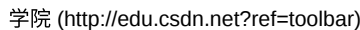


立即体

驗



更多 ▼



1

## +关注



elasticsearch (<http://so.csdn.net/so/search/s.do?q=elasticsearch&t=blog>)

 3336

从网上了解到，scroll类似于数据库中的游标，不考虑排序的时候，可以结合SearchType.SCAN使用。

**school**  
size: 1.39Mi (1.39Mi)  
docs: 9,736 (9,736)

信息 ▼

动作 ▼



## ■ 在线课程



Apache Weex: 移动研发的进阶之路



80

## 热门文章

 23662 16569

内容举报

 8295

04657) [返回顶部](#)

 7111

QueryBuilder简单查询 (/lom9357bye/article/details/52852533)

```

1.  /**
2.   * 创建假数据
3.   * @param count 数据量
4.   */
5.  public static void insertData(int count) {
6.      System.out.println("-----begin-----");
7.      Student student=new Student();
8.      for (int i=1;i<=count;i++){
9.          student.setName("name"+i);
10.         student.setId(i);
11.         String id="id_"+i;//设置存入ES中的ID
12.         //插入数据(数据量大时,最好使用批量插入,此处为单条插入)
13.         esUtils.insert("school","student",id,new Gson().toJson(student));
14.     }
15.     System.out.println("-----end-----");
16. }

```

Student类

```
[java]
1. package com.es.scroll.bean;
2.
3.
4. public class Student {
5.     private String name;
6.     private int id;
7.
8.     public String getName() {
9.         return name;
10.    }
```

6963

```

11.
12.     public void setName(String name) {
13.         this.name = name;
14.     }
15.
16.     public int getId() {
17.         return id;
18.     }
19.
20.     public void setId(int id) {
21.         this.id = id;
22.     }
23. }

```

## ElasticsearchUtils

```

[java]
1. package com.es.scroll.esutil;
2.
3. import org.elasticsearch.action.index.IndexRequestBuilder;
4. import org.elasticsearch.client.Client;
5. import org.elasticsearch.client.transport.TransportClient;
6. import org.elasticsearch.common.settings.ImmutableSettings;
7. import org.elasticsearch.common.settings.Settings;
8. import org.elasticsearch.common.transport.InetSocketTransportAddress;
9.
10.
11. public class ElasticsearchUtils {
12.
13.     private Client client;
14.
15.     public ElasticsearchUtils(String clusterName, String ipAddress, int port) {
16.         Settings settings = ImmutableSettings
17.             .settingsBuilder()
18.             //设置集群名称
19.             .put("cluster.name", clusterName)
20.             .put("client.transport.ignore_cluster_name", false)
21.             .put("node.client", true).put("client.transport.sniff", true)
22.             .build();
23.         client = new TransportClient(settings)
24.             .addTransportAddress(new InetSocketTransportAddress(ipAddress,
25.                 port));
26.     }
27.
28.     public Client getClient(){
29.         return this.client;
30.     }
31.
32.
33.     /**
34.      * 创建索引
35.      * @param indexName 索引名称, 相当于数据库名称
36.      * @param typeName 索引类型, 相当于数据库中的表名
37.      * @param id id名称, 相当于每个表中某一行记录的标识
38.      * @param jsonData json数据
39.      */
40.     public void insert(String indexName, String typeName, String id,
41.         String jsonData) {
42.         IndexRequestBuilder requestBuilder = client.prepareIndex(indexName,
43.             typeName, id).setRefresh(true); //设置索引名称, 索引类型, id
44.         requestBuilder.setSource(jsonData).execute().actionGet(); //创建索引
45.     }
46. }

```

数据如下:

