Apache Solr — Cheat Sheet

Solr 9.x line · verified against solr.apache.org/guide · irurueta.com

REQUEST SHAPE (CURL / HTTP)

```
# select handler (GET, query string)
curl
"http://localhost:8983/solr/techproducts/select?
q=*:*&rows=10&wt=json"

# POST with JSON body (v1 API)
curl
http://localhost:8983/solr/techproducts/select \
  -H 'Content-Type: application/json' \
  -d '{"query":"*:*","limit":10}'

# response wrapper
{ "responseHeader": {"status":0,"QTime":1},
  "response": {"numFound":32,"docs":[ ... ]} }

# common params
q, fq, fl, rows, start, sort, wt=json|xml
```

SCHEMA: FIELDS, TYPES, KEY

```
<schema name="techproducts" version="1.6">

  <field name="id"    type="string" indexed="true"
         stored="true" required="true"/>
  <field name="title" type="text_general"
         indexed="true" stored="true"/>
  <field name="price" type="pfloat"
         indexed="true" stored="true"/>
  <field name="cat"   type="string"
         indexed="true" stored="true"
         multiValued="true"/>

  <fieldType name="text_general"
class="solr.TextField"
            positionIncrementGap="100">
    <analyzer> ... </analyzer>
  </fieldType>

  <uniqueKey>id</uniqueKey>
</schema>
# Schema API (managed schema, no restart)
POST /solr/techproducts/schema {"add-field":{...}}
```

ANALYSIS CHAIN

char filter → tokenizer → token filter(s)

```
<analyzer type="index">
  <charFilter
class="solr.HTMLStripCharFilterFactory"/>
  <tokenizer
class="solr.StandardTokenizerFactory"/>
  <filter class="solr.LowerCaseFilterFactory"/>
  <filter class="solr.StopFilterFactory"
          words="stopwords.txt"/>
  <filter class="solr.PorterStemFilterFactory"/>
</analyzer>
<analyzer type="query"> ... </analyzer>

# Analysis screen / API
GET /solr/techproducts/analysis/field
    ?
analysis.fieldtype=text_general&analysis.fieldvalu
e=Foxes
```

UPDATE / COMMIT

```
# JSON docs update handler
POST /solr/techproducts/update/json/docs
{ "id": "sp2500", "title": "Solr in Action",
  "price": 39.99, "cat": ["book","search"] }

# Solr update format (add/delete/commit)
POST /solr/techproducts/update
[ {"id":"sp2500","title":{"set":"Solr in Action"}}
]

# commit control
POST .../update?commitWithin=5000
POST .../update?commit=true
DELETE: {"delete":{"id":"sp2500"}}
       {"delete":{"query":"cat:book"}}

# solrconfig.xml soft-commit / autocommit
<autoSoftCommit><maxTime>1000</maxTime>
</autoSoftCommit>
<autoCommit><maxTime>15000</maxTime>
  <openSearcher>false</openSearcher></autoCommit>
```

EDISMAX PARAMETERS

```
q={!edismax qf="title^3 body" pf="title^5"
   bq="inStock:true^2"
bf="recip(price,1,1000,1000)"
   mm="2<-1 5<-2" tie="0.1"}solr search

qf   fields + boosts searched (title^3 body)
pf   phrase-boost fields (adjacent term proximity)
bq   boost query (additive score bump)
bf   boost function (function-query score bump)
mm   minimum should match (2<-1 5<-2 = clause
rules)
tie  tie-breaker for dismax max-of-fields scoring
```

LOCAL PARAMS SYNTAX

```
{!type key=value key2=value2}query-text

q={!dismax qf="title body" mm=2}solr cloud
fq={!term f=cat}book
q={!lucene df=title v=$qq}&qq=solr

# common types
lucene, dismax, edismax, term, raw, frange, knn,
join
```

JSON REQUEST API

```
POST /solr/techproducts/query
{
  "query": "memory",
  "filter": [ "inStock:true", "price:[10 TO 100]"
],
  "fields": [ "id", "title", "price", "score" ],
  "limit": 10,
  "offset": 0,
  "sort": "score desc, id asc",
  "params": { "qf": "title^2 body", "defType":
"edismax" }
}
```

