第三版：Elasticsearch 28 道

你之前公司的 ElasticSearch 集群，一个 Node 一般会分配几个分片?

我们遵循官方建议，一个 Node 最好不要多于三个 shards.

ElasticSearch 是如何实现 Master 选举的?

ElasticSearch 的选举是 ZenDiscovery 模块负责的，主要包含 Ping（节点之间通过这个 RPC 来发现彼此）和 Unicast（单播模块包含一个主机列表以控制哪些节点需要 ping 通）这两部分；

对所有可以成为 master 的节点（node.master: true）根据 nodeId 字典排序，每次选举每个节点都把自己所知道节点排一次序，然后选出第一个（第 0 位）节点，暂且认为它是 master 节点。

如果对某个节点的投票数达到一定的值（可以成为 master 节点数 n/2+1）并且该节点自己也选举自己， 那这个节点就是 master。否则重新选举一直到满足上述条件。

你是如何做 ElasticSearch 写入调优的?

1）写入前副本数设置为 0；

2）写入前关闭 refresh_interval 设置为-1，禁用刷新机制；

3）写入过程中：采取 bulk 批量写入；

4） 写入后恢复副本数和刷新间隔；

5） 尽量使用自动生成的 id。

ElasticSearch 如何避免脑裂?

可以通过设置最少投票通过数量（discovery.zen.minimum_master_nodes）超过所有候选节点一半以上，来解决脑裂问题。

ElasticSearch 对于大数据量（上亿量级）的聚合如何实现?

ElasticSearch 提供的首个近似聚合是 cardinality 度量。它提供一个字段的基数，即该字段的 distinct 或者 unique 值的数目。它是基于 HLL 算法的。HLL 会先对我们的输入做哈希运算，然后根据哈希运算结果中的 bits 做概率估算从而得到基数。其特点是：

可配置的精度，用来控制内存的使用（更精确 = 更多内存），小的数据集精度是非常高的；我们可以通过配置参数来设置去重需要的固定内存使用量，无论数千还是数十亿的唯一值，内存使用量只与你配置的精确度相关 。

图片

ElasticSearch 主分片数量可以在后期更改吗？为什么?

不可以，因为根据路由算法 shard = hash(document_id) % (num_of_primary_shards)，当主分片数量变化时会影响数据被路由到哪个分片上。

ElasticSearch 如何监控集群状态?

Marvel 让你可以很简单的通过 Kibana 监控 Elasticsearch。你可以实时查看你的集群健康状态和性能，也可以分析过去的集群、索引和节点指标。

ElasticSearch 中的副本是什么?

一个索引被分解成碎片以便于分发和扩展，副本是分片的副本。一个节点是一个属于一个集群的 ElasticSearch 的运行实例，一个集群由一个或多个共享相同集群名称的节点组成。

ES 更新数据的执行流程?

(1) 将原来的 doc 标识为 deleted 状态，然后新写入一条数据。

(2) buffer 每 refresh 一次，就会产生一个 segmentfile，所以默认情况下是 1s 一个 segmentfile，segmentfile 会越来越多，此时会定期执行 merge。

(3) 每次 merge 时,会将多个 segmentfile 合并成一个，同时这里会将标识为 deleted 的 doc 给物理删除掉，然后将新的 segmentfile 写入磁盘，这里会写一个 commitpoint，标识所有新的 segmentfile，然后打开 segmentfile 供搜索使用，同时删除旧的 segmentfile。

ElasticSearch 中的分析器是什么?

在 ElasticSearch 中索引数据时，数据由为索引定义的 Analyzer 在内部进行转换。分析器由一个 Tokenizer 和零个或多个 TokenFilter 组成。编译器可以在一个或多个 CharFilter 之前，分析模块允许你在逻辑名称下注册分析器，然后可以在映射定义或某些 API 中引用它们。ElasticSearch 附带了许多可以随时使用的预建分析器。或者，你可以组合内置的字符过滤器，编译器和过滤器来创建自定义分析器。

客户端在和集群连接时，是如何选择特定的节点执行请求的?

TransportClient 利用 transport 模块远程连接一个 ElasticSearch 集群。它并不加入到集群中，只是简单的获得一个或者多个初始化的 transport 地址，并以轮询的方式与这些地址进行通信。

ElasticSearch 中的倒排索引是什么?

倒排索引是搜索引擎的核心，搜索引擎的主要目标是在查找发生搜索条件的文档时提供快速搜索。倒排索引是一种像数据结构一样的散列图，可将用户从单词导向文档或网页，它是搜索引擎的核心。其主要目标是快速搜索从数百万文件中查找数据。

什么是 ElasticSearch 脑裂?