 内容举报

 返回顶部

▼ 查询 5 个分片中用的 5 个, 10000 命中, 耗时 0.011 秒

_index	_type	_id	_score ▼	name	id
school	student	id_11	1	name11	11
school	student	id_16	1	name16	16
school	student	id_23	1	name23	23
school	student	id_28	1	name28	28
school	student	id_30	1	name30	30
school	student	id_35	1	name35	35
school	student	id_42	1	name42	42
school	student	id_47	1	name47	47
school	student	id_2	1	name2	2
school	student	id_7	1	name7	7
school	student	id_54	1	name54	54
school	student	id_59	1	name59	59
school	student	id_61	1	name61	61
school	student	id_66	1	name66	66
school	student	id_73	1	name73	73
school	student	id_78	1	name78	78
school	student	id_80	1	name80	80
school	student	id_85	1	name85	85
school	student	id_92	1	name92	92
school	student	id_102	1	name102	102
school	student	id_107	1	name107	107
school	student	id_114	1	name114	114
school	student	id_119	1	name119	119
school	student	id_121	1	name121	121
school	student	id_126	1	name126	126
school	student	id_133	1	name133	133

### 3.使用scroll结合scan遍历所有数据

```
[java]
1. package com.es.scroll;
2.
3. import com.es.scroll.bean.Student;
4. import com.es.scroll.esutil.ElasticsearchUtils;
5. import com.google.gson.Gson;
6. import org.elasticsearch.action.search.SearchResponse;
7. import org.elasticsearch.action.search.SearchType;
8. import org.elasticsearch.common.unit.TimeValue;
9. import org.elasticsearch.search.SearchHit;
10. import org.elasticsearch.search.SearchHits;
11.
12.
13.
14.
15. public class MainApplication {
16.     private static int i=0;
17.     private static ElasticsearchUtils esUtils;
18.
19.     public static void main(String[] args) {
20.
21.         //创建ElasticsearchUtils对象, 设置集群名称, IP地址, 端口号
22.         esUtils = new ElasticsearchUtils("im_shanmenglu", "localhost", 9300);
23.         // insertData(10000);
24.         //获取Client对象, 设置索引名称, 搜索类型(SearchType.SCAN), 搜索数量, 发送请求
25.         SearchResponse searchResponse = esUtils.getClient()
26.             .prepareSearch("school").setSearchType(SearchType.SCAN)
27.             .setSize(10).setScroll(new TimeValue(20000)).execute()
28.             .actionGet(); //注意: 首次搜索并不包含数据
29.         //获取总数量
30.         long totalCount = searchResponse.getHits().getTotalHits();
31.         int page=(int)totalCount/(5*10); //计算总页数, 每次搜索数量为分片数*设置的size大小
32.         System.out.println(totalCount);
33.         for (int i = 0; i <= page; i++) {
34.             //再次发送请求, 并使用上次搜索结果的ScrollId
35.             searchResponse = esUtils.getClient()
36.                 .prepareSearchScroll(searchResponse.getScrollId())
37.                 .setScroll(new TimeValue(20000)).execute()
38.                 .actionGet();
39.             parseSearchResponse(searchResponse);
40.         }
41.     }

```



内容举报



返回顶部

```

42.     }
43.
44.     public static void parseSearchResponse(SearchResponse searchResponse) {
45.         SearchHits hits = searchResponse.getHits();
46.         System.out.println("-----begin-----");
47.         for (SearchHit searchHit : hits.getHits()) {
48.             try {
49.                 i++;
50.                 String id = searchHit.getId();
51.                 System.out.println("第" + i + "条数据:" + id);
52.             } catch (Exception e) {
53.                 e.printStackTrace();
54.             }
55.         }
56.         System.out.println("-----end-----");
57.     }
58.
59.     /**
60.      * 创造假数据
61.      * @param count 数据量
62.      */
63.     public static void insertData(int count) {
64.         System.out.println("-----begin-----");
65.         Student student=new Student();
66.         for (int i=1;i<=count;i++){
67.             student.setName("name"+i);
68.             student.setId(i);
69.             String id="id_"+i;//设置存入ES中的ID
70.             //插入数据(数据量大时,最好使用批量插入,此处为单条插入)
71.             esUtils.insert("school", "student", id, new Gson().toJson(student));
72.         }
73.         System.out.println("-----end-----");
74.     }
75.
76.
77. }
78.
79. /**部分数据
80.  * 10000
81.  -----begin-----
82.  第1条数据:id_11
83.  第2条数据:id_16
84.  第3条数据:id_23
85.  第4条数据:id_28
86.  第5条数据:id_30
87.  第6条数据:id_35
88.  第7条数据:id_42
89.  第8条数据:id_47
90.  第9条数据:id_2
91.  第10条数据:id_7
92.  第11条数据:id_12
93.  第12条数据:id_17
94.  第13条数据:id_24
95.  第14条数据:id_29
96.  第15条数据:id_31
97.  第16条数据:id_36
98.  第17条数据:id_43
99.  第18条数据:id_48
100. 第19条数据:id_3
101. 第20条数据:id_8
102. 第21条数据:id_13
103. 第22条数据:id_18
104. 第23条数据:id_20
105. 第24条数据:id_25
106. 第25条数据:id_32
107. 第26条数据:id_37
108. 第27条数据:id_44
109. 第28条数据:id_49
110. 第29条数据:id_4
111. 第30条数据:id_9
112. 第31条数据:id_14
113. 第32条数据:id_19
114. 第33条数据:id_21
115. 第34条数据:id_26
116. 第35条数据:id_33
117. 第36条数据:id_38
118. 第37条数据:id_40
119. 第38条数据:id_45
120. 第39条数据:id_52
121. 第40条数据:id_5

