集群内的原理

> **充章**
>
> 如前文所述， 是 充章 中第一篇介 Elasticsearch 在分布式 境中的 行原理。 在 个章 中，我 将会介 *cluster* 、 *node* 、 *shard* 等常用 ，Elastisearch 的 容机制， 以及如何 理硬件故障的内容。
>
> 然 个章 不是必 的— 完全可以在不 注分片、副本和失效切 等内容的情况下 期使用Elasticsearch-- 但是 将 助 了解 Elasticsearch 的内部工作 程。 可以先快速 章 ，将来有需要 再次 看。

ElasticSearch 的主旨是随 可用和按需 容。 而 容可以通 性能更 大（ 垂直 容，或 向 容 ）或者数量更多的服 器（ 水平 容，或 横向 容）来 。

然 Elasticsearch 可以 益于更 大的硬件 ，但是垂直 容是有 限的。 真正的 容能力是来自于水平 容— 集群添加更多的 点，并且将 力和 定性分散到 些 点中。

于大多数的数据 而言，通常需要 用程序 行非常大的改 ，才能利用上横向 容的新 源。 与之相反的是，ElastiSearch天生就是 分布式的 ，它知道如何通 管理多 点来提高 容性和可用性。 也意味着 的 用无需 注 个 。

本章将 述如何按需配置集群、 点和分片，并在硬件故障 保数据安全。

空集群

如果我 了一个 独的 点，里面不包含任何的数据和索引，那我 的集群看起来就是一个 包含空内容 点的集群。

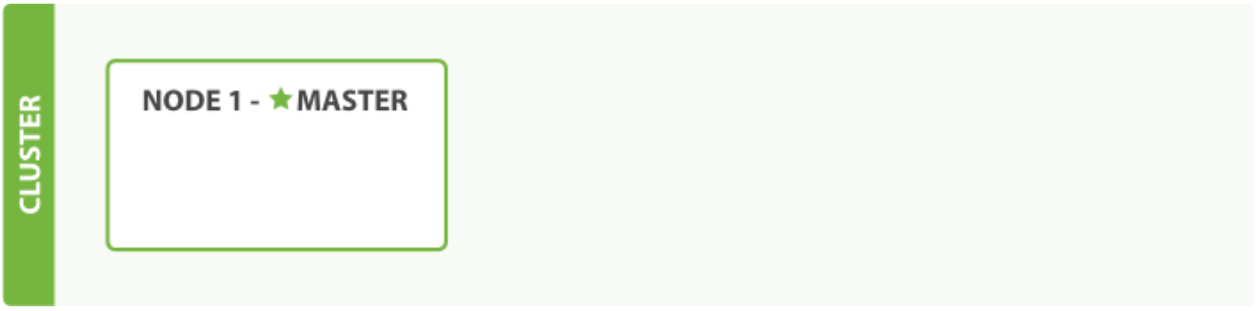

Figure 1. 包含空内容 点的集群

一个 行中的 Elasticsearch 例称 一个 点，而集群是由一个或者多个 有相同 `cluster.name` 配置的 点 成， 它 共同承担数据和 的 力。当有 点加入集群中或者从集群中移除 点 ，集群将会重新平均分布所有的数据。

当一个 点被 成 主 点 ， 它将 管理集群 内的所有 更，例如 加、 除索引，或者 加、 除 点等。 而主 点并不需要 及到文 的 更和搜索等操作，所以当集群只 有一个主 点的情况下，即使流量的 加它也不会成 瓶 。 任何 点都可以成 主 点。我

的示例集群就只有一个 点，所以它同 也成 了主 点。

作 用 ，我 可以将 求 送到 集群中的任何 点 ，包括主 点。 个 点都知道任意文 所 的位置，并且能 将我 的 求直接 到存 我 所需文 的 点。 无 我 将 求 送到 个 点，它都能 从各个包含我 所需文 的 点收集回数据，并将最 果返回給客 端。 Elasticsearch 一切的管理都是透明的。