一个正常 es 集群中只有一个主节点，主节点负责管理整个集群，集群的所有节点都会选择同一个节点作 为主节点所以无论访问那个节点都可以查看集群的状态信息。而脑裂问题的出现就是因为从节点在选择 主节点上出现分歧导致一个集群出现多个主节点从而使集群分裂，使得集群处于异常状态。

什么是 ElasticSearch 索引?

索引（名词）： 一个索引(index)就像是传统关系数据库中的数据库，它是相关文档存储的地方，index 的复数是 indices 或 indexes。

索引（动词）：「索引一个文档」表示把一个文档存储到索引（名词）里，以便它可以被检索或者查询。这很像 SQL 中的 INSERT 关键字，差别是，如果文档已经存在，新的文档将覆盖旧的文档。

详细描述一下 ElasticSearch 更新和删除文档的过程

删除和更新都是写操作，但是 ElasticSearch 中的文档是不可变的，因此不能被删除或者改动以展示其变更。

磁盘上的每个段都有一个相应的.del 文件。当删除请求发送后，文档并没有真的被删除，而是在.del 文件中被标记为删除。该文档依然能匹配查询，但是会在结果中被过滤掉。当段合并时，在.del 文件中被标记为删除的文档将不会被写入新段。

在新的文档被创建时，ElasticSearch 会为该文档指定一个版本号，当执行更新时，旧版本的文档在.del 文件中被标记为删除，新版本的文档被索引到一个新段。旧版本的文档依然能匹配查询，但是会在结果中会被过滤掉。

Master 节点和 候选 Master 节点有什么区别?

主节点负责集群相关的操作，例如创建或删除索引，跟踪哪些节点是集群的一部分，以及决定将哪些分片分配给哪些节点。

拥有稳定的主节点是衡量集群健康的重要标志。

而候选主节点是被选具备候选资格，可以被选为主节点的那些节点。

Elasticsearch 中的属性 enabled, index 和 store 的功能是什么?

enabled：false，启用的设置仅可应用于顶级映射定义和 Object 对象字段，导致 Elasticsearch 完全跳过对字段内容的解析。

仍然可以从 _source 字段中检索 JSON，但是无法搜索或以其他任何方式存储 JSON。

如果对非全局或者 Object 类型，设置 enable : false 会报错如下：

```
"type": "mapper_parsing_exception",
"reason": "Mapping definition for [user_id] has unsupported parameters: [enabled : false]"
```

index：false，索引选项控制是否对字段值建立索引。它接受 true 或 false，默认为 true。未索引的字段不可查询。

如果非要检索，报错如下：

```
"type": "search_phase_execution_exception",
"reason": "Cannot search on field [user_id] since it is not indexed."
```

store：

某些特殊场景下，如果你只想检索单个字段或几个字段的值，而不是整个_source的值，则可以使用源过滤来实现；

这个时候，store 就派上用场了。

Elasticsearch Analyzer 中的字符过滤器如何利用?

字符过滤器将原始文本作为字符流接收，并可以通过添加，删除或更改字符来转换字符流。

字符过滤分类如下：

HTML Strip Character Filter.

用途：删除 HTML 元素，如，**并解码 HTML 实体，如＆amp** 。

Mapping Character Filter

用途：替换指定的字符。

Pattern Replace Character Filter

用途：基于正则表达式替换指定的字符。

请解释有关 Elasticsearch 的 NRT?

从文档索引（写入）到可搜索到之间的延迟默认一秒钟，因此 Elasticsearch 是近实时（NRT）搜索平台。

也就是说：文档写入，最快一秒钟被索引到，不能再快了。

写入调优的时候，我们通常会动态调整：refresh_interval = 30s 或者更达值，以使得写入数据更晚一点时间被搜索到。

Elasticsearch 对于大数据量（上亿量级）的聚合如何实现?

Elasticsearch 提供的首个近似聚合是 cardinality 度量。它提供一个字段的基数，即该字段的 distinct 或者 unique 值的数目。它是基于 HLL 算法的。HLL 会先对我们的输入作哈希运算，然后根据哈希运算的结果中的 bits 做概率估算从而得到基数。

其特点是：可配置的精度，用来控制内存的使用（更精确 = 更多内存）；

小的数据集精度是非常高的；我们可以通过配置参数，来设置去重需要的固定内存使用量。无论数千还是数十亿的唯一值，内存使用量只与你配置的精确度相关。

详细描述一下 Elasticsearch 搜索的过程?

