容

一些公司 天使用 Elasticsearch 索引 索 PB 数据， 但我 中的大多数都起 于 模 的 目。即使我 立志成 下一个 Facebook，我 的 行 余 却也跟不上梦想的脚 。 我 需要 今日所需而 建，但也要允 我 可以 活而又快速地 行水平 展。

Elasticsearch 了可 展性而生。它可以良好地 行于 的 本 又或者一个 有数百 点的集群，同 用 体 基本相同。 由小 模集群 大 模集群的 程几乎完全自 化并且无痛。由大 模集群 超大 模集群需要一些 和 ，但 是相 地无痛。

当然 一切并不是魔法。Elasticsearch 也有它的局限性。如果 了解 些局限性并能 与之相 ，集群 容的 程将会是愉快的。 如果 Elasticsearch 理不当，那 将 于一个充 痛苦的世界。

Elasticsearch 的 置会伴 走 很 的一段路，但 了 它最大的效用， 需要考 数据是如何流 的系 的。 我 将 常 的数据流： 序数据（ 相 性，例如日志或社交 数据流），以及基于用 的数据（ 有很大的文 集但可以按用 或客 分）。

一章将 助 在遇到不愉快之前做出正 的 。

容的 元

在 [dynamic-indices]，我 介 了一个分片即一个 *Lucene* 索引 ，一个 Elasticsearch 索引即一系列分片的集合。 的 用程序与索引 行交互，Elasticsearch 助 将 求路由至相 的分片。

一个分片即 容的 元 。一个最小的索引 有一个分片。 可能已 完全 足 的需求了 — 个分片即可存 大量的数据 — 但 限制了 的可 展性。

想象一下我 的集群由一个 点 成，在集群内我 有一个索引， 个索引只含一个分片：

```
PUT /my_index
{
  "settings": {
    "number_of_shards":   1, ①
    "number_of_replicas": 0
  }
}
```

① 建一个 有 1 主分片 0 个副本分片的索引.

个 置 也 很小，但它 足我 当前的需求而且 行代 低。

> **NOTE** 当前我 只 主 分片。我 将在 副本分片 副本 分片。

在美好的一天，互 了我 ，一个 点再也承受不了我 的流量。 我 决定根据 一个只有一个分片的索引无 容因子 添加一个 点。 将会 生什 ？

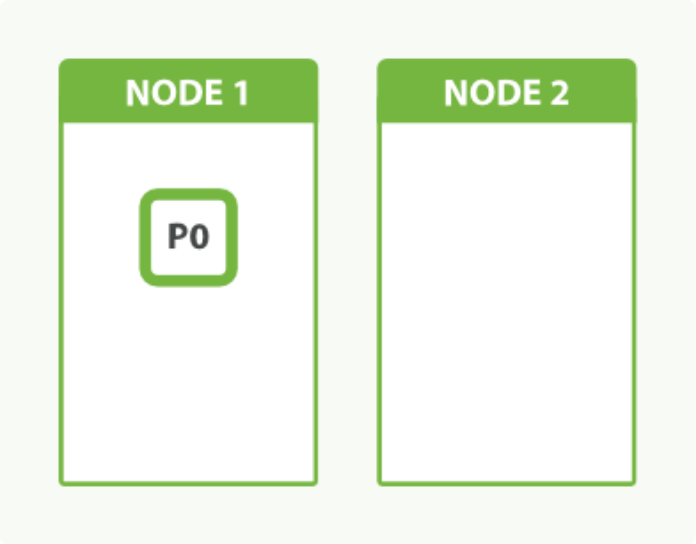

Figure 1. 一个只有一个分片的索引无 容因子

答案是：什 都不会 生。因 我 只有一个分片，已 没有什 可以放在第二个 点上的了。 我 不能 加索引的分片数因 它是 route documents to shards 算法中的重要元素：

```
shard = hash(routing) % number_of_primary_shards
```

我 当前的 只有一个就是将数据重新索引至一个 有更多分片的一个更大的索引，但 做将消耗的 是我 无法提供的。通 事先 ，我 可以使用 分配 的方式来完全避免 个 。

分片 分配

一个分片存在于 个 点，但一个 点可以持有多个分片。想象一下我 建 有 个主分片的索引而不是一个：

```
PUT /my_index
{
  "settings": {
    "number_of_shards":   2, ①
    "number_of_replicas": 0
  }
}
```

① 建 有 个主分片无副本分片的索引。

当只有一个 点 ， 个分片都将被分配至相同的 点。 从我 用程序的角度来看，一切都和之前一 作着。 用程序和索引 行通 ，而不是分片， 在 是只有一个索引。

，我 加入第二个 点，Elasticsearch 会自 将其中一个分片移 至第二个 点，如 一个 有 个分片的索引可以利用第二个 点 描 的那 ， 当重新分配完成后， 个分片都将接近至 倍于之前的 算能力。