集群健康

Elasticsearch 的集群 控信息中包含了 多的 数据，其中最 重要的一 就是 集群健康 ， 它在 `status` 字段中展示 `green` 、 `yellow` 或者 `red` 。

```
GET /_cluster/health
```

在一个不包含任何索引的空集群中，它将会有一个 似于如下所示的返回内容：

```
{
   "cluster_name":          "elasticsearch",
   "status":                "green", ①
   "timed_out":             false,
   "number_of_nodes":       1,
   "number_of_data_nodes":  1,
   "active_primary_shards": 0,
   "active_shards":         0,
   "relocating_shards":     0,
   "initializing_shards":   0,
   "unassigned_shards":     0
}
```

① `status` 字段是我 最 心的。

`status` 字段指示着当前集群在 体上是否工作正常。它的三 色含 如下：

`green`
所有的主分片和副本分片都正常 行。

`yellow`
所有的主分片都正常 行，但不是所有的副本分片都正常 行。

`red`
有主分片没能正常 行。

在本章 剩余的部分，我 将解 什 是 主 分片和 副本 分片，以及上面提到的 些 色的 意 。

添加索引

我 往 Elasticsearch 添加数据 需要用到 索引 —— 保存相 数据的地方。 索引 上是指向一个或者多个物理 分片 的 命名空 。

一个 分片 是一个底 的 工作 元 ，它 保存了全部数据中的一部分。 在分片内部机制中，我 将 介 分片是如何工作的，而 在我 只需知道一个分片是一个 Lucene 的 例，以及它本身就是一个完整的搜索引 。 我 的文 被存 和索引到分片内，但是 用程序是直接与索引而不是与分片 行交互。

Elasticsearch 是利用分片将数据分 到集群内各 的。分片是数据的容器，文 保存在分片内，分片又被分配到集群内的各个 点里。 当 的集群 模 大或者 小 ， Elasticsearch 会自 的在各 点中 移分片，使得数据 然均 分布在集群里。

一个分片可以是 主 分片或者 副本 分片。 索引内任意一个文 都 属于一个主分片，所以主分片的数目决定着索引能 保存的最大数据量。

NOTE	技 上来 ，一个主分片最大能 存 Integer.MAX_VALUE - 128 个文 ，但是 最大 需要参考 的使用 景：包括 使用的硬件， 文 的大小和 程度，索引和 文 的方式以及 期望的 。

一个副本分片只是一个主分片的拷 。副本分片作 硬件故障 保 数据不 失的冗余 ，并 搜索和返回文 等 操作提供服 。

在索引建立的 候就已 定了主分片数，但是副本分片数可以随 修改。

我 在包含一个空 点的集群内 建名 `blogs` 的索引。 索引在 情况下会被分配5个主分片， 但是 了演示目的，我 将分配3个主分片和一 副本（ 个主分片 有一个副本分片）：

```
PUT /blogs
{
   "settings" : {
      "number_of_shards" : 3,
      "number_of_replicas" : 1
   }
}
```

