是，将来进行增删改的时候，不用去移动数据，直接拿新的数据把原来位置进行复写就可以了。

明确：数据和索引都是在硬盘里的

内存是速度最快的地方，在内存里准备一个B+Tree，B+Tree树干是在内存里的，只存一些区间，数据区间和偏移，sql的where条件里只要命中索引了，查询在B+Tree会找到某一个页，这个页刚好在这个B+Tree区间里，然后把索引从磁盘读到内存，然后就找到哪个data page，把这个data page读到内存里，就找到记录了。

索引和数据在磁盘，B+Tree在内存，分而治之。减少磁盘IO的流量。

如果数据库表有几千万行，表很大，性能降低？如果表有索引

增删改变慢：需要找到索引，调整位置

查询速度？假设表有1T，B+Tree也能在内存装下，哪都没有溢出，没有问题

1. 一个人或者少量来查询，简单sql，where条件能够命中索引，查询依然很快。还是从内存的B+Tree，找到索引文件，再找到data page，带到内存。并不会因为数据量的变大，把别的data page也带到内存里去。
2. 并发大 的时候：会受到**硬盘带宽影响**速度
   1. 大量sql查询不同的data page，每个data page的4K，是挨个的向内存去走，是并行的。（MySQL8.0引入了并行查询虽然还比较初级）此时就会有一个部分查询等待。
   2. 或者复杂的sql到达：查询的时候不是要获取一个data page到内存了，而是多个data page到内存。

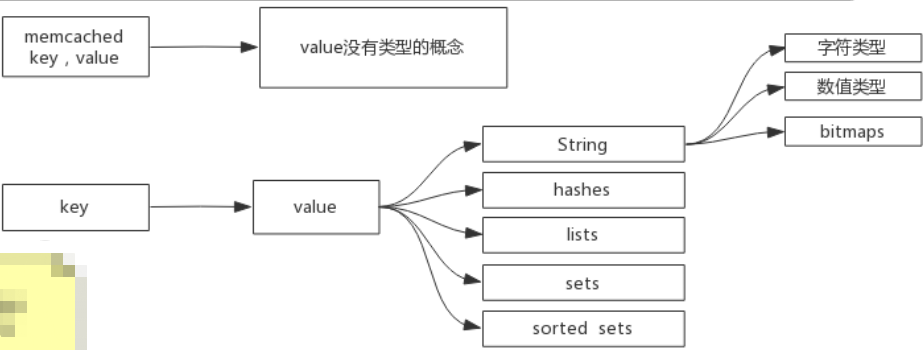
商业内存关系数据库

SAP公司的HANA数据库。内存级别的关系型数据库，内存约2T，硬件软件服务总和约2亿一个套餐。

数据在磁盘和内存中体积不一样。磁盘没有指针的概念。同个文件内存占用的更少。

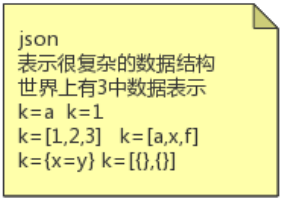
## **缓存**

Redis 是一个开源（BSD许可）的，内存中的数据结构存储系统，它可以用作数据库、缓存和消息中间件。 它支持多种类型的数据结构，如 [字符串（strings）](http://redis.cn/topics/data-types-intro.html" \l "strings)， [散列（hashes）](http://redis.cn/topics/data-types-intro.html" \l "hashes)， [列表（lists）](http://redis.cn/topics/data-types-intro.html" \l "lists)， [集合（sets）](http://redis.cn/topics/data-types-intro.html" \l "sets)， [有序集合（sorted sets）](http://redis.cn/topics/data-types-intro.html" \l "sorted-sets) 与范围查询， [bitmaps](http://redis.cn/topics/data-types-intro.html" \l "bitmaps)， [hyperloglogs](http://redis.cn/topics/data-types-intro.html" \l "hyperloglogs) 和[地理空间（geospatial）](http://redis.cn/commands/geoadd.html) 索引半径查询。 Redis 内置了 [复制（replication）](http://redis.cn/topics/replication.html)，[LUA脚本（Lua scripting）](http://redis.cn/commands/eval.html)， [LRU驱动事件（LRU eviction）](http://redis.cn/topics/lru-cache.html)，[事务（transactions）](http://redis.cn/topics/transactions.html) 和不同级别的 [磁盘持久化（persistence）](http://redis.cn/topics/persistence.html)， 并通过 [Redis哨兵（Sentinel）](http://redis.cn/topics/sentinel.html)和自动 [分区（Cluster）](http://redis.cn/topics/cluster-tutorial.html)提供高可用性（high availability）。



