Elasticsearch 索引设计实战指南

Part 1

索引设计的重要性

“原则七：设计优先”

——美团写给工程师的十条精进原则

* 好的索引设计在整个集群规划中占据举足轻重的作用，索引的设计直接影响集群设计的好坏和复杂度。
* 好的索引设计应该是充分结合业务场景的时间维度和空间维度，结合业务场景充分考量增、删、改、查等全维度设计的。
* 好的索引设计是完全基于“设计先行，编码在后”的原则，前期会花很长时间，为的是后期工作更加顺畅，避免不必要的返工。

1、PB 级别的大索引如何设计？

1.0 普通数据的索引

如果不考虑增量数据，基本上普通索引就能够满足性能要求。

1.1 大索引的缺陷

如果每天亿万+的实时增量数据，那么需要考虑以下问题

1.1.1 存储大小限制维度

单个分片（Shard）实际是 Lucene 的索引，单分片能存储的最大文档数是：2,147,483,519 (= Integer.MAX\_VALUE - 128)。如下命令能查看全部索引的分隔分片的文档大小：

GET \_cat/shards

app\_index 2 p STARTED 9443 2.8mb 127.0.0.1 Hk9wFwU

app\_index 2 r UNASSIGNED

app\_index 3 p STARTED 9462 2.7mb 127.0.0.1 Hk9wFwU

app\_index 3 r UNASSIGNED

app\_index 4 p STARTED 9520 3.5mb 127.0.0.1 Hk9wFwU

app\_index 4 r UNASSIGNED

app\_index 1 p STARTED 9453 2.4mb 127.0.0.1 Hk9wFwU

app\_index 1 r UNASSIGNED

app\_index 0 p STARTED 9365 2.3mb 127.0.0.1 Hk9wFwU

app\_index 0 r UNASSIGNED

1.1.2 性能维度

一个索引很大的话，数据写入和查询性能都会变差。

对于日志文件：分日期建立索引

1.1.3 风险维度

一旦一个大索引出现故障，相关的数据都会受到影响。而分成滚动索引的话，相当于做了物理隔离。

1.2 PB 级索引设计实现

大索引设计建议：使用模板+Rollover+Curator动态创建索引。

1.2.1 使用模板统一配置索引

目的：统一管理索引，相关索引字段完全一致。

1.2.2 使用 Rollver 增量管理索引

目的：按照日期、文档数、文档存储大小三个维度进行更新索引。使用举例：

POST /logs\_write/\_rollover

{

"conditions": {

"max\_age": "7d",

"max\_docs": 1000,

"max\_size": "5gb"

}

}

1.2.3 索引增量更新原理

在索引模板设计阶段，模板定义一个全局别名：用途是全局检索，如图所示的别名：index\_all。

每次更新到新的索引后，新索引指向一个用于实时新数据写入的别名，如图所示的别名：index\_latest。同时将旧索引的别名 index\_latest 移除。

POST /\_aliases

{

"actions" : [

{ "remove" : { "index" : "index\_2019-01-01-000001", "alias" : "index\_latest" } },

{ "add" : { "index" : "index\_2019-01-02-000002", "alias" : "index\_latest" } }

]

}

1.2.4 使用 curator 高效清理历史数据

目的：按照日期定期删除、归档历史数据。

curator 工具通过简单的配置文件结合定义任务 crontab 一键实现。

注意：7.X高版本借助iLM实现更为简单。

举例，一键删除 30 天前的历史数据：

[root@localhost .curator]# cat action.yml

actions:

1:

action: delete\_indices

description: >-

Delete indices older than 30 days (based on index name), for logstash-

prefixed indices. Ignore the error if the filter does not result in an

actionable list of indices (ignore\_empty\_list) and exit cleanly.

options:

ignore\_empty\_list: True

disable\_action: False

filters:

- filtertype: pattern

kind: prefix

value: logs\_

- filtertype: age

source: name

direction: older

timestring: '%Y.%m.%d'

unit: days

unit\_count: 30

2、分片数和副本数如何设计？

2.1 Shards & Replicas（分片/副本）

* 分片：分片本身都是一个功能齐全且独立的“索引”，可以托管在集群中的任何节点上。

数据切分分片的主要目的：

（1）水平分割/缩放内容量 。

（2）跨分片（可能在多个节点上）分布和并行化操作，提高性能/吞吐量。

注意：分片一旦创建，不可以修改大小。

* 副本：它在分片/节点出现故障时提供高可用性。

副本的好处：因为可以在所有副本上并行执行搜索——因此扩展了搜索量/吞吐量。

注意：副本分片与主分片存储在集群中不同的节点。副本的大小可以通过：number\_of\_replicas动态修改。

2.2 分片和副本实战中设计

2.2.1 索引设置多少分片？

Shard 大小官方推荐值为 20-40GB，在这个范围内查询和写入性能较好。

评估具体场景的分片数

* 第一步：预估一下数据量的规模。一共要存储多久的数据，每天新增多少数据？两者的乘积就是总数据量。
* 第二步：预估分多少个索引存储。索引的划分可以根据业务需要。
* 第三步：考虑和衡量可扩展性，预估需要搭建几台机器的集群。存储主要看磁盘空间，假设每台机器2TB，可用：2TB0.85(磁盘实际利用率）0.85(ES 警戒水位线）。
* 第四步：单分片的大小建议最大设置为 30GB。此处如果是增量索引，可以结合大索引的设计部分的实现一起规划。

