# 第二章 Elasticsearch高级

## Elasticsearch架构

### Elasticsearch的节点类型

在Elasticsearch有两类节点，一类是Master，一类是DataNode。

#### Master节点

在Elasticsearch启动时，会选举出来一个Master节点。当某个节点启动后，然后使用Zen Discovery机制找到集群中的其他节点，并建立连接。

discovery.seed\_hosts: ["node1.itcast.cn", "node2.itcast.cn", "node3.itcast.cn"]

并从候选主节点中选举出一个主节点。

cluster.initial\_master\_nodes: ["node1.itcast.cn", "node2.itcast.cn"]

Master节点主要负责：

* 管理索引（创建索引、删除索引）、分配分片
* 维护元数据
* 管理集群节点状态
* 不负责数据写入和查询，比较轻量级

一个Elasticsearch集群中，只有一个Master节点。在生产环境中，内存可以相对小一点，但机器要稳定。

#### DataNode节点

在Elasticsearch集群中，会有N个DataNode节点。DataNode节点主要负责：

* 数据写入、数据检索，大部分Elasticsearch的压力都在DataNode节点上
* 在生产环境中，内存最好配置大一些

### 分片（Shard）

Elasticsearch是一个分布式的搜索引擎，索引的数据也是分成若干部分，分布在不同的服务器节点中。而分布在不同服务器节点中的索引数据，就是分片（Shard）。Elasticsearch会自动管理分片，在必要的时候会将分片数据进行负载均衡分配。

* 一个索引（index）由多个shard（分片）组成，而分片是分布在不同的服务器上的

PS：

* 可以把分片理解为Kafka的分区，一个topic由多个partition组成
* Elasticsearch中的索引由一个个的分片组成

### 副本

为了对Elasticsearch的分片进行容错，假设某个节点不可用，会导致整个索引库都将不可用。所以，需要对分片进行副本容错。每一个分片都会有对应的副本。在Elasticsearch中，默认创建的索引为1个分片、每个分片有1个主分片和1个副本分片。

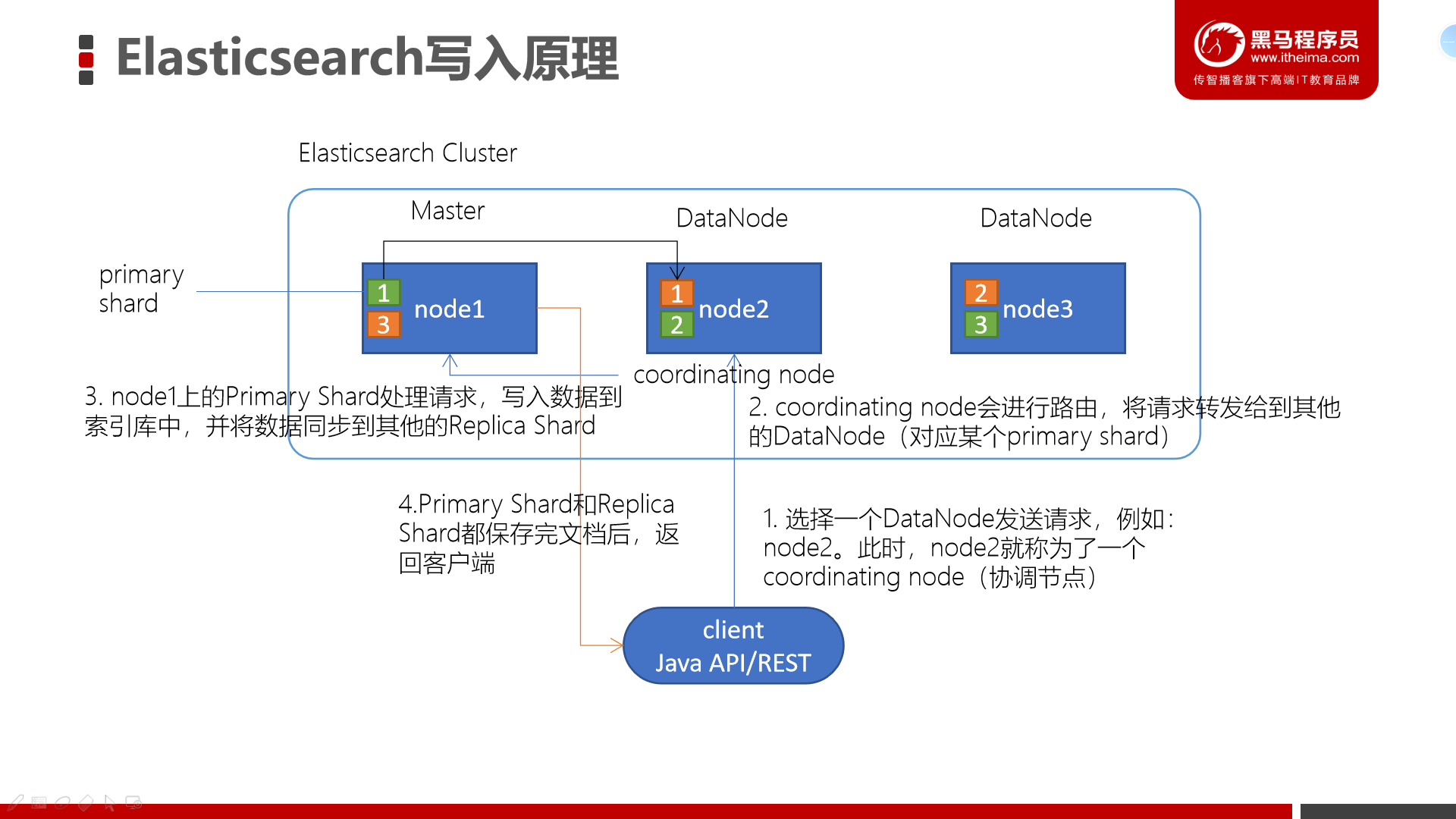
* 每个分片都会有一个Primary Shard（主分片），也会有若干个Replica Shard（副本分片）
* 副本的概念也与Kafka的副本类似。Kafka中分区有leader、也会有若干个Replica
* Primary Shard和Replica Shard不在同一个节点上