```



内容举报



返回顶部

```
122. 第41条数据:id_10
123. 第42条数据:id_15
124. 第43条数据:id_22
125. 第44条数据:id_27
126. 第45条数据:id_34
127. 第46条数据:id_39
128. 第47条数据:id_41
129. 第48条数据:id_46
130. 第49条数据:id_1
131. 第50条数据:id_6
132. -----end-----
133. -----begin-----
134. 第51条数据:id_54
135. 第52条数据:id_59
136. 第53条数据:id_61
137. 第54条数据:id_66
138. 第55条数据:id_73
139. 第56条数据:id_78
140. 第57条数据:id_80
141. 第58条数据:id_85
142. 第59条数据:id_92
143. 第60条数据:id_102
144. 第61条数据:id_50
145. 第62条数据:id_55
146. 第63条数据:id_62
147. 第64条数据:id_67
148. 第65条数据:id_74
149. 第66条数据:id_79
150. 第67条数据:id_81
151. 第68条数据:id_86
152. 第69条数据:id_93
153. 第70条数据:id_103
154. 第71条数据:id_51
155. 第72条数据:id_56
156. 第73条数据:id_63
157. 第74条数据:id_68
158. 第75条数据:id_70
159. 第76条数据:id_75
160. 第77条数据:id_82
161. 第78条数据:id_87
162. 第79条数据:id_94
163. 第80条数据:id_104
164. 第81条数据:id_57
165. 第82条数据:id_64
166. 第83条数据:id_69
167. 第84条数据:id_71
168. 第85条数据:id_76
169. 第86条数据:id_83
170. 第87条数据:id_88
171. 第88条数据:id_90
172. 第89条数据:id_95
173. 第90条数据:id_100
174. 第91条数据:id_53
175. 第92条数据:id_58
176. 第93条数据:id_60
177. 第94条数据:id_65
178. 第95条数据:id_72
179. 第96条数据:id_77
180. 第97条数据:id_84
181. 第98条数据:id_89
182. 第99条数据:id_91
183. 第100条数据:id_101
184. -----end-----
185. -----begin-----
186. 第101条数据:id_107
187. 第102条数据:id_114
188. 第103条数据:id_119
189. 第104条数据:id_121
190. 第105条数据:id_126
191. 第106条数据:id_133
192. 第107条数据:id_138
193. 第108条数据:id_97
194. 第109条数据:id_171
195. 第110条数据:id_176
196. 第111条数据:id_108
197. 第112条数据:id_110
198. 第113条数据:id_115
199. 第114条数据:id_122
200. 第115条数据:id_127
201. 第116条数据:id_134
```