前三步能得出一个索引的大小。分片数考虑维度：

* 1）分片数 = 索引大小/分片大小经验值 30GB 。
* 2）分片数建议和节点数一致。设计的时候1）、2）两者权衡考虑+rollover 动态更新索引结合。

2.2.2 索引设置多少副本？

结合集群的规模，对于集群数据节点 >=2 的场景：建议副本至少设置为 1。

3、Mapping(映射) 如何设计？

3.1 Mapping 认知

[Mapping](http://doc.codingdict.com/elasticsearch/287/#mapping) 是定义文档及其包含的字段的存储和索引方式的过程。例如，使用映射来定义：

* 应将哪些字符串字段定义为全文检索字段；
* 哪些字段包含数字，日期或地理位置；
* 定义日期值的格式（时间戳还是日期类型等）；
* 用于控制动态添加字段的映射的自定义规则。

3.2 设计 Mapping 的注意事项

ES 支持

增加字段 /新增字段

ES 不支持

直接删除字段

直接修改字段

直接修改字段类型

3.3 Mapping 字段的设置流程

索引分为静态 Mapping（自定义字段）+动态 Mapping（ES 自动根据导入数据适配）。

实战业务场景建议：选用静态 Mapping，根据业务类型自己定义字段类型。

设置字段的时候，务必过一下如下图示的流程。

核心参数含义

3.4 Mapping 建议结合模板定义

索引 Templates——索引模板允许您定义在创建新索引时自动应用的模板。模板包括settings和Mappings以及控制是否应将模板应用于新索引。

注意：模板仅在索引创建时应用。更改模板不会对现有索引产生影响

核心配置

* index.numberofreplicas 每个主分片具有的副本数。默认为 1（7.X 版本，低于 7.X 为 5）。
* index.maxresultwindow 深度分页 rom + size 的最大值—— 默认为 10000。
* index.refresh\_interval 默认 1s：代表最快 1s 搜索可见；实战业务如果对实时性要求不高，建议设置为 30s 或者更高。根据需要可以设置为 -1，提高写入性能；

3.5 包含 Mapping 的 template 设计万能模板

PUT \_template/test\_template

{

"index\_patterns": [

"test\_index\_\*",

"test\_\*"

],

"settings": {

"number\_of\_shards": 1,

"number\_of\_replicas": 1,

"max\_result\_window": 100000,

"refresh\_interval": "30s"

},

"mappings": {

"properties": {

"id": {

"type": "long"

},

"title": {

"type": "keyword"

},

"content": {

"analyzer": "ik\_max\_word",

"type": "text",

"fields": {

"keyword": {

"ignore\_above": 256,

"type": "keyword"

}

}

},

"available": {

"type": "boolean"

},

"review": {

"type": "nested",

"properties": {

"nickname": {

"type": "text"

},

"text": {

"type": "text"

},

"stars": {

"type": "integer"

}

}

},

"publish\_time": {

"type": "date",

"format": "yyyy-MM-dd HH:mm:ss||yyyy-MM-dd||epoch\_millis"

},

"expected\_attendees": {

"type": "integer\_range"

},

"ip\_addr": {

"type": "ip"

},

"suggest": {

"type": "completion"

}

}

}

}

附录

美团写给工程师的十条精进原则

原则一：Owner意识

认真负责是工作的底线。

积极主动是“Owner意识”更高一级的要求。

原则二：时间观念

工作安排要有计划性。

工作安排要分清楚主次。

原则三：以终为始

先想清楚目标，然后努力实现

根据问题设定目标，再进行优化。

带着目标去学习

原则四：闭环思维

即时反馈闭环

沟通要有结论，通知要有反馈，ToDo要有验收。

原则五：保持敬畏

尽快学习团队既有的规范，并且让自己与团队保持一致

让规范与约定与时俱进

原则六：事不过二

所有的评审与问题讨论，不要超过两次

所有的评审最多两次

同样的错误不能犯第二次

原则七：设计优先

架构设计，并不仅仅关系到系统的质量，还关乎团队的效能问题

原则八：P/PC平衡

产出与产能必须平衡，才能达到真正的高效能

原则九：善于提问

原则十：空杯心态

时刻进行自我检视与反省

附录

**Rollover Index/**滚动索引

当你判断现有索引太大或太旧时，可以使用滚动索引API将别名滚动到新的索引。

方式一：直接指定别名

方式二：指定条件

# 创建索引 logs-0000001 别名为 logs\_write

curl -XPUT 'localhost:9200/logs-000001 ?pretty' -d'

{

"aliases": {

"logs\_write": {}

}

}’

#如果logs\_write 指向的索引是在7天以前创建的，或者包含1000个以上的文档，则会创建 logs-000002索引，并更新logs\_write别名以指向logs-000002. | Add > 1000 documents to logs-000001

curl -XPOST 'localhost:9200/logs\_write/\_rollover ?pretty' -d'

{

"conditions": {

"max\_age": "7d",

"max\_docs": 1000

}

}’

上述可能会返回如下的响应:

{

"acknowledged": true,

"shards\_acknowledged": true,

"old\_index": "logs-000001",

"new\_index": "logs-000002",

"rolled\_over": true, //index 是否被滚动.