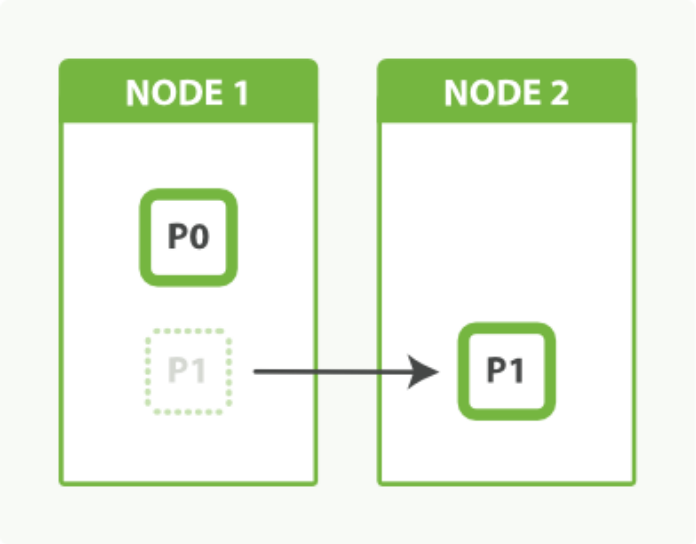

Figure 2. 一个 有 个分片的索引可以利用第二个 点

我 已 可以通 地将一个分片通 制到一个新的 点来加倍我 的 理能力。 最棒的是，我 零停机地做到了 一点。在分片移 程中，所有的索引搜索 求均在正常 行。

在 Elasticsearch 中新添加的索引 被指定了五个主分片。 意味着我最多可以将那个索引分散到五个 点上， 个 点一个分片。 它具有很高的 理能力， 未等 去思考一切就已 做到了！

分片分裂

用 常在 ， 什 Elasticsearch 不支持 分片分裂（shard-splitting）— 将 个分片分裂 个或更多部分的能力。 原因就是分片分裂是一个糟 的想法：

- 分裂一个分片几乎等于重新索引 的数据。它是一个比 将分片从一个 点 制到 一个 点更重量 的操作。
- 分裂是指数的。起初 有一个分片，然后分裂 个，然后四个，八个，十六个，等等。分裂并不会 好地把 的 理能力提升 50%。
- 分片分裂需要 有足 的能力支 一 索引的拷 。通常来 ，当 意 到 需要横向 展 ， 已 没有足 的剩余空 来做分裂了。

Elasticsearch 通 一 方式来支持分片分裂。 是可以把 的数据重新索引至一个 有 当分片个数的新索引（参 [reindex]）。 和移 分片比起来 依然是一个更加密集的操作，依然需要足 的剩余空 来完成，但至少 可以控制新索引的分片个数了。

海量分片

当新手 在了解 分片 分配 之后做的第一件事就是 自己 ：

> 我不知道 个索引将来会 得多大，并且 后我也不能更改索引的大小，所以 了保 起 ， 是 它 1000 个分片 ...
>
> — 一个新手的

一千个分片——当真？在　来　一千个　点　之前，　不　得　可能需要再三思考　的数据模型然后将它　重新索引　？

一个分片并不是没有代　的。　住：

- 一个分片的底　即　一个 Lucene 索引，会消耗一定文件句柄、内存、以及 CPU　　。
- 　一个搜索　求都需要命中索引中的　一个分片，如果　一个分片都　于不同的　点　好，但如果多个分片都需要在同一个　点上　争使用相同的　源就有些糟　了。
- 用于　算相　度的　　信息是基于分片的。如果有　多分片，　一个都只有很少的数据会　致很低的相　度。

TIP

　当的　分配是好的。但上千个分片就有些糟　。我　很　去定　分片是否　多了，　取决于它　的大小以及如何去使用它　。　　一百个分片但很少使用　好，　个分片但非常　繁地使用有可能就有点多了。　　控　的　点保　它　留有足　的空　　源来　理一些特殊情况。

横向　展　当分　段　行。　下一　段准　好足　的　源。　　只有当　　入到下一个　段，　才有　思考需要作出　些改　来　到　个　段。

容量

如果一个分片太少而　　1000　　个又太多，那　我　　知道我需要多少分片　？　　一般情况下是一个无法回答的　　。因　　在有太多相　的因素了：　使用的硬件、文　的大小和　　度、文　的索引分析方式、　行的　　型、　行的聚合以及　的数据模型等等。

幸　的是，在特定　景下　是一个容易回答的　　，尤其是　自己的　景：

1. 基于　准　用于生　　境的硬件　建一个　有　个　点的集群。
2. 　建一个和　准　用于生　　境相同配置和分析器的索引，但　它只有一个主分片无副本分片。
3. 索引　　的文　（或者尽可能接近　　）。
4. 　行　　的　　和聚合（或者尽可能接近　　）。