### 指定分片、副本数量

|  |
| --- |
| // 创建指定分片数量、副本数量的索引  PUT /job\_idx\_shard  {      "mappings": {          "properties": {              "id": { "type": "long", "store": true },              "area": { "type": "keyword", "store": true },              "exp": { "type": "keyword", "store": true },              "edu": { "type": "keyword", "store": true },              "salary": { "type": "keyword", "store": true },              "job\_type": { "type": "keyword", "store": true },              "cmp": { "type": "keyword", "store": true },              "pv": { "type": "keyword", "store": true },              "title": { "type": "text", "store": true },              "jd": { "type": "text"}          }      },      "settings": {          "number\_of\_shards": 3,          "number\_of\_replicas": 2      }  }  // 查看分片、主分片、副本分片  GET /\_cat/indices?v |

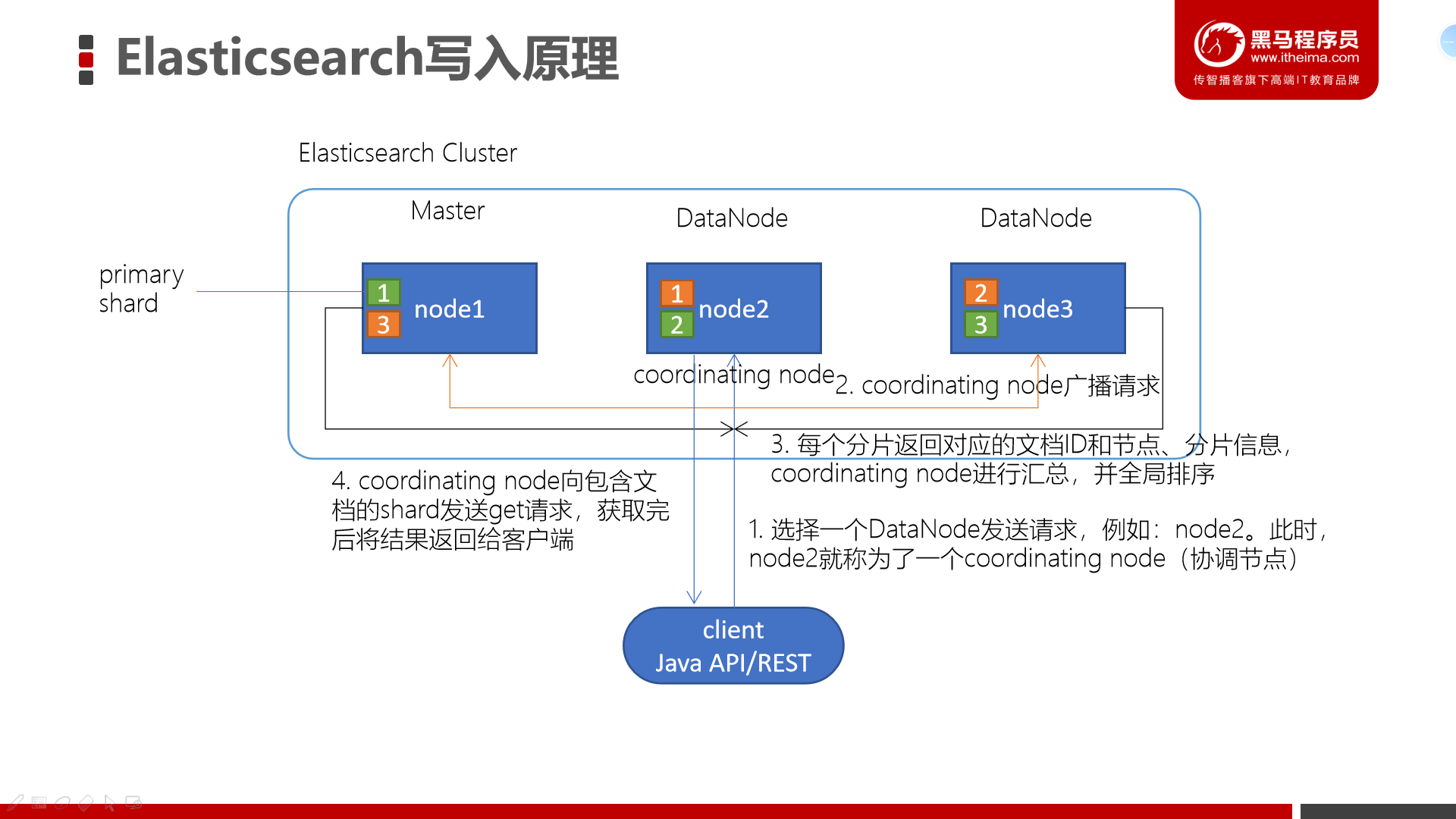
## Elasticsearch重要工作流程

### Elasticsearch文档写入原理



1. 选择任意一个DataNode发送请求，例如：node2.itcast.cn。此时，node2.itcast.cn就成为一个 coordinating node（协调节点）
2. coordinating node会进行路由，将请求转发给对应的primary shard所在的DataNode（假设 primary shard在node1.itcast.cn、replica shard在node2.itcast.cn）
3. node1.itcast.cn节点上的Primary Shard处理请求，写入数据到索引库中，并将数据同步到 Replica shard
4. Primary Shard和Replica Shard都保存好了文档，返回client