面试官：想了解 ES 搜索的底层原理，不再只关注业务层面了。

搜索拆解为“query then fetch” 两个阶段。

query 阶段的目的：定位到位置，但不取。

步骤拆解如下：

1、 假设一个索引数据有 5 主+1 副本 共 10 分片，一次请求会命中（主或者副本分片中）的一个。

2、 每个分片在本地进行查询，结果返回到本地有序的优先队列中。

3、 第 2）步骤的结果发送到协调节点，协调节点产生一个全局的排序列表。

fetch 阶段的目的：取数据。

路由节点获取所有文档，返回给客户端。

详细描述一下 Elasticsearch 更新和删除文档的过程。

1、 删除和更新也都是写操作，但是 Elasticsearch 中的文档是不可变的，因此不能被删除或者改动以展示其变更；

2、 磁盘上的每个段都有一个相应的.del 文件。当删除请求发送后，文档并没有真的被删除，而是在.del 文件中被标记为删除。该文档依然能匹配查询，但是会在结果中被过滤掉。当段合并时，在.del 文件中被标记为删除的文档将不会被写入新段。

3、 在新的文档被创建时，Elasticsearch 会为该文档指定一个版本号，当执行更新时，旧版本的文档在.del 文件中被标记为删除，新版本的文档被索引到一个新段。旧版本的文档依然能匹配查询，但是会在结果中被过滤掉。

详细描述一下 Elasticsearch 索引文档的过程。

协调节点默认使用文档 ID 参与计算（也支持通过 routing），以便为路由提供合适的分片。

shard = hash(document_id) % (num_of_primary_shards)

1、 当分片所在的节点接收到来自协调节点的请求后，会将请求写入到 MemoryBuffer，然后定时（默认是每隔 1 秒）写入到 Filesystem Cache，这个从 MomeryBuffer 到 Filesystem Cache 的过程就叫做 refresh；

2、 当然在某些情况下，存在 Momery Buffer 和 Filesystem Cache 的数据可能会丢失，ES 是通过 translog 的机制来保证数据的可靠性的。其实现机制是接收到请求后，同时也会写入到 translog 中，当 Filesystem cache 中的数据写入到磁盘中时，才会清除掉，这个过程叫做 flush；

3、 在 flush 过程中，内存中的缓冲将被清除，内容被写入一个新段，段的 fsync 将创建一个新的提交点，并将内容刷新到磁盘，旧的 translog 将被删除并开始一个新的 translog。

4、 flush 触发的时机是定时触发（默认 30 分钟）或者 translog 变得太大（默认为 512M）时；

Elasticsearch 是如何实现 master 选举的?

面试官：想了解 ES 集群的底层原理，不再只关注业务层面了。

前置前提：

1、 只有候选主节点（master：true）的节点才能成为主节点。

1、 最小主节点数（min_master_nodes）的目的是防止脑裂。

这个我看了各种网上分析的版本和源码分析的书籍，云里雾里。
核对了一下代码，核心入口为 findMaster，选择主节点成功返回对应 Master，否则返回 null。选举流程大致描述如下：
第一步：确认候选主节点数达标，elasticsearch.yml 设置的值 discovery.zen.minimum_master_nodes；
第二步：比较：先判定是否具备 master 资格，具备候选主节点资格的优先返回；若两节点都为候选主节点，则 id 小的值会主节点。注意这里的 id 为 string 类型。

题外话：获取节点 id 的方法。

1、 GET /_cat/nodes?v&h=ip,port,heapPercent,heapMax,id,name

2、 ip　　port heapPercent heapMax id　name

3、 127.0.0.1 9300　　39　1.9gb Hk9w Hk9wFwU

补充：

1、 Elasticsearch 的选主是 ZenDiscovery 模块负责的，主要包含 Ping（节点之间通过这个 RPC 来发现彼此）和 Unicast（单播模块包含一个主机列表以控制哪些节点需要 ping 通）这两部分；

2、 对所有可以成为 master 的节点（node.master: true）根据 nodeId 字典排序，每次选举每个节点都把自己所知道节点排一次序，然后选出第一个（第 0 位）节点，暂且认为它是 master 节点。

3、 如果对某个节点的投票数达到一定的值（可以成为 master 节点数 n/2+1）并且该节点自己也选举自己，那这个节点就是 master。否则重新选举一直到满足上述条件。

4、 master 节点的职责主要包括集群、节点和索引的管理，不负责文档级别的管理；data 节点可以关闭 http 功能。