基本来　，　需要　制真　　境的使用方式并将它　全部　　到　个分片上直到它``挂掉。"　　上　挂掉的定　也取决于　：一些用　需要所有　　在 50　秒内返回；　一些　　于等上 5 秒　。

一旦　定　好了　个分片的容量，很容易就可以推算出整个索引的分片数。　　用　需要索引的数据　数加上一部分　期的　　，除以　个分片的容量，　果就是　需要的主分片个数。

TIP

容量　　不　当作　　的第一　。

先看看有没有　法　化　　Elasticsearch　　的使用方式。也　　有低效的　　，　少足　的内存，又或者　　了 swap？

我　　一些新手　于初始性能感到沮　，立即就着手　　回收又或者是　程数，而不是　理　　例如去掉通配符　　。

副本分片

目前 止我 只 主分片，但我 身 有 一个工具：副本分片。 副本分片的主要目的就是了故障 移，正如在 [distributed-cluster] 中 的：如果持有主分片的 点挂掉了，一个副本分片就会晋升 主分片的角色。

在索引写入 ，副本分片做着与主分片相同的工作。新文 首先被索引 主分片然后再同 到其它所有的副本分片。 加副本数并不会 加索引容量。

无 如何，副本分片可以服 于 求，如果 的索引也如常 的那 是偏向 使用的，那 可以通加副本的数目来提升 性能，但也要 此 加 外的硬件 源。

我 回到那个有着 个主分片索引的例子。我通 加第二个 点来提升索引容量。 加 外的点不会 助我 提升索引写入能力，但我 可以通 加副本数在搜索 利用 外的硬件：

```
PUT /my_index/_settings
{
  "number_of_replicas": 1
}
```

有 个主分片，加上 个主分片的一个副本， 共 予我 四个分片： 个 点一个，如 所示 一个有 个主分片一 副本的索引可以在四个 点中横向 展。

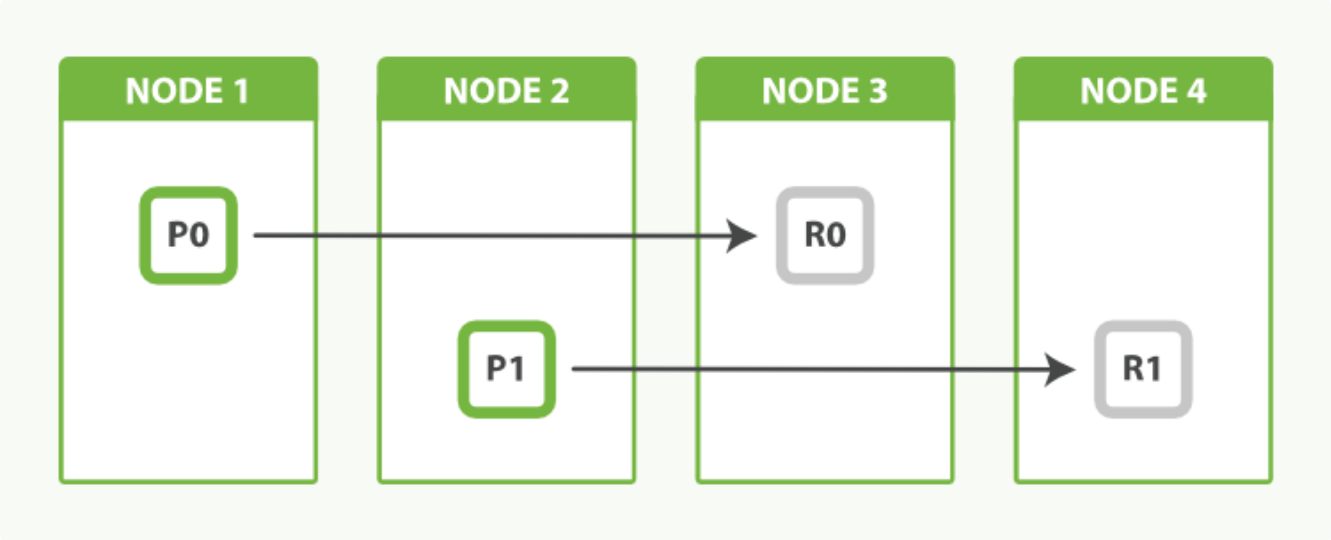

Figure 3. 一个 有 个主分片一 副本的索引可以在四个 点中横向 展

通 副本 行 均衡

搜索性能取决于最慢的 点的 ，所以 均衡所有 点的 是一个好想法。 如果我 只是 加一个 点而不是 个，最 我 会有 个 点各持有一个分片，而 一个持有 个分片做着 倍的工作。

我 可以通 整副本数量来平衡 些。通 分配 副本而不是一个，最 我 会 有六个分片， 好可以平均分 三个 点，如 所示 通 整副本数来均衡 点 ：

```
PUT /my_index/_settings
{
  "number_of_replicas": 2
}
```