我 的集群 在是 有一个索引的 点集群。所有3个主分片都被分配在 `Node 1` 。

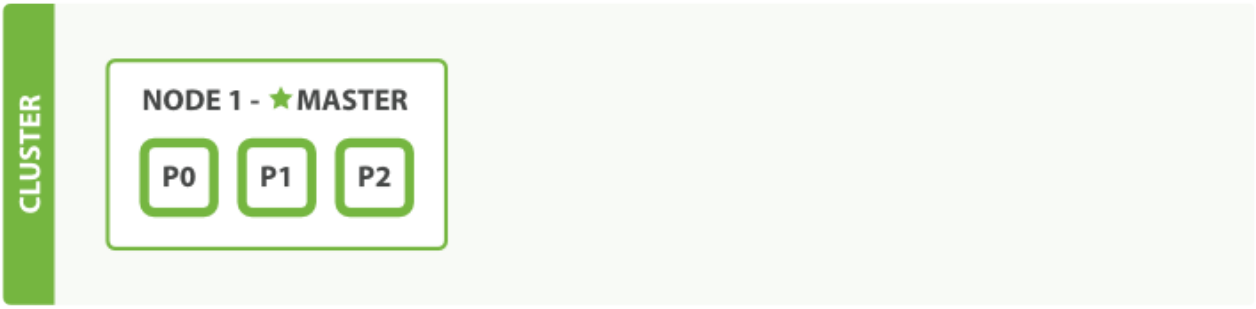

Figure 2. 有一个索引的 点集群

如果我 在 看集群健康，我 将看到如下内容：

```
{
  "cluster_name": "elasticsearch",
  "status": "yellow", ①
  "timed_out": false,
  "number_of_nodes": 1,
  "number_of_data_nodes": 1,
  "active_primary_shards": 3,
  "active_shards": 3,
  "relocating_shards": 0,
  "initializing_shards": 0,
  "unassigned_shards": 3, ②
  "delayed_unassigned_shards": 0,
  "number_of_pending_tasks": 0,
  "number_of_in_flight_fetch": 0,
  "task_max_waiting_in_queue_millis": 0,
  "active_shards_percent_as_number": 50
}
```

① 集群 `status` `yellow` 。

② 没有被分配到任何 点的副本数。

集群的健康状况 `yellow` 表示全部 主 分片都正常 行（集群可以正常服 所有 求），但是 副本分片没有全部 在正常状 。 上，所有3个副本分片都是 `unassigned` —— 它 都没有被分配到任何 点。 在同一个 点上既保存原始数据又保存副本是没有意 的，因 一旦失去了那个 点，我 也将失 点上的所有副本数据。

当前我 的集群是正常 行的，但是在硬件故障 有 失数据的 。

添加故障 移

当集群中只有一个 点在 行 ，意味着会有一个 点故障 ——没有冗余。 幸 的是，我 只需再 一个 点即可防止数据 失。

> **第二个 点**
>
> 了 第二个 点 后的情况， 可以在同一个目 内，完全依照 第一个 点的方式来 一个新 点（参考[running-elasticsearch]）。多个 点可以共享同一个目 。
>
> 当 在同一台机器上 了第二个 点 ，只要它和第一个 点有同 的 `cluster.name` 配置，它就会自 集群并加入到其中。 但是在不同机器上 点的 候， 了加入到同一集群， 需要配置一个可 接到的 播主机列表。 信息 看[unicast]

