ElasticSearch  nested parent/child join

ElasticSearch  7.2.0

1、创建索引含有nested类型的索引

2、添加数据（nested）

3、查询（nested）

4、创建含有关联关系的索引

5、添加数据（parent/child join）

   5.1 添加父文档

   5.2 添加子文档

6、查询（parent/child join）

   6.1 Parent Id 查询

   6.2 Has Child 查询

   6.3 Has Parent 查询

   6.4 通过ID和routing

7、nested VS parent/child join

对人工智能感兴趣点下面链接

现在人工智能非常火爆，很多朋友都想学，但是一般的教程都是为博硕生准备的，太难看懂了。最近发现了一个非常适合小白入门的教程，不仅通俗易懂而且还很风趣幽默。所以忍不住分享一下给大家。点这里可以跳转到教程。

https://www.cbedai.net/u014646662

1、创建索引含有nested类型的索引

PUT my\_blogs\_nested

{

"settings": {

"number\_of\_shards": 3,

"number\_of\_replicas": 0

},

"mappings": {

"properties": {

"blog\_comments\_relation":{

"type" : "nested",

"properties" : {

"comment" : {

"type" : "text",

"norms" : false,

"fields" : {

"keyword" : {

"type" : "keyword"

}

}

},

"username" : {

"type" : "text",

"norms" : false,

"fields" : {

"keyword" : {

"type" : "keyword"

}

}

}

}

},

"content":{

"type":"text"

},

"title":{

"type":"text",

"norms":false,

"fields": {

"keyword":{

"type":"keyword"

}

}

}

}

}

}

2、添加数据（nested）

PUT my\_blogs\_nested/\_doc/blog1

{

"title":"Learning ElasticSearch",

"content":"Learning Elastic Stack",

"blog\_comments\_relation":[

{

"comment":"I am learning Elatsic Stack",

"username":"Lucky"

},{

"comment":"Hello ElasticSearch",

"username":"Bob"

}

]

}

PUT my\_blogs\_nested/\_doc/blog2

{

"title":"Leaning Hadoop",

"content":"Leaning Hadoop",

"blog\_comments\_relation":[

{

"comment":"I am learning Hadoop",

"username":"Lucy"

},{

"comment":"Hello Hadoop",

"username":"Bob"

}

]

}

3、查询（nested）

#查看索引

GET my\_blogs\_nested

#查询文档

GET my\_blogs\_nested/\_search

{

"query": {

"nested": {

"path": "blog\_comments\_relation",

"query": {

"match": {

"blog\_comments\_relation.username": "lucy"

}

}

}

}

}

4、创建含有关联关系的索引

"blog":"comment"，即父文档是blog，子文档是comment，子文档可以有多个（一对多的关系）

PUT my\_blogs

{

"settings": {

"number\_of\_shards": 3,

"number\_of\_replicas": 0

},

"mappings": {

"properties": {

"blog\_comments\_relation":{

"type": "join",

"relations":{

"blog":"comment"

}

},

"content":{

"type":"text"

},

"title":{

"type":"text",

"norms":false,

"fields": {

"keyword":{

"type":"keyword"

}

}

}

}

}

}

5、添加数据（parent/child join）

5.1 添加父文档

"name":"blog" 父文档名

PUT my\_blogs/\_doc/blog1

{

"title":"Learning ElasticSearch",

"content":"Learning Elastic Stack",

"blog\_comments\_relation":{

"name":"blog"

}

}

PUT my\_blogs/\_doc/blog2

{

"title":"Leaning Hadoop",

"content":"Leaning Hadoop",

"blog\_comments\_relation":{

"name":"blog"

}

}

5.2 添加子文档

"name":"comment", 子文档名

 "parent":"blog1"，指定父文档的ID

routing=blog1，子文档和父文档必须在同一个分片上，路由到父文档的ID

PUT my\_blogs/\_doc/comment3?routing=blog2

{

"comment":"Hello Hadoop",

"username":"Bob",

"blog\_comments\_relation":{

"name":"comment",

"parent":"blog2"

}

}

PUT my\_blogs/\_doc/comment4?routing=blog1

{

"comment":"Hello ElasticSearch",

"username":"Bob",

"blog\_comments\_relation":{

"name":"comment",

"parent":"blog1"

}

}

6、查询（parent/child join）

6.1 Parent Id 查询

GET my\_blogs/\_search

{

"query": {

"parent\_id":{

"type":"comment",

"id":"blog2"

}

}

}

6.2 Has Child 查询,返回父文档

GET my\_blogs/\_search

{

"query": {

"has\_child": {

"type": "comment",

"query": {

"match": {

"username": "lucky"

}

}

}

}

}

6.3 Has Parent 查询，返回相关的子文档

GET my\_blogs/\_search

{

"query": {

"has\_parent": {

"parent\_type": "blog",

"query": {

"match": {

"content": "elastic"

}

}

}

}

}

6.4 通过ID和routing ，访问子文档(不加routing查不到)

GET my\_blogs/\_doc/comment3?routing=blog2

7、 nested VS parent/child join

  nested parent/child join

优点 读取性能高 父子文档可以独立更新

缺点 每次更新需要更新整个文档 关联关系，需要额外的内存，查询效率相对较差

场景 频繁查询 频繁更新

————————————————

版权声明：本文为CSDN博主「周天祥」的原创文章，遵循CC 4.0 BY-SA版权协议，转载请附上原文出处链接及本声明。

原文链接：https://blog.csdn.net/u014646662/java/article/details/100081257