CSV Formatted Index Updates

CSV formatted update requests may be sent to Solr’s /update handler using Content-Type: application/csv or Content-Type: text/csv.

A sample CSV file is provided at example/exampledocs/books.csv that you can use to add some documents to the Solr techproducts example:

curl 'http://localhost:8983/solr/my\_collection/update?commit=true' --data-binary @example/exampledocs/books.csv -H 'Content-type:application/csv'

CSV Update Convenience Paths

In addition to the /update handler, there is an additional CSV specific request handler path available by default in Solr, that implicitly override the behavior of some request parameters:

| **Path** | **Default Parameters** |
| --- | --- |
| /update/csv | stream.contentType=application/csv |

The /update/csv path may be useful for clients sending in CSV formatted update commands from applications where setting the Content-Type proves difficult.

CSV Update Parameters

The CSV handler allows the specification of many parameters in the URL in the form: f.*parameter*.*optional\_fieldname*=*value* .

The table below describes the parameters for the update handler.

| **Parameter** | **Usage** | **Global (g) or Per Field (f)** | **Example** |
| --- | --- | --- | --- |
| separator | Character used as field separator; default is "," | g,(f: see split) | separator=%09 |
| trim | If true, remove leading and trailing whitespace from values. Default=false. | g,f | f.isbn.trim=true trim=false |
| header | Set to true if first line of input contains field names. These will be used if the **fieldnames** parameter is absent. | g |  |
| fieldnames | Comma separated list of field names to use when adding documents. | g | fieldnames=isbn,price,title |
| literal.<field\_name> | A literal value for a specified field name. | g | literal.color=red |
| skip | Comma separated list of field names to skip. | g | skip=uninteresting,shoesize |
| skipLines | Number of lines to discard in the input stream before the CSV data starts, including the header, if present. Default=0. | g | skipLines=5 |
| encapsulator | The character optionally used to surround values to preserve characters such as the CSV separator or whitespace. This standard CSV format handles the encapsulator itself appearing in an encapsulated value by doubling the encapsulator. | g,(f: see split) | encapsulator=" |
| escape | The character used for escaping CSV separators or other reserved characters. If an escape is specified, the encapsulator is not used unless also explicitly specified since most formats use either encapsulation or escaping, not both | g | escape=\ |
| keepEmpty | Keep and index zero length (empty) fields. Default=false. | g,f | f.price.keepEmpty=true |
| map | Map one value to another. Format is value:replacement (which can be empty.) | g,f | map=left:right f.subject.map=history:bunk |
| split | If true, split a field into multiple values by a separate parser. | f |  |
| overwrite | If true (the default), check for and overwrite duplicate documents, based on the uniqueKey field declared in the Solr schema. If you know the documents you are indexing do not contain any duplicates then you may see a considerable speed up setting this to false. | g |  |
| commit | Issues a commit after the data has been ingested. | g |  |
| commitWithin | Add the document within the specified number of milliseconds. | g | commitWithin=10000 |
| rowid | Map the rowid (line number) to a field specified by the value of the parameter, for instance if your CSV doesn’t have a unique key and you want to use the row id as such. | g | rowid=id |
| rowidOffset | Add the given offset (as an int) to the rowid before adding it to the document. Default is 0 | g | rowidOffset=10 |

Indexing Tab-Delimited files

The same feature used to index CSV documents can also be easily used to index tab-delimited files (TSV files) and even handle backslash escaping rather than CSV encapsulation.

For example, one can dump a MySQL table to a tab delimited file with:

**SELECT** \* **INTO** OUTFILE '/tmp/result.txt' **FROM** mytable;

This file could then be imported into Solr by setting the separator to tab (%09) and the escape to backslash (%5c).

curl 'http://localhost:8983/solr/my\_collection/update/csv?commit=true&separator=%09&escape=%5c' --data-binary @/tmp/result.txt