如果 了第二个 点，我 的集群将会如 有 个 点的集群——所有主分片和副本分片都已被分配所示 。

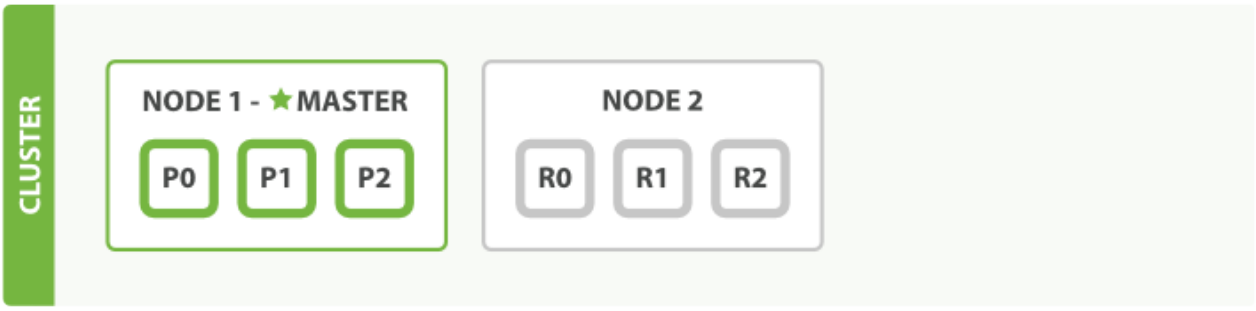

Figure 3. 有 个 点的集群——所有主分片和副本分片都已被分配

当第二个 点加入到集群后，3个 副本分片 将会分配到 个 点上—— 个主分片 一个副本分片。 意味着当集群内任何一个 点出 ，我 的数据都完好无 。

所有新近被索引的文 都将会保存在主分片上，然后被并行的 制到 的副本分片上。 就保 了我 既可以从主分片又可以从副本分片上 得文 。

`cluster-health` 在展示的状 `green` ， 表示所有6个分片（包括3个主分片和3个副本分片）都在正常 行。

```
{
  "cluster_name": "elasticsearch",
  "status": "green", ①
  "timed_out": false,
  "number_of_nodes": 2,
  "number_of_data_nodes": 2,
  "active_primary_shards": 3,
  "active_shards": 6,
  "relocating_shards": 0,
  "initializing_shards": 0,
  "unassigned_shards": 0,
  "delayed_unassigned_shards": 0,
  "number_of_pending_tasks": 0,
  "number_of_in_flight_fetch": 0,
  "task_max_waiting_in_queue_millis": 0,
  "active_shards_percent_as_number": 100
}
```

① 集群 `status` `green` 。

我 的集群 在不 是正常 行的，并且 于 始 可用 的状 。

水平 容

我 的正在 中的 用程序按需 容 ？ 当 了第三个 点，我 的集群将会看起来如有三个 点的集群—— 了分散 而 分片 行重新分配所示。

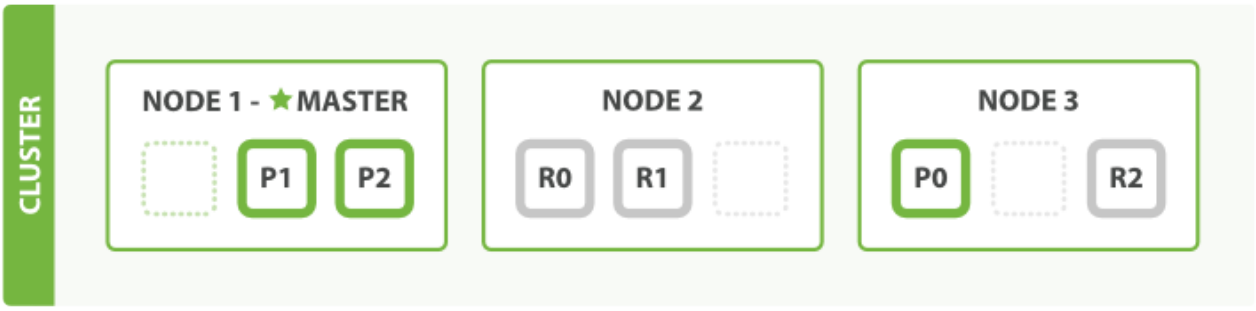

Figure 4. 有三个 点的集群—— 了分散 而 分片 行重新分配

Node 1 和 Node 2 上各有一个分片被 移到了新的 Node 3 点， 在 个 点上都 有2个分片，而不是之前的3个。 表示 个 点的硬件 源（CPU， RAM， I/O）将被更少的分片所共享， 个分片的性能将会得到提升。

分片是一个功能完整的搜索引 ，它 有使用一个 点上的所有 源的能力。 我 个 有6个分片（3个主分片和3个副本分片）的索引可以最大 容到6个 点， 个 点上存在一个分片，并且 个分片 有所在 点的全部 源。

更多的 容

但是如果我 想要 容超 6个 点 ？

主分片的数目在索引 建 就已 定了下来。 上， 个数目定 了 个索引能 存的最大数据量。（ 大小取决于 的数据、硬件和使用 景。） 但是， 操作——搜索和返回数据——可以同 被主分片 或 副本分片所 理，所以当 有越多的副本分片 ，也将 有越高的 吐量。