作　　励，我　同　提升了我　的可用性。我　可以容忍　失　个　点而　然保持一　完整数据的拷　。

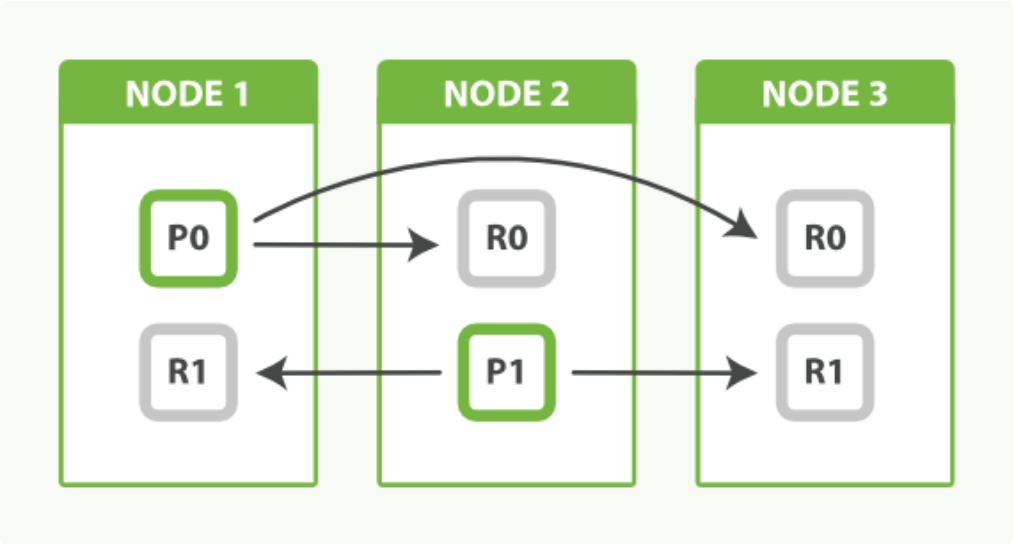

Figure 4. 通　　整副本数来均衡　点

NOTE	事　上　点　　3　　持有　个副本分片，然而没有主分片并不重要。副本分片与主分片做着相同的工作；它　只是扮演着略微不同的角色。没有必要　保主分片均　地分布在所有　点中。

多索引

最后，　住没有任何　　限制　的　用程序只使用一个索引。　　当我　　起一个搜索　求　，它被　　至索引中　个分片的一　拷　（一个主分片或一个副本分片），如果我　向多个索引　出同　的　求，会　生完全相同的事情——只不　会　及更多的分片。

TIP	搜索　1　个有着　50　个分片的索引与搜索　50　个　个都有　1　个分片的索引完全等　：搜索　求均命中 50 个分片。

当　需要在不停服　的情况下　加容量　，下面有一些有用的建　。相　于将数据　移到更大的索引中，可以　　做下面　些操作：

- 　建一个新的索引来存　新的数据。
- 同　搜索　个索引来　取新数据和旧数据。

　上，通　一点　先　　，添加一个新索引可以通　一　完全透明的方式完成，　的　用程序根本不会察到任何的改　。

在　　[index-aliases]，我　提到　使用索引　名来指向当前版本的索引。　　例来　，　　的索引命名 `tweets_v1` 而不是 `tweets` 。　的　用程序会与 `tweets`　行交互，但事　上它是一个指向 `tweets_v1` 的　名。　允　　将　名切　至一个更新版本的索引而保持服　　。

我 可以使用一个 似的技 通 加一个新索引来 展容量。 需要一点点 ，因 需要 个 名：一个用于搜索 一个用于索引数据：

```
PUT /tweets_1/_alias/tweets_search ①
PUT /tweets_1/_alias/tweets_index ①
```

① tweets_search 与 tweets_index 个 名都指向索引 tweets_1 。

新文 当索引至 tweets_index ，同 ，搜索 求 当 名 tweets_search 出。目前， 个名指向同一个索引。

当我 需要 外容量 ，我 可以 建一个名 tweets_2 的索引，并且像 更新 名：

```
POST /_aliases
{
  "actions": [
    { "add":    { "index": "tweets_2", "alias": "tweets_search" }}, ①
    { "remove": { "index": "tweets_1", "alias": "tweets_index"  }}, ②
    { "add":    { "index": "tweets_2", "alias": "tweets_index"  }}  ②
  ]
}
```

① 添加索引 tweets_2 到 名 tweets_search 。

② 将 名 tweets_index 由 tweets_1 切 至 tweets_2 。

一个搜索 求可以以多个索引 目 ，所以将搜索 名指向 tweets_1 以及 tweets_2 是完全有效的。然而，索引写入 求只能以 个索引 目 。因此，我 必 将索引写入的 名只指向新的索引。