"dry\_run": false, // 干运行模式：即只检查请求条件，不执行实际滚动

"conditions": {

"[max\_age: 7d]": false,

"[max\_docs: 1000]": true

}

}

**Using date math with the rolllover API/**使用滚动**API**的日期计算

使用[日期计算](https://www.elastic.co/guide/en/elasticsearch/reference/current/date-math-index-names.html): 根据索引滚动的日期来命名滚动索引是有用的技术，例如  logstash-2016.02.03.。 滚动API支持日期，但要求索引名称以一个破折号后跟一个数字，例如 logstash-2016.02.03-1，每次索引滚动时都会增加。 例如

*# PUT /<logs-{now/d}-1> with URI encoding:*

curl -XPUT 'localhost:9200/%3Clogs-%7Bnow%2Fd%7D-1%3E ?pretty' -d'

{

"aliases": {

"logs\_write": {}

}

}'

curl -XPUT 'localhost:9200/logs\_write/log/1?pretty' -d'

{

"message": "a dummy log"

}'

*# Wait for a day to pass*

curl -XPOST 'localhost:9200/logs\_write/\_rollover ?pretty' -d'

{

"conditions": {

"max\_docs": "1"

}

}'

|  | 创建当日的索引 logs-2016.10.31-1 | |  | 当日索引滚动, 立即生成如. logs-2016.10.31-000002 , 或者 logs-2016.11.01-000002 24小时后 |

然后可以按照 [日期数学文档](https://www.elastic.co/guide/en/elasticsearch/reference/current/date-math-index-names.html) 中的描述来引用这些索引。 例如，要搜索过去三天创建的索引，您可以执行以下操作：

*# GET /<logs-{now/d}-\*>,<logs-{now/d-1d}-\*>,<logs-{now/d-2d}-\*>/\_search*

curl -XGET 'localhost:9200/%3Clogs-%7Bnow%2Fd%7D-\*%3E%2C%3Clogs-%7Bnow%2Fd-1d%7D-\*%3E%2C%3Clogs-%7Bnow%2Fd-2d%7D-\*%3E/\_search?pretty'

附录

[Curator](https://www.elastic.co/guide/en/elasticsearch/client/curator/index.html)

[Curator](https://www.elastic.co/guide/en/elasticsearch/client/curator/index.html) 是elasticsearch 官方的一个索引管理工具，可以通过配置文件的方式帮助我们对指定的一批索引进行创建/删除、打开/关闭、快照/恢复等管理操作。

安装

首先，Curator是基于python实现的，我们可以直接通过pip来安装，这种方式最简单。

pip install elasticsearch-curator

基本配置

接下来，需要为 Curator 配置es连接:

# ~/.curator/curator.yml

client:

hosts:

- 127.0.0.1

port: 9200

logging:

loglevel: INFO

动作配置

然后需要配置我们需要执行的动作，每个动作会按顺序执行：

# /etc/curator/actions/maintain\_log.yml

actions:

1:

#创建第二天的索引

action: create\_index

description: "create new time-based index for log-\*"

options:

name: '<log-{now/d+1d}>'

2:

#删除3天前的索引

action: delete\_indices

description: "delete outdated indices for log-\*"

filters:

- filtertype: pattern

kind: prefix

value: log

- filtertype: age

source: name

direction: older

timestring: '%Y.%m.%d'

unit: days

unit\_count: 3

**action** 定义了需要执行的动作，curator支持十多种动作，可以在[官方文档](https://www.elastic.co/guide/en/elasticsearch/client/curator/current/actions.html)查看完整的动作列表。

**options** 定义了执行动作所需的参数，不同动作的参数也不尽相同，具体文档中都有写明。

**filters** 定义了动作的执行对象，通过设置filter，可以过滤出我们需要操作的索引。同一个action下的filter之间是且的关系。比如在上面的定义中，delete\_indices下定义了两个filters：

* 模式匹配：匹配前缀为log的索引
* “年龄”匹配：根据索引名中“%Y.%m.%d”时间格式，过滤出3天以前的索引

curator支持十多种filter，可以在[官方文档](https://www.elastic.co/guide/en/elasticsearch/client/curator/current/filters.html)查看完整列表。

执行

最后，我们通过curator命令行工具来执行：

curator --config /etc/curator/curator.yml /etc/curator/actions/maintain\_log.yml

参考

<https://cloud.tencent.com/developer/article/1587375>

<https://tech.meituan.com/2018/08/16/10-principles-for-engineers.html>

<http://doc.codingdict.com/elasticsearch/191/>

<https://elasticsearch.cn/article/779>

<http://doc.codingdict.com/elasticsearch/1/>

<http://doc.codingdict.com/elasticsearch/287/#mapping>