FieldCollapsing

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

Result Grouping / Field Collapsing

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.

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:

```
[...]
  "grouped":{
    "manu_exact":{
      "matches":6,
      "groups":[{
          "groupValue":"Apache Software Foundation",
          "doclist":{"numFound":1,"start":0,"docs":[
              {
                "id":"SOLR1000",
                "name":"Solr, the Enterprise Search Server"}]
          }},
        {
          "groupValue":"Corsair Microsystems Inc.",
          "doclist":{"numFound":2,"start":0,"docs":[
              {
                "id":"VS1GB400C3",
                "name":"CORSAIR ValueSelect 1GB 184-Pin DDR SDRAM Unbuffered DDR 400 (PC 3200) System Memory -
Retail"}]
          }},
        {
          "groupValue":"A-DATA Technology Inc.",
          "doclist":{"numFound":1,"start":0,"docs":[
              {
                "id":"VDBDB1A16",
                "name":"A-DATA V-Series 1GB 184-Pin DDR SDRAM Unbuffered DDR 400 (PC 3200) System Memory -
OEM"}]
          }},
        {
          "groupValue":"Canon Inc.",
          "doclist":{"numFound":1,"start":0,"docs":[
              {
                "id":"0579B002",
                "name":"Canon PIXMA MP500 All-In-One Photo Printer"}]
          }},
        {
          "groupValue":"ASUS Computer Inc.",
          "doclist":{"numFound":1,"start":0,"docs":[
              {
                "id":"EN7800GTX/2DHTV/256M",
                "name":"ASUS Extreme N7800GTX/2DHTV (256 MB)"}]
          }}]}}
```

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]

```
[...]
  "grouped":{
    "price:[0 TO 99.99]":{
      "matches":5,
      "doclist":{"numFound":1,"start":0,"docs":[
          {
            "name":"CORSAIR ValueSelect 1GB 184-Pin DDR SDRAM Unbuffered DDR 400 (PC 3200) System Memory -
Retail",
            "price":74.99}]
      }},
    "price:[100 TO *]":{
      "matches":5,
      "doclist":{"numFound":3,"start":0,"docs":[
          {
            "name":"Canon PIXMA MP500 All-In-One Photo Printer",
            "price":179.99},
          {
            "name":"CORSAIR  XMS 2GB (2 x 1GB) 184-Pin DDR SDRAM Unbuffered DDR 400 (PC 3200) Dual Channel Kit
System Memory - Retail",
            "price":185.0},
          {
            "name":"ASUS Extreme N7800GTX/2DHTV (256 MB)",
            "price":479.95}]
      }}
[...]
```

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

```
  "response":{"numFound":6,"start":0,"docs":[
      {
        "id":"SOLR1000",
        "name":"Solr, the Enterprise Search Server",
        "manu":"Apache Software Foundation"},
      {
        "id":"VS1GB400C3",
        "name":"CORSAIR ValueSelect 1GB 184-Pin DDR SDRAM Unbuffered DDR 400 (PC 3200) System Memory - Retail",
        "manu":"Corsair Microsystems Inc."},
      {
        "id":"VDBDB1A16",
        "name":"A-DATA V-Series 1GB 184-Pin DDR SDRAM Unbuffered DDR 400 (PC 3200) System Memory - OEM",
        "manu":"A-DATA Technology Inc."},
      {
        "id":"0579B002",
        "name":"Canon PIXMA MP500 All-In-One Photo Printer",
        "manu":"Canon Inc."},
      {
        "id":"EN7800GTX/2DHTV/256M",
        "name":"ASUS Extreme N7800GTX/2DHTV (256 MB)",
        "manu":"ASUS Computer Inc."}]
  }
```