TIP

一个文 GET 求，像一个索引写入 求那 ，只能以 个索引 目 。 致在通 ID 取文 的 景下有一点 。作 代替， 可以 tweets_1 以及 tweets_2 行一个 {ref}/query-dsl-ids-query.html[ids] 搜索 求， 或者 {ref}/docs-multi-get.html[multi-get] 求。

在服 行中使用多索引来 展索引容量 于一些使用 景有着特 的好 ，像我 将在下一 中 的基于 的数据例如日志或社交事件流。

基于 的数据

Elasticsearch 的常用案例之一便是日志 ， 它 在太常 了以至于 Elasticsearch 提供了一个集成的日志平台叫做 ELK stack— Elasticsearch，Logstash，以及 Kibana ——来 工作 得 。

Logstash 采集、解析日志并在将它 写入Elasticsearch之前格式化。 Elasticsearch 扮演了一个集中式的日志服 角色， Kibana 是一个 形化前端可以很容易地 以及可 化的 化。

搜索引 中大多数使用 景都是 慢相 定的文 集合。搜索 最相 的文 ，而不 心它是何 建的。

日志——以及其他基于　的数据流例如社交　活　——　上有很大不同。　索引中文　数量迅速　，通常随　加速。　文　几乎不会更新，基本以最近文　搜索目　。随着　推移，文　逐　失去　。

我　需要　整索引　使其能　工作于　基于　的数据流。

按　索引

如果我　此　型的文　建立一个超大索引，我　可能会很快耗尽存　空　。日志事件会不断的　来，不会停　也不会中断。　我　可以使用 `scroll`　和批量　除来　除旧的事件。但　方法　非常低效　。当　除一个文　，它只会被　被　除（参　[deletes-and-updates]）。在包含它的段被合并之前不会被物理　除。

替代方案是，我　使用一个　索引。　可以着手于一个按年的索引　(`logs_2014`)　或按月的索引(`logs_2014-10`)。　也　当　的　得十分繁忙　，　需要切　到一个按天的索引（`logs_2014-10-24`）。　除旧数据十分　：只需要　除旧的索引。

　方法有　的　点，允　在需要的　候　行　容。　不需要　先做任何　的决定。　天都是一个新的机会来　整　的索引　来　当前需求。　用相同的　到决定　个索引的大小上。起初也　需要的　是　周一个主分片。　一　子，也　需要　天五个主分片。　都不重要——任何　都可以　整到新的　境。

　名可以　助我　更加透明地在索引　切　。　当　建索引　，　可以将　`logs_current` 指向当前索引来接收新的日志事件，　当　索　，更新 `last_3_months` 来指向所有最近三个月的索引：

```
POST /_aliases
{
  "actions": [
    { "add":    { "alias": "logs_current",  "index": "logs_2014-10" }}, ①
    { "remove": { "alias": "logs_current",  "index": "logs_2014-09" }}, ①
    { "add":    { "alias": "last_3_months", "index": "logs_2014-10" }}, ②
    { "remove": { "alias": "last_3_months", "index": "logs_2014-07" }}  ②
  ]
}
```

① 将 `logs_current` 由九月切　至十月。

② 将十月添加到 `last_3_months` 并且　掉七月。

索引模板

Elasticsearch　不要求　在使用一个索引前　建它。　于日志　用，依　于自　建索引比手　建要更加方便。

Logstash 使用事件中的　来生成索引名。　天被索引至不同的索引中，因此一个 `@timestamp` `2014-10-01　00:00:01` 的事件将被　送至索引　`logstash-2014.10.01`　中。如果那个索引不存在，它将被自　建。

通常我　想要控制一些新建索引的　置（settings）和映射（mappings）。也　我　想要限制分片数　`1`，并且禁用 `_all` 域。索引模板可以用于控制何　置（settings）　当被　用于新　建的索引：

```
PUT /_template/my_logs ①
{
  "template": "logstash-*", ②
  "order":    1, ③
  "settings": {
    "number_of_shards": 1 ④
  },
  "mappings": {
    "_default_": { ⑤
      "_all": {
        "enabled": false
      }
    }
  },
  "aliases": {
    "last_3_months": {} ⑥
  }
}
```

① 建一个名 my_logs 的模板。

② 将 个模板 用于所有以 logstash- 起始的索引。

③ 个模板将会覆 的 logstash 模板，因 模板的 order 更低。

④ 限制主分片数量 1 。

⑤ 所有 型禁用 _all 域。

⑥ 添加 个索引至 last_3_months 名中。

个模板指定了所有名字以 logstash- 起始的索引的 置，不 它是手 是自 建的。如果我 明天的索引需要比今天更大的容量，我 可以更新 个索引以使用更多的分片。