在 行中的集群上是可以 整副本分片数目的，我 可以按需伸 集群。 我 把副本数从 的 1 加到 2 ：

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

如将参数 number_of_replicas 大到 2所示， blogs 索引 在 有9个分片：3个主分片和6个副本分片。 意味着我 可以将集群 容到9个 点， 个 点上一个分片。相比原来3个 点 ，集群搜索性能可以提升 *3* 倍。

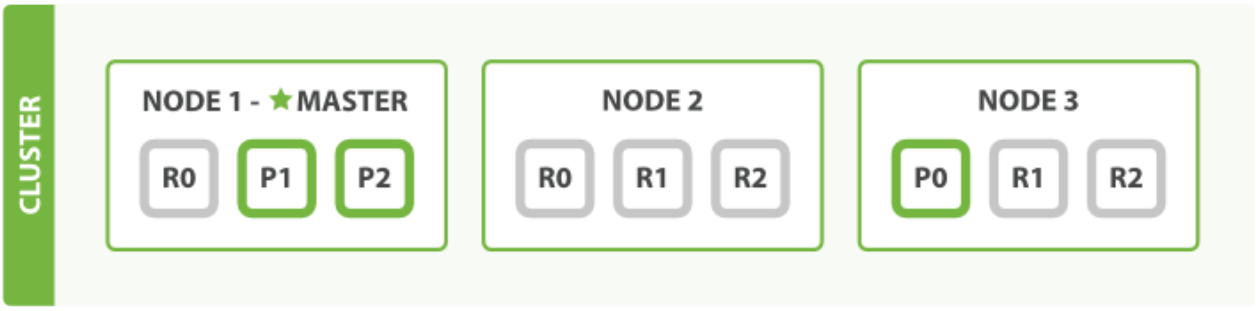

Figure 5. 将参数 number_of_replicas 大到 *2*

NOTE

当然，如果只是在相同 点数目的集群上 加更多的副本分片并不能提高性能，因 个分片从 点上 得的 源会 少。 需要 加更多的硬件 源来提升 吐量。

但是更多的副本分片数提高了数据冗余量：按照上面的 点配置，我 可以在失去2个 点的情况下不 失任何数据。

故障

我 之前 Elasticsearch 可以 点故障，接下来 我 下 个功能。 如果我 第一个 点， 集群的状 了一个 点后的集群

Figure 6. 了一个 点后的集群

我 的 点是一个主 点。而集群必 有一个主 点来保 正常工作，所以 生的第一件事情就是 一个新的主 点： Node 2 。

在我 Node 1 的同 也失去了主分片 1 和 2 ，并且在 失主分片的 候索引也不能正常工作。如果此 来 集群的状况，我 看到的状 将会 red ：不是所有主分片都在正常工作。

幸 的是，在其它 点上存在着 个主分片的完整副本， 所以新的主 点立即将 些分片在 Node 2 和 Node 3 上 的副本分片提升 主分片， 此 集群的状 将会 yellow 。 个提升主分片的 程是瞬 生的，如同按下一个 一般。

什 我 集群状 是 yellow 而不是 green ？ 然我 有所有的三个主分片，但是同 置了个主分片需要 2 副本分片，而此 只存在一 副本分片。 所以集群不能 green 的状 ，不 我不必 于担心：如果我 同 了 Node 2 ，我 的程序 依然 可以保持在不 任何数据的情况下行，因 Node 3 一个分片都保留着一 副本。

如果我 重新 Node 1 ，集群可以将 失的副本分片再次 行分配，那 集群的状 也将如将参数 number_of_replicas 大到 2所示。 如果 Node 1 依然 有着之前的分片，它将 去重用它 ，同 从主分片 制 生了修改的数据文件。

到目前 止， 分片如何使得 Elasticsearch 行水平 容以及数据保障等知 有了一定了解。接下来我 将 述 于分片生命周期的更多 。