FACETING + JSON FACET API

```
# classic faceting (request params)
GET .../select?q=*:*&facet=true

&facet.field=cat&facet.mincount=1&facet.limit=20

# JSON Facet API (nested, more powerful)
"facet": {
  "categories": { "type": "terms", "field": "cat",
                  "limit": 10, "mincount": 1,
    "facet": { "avg_price": "avg(price)" } },
  "price_ranges": { "type": "range", "field":
"price",
                    "start": 0, "end": 500,
"gap": 50 }
}
```

HIGHLIGHTING

```
GET .../select?q=title:solr
    &hl=true&hl.fl=title,body
    &hl.method=unified
    &hl.snippets=3&hl.fragsize=120
    &hl.simple.pre=<em>&hl.simple.post=</em>

# hl.method: unified | original | fastVector
```

DENSE VECTOR / KNN QUERY

```
<field name="vector" type="knn_vector"
       indexed="true" stored="true"/>
<fieldType name="knn_vector"
class="solr.DenseVectorField"
           vectorDimension="4"
similarityFunction="cosine"/>

# {!knn} local params query
q={!knn f=vector topK=10}[0.1,0.2,0.3,0.4]

# combine with a filter query
q={!knn f=vector topK=10}
[0.1,0.2,0.3,0.4]&fq=inStock:true
```

STREAMING EXPR. / PARALLEL SQL

```
# streaming expression (/stream handler)
search(collection1, q="*:*", fl="id,price",
       sort="price asc", qt="/export")

top(n=5, sort="price desc",
    search(collection1, q="cat:book",
fl="id,price"))

# Parallel SQL (/sql handler, JDBC-friendly)
SELECT id, price FROM collection1
WHERE price > 10 ORDER BY price DESC LIMIT 20
```

SOLRCLOUD TOPOLOGY

```
Collection "products"
 |
 +-- Shard 1 (hash range) ----+
 |    +-- Replica (leader)  -> Node A
 |    +-- Replica (follower)-> Node B
 |
 +-- Shard 2 (hash range) ----+
      +-- Replica (leader)  -> Node C
      +-- Replica (follower)-> Node A

ZooKeeper ensemble (odd # nodes, quorum)
 ZK-1  ZK-2  ZK-3
  |     |     |
  +--- cluster state, leader election,
       configs (configsets), overseer
```

COLLECTIONS API

```
GET /solr/admin/collections?action=CREATE
    &name=products&numShards=2&replicationFactor=2
    &collection.configName=products_config

...&action=DELETE&name=products
...&action=RELOAD&name=products
...&action=SPLITSHARD&collection=products&shard=sh
ard1
...&action=ADDREPLICA&collection=products&shard=sh
ard1
    &node=node2:8983_solr
...&action=BACKUP&name=bak1&collection=products
    &location=/backups
...&action=RESTORE&name=bak1&collection=products2
    &location=/backups
```

CACHES / SOLRCONFIG.XML

```
<filterCache class="solr.CaffeineCache"
             size="512" initialSize="512"
             autowarmCount="128"/>

<queryResultCache class="solr.CaffeineCache"
                  size="512" autowarmCount="32"/>

<documentCache class="solr.CaffeineCache"
               size="1024" autowarmCount="0"/>

<queryResultWindowSize>20</queryResultWindowSize>
# filterCache: fq bitsets; queryResultCache:
(q,sort)->
# ordered doc-id list; documentCache: stored
fields
```

SECURITY.JSON (AUTH PLUGIN)

```
{
  "authentication": {
    "class": "solr.BasicAuthPlugin",
    "credentials": {
      "solr": "CHANGE_ME_HASH!CHANGE_ME_SALT="
    }
  },
  "authorization": {
    "class": "solr.RuleBasedAuthorizationPlugin",
    "permissions": [
      { "name": "collection-admin-edit",
        "role": "admin" }
    ],
    "user-role": { "solr": "admin" }
  }
}
```

SOLRJ (JAVA CLIENT)

```
Http2SolrClient client = new
Http2SolrClient.Builder(
    "http://localhost:8983/solr").build();

SolrInputDocument doc = new SolrInputDocument();
doc.addField("id", "sp2500");
doc.addField("title", "Solr in Action");
client.add("techproducts", doc);
client.commit("techproducts");

SolrQuery query = new SolrQuery("title:solr");
query.setRows(10).addFilterQuery("inStock:true");
QueryResponse rsp = client.query("techproducts",
query);
SolrDocumentList docs = rsp.getResults();
```