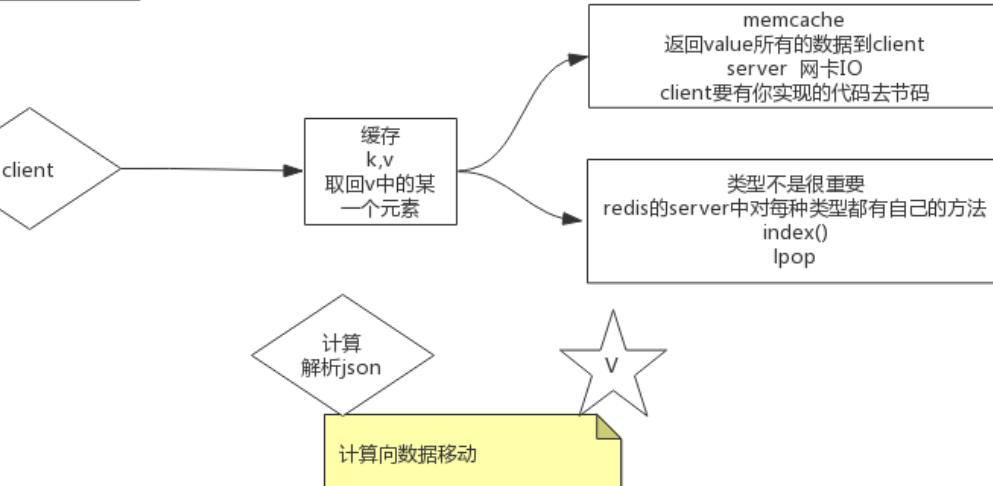
Redis ：秒级10w次操作

JSON格式可以表示很复杂的数据格式



问，Memcached也可以把各种类型转换成Json字符串存，为什么还要Redis？

复杂度不一样。



## 安装install

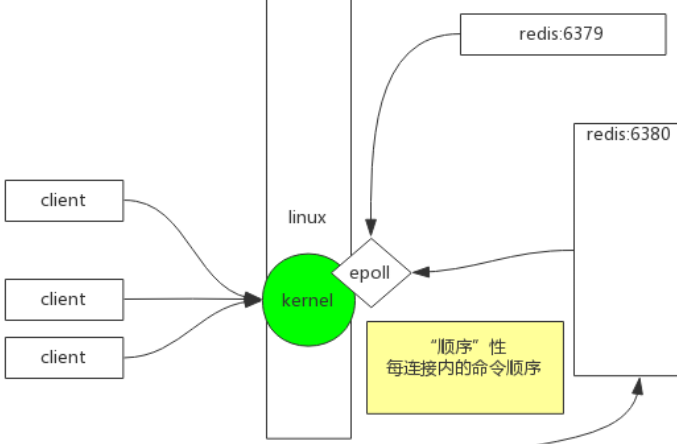


一个Linux可以安装多个redis实例（进程），根据port区分

**Redis是单进程，单线程，单实例，并发请求处理为什么这么快？**

秒级10w次操作，并发很多请求。

操作系统



插曲

|  |
| --- |
| 早期的计算机，Socket是BIO的    内核发展，同步非阻塞NIO  在用户空间，一个where(true) 死循环，单线程的，一直调用read() 函数，读取文件描述符fd8 fd9    **多路复用 NIO**  **存在问题：用户态内核态fd相关数据考来考去** |