内容举报



返回顶部

202. 第117条数据:id\_139  
203. 第118条数据:id\_98  
204. 第119条数据:id\_172  
205. 第120条数据:id\_177  
206. 第121条数据:id\_109  
207. 第122条数据:id\_111  
208. 第123条数据:id\_116  
209. 第124条数据:id\_123  
210. 第125条数据:id\_128  
211. 第126条数据:id\_130  
212. 第127条数据:id\_135  
213. 第128条数据:id\_99  
214. 第129条数据:id\_166  
215. 第130条数据:id\_173  
216. 第131条数据:id\_105  
217. 第132条数据:id\_112  
218. 第133条数据:id\_117  
219. 第134条数据:id\_124  
220. 第135条数据:id\_129  
221. 第136条数据:id\_131  
222. 第137条数据:id\_136  
223. 第138条数据:id\_143  
224. 第139条数据:id\_179  
225. 第140条数据:id\_181  
226. 第141条数据:id\_106  
227. 第142条数据:id\_113  
228. 第143条数据:id\_118  
229. 第144条数据:id\_120  
230. 第145条数据:id\_125  
231. 第146条数据:id\_132  
232. 第147条数据:id\_137  
233. 第148条数据:id\_96  
234. 第149条数据:id\_168  
235. 第150条数据:id\_170  
236. -----end-----  
237. -----begin-----  
238. 第151条数据:id\_183  
239. 第152条数据:id\_140  
240. 第153条数据:id\_145  
241. 第154条数据:id\_152  
242. 第155条数据:id\_157  
243. 第156条数据:id\_164  
244. 第157条数据:id\_169  
245. 第158条数据:id\_188  
246. 第159条数据:id\_190  
247. 第160条数据:id\_195  
248. 第161条数据:id\_184  
249. 第162条数据:id\_141  
250. 第163条数据:id\_146  
251. 第164条数据:id\_153  
252. 第165条数据:id\_158  
253. 第166条数据:id\_160  
254. 第167条数据:id\_165  
255. 第168条数据:id\_189  
256. 第169条数据:id\_191  
257. 第170条数据:id\_196  
258. 第171条数据:id\_178  
259. 第172条数据:id\_180  
260. 第173条数据:id\_142  
261. 第174条数据:id\_147  
262. 第175条数据:id\_154  
263. 第176条数据:id\_159  
264. 第177条数据:id\_161  
265. 第178条数据:id\_185  
266. 第179条数据:id\_192  
267. 第180条数据:id\_197  
268. 第181条数据:id\_186  
269. 第182条数据:id\_148  
270. 第183条数据:id\_150  
271. 第184条数据:id\_155  
272. 第185条数据:id\_162  
273. 第186条数据:id\_167  
274. 第187条数据:id\_174  
275. 第188条数据:id\_193  
276. 第189条数据:id\_198  
277. 第190条数据:id\_201  
278. 第191条数据:id\_175  
279. 第192条数据:id\_182  
280. 第193条数据:id\_144  
281. 第194条数据:id\_149



1



  
内容举报

  
返回顶部

```
282. 第195条数据:id_151
283. 第196条数据:id_156
284. 第197条数据:id_163
285. 第198条数据:id_187
286. 第199条数据:id_194
287. 第200条数据:id_199
288. -----end-----
289. */
```



此处截取了部分输出数据，可以看出使用scroll进行分页查询，每次查询的数据量为分片的数量\*首次查询设置的size大小，TimeValue表示需要保持搜索的上下文时间。



参考：<http://www.jianshu.com/p/14aa8b09c789> (<http://www.jianshu.com/p/14aa8b09c789>)  
<http://www.pig66.com/doc/pc/web/2016-03-20/782672.html>  
(<http://www.pig66.com/doc/pc/web/2016-03-20/782672.html>)


源码下载：<http://download.csdn.net/detail/lom9357bye/9789003>  
(<http://download.csdn.net/detail/lom9357bye/9789003>)

版权声明：



发表你的评论

(<http://my.csdn.net/u013411339>)

benbendehuoze (/benbendehuoze) 2017-10-10 16:48 1楼


(/benbendehuoze) 你这样查询，100条好像没查询出来吧

回复

相关文章推荐


elasticsearch-利用游标查询 'Scroll' 来做分页查询 (/chuan442616909/article/details/551...