个模板 将新建索引添加至了 last_3_months 名中，然而从那个 名中 除旧的索引 需要手 行。

数据 期

随着 推移，基于 数据的相 度逐 降低。 有可能我 会想要 看上周、上个月甚至上一年度 生了什 ，但是大多数情况，我 只 心当前 生的。

按 索引 来的一个好 是可以方便地 除旧数据：只需要 除那些 得不重要的索引就可以了。

```
DELETE /logs_2013*
```

除整个索引比 除 个文 要更加高效：Elasticsearch 只需要 除整个文件 。

但是 除索引是 手段。在我 决定完全 除它之前 有一些事情可以做来 助数据更加 雅地 期。

移旧索引

随着数据被 ，很有可能存在一个 点 索引——今日的索引。 所有新文 都会被加到那个索引，几乎所有 都以它 目 。那个索引 当使用 最好的硬件。

Elasticsearch 是如何得知 台是 最好的服 器 ？ 可以通 台服 器指定任意的 来告 它。例如， 可以像 一个 点：

```
./bin/elasticsearch --node.box_type strong
```

`box_type` 参数是完全随意的—— 可以将它随意命名只要 喜 ——但 可以用 些任意的 来告 Elasticsearch 将一个索引分配至何 。

我 可以通 按以下配置 建今日的索引来 保它被分配到我 最好的服 器上：

```
PUT /logs_2014-10-01
{
  "settings": {
    "index.routing.allocation.include.box_type" : "strong"
  }
}
```

昨日的索引不再需要我 最好的服 器了，我 可以通 更新索引 置将它移 到 `medium` 的 点上：

```
POST /logs_2014-09-30/_settings
{
  "index.routing.allocation.include.box_type" : "medium"
}
```

索引 化（Optimize）

昨日的索引不大可能会改 。 日志事件是静 的：已 生的 往不会再改 了。如果我 将 个分片合并至一个段（Segment），它会占用更少的 源更快地 。 我 可以通 [optimize-api]来做到。

分配在 `strong` 主机上的索引 行 化（Optimize）操作将会是一个糟 的想法， 因 化操作将消耗 点上大量 I/O 并 索引今日日志造成冲 。但是 `medium` 的 点没有做太多 似的工作，我 可以安全地在上面 行 化。

昨日的索引有可能 有副本分片。如果我 下 一个 化（Optimize） 求， 它会 化主分片和副本分片， 有些浪 。然而，我 可以 移除副本分片， 行 化，然后再恢 副本分片：

```
POST /logs_2014-09-30/_settings
{ "number_of_replicas": 0 }

POST /logs_2014-09-30/_optimize?max_num_segments=1

POST /logs_2014-09-30/_settings
{ "number_of_replicas": 1 }
```

当然，没有副本我 将面 磁 故障而 致 失数据的 。 可能想要先通 {ref}/modules-snapshots.html[snapshot-restore API] 数据。

旧索引

当索引 得更“老”，它 到 一个几乎不会再被 的 点。 我 可以在 个 段 除它 ，但也 想将它 留在 里以防万一有人在半年后 想要 它 。

些索引可以被 。它 会存在于集群中，但它 不会消耗磁 空 以外的 源。重新打 一个索引要比从 中恢 快得多。

在 之前， 得我 去刷写索引来 保没有事 残留在事 日志中。一个空白的事 日志会使得索引在重新打 恢 得更快：

```
POST /logs_2014-01-*/_flush ①
POST /logs_2014-01-*/_close ②
POST /logs_2014-01-*/_open ③
```

① 刷写（Flush）所有一月的索引来清空事 日志。

② 所有一月的索引.

③ 当 需要再次 它 ，使用 open API 来重新打 它 。

旧索引

最后，非常旧的索引可以通 {ref}/modules-snapshots.html[snapshot-restore API] 至 期存 例如共享磁 或者 Amazon S3，以防日后 可能需要 它 。 当存在 我 就可以将索引从集群中 除了。

基于用 的数据

通常来 ，用 使用 Elasticsearch 的原因是他 需要添加全文 索或者需要分析一个已 存在的 用。他 建一个索引来存 所有文 。公司里的其他人也逐 了 Elasticsearch 来的好 ，也想把他 的数据添加到 Elasticsearch 中去。

幸 的是，Elasticsearch 支持http://en.wikipedia.org/wiki/Multitenancy[多租]所以 个用 可以在相同的集群中 有自己的索引。 有人偶 会想要搜索所有用 的文 ， 情况可以通 搜索所有索引 ，但大多数情况下用 只 心它 自己的文 。