Request Parameters

param name	param value	description

group	true/false	if true, turn on result grouping			
<ac:structured-macro ac:name="unmigrated-wiki-markup" ac:schema-version="1" ac:macro-id="7e3890fe-ace7-4f48-8f4c-8d221d2c067b"><ac:plain-text-body><![CDATA[	group.field	[fieldname]	Group based on the unique values of a field. The field must currently be single-valued and must be either indexed, or be another field type that has a value source and works in a function query - such as [ExternalFileField	http://lucene.apache.org/solr/api/org/apache/solr/schema/ExternalFileField.html]. Note: for Solr 3.x versions the field must by a string like field such as [StrField] or [TextField], otherwise a http status 400 is returned.	]]></ac:plain-text-body></ac:structured-macro>
<ac:structured-macro ac:name="unmigrated-wiki-markup" ac:schema-version="1" ac:macro-id="f2886415-2d68-4999-9d9b-215efb31286a"><ac:plain-text-body><![CDATA[	group.func	[function query]	Group based on the unique values of a function query. ⚠ [Solr4.0] This parameter only is supported on 4.0	]]></ac:plain-text-body></ac:structured-macro>	
<ac:structured-macro ac:name="unmigrated-wiki-markup" ac:schema-version="1" ac:macro-id="7836bcc3-7a2d-4aa4-881b-6685e8690789"><ac:plain-text-body><![CDATA[	group.query	[query]	Return a single group of documents that also match the given query.	]]></ac:plain-text-body></ac:structured-macro>	
<ac:structured-macro ac:name="unmigrated-wiki-markup" ac:schema-version="1" ac:macro-id="26ba566a-1fa7-40e5-8fd8-619fe36c1bbd"><ac:plain-text-body><![CDATA[	rows	[number]	The number of groups to return. Defaults to 10.	]]></ac:plain-text-body></ac:structured-macro>	
<ac:structured-macro ac:name="unmigrated-wiki-markup" ac:schema-version="1" ac:macro-id="8fb4b934-953d-4958-aa42-22a884fe0a0e"><ac:plain-text-body><![CDATA[	start	[number]	The offset into the list of groups.	]]></ac:plain-text-body></ac:structured-macro>	
<ac:structured-macro ac:name="unmigrated-wiki-markup" ac:schema-version="1" ac:macro-id="2901a182-eef0-4158-a5c2-8af1167e3a42"><ac:plain-text-body><![CDATA[	group.limit	[number]	The number of results (documents) to return for each group. Defaults to 1.	]]></ac:plain-text-body></ac:structured-macro>	
<ac:structured-macro ac:name="unmigrated-wiki-markup" ac:schema-version="1" ac:macro-id="3c086348-66e9-4442-95c9-efee2f380d04"><ac:plain-text-body><![CDATA[	group.offset	[number]	The offset into the document list of each group.	]]></ac:plain-text-body></ac:structured-macro>	
<ac:structured-macro ac:name="unmigrated-wiki-markup" ac:schema-version="1" ac:macro-id="0d05412a-d1c4-4f6f-9bf9-eb140d541410"><ac:plain-text-body><![CDATA[	sort	[sortspec]	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".	]]></ac:plain-text-body></ac:structured-macro>	
<ac:structured-macro ac:name="unmigrated-wiki-markup" ac:schema-version="1" ac:macro-id="f2627ae2-1be2-4fd0-bde8-65624f02817a"><ac:plain-text-body><![CDATA[	group.sort	[sortspec]	How to sort documents within a single group. Defaults to the same value as the `sort` parameter.	]]></ac:plain-text-body></ac:structured-macro>	
group.format	grouped/simple	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.			
group.main	true/false	If true, the result of the last field grouping command is used as the main result list in the response, using group.format=simple			
group.ngroups	true/false	If true, includes the number of groups that have matched the query. Default is false. ⚠ Solr4.1 **WARNING:** If 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"			
group.truncate	true/false	If true, facet counts are based on the most relevant document of each group matching the query. Same applies for StatsComponent. Default is false. ⚠ Solr3.4 Supported from Solr 3.4 and up.			
group.facet	true/false	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 counts are computed for each token). Grouped faceting supports single and multivalued fields. Default is false. ⚠ Solr4.0 **WARNING:** If 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"			

<ac:structured-macro ac:name="unmigrated-wiki-markup" ac:schema-version="1" ac:macro-id="c584d873-8e69-4067-8fcf-fa79ce80cab7"><ac:plain-text-body><![CDATA[	group.cache.percent	[0-100]	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	]]></ac:plain-text-body></ac:structured-macro>

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!