游标查询 'Scroll' scroll 查询 可以用来对 Elasticsearch 有效地执行大批量的文档查询，而又不付出深度分页那种代价。游标查询允许我们先做查询初始化，然...

 chuan442616909 (<http://blog.csdn.net/chuan442616909>) 2017-02-15 14:12 2514


ES 分页查询 (/u012307002/article/details/50457749)

<http://eggtwo.com/news/detail/147>

 u012307002 (<http://blog.csdn.net/u012307002>) 2016-01-04 18:36 1010

elasticsearch分页查询 (/lsshls/article/details/50302069)

新博客地址主要有两种方式from/sizefrom 偏移，默认为0size 返回的结果数，默认为10在数据量不大的情况下我们一般会使用from/size，而在深度分页的情况下效率极低，该命令会把fro...


 lsshls (<http://blog.csdn.net/lsshls>) 2015-12-14 22:51 7549

es 分页两种方法 (/cottonduke/article/details/70231119)

Elasticsearch——分页查询From&Size VS scroll Elasticsearch中数据都存储在分片中，当执行搜索时每个分片独立搜索后，数据再经过整合返...






 youyonghu001 (<http://blog.csdn.net/youyonghu001>) 2015-04-21 14:50 799

### 使用ajax进行分页查询因连接池耗尽导致请求被挂起（备忘） (/crazypandariy/article/detail...

本人在项目中，实现分页查询时，遇到了这个问题，当多次翻页后，页面请求被挂起。项目使用spring MVC+hibernate。刷新页面没有反应，从新登陆同意不起作用，只有重启tomcat才能解决。经过...


 crazypandariy (<http://blog.csdn.net/crazypandariy>) 2013-12-10 22:34 1216



**采用JDBC进行数据库分页查询 (<http://download.csdn.net/detail/zengw...>**




[下载](#)



2008-12-14 00:41 22KB


### OracleHelper(对增删改查分页查询操作进行了面向对象的封装，对批量增删改操作的事务封...

公司的一个新项目使用ASP.NET MVC开发，经理让我写个OracleHelper，我从网上找了一个比较全的OracleHelper类，缺点是查询的时候返回DataSet，数据增删改要写很多代码(当...

 shan1774965666 (<http://blog.csdn.net/shan1774965666>) 2015-01-01 20:11 1090


### 一个 JDBC 实现对 mysql 进行分页查询的 实例 (/dxr970110/article/details/76165747)

要在web页面显示分页后的学生信息，首先我们在navicat创建一个学生信息表，在这里我在表里插入了30 条数据1，创建Student实体类package com.lanou.controller;...

 dxr970110 (<http://blog.csdn.net/dxr970110>) 2017-07-26 21:21 134


### mysql 数据库 分表后 怎么进行分页查询？ Mysql分库分表方案? (/ahjxhy2010/article/detail...

1.如果只是为了分页，可以考虑这种分表，就是表的id是范围性的，且id是连续的，比如第一张表id是1到10万，第二张是10万到20万，这样分页应该没什么问题。 2.如果是其他的分表方式，建议用s...

 ahjxhy2010 (<http://blog.csdn.net/ahjxhy2010>) 2016-01-19 09:47 8412


### Oracle的分页查询语句基本上可以按照本文给出的格式来进行套用。 (/lihaihuai3yy/article/d...

Oracle的分页查询语句基本上可以按照本文给出的格式来进行套用。 分页查询格式： SELECT \* FROM ( SELECT A.\*,ROWNUM RN FROM (SEL...

 lihaihuai3yy (<http://blog.csdn.net/lihaihuai3yy>) 2011-10-18 14:26 150

### Elasticsearch Java API 的使用（8） — Scroll (游标)API详解 (/zx711166/article/details/75...

Elasticsearch中进行大数据量查询时，往往因为设备、网络传输问题影响查询数据的效率；Elasticsearch中提供了Scroll（游标）的方式对数据进行少量多批次的查询，来提高查询效率。p...

 zx711166 (<http://blog.csdn.net/zx711166>) 2017-07-14 17:31 75