一些用 有着比其他人更多的文 ，一些用 可能有比其他人更多的搜索次数， 所以 指定

个索引主分片和副本分片数量能力的需要　很　合使用“一个用　一个索引”的模式。　似地，　繁忙的索引可以通　分片分配　指定到高配的　点。（参　移旧索引。）

TIP	不要　个索引都使用　的主分片数。想想看它需要存　多少数据。有可能　需要一个分片——再多的都只是浪　源。

大多数　Elasticsearch　的用　到　里就已　了。　的“一个用　一个索引”　大多数　景都可以　足了。

于例外的　景，　可能会　需要支持很大数量的用　，都是相似的需求。一个例子可能是　一个　有几千个　箱　的　提供搜索服　。　一些　可能有巨大的流量，但大多数都很小。将一个有着　个分片的索引用于一个小　模　已　是足　的了——一个分片可以承　很多个　的数据。

我　需要的是一　可以在用　共享　源的方法，　个用　他　有自己的索引　印象，而不在小用　上浪　源。

共享索引

我　可以　多的小　使用一个大的共享的索引，将　索引　一个字段并且将它用作一个　器：

```
PUT /forums
{
  "settings": {
    "number_of_shards": 10 ①
  },
  "mappings": {
    "post": {
      "properties": {
        "forum_id": { ②
          "type":  "string",
          "index": "not_analyzed"
        }
      }
    }
  }
}

PUT /forums/post/1
{
  "forum_id": "baking", ②
  "title":    "Easy recipe for ginger nuts",
  ...
}
```

① 建一个足　大的索引来存　数千个小　的数据。

② 个帖子都必　包含一个 `forum_id` 来　它属于　个　。

我　可以把　`forum_id`　用作一个　器来　个　行搜索。　个　器可以排除索引中　大部分的数据（属于其它　的数据），　存会保　快速的　：

```
GET /forums/post/_search
{
  "query": {
    "bool": {
      "must": {
        "match": {
          "title": "ginger nuts"
        }
      },
      "filter": {
        "term": {
          "forum_id": {
            "baking"
          }
        }
      }
    }
  }
}
```

个 法行得通，但我 可以做得更好。 来自于同一个 的帖子可以 地容 于 个分片，但它 在被打散到了 个索引的所有十个分片中。 意味着 个搜索 求都必 被 至所有十个分片的一个主分片或者副本分片。 如果能 保 所有来自于同一个 的所有帖子都被存于同一个分片可能会是个好想法。

在 [routing-value]，我 一个文 将通 使用如下公式来分配到一个指定分片：

```
shard = hash(routing) % number_of_primary_shards
```

routing 的 文 的 _id ，但我 可以覆 它并且提供我 自己自定 的路由 ，例如 forum_id 。所有有着相同 routing 的文 都将被存 于相同的分片：

```
PUT /forums/post/1?routing=baking ①
{
  "forum_id": "baking", ①
  "title":    "Easy recipe for ginger nuts",
  ...
}
```

① 将 forum_id 用于路由 保 所有来自相同 的帖子都存 于相同的分片。

当我 搜索一个指定 的帖子 ，我 可以 相同的 routing 来保 搜索 求 在存有我 文 的分片上 行：

```
GET /forums/post/_search?routing=baking ①
{
  "query": {
    "bool": {
      "must": {
        "match": {
          "title": "ginger nuts"
        }
      },
      "filter": {
        "term": { ②
          "forum_id": {
            "baking"
          }
        }
      }
    }
  }
}
```

① 求 在 于 routing 的分片上 行。

② 我 是需要 （Filter） ，因 一个分片可以存 来自于很多 的帖子。

多个 可以通 一个逗号分隔的列表来指定 routing ，然后将 个 forum_id 包含于一个 terms :

```
GET /forums/post/_search?routing=baking,cooking,recipes
{
  "query": {
    "bool": {
      "must": {
        "match": {
          "title": "ginger nuts"
        }
      },
      "filter": {
        "terms": {
          "forum_id": {
            [ "baking", "cooking", "recipes" ]
          }
        }
      }
    }
  }
}
```

方式从技 上来 比 高效，由于要 一个 或者索引 求指定 routing 和 terms 的看起来有一点的 拙。索引 名可以 解决 些！

利用　名　　一个用　一个索引

　了保持　　的　　，我　想　我　的　用　　我　　个用　都有一个　　的索引——或者按照我　的例子　个　　一个——尽管　　上我　用的是一个大的shared index。　因此，我　需要一　方式将 routing　及　　器　含于 forum_id 中。

索引　名可以　　做到　些。当　将一个　名与一个索引　　起来，　可以指定一个　　器和一个路由　：

```
PUT /forums/_alias/baking
{
  "routing": "baking",
  "filter": {
    "term": {
      "forum_id": "baking"
    }
  }
}
```

　在我　可以将　baking　　名　　一个　独的索引。索引至　baking　　名的文　会自　地　用我　自定的路由　：

```
PUT /baking/post/1 ①
{
  "forum_id": "baking", ①
  "title":    "Easy recipe for ginger nuts",
  ...
}
```