### Elasticsearch检索原理



* client发起查询请求，某个DataNode接收到请求，该DataNode就会成为协调节点（Coordinating Node）
* 协调节点（Coordinating Node）将查询请求广播到每一个数据节点，这些数据节点的分片会处理该查询请求。协调节点会轮询所有的分片来自动进行负载均衡
* 每个分片进行数据查询，将符合条件的数据放在一个优先队列中，并将这些数据的文档ID、节点信息、分片信息返回给协调节点
* 协调节点将所有的结果进行汇总，并进行全局排序
* 协调节点向包含这些文档ID的分片发送get请求，对应的分片将文档数据返回给协调节点，最后协调节点将数据返回给客户端

## Elasticsearch SQL

Elasticsearch SQL允许执行类SQL的查询，可以使用REST接口、命令行或者是JDBC，都可以使用SQL来进行数据的检索和数据的聚合。

Elasticsearch SQL特点：

* 本地集成
  + Elasticsearch SQL是专门为Elasticsearch构建的。每个SQL查询都根据底层存储对相关节点有效执行。
* 没有额外的要求
  + 不依赖其他的硬件、进程、运行时库，Elasticsearch SQL可以直接运行在Elasticsearch集群上
* 轻量且高效
  + 像SQL那样简洁、高效地完成查询

### SQL与Elasticsearch对应关系

|  |  |
| --- | --- |
| SQL | Elasticsearch |
| column（列） | field（字段） |
| row（行） | document（文档） |
| table（表） | index（索引） |
| schema（模式） | N/A |
| database（数据库） | Elasticsearch集群实例 |

### Elasticsearch SQL语法

|  |
| --- |
| SELECT select\_expr [, ...]  [ FROM table\_name ]  [ WHERE condition ]  [ GROUP BY grouping\_element [, ...] ]  [ HAVING condition]  [ ORDER BY expression [ ASC | DESC ] [, ...] ]  [ LIMIT [ count ] ]  [ PIVOT ( aggregation\_expr FOR column IN ( value [ [ AS ] alias ] [, ...] ) ) ] |

* 目前FROM只支持一个表

### 查询案例

#### 查询职位索引库中的一条数据

format：表示指定返回的数据类型

|  |
| --- |
| // 1. 查询职位信息  GET /\_sql?format=txt  {      "query": "SELECT \* FROM job\_idx limit 1"  } |

