

# BIBTOOL Quick Reference Card

for BIBTOOL version 2.57 — see also <http://www.gerd.neugebauer.de/software/TeX/BibTool/>  
©2019 Gerd Neugebauer (gene@gerd-neugebauer.de)

---

## Command line options

`-- rsc_command`  
Perform resource command as if given in a file.

`-A type`  
Determine key disambiguation. *type* in 0, a, A,

`-d`  
Check double entries.

`-f key_format`  
Generate keys according to *key\_format*

`-F`  
Enable key generation with free key format.

`-h`  
Print short help and exit.

`-i input_file`  
Mark a file to be processed later.

`-k`  
Make keys with the short format.

`-K`  
Make keys with the long format.

`-o output_file`  
Send the output to *output\_file*.

`-q`  
Suppress warning messages.

`-r resource_file`  
Read the resource file *resource\_file*.

`-R`  
Load the default resource file now.

`-s`  
Sort the result.

`-S`  
Sort the result in reverse order.

`-v`  
Turn on verbose messages about the actions performed.

`-x aux_file`  
Extract those entries mentioned in *aux\_file*.

`-X regex`  
Extract entries matching *regex*.

## Libraries

`check_y` Check the value of the year.

`default` All default settings.

`field` Redefine field names.

`brace` Use braces as delimiters.

`improve` Apply improvements.

`iso2tex` Translate ISO 8859/1 characters.

`iso_def` Define ISO 8859/1 characters for formatting.

`month` Introduce strings for month names.

`opt` Remove OPT in field names.

`sort_fld` Specify sort order for fields.

`tex_def` Define T<sub>E</sub>X macros for formatting.

`biblatex` Capitalize fields known to bibL<sub>A</sub>T<sub>E</sub>X.

## General

`resource.search.path = {dir1:dir2...}`

`resource {file}`

`bibtex.search.path = {dir1:dir2...}`

`bibtex.env.name = {ENV_NAME}`

`env.separator = {c}`

`dir.file.separator = {c}`

`print {message}`

`quiet = OnOff`

`verbose = OnOff`

`crossref.limit = {n}`

## Reading and Printing

`input {bib_file}`

`output.file = {file}`

`pass.comments = OnOff`

`new.entry.type {type}`

`print.align = n`

`print.align.key = n`

`print.align.preamble = n`

`print.align.comment = n`

`print.braces = OnOff`

`print.comma.at.end = OnOff`

`print.deleted.entries = OnOff`

`print.deleted.prefix = {prefix}`

`print.indent = n`

`print.line.length = n`

`print.newline = n`

`print.parentheses = OnOff`

`print.terminal.comma = OnOff`

`print.use.tab = OnOff`

`print.wide.equal = OnOff`

`suppress.initial.newline = OnOff`

`new.field.type {new=old}`

`symbol.type = type`

upper, lower, cased

## Sorting

`sort = OnOff`

`sort.cased = OnOff`

`sort.reverse = OnOff`

`sort.format = {format}`

`sort.order {...}`

`sort.macros = OnOff`

## Searching (Extraction)

`tex.define {macro[arg]=text}`

`extract.file {file}`

`select {field1...fieldn "regex"}`

`select {type1...typen}`

`select.by.string {field1...fieldn "regex"}`

`select.by.string.ignore {chars}`

`select.case.sensitive = OnOff`

`select.fields = {field1,field2,...}`

## Field Manipulation

`add.field {field=value}`

`delete.field {field}`

`rewrite.rule { pattern }`  
delete all matching fields

`rewrite.rule { pattern # replacement }`  
rewrite all fields

`rewrite.rule {f1...fn # pattern # replacement }`

rewrite some fields

`rewrite.case.sensitive = OnOff`

`rewrite.limit = {n}`

## Checks

`check.double = OnOff`

`check.do.delete = OnOff`

`check.rule {field # pattern # message}`

`check.case.sensitive = OnOff`

## Strings

`macro.file {file}`

`print.all.strings = OnOff`

`expand.macros = OnOff`

`expand.crossref = OnOff`

---

---

## BI<sub>T</sub>E<sub>X</sub>1.0

apply.alias = *OnOff*  
apply.include = *OnOff*  
apply.modify = *OnOff*  
key.make.alias = *OnOff*

## Counting

count.all = *OnOff*  
count.used = *OnOff*

## Key Generation

preserve.keys = *OnOff*  
preserve.key.case = *OnOff*  
key.format = {*format*}  
    special values: short, long, short.need,  
    long.need, empty  
key.generation = *OnOff*  
default.key = {*key*}  
key.base = *base*  
    values: upper, lower, digit  
key.number.separator = {*s*}  
key.expand.macros = *OnOff*  
fmt.name.title = {*s*}  
fmt.title.title = {*s*}  
fmt.name.name = {*s*}  
fmt.inter.name = {*s*}

fmt.name.pre = {*s*}  
fmt.et.al = {*s*}  
fmt.word.separator = {*s*}  
new.format.type = {*n*=*"spec"*}

## Name Formatting Specification

Use *n* letters. Use *m* name parts. Insert *pre* before, *mid* between, and *post* after the words. Translate according to the *s* parameter ('+', '-', '\*', ').

%*sn.mf*[*mid*][*pre*][*post*]  
    format first names.  
%*sn.mv*[*mid*][*pre*][*post*]  
    format “von” part.  
%*sn.ml*[*mid*][*pre*][*post*]  
    format last name.  
%*sn.mj*[*mid*][*pre*][*post*]  
    format “junior” part.

## Format Specifications

### Pseudo fields:

\$key  
\$default.key  
\$sortkey  
\$source  
\$type  
@type

\$day  
\$month  
\$mon  
\$year  
\$hour  
\$minute  
\$second  
\$user  
\$hostname

### Formatting Fields:

%±*x.y* *n*(*field*)  
    format *y* characters of *x* last names.  
%±*x.y* *N*(*field*)  
    format *y* characters of *x* names.  
%±*x.y* *p*(*field*)  
    format *x* names according to the name format *y*.  
%±*x.y* *d*(*field*)  
    format at most *x* digits of the *y*<sup>th</sup> number.  
%±*x.y* *D*(*field*)  
    format *x* digits of the *y*<sup>th</sup> number without truncation.  
%±*x* *s*(*field*)  
    format *x* string characters.  
%±*x.y* *t*(*field*)  
    format *x* sentence words of length *y*.  
%±*x.y* *T*(*field*)  
    format *x* sentence words of length *y*.  
    (Words ignored)

%±*x.y* *w*(*field*)  
    format *x* words of length *y*.  
%±*x* *W*(*field*)  
    format *x* words of length *y*. (Words ignored)  
%±*x.y* #*n*(*field*)  
    test whether the number of names is between *x* and *y*.  
%±*x.y* #*N*(*field*)  
    test whether the number of names is between *x* and *y*.  
%±*x.y* #*p*(*field*)  
    test whether the number of names is between *x* and *y*.  
%±*x.y* #*s*(*field*)  
    test whether the number of characters is between *x* and *y*.  
%±*x.y* #*t*(*field*)  
    test whether the number of words is between *x* and *y*.  
%±*x.y* #*T*(*field*)  
    test whether the number of not ignored words is between *x* and *y*.  
%±*x.y* #*w*(*field*)  
    test whether the number of words is between *x* and *y*.  
%±*x.y* #*W*(*field*)  
    test whether the number of not ignored words is between *x* and *y*.

---