① 我　　是需要　　器指定 forumn_id 字段，但自定　路由　已　是　含的了。

　baking　　名上的　　只会在自定　路由　　的分片上　行，并且　果也自　按照我　指定的　　器行了　　：

```
GET /baking/post/_search
{
  "query": {
    "match": {
      "title": "ginger nuts"
    }
  }
}
```

当　多个　　行搜索　可以指定多个　名：

```
GET /baking,recipes/post/_search ①
{
  "query": {
    "match": {
      "title": "ginger nuts"
    }
  }
}
```

① 个 routing 的 都会 用，返回 果会匹配任意一个 器。

一个大的用

大 模流行 都是从小 起 的。 有一天我 会 我 共享索引中的一个分片要比其它分片更加繁忙，因 个分片中一个 的文 得更加 。 ，那个 需要属于它自己的索引。

我 用来提供一个用 一个索引的索引 名 了我 一个 的 移 方式。

第一 就是 那个 建一个新的索引，并 其分配合理的分片数，可以 足一定 期的数据 ：

```
PUT /baking_v1
{
  "settings": {
    "number_of_shards": 3
  }
}
```

第二 就是将共享的索引中的数据 移到 用的索引中，可以通 scroll 和bulk API来 。 当 移完成 ，可以更新索引 名指向那个新的索引：

```
POST /_aliases
{
  "actions": [
    { "remove": { "alias": "baking", "index": "forums"    }},
    { "add":    { "alias": "baking", "index": "baking_v1" }}
  ]
}
```

更新索引 名的操作是原子性的；就像在 一个 。 的 用程序 是在与 baking API 交互并且 于它已 指向一个 用的索引 无感知。

用的索引不再需要 器或者自定 的路由 了。我 可以依 于 Elasticsearch 使用的 _id 字段来做分区。

最后一 是从共享的索引中 除旧的文 ，可以通 搜索之前的路由 以及 ID 然后 行批量 除操作来 。

一个用　一个索引模型的　雅之　在于它允　　少　源消耗，保持快速的　　，同　　有在需要　零宕机　　容的能力。

容并不是无限的

　整个章　我　　了多　　Elasticsearch　可以做到的　容方式。　大多数的　容　　可以通　添加　点来解决。但有一　　源是有限制的，因此　得我　　真　待：集群状　。

集群状　是一　数据　　，　存下列集群　　的信息：

- 集群　　的　置
- 集群中的　点
- 索引以及它　的　置、映射、分析器、　　器（Warmers）和　名
- 与　个索引　　的分片以及它　分配到的　点

　可以通　如下　求　看当前的集群状　：

```
GET /_cluster/state
```

集群状　存在于集群中的　个　点，包括客　端　点。　　就是　什　任何一个　点都可以将　求直接　至被　求数据的　点——　个　点都知道　个文　　在　里。

只有主　点被允　更新集群状　。想象一下一个索引　求引入了一个之前未知的字段。持有那个文　的主分片所在的　点必　将新的映射　　到主　点上。　　主　点把更改合并到集群状中，然后向所有集群中的所有　点　布一个新的版本。

搜索　求　　使用　　集群状　，但它　不会　生修改。同　，文　　的　　改　　求也不会　集群状　　生修改。当然，除非它　引入了一个需要更新映射的新的字段了。　　的来　，集群状　是静的不会成　瓶　。

然而，需要　住的是相同的数据　　需要在　个　点的内存中保存，并且当它　生更改　必　　布到　一个　点。 集群状　的数据量越大，　个操作就会越久。

我　　最常　的集群状　　就是引入了太多的字段。一个用　可能会决定　　一个　IP　地址或者　个referer　　URL　　使用一个　独的字段。　　下面　个例子通　　一个唯一的　　referer使用一个不同的字段名来保持　　面　　量的　数：

```
POST /counters/pageview/home_page/_update
{
  "script": "ctx._source[referer]++",
  "params": {
    "referer": "http://www.foo.com/links?bar=baz"
  }
}
```

　方式十分的糟　！它会生成数百万个字段，　些都需要被存　在集群状　中。　　当　到一个新的

referer ，都有一个新的字段需要加入那个已 膨 的集群状 中， 都需要被 布到集群的 个点中去。

更好的方式是使用nested objects， 它使用一个字段作参数名—`referer`— 一个字段作 的 —`count` ：

```
    "counters": [
      { "referer": "http://www.foo.com/links?bar=baz",  "count": 2 },
      { "referer": "http://www.linkbait.com/article_3", "count": 10 },
      ...
    ]
```

嵌套的方式有可能会 加文 数量，但 Elasticsearch 生来就是 了解决它的。重要的是保持集群状小而敏捷。

最 ，不管 的初衷有多好， 可能会 集群 点数量、索引、映射 于一个集群来 是太大了。 此 ，可能有必要将 个 拆分到多个集群中了。感 {ref}/modules-tribe.html[tribe nodes]，甚至可以向多个集群 出搜索 求，就好像我 有一个巨大的集群那 。