除了txt类型，Elasticsearch SQL还支持以下类型，

|  |  |
| --- | --- |
| 格式 | 描述 |
| csv | 逗号分隔符 |
| json | JSON格式 |
| tsv | 制表符分隔符 |
| txt | 类cli表示 |
| yaml | YAML人类可读的格式 |

#### 将SQL转换为DSL

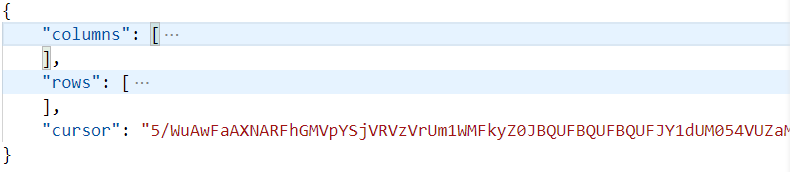
|  |
| --- |
| GET /\_sql/translate  {      "query": "SELECT \* FROM job\_idx limit 1"  } |

#### 职位scroll分页查询

##### 第一次查询

|  |
| --- |
| // 2. scroll分页查询  GET /\_sql?format=json  {      "query": "SELECT \* FROM job\_idx",      "fetch\_size": 10  } |

fetch\_size表示每页显示多少数据，而且当我们指定format为Json格式时，会返回一个cursor ID。



##### 第二次查询

|  |
| --- |
| GET /\_sql?format=json  {      "cursor": "5/WuAwFaAXNARFhGMVpYSjVRVzVrUm1WMFkyZ0JBQUFBQUFBQUFJZ1dUM054VUZaMk9YVlJWalowYkVJeFowUkdVak10ZHc9Pf////8PCgFmBGFyZWEBBGFyZWEBB2tleXdvcmQBAAABZgNjbXABA2NtcAEHa2V5d29yZAEAAAFmA2VkdQEDZWR1AQdrZXl3b3JkAQAAAWYDZXhwAQNleHABB2tleXdvcmQBAAABZgJpZAECaWQBBGxvbmcAAAABZgJqZAECamQBBHRleHQAAAABZghqb2JfdHlwZQEIam9iX3R5cGUBB2tleXdvcmQBAAABZgJwdgECcHYBB2tleXdvcmQBAAABZgZzYWxhcnkBBnNhbGFyeQEHa2V5d29yZAEAAAFmBXRpdGxlAQV0aXRsZQEEdGV4dAAAAAL/Aw=="  } |

##### 清除游标

|  |
| --- |
| POST /\_sql/close  {      "cursor": "5/WuAwFaAXNARFhGMVpYSjVRVzVrUm1WMFkyZ0JBQUFBQUFBQUFJZ1dUM054VUZaMk9YVlJWalowYkVJeFowUkdVak10ZHc9Pf////8PCgFmBGFyZWEBBGFyZWEBB2tleXdvcmQBAAABZgNjbXABA2NtcAEHa2V5d29yZAEAAAFmA2VkdQEDZWR1AQdrZXl3b3JkAQAAAWYDZXhwAQNleHABB2tleXdvcmQBAAABZgJpZAECaWQBBGxvbmcAAAABZgJqZAECamQBBHRleHQAAAABZghqb2JfdHlwZQEIam9iX3R5cGUBB2tleXdvcmQBAAABZgJwdgECcHYBB2tleXdvcmQBAAABZgZzYWxhcnkBBnNhbGFyeQEHa2V5d29yZAEAAAFmBXRpdGxlAQV0aXRsZQEEdGV4dAAAAAL/Aw=="  } |

#### 职位全文检索

##### 需求

检索title和jd中包含hadoop的职位。

##### MATCH函数

在执行全文检索时，需要使用到MATCH函数。

|  |
| --- |
| MATCH(  field\_exp,  constant\_exp  [, options]) |

* field\_exp：匹配字段
* constant\_exp：匹配文档

##### 实现

|  |
| --- |
| GET /\_sql?format=txt  {      "query": "select \* from job\_idx where MATCH(title, 'hadoop') or MATCH(jd, 'hadoop') limit 10"  } |

#### 统计不同支付方式订单数

|  |
| --- |
| // 统计不同支付方式订单数量  GET /\_sql?format=txt  {      "query": "select payway, count(\*) from order\_idx group by payway"  } |

### Elasticsearch SQL目前的一些限制

目前Elasticsearch SQL还存在一些限制。例如：不支持JOIN、不支持较复杂的子查询。所以，有一些相对复杂一些的功能，还得借助于DSL方式来实现。