FieldCollapsing

<!> Solr4.0 (!) Solr3.3

Result Grouping / Field Collapsing

- Result Grouping / Field Collapsing
- Introduction
- Quick Start
- Request Parameters
- Known Limitations

Introduction

Field Collapsing and Result Grouping are different ways to think about the same Solr feature.

Field Collapsing collapses a group of results with the same field value down to a single (or fixed number) of entries. For example, most search engines such as Google collapse on site so only one or two entries are shown, along with a link to click to see more results from that site. Field collapsing can also be used to suppress duplicate documents.

Result Grouping groups documents with a common field value into groups, returning the top documents per group, and the top groups based on what documents are in the groups. One example is a search at Best Buy for a common term such as DVD, that shows the top 3 results for each category ("TVs & Video","Movies","Computers", etc)

Quick Start

If you haven't already, get a recent nightly build of solr tutorial] or Solr3.3, start the example server and index the example data as shown in the [http://lucene.apache.org/solr/tutorial.html](http://lucene.apache.org/solr/tutorial.html).

Now send a query request to solr and turn on result grouping. We'll first try grouping on the manufacturer name (the manu_exact field). You can currently only group on single-valued fields!

...&q=solr+memory&group=true&group.field=manu_exact

And the grouped response is returned:
The response indicates that there are 6 total matches to our query. For each unique value of group.field (manufacturer names in this example) a docList with the top scoring document is returned. The docList also gives the total number of matches in that group as "numFound". The groups themselves are also sorted by the score of the top document within each group.

We can find the top documents that also match arbitrary queries with the group.query command (much like facet.query). For example, we could use this to find the top 3 documents with in different price ranges:

[http://localhost:8983/solr/select?wt=json&indent=true&fl=name,price&q=memory&group=true&group.query=price:[0 TO 99.99]&group.query=price:[100 TO *]&group.limit=3]
From the response above, we can see that 5 documents matched our base query of "memory". Of those, 1 had a price below $100, 3 had a price above $100. This does not add up to 5 because one document was missing a price and hence did not match either group.query.

We can optionally use the results of a group command as the "main" result (i.e. a single flat document list that would normally be produced by a non-grouped query request) by adding the parameter group.main=true. Although this result format does not have as much information, it may be easier for existing solr clients to parse.

...&q=solr+memory&group=true&group.field=manu_exact&group.main=true

Request Parameters

<table>
<thead>
<tr>
<th>param name</th>
<th>parameter value</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Solr, the Enterprise Search Server</td>
<td>Apache Software Foundation</td>
<td></td>
</tr>
<tr>
<td>CORSAIR ValueSelect 1GB 184-Pin DDR SDRAM Unbuffered DDR 400 (PC 3200) System Memory - Retail</td>
<td>Corsair Microsystems Inc.</td>
<td></td>
</tr>
<tr>
<td>A-DATA V-Series 1GB 184-Pin DDR SDRAM Unbuffered DDR 400 (PC 3200) System Memory - OEM</td>
<td>A-DATA Technology Inc.</td>
<td></td>
</tr>
<tr>
<td>Canon PIXMA MP500 All-In-One Photo Printer</td>
<td>Canon Inc.</td>
<td></td>
</tr>
<tr>
<td>ASUS Extreme N7800GTX/2DHTV (256 MB)</td>
<td>ASUS Computer Inc.</td>
<td></td>
</tr>
</tbody>
</table>
### Grouping Fields

When grouping documents, you can specify grouping criteria based on fields or functions. Here's a breakdown of the parameters available for grouping:

- **group**: A Boolean parameter indicating if true, the result of the last field grouping command is used as the main result list in the response, using group.format=simple.

#### Facilities

The following facilities are available:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>group.facet</td>
<td>Whether to compute grouped facets for the field facets specified in facet.field parameters. Grouped facets are computed based on the first specified group. Just like normal field faceting, fields shouldn't be tokenized (otherwise the count is computed for each token). Grouped faceting supports single and multivalued fields. Default is false. <img src="http://lucene.apache.org/solr/api/org/apache/solr/schema/ExternalFileField.html" alt="Solr 3.4 WARNING" />. Note for Solr 3.x versions the field must be a string-like field such as [StrField] or [TextField], otherwise a http status 400 is returned.</td>
</tr>
<tr>
<td>group</td>
<td>If true, the result of the last field grouping command is used as the main result list in the response, using group.format=simple.</td>
</tr>
<tr>
<td>group.ngroups</td>
<td>If true, includes the number of groups that have matched the query. Default is false. <img src="http://lucene.apache.org/solr/api/org/apache/solr/schema/ExternalFileField.html" alt="Solr 4.1 WARNING" />. Note for Solr 3.x versions this parameter is set to true on a sharded environment, all the documents that belong to the same group have to be located in the same shard, otherwise the count will be incorrect. If you are using SolrCloud, consider using &quot;custom hashing&quot;.</td>
</tr>
<tr>
<td>group.format</td>
<td>How to format the grouping result. If simple, the grouped documents are presented in a single flat list. The start and rows parameters refer to numbers of documents instead of numbers of groups.</td>
</tr>
<tr>
<td>group</td>
<td>If true, the result of the last field grouping command is used as the main result list in the response, using group.format=simple.</td>
</tr>
<tr>
<td>group.truncate</td>
<td>If true, facet counts are based on the most relevant document of each group matching the query. Same applies for StaticComponent. Default is false. <img src="http://lucene.apache.org/solr/api/org/apache/solr/schema/ExternalFileField.html" alt="Solr 3.4.0 WARNING" />. Supported from Solr 3.4 and up.</td>
</tr>
<tr>
<td>group.main</td>
<td>If true, the result of the last field grouping command is used as the main result list in the response, using group.format=simple.</td>
</tr>
</tbody>
</table>

#### Fields

- **fieldname**: A string parameter specifying the name of the field to be used for grouping.
- **query**: A string parameter specifying the query to be used for grouping.
- **function query**: A string parameter specifying the function query to be used for grouping.
- **number**: A long parameter specifying the number of groups to return. Defaults to 10.
- **start**: A long parameter specifying the offset into the list of groups.
- **offset**: A long parameter specifying the offset into the document list of each group.
- **sort**: A string parameter specifying how to sort the groups relative to each other. For example, `sort=popularity desc` will cause the groups to be sorted according to the highest popularity doc in each group. Defaults to "score desc".
- **sortspec**: A string parameter specifying how to sort documents within a single group. Defaults to the same value as the sort parameter.

#### Additional Parameters

- **custom hashing**: If true, include custom hashing for the groups. Otherwise, the count will be incorrect. If you are using SolrCloud, consider using "custom hashing". ![](http://lucene.apache.org/solr/api/org/apache/solr/schema/ExternalFileField.html). Note for Solr 3.x versions this parameter is set to true on a sharded environment, all the documents that belong to the same group have to be located in the same shard, otherwise the count will be incorrect. If you are using SolrCloud, consider using "custom hashing". |
If > 0 enables grouping cache. Grouping is executed actual two searches. This option caches the second search. A value of 0 disables grouping caching. Default is 0. Tests have shown that this cache only improves search time with boolean queries, wildcard queries and fuzzy queries. For simple queries like a term query or a match all query this cache has a negative impact on performance.

Notes:
- Any number of group commands (group.field, group.func, group.query) may be specified in a single request.
- Grouping is also supported for distributed searches from version Solr3.5 and from version Solr4.0. Currently group.truncate and group.func are the only parameters that aren't supported for distributed searches.

Known Limitations
- Support for grouping on a multi-valued field has not yet been implemented.
- Further performance improvements are planned soon!