etextools The `etextools` package provides enhancements to list macros defined by `etoolbox` and a few other tools for command definitions. The package redefines list handling macros in a way incompatible with `biblatex`.

If you must load the `etextools` package at all costs, define the control sequence `\blx@noerroretextools` before you load `biblatex`. If `\blx@noerroretextools` is defined, no error will be issued if `etextools` is loaded, the message is degraded to a warning instead. In that case you need to make sure that all redefined macros used by `biblatex` (currently only `\forlistloop`) have their original `etoolbox` definitions when `biblatex` is loaded.

1.5.6 Compatibility Matrix for biber

`biber` versions are closely coupled with `biblatex` versions. You need to have the right combination of the two. `biber` will throw a fatal error during processing if it encounters information which comes from a `biblatex` version which is incompatible. Table 1 shows a compatibility matrix for the recent versions.

2 Database Guide

This section describes the default data model defined in the `blx-dm.def` file which is part of `biblatex`. The data model is defined using the macros documented in § 4.5.4. It is possible to redefine the data model which both `biblatex` and `biber` use so that datasources can contain new entrytypes and fields (which of course will need style support). The data model specification also allows for constraints to be defined so that data sources can be validated against the data model (using `biber`'s `--validate-datamodel` option). Users who want to customise the data model need to look at the `blx-dm.def` file and to read § 4.5.4.

All entry types and field names are given in all-lowercase form here. This is how the entry types and field names are given in the data model. While the `biber`/BibTeX input side is case insensitive, the LaTeX side is case sensitive and uses the exact capitalisation from the data model. This means that the input in the `bib` file may use any capitalisation of entry types and field names, but when the fields are used in the LaTeX document—for example in `\citefield`—the capitalisation must match the capitalisation in the data model, for standard types and fields that would be all lowercase.

2.1 Entry Types

This section gives an overview of the entry types supported by the default `biblatex` data model along with the fields supported by each type.

2.1.1 Regular Types

The lists below indicate the fields supported by each entry type. Note that the mapping of fields to an entry type is ultimately at the discretion of the bibliography style. The lists below therefore serve two purposes. They indicate the fields supported by the standard styles which come with this package and they also serve as a model for custom styles. Note that the 'required' fields are not strictly required in all cases, see § 2.3.2 for details. The fields marked as 'optional' are optional in a technical sense. Bibliographical formatting rules usually require more than just the 'required' fields. The default data model defined a few constraints for the format of date fields, ISBNs and some special fields like `gender` but the constraints are only used if validating

Table 1: biber/biblatex compatibility matrix

Biber version	biblatex version
2.21	3.21
2.20	3.20
2.19	3.19
2.18	3.18
2.17	3.17
2.16	3.16
2.15	3.15
2.14	3.14
2.13	3.13
2.12	3.12
2.11	3.11
2.10	3.10
2.9	3.9
2.8	3.8
2.7	3.7
2.6	3.5, 3.6
2.5	3.4
2.4	3.3
2.3	3.2
2.2	3.1
2.1	3.0
2.0	3.0
1.9	2.9
1.8	2.8
1.7	2.7
1.6	2.6
1.5	2.5
1.4	2.4
1.3	2.3
1.2	2.1, 2.2
1.1	2.1
1.0	2.0
0.9.9	1.7x
0.9.8	1.7x
0.9.7	1.7x
0.9.6	1.7x
0.9.5	1.6x
0.9.4	1.5x
0.9.3	1.5x
0.9.2	1.4x
0.9.1	1.4x
0.9	1.4x

against the data model with biber's --validate-datamodel option. Generic fields like abstract and annotation or label and shorthand are not included in the lists below because they are independent of the entry type. The special fields discussed in § 2.2.3, which are also independent of the entry type, are not included in the lists either. See the default data model specification in the file blx-dm.def which comes with biblatex for a complete specification.

The 'alias' relation referred to in this subsection is the 'soft alias' defined with \DeclareBibliographyAlias. That means that the alias will use the same bibliography driver as the type it is aliased to, but that its type-specific formatting is still handled independently of the aliased type.

article An article in a journal, magazine, newspaper, or other periodical which forms a self-contained unit with its own title. The title of the periodical is given in the journaltitle field. If the issue has its own title in addition to the main title of the periodical, it goes in the issuetitle field. Note that editor and related fields refer to the journal while translator and related fields refer to the article.

Required fields: author, title, journaltitle, year/date

Optional fields: translator, annotator, commentator, subtitle, titleaddon, editor, editora, editorb, editorc, journalsubtitle, journaltitleaddon, issuetitle, issuesubtitle, issuetitleaddon, language, origlanguage, series, volume, number, eid, issue, month, pages, version, note, issn, addendum, pubstate, doi, eprint, eprintclass, eprinttype, url, urldate

book A single-volume book with one or more authors where the authors share credit for the work as a whole. This entry type also covers the function of the @inbook type of traditional BibTeX, see § 2.3.1 for details.

Required fields: author, title, year/date

Optional fields: editor, editora, editorb, editorc, translator, annotator, commentator, introduction, foreword, afterword, subtitle, titleaddon, maintitle, mainsubtitle, maintitleaddon, language, origlanguage, volume, part, edition, volumes, series, number, note, publisher, location, isbn, eid, chapter, pages, pagetotal, addendum, pubstate, doi, eprint, eprintclass, eprinttype, url, urldate

mvbook A multi-volume @book. For backwards compatibility, multi-volume books are also supported by the entry type @book. However, it is advisable to make use of the dedicated entry type @mvbook.

Required fields: author, title, year/date

Optional fields: editor, editora, editorb, editorc, translator, annotator, commentator, introduction, foreword, afterword, subtitle, titleaddon, language, origlanguage, edition, volumes, series, number, note, publisher, location, isbn, pagetotal, addendum, pubstate, doi, eprint, eprintclass, eprinttype, url, urldate

inbook A part of a book which forms a self-contained unit with its own title. Note that the profile of this entry type is different from standard BibTeX, see § 2.3.1.

Required fields: author, title, booktitle, year/date

Optional fields: `bookauthor`, `editor`, `editora`, `editorb`, `editorc`, `translator`, `annotator`, `commentator`, `introduction`, `foreword`, `afterword`, `subtitle`, `titleaddon`, `maintitle`, `mainsubtitle`, `maintitleaddon`, `booksubtitle`, `booktitleaddon`, `language`, `origlanguage`, `volume`, `part`, `edition`, `volumes`, `series`, `number`, `note`, `publisher`, `location`, `isbn`, `eid`, `chapter`, `pages`, `addendum`, `pubstate`, `doi`, `eprint`, `eprintclass`, `eprinttype`, `url`, `urldate`

bookinbook This type is similar to `@inbook` but intended for works originally published as a stand-alone book. A typical example are books reprinted in the collected works of an author.

suppbook Supplemental material in a `@book`. This type is closely related to the `@inbook` entry type. While `@inbook` is primarily intended for a part of a book with its own title (e. g., a single essay in a collection of essays by the same author), this type is provided for elements such as prefaces, introductions, forewords, afterwords, etc. which often have a generic title only. Style guides may require such items to be formatted differently from other `@inbook` items. The standard styles will treat this entry type as an alias for `@inbook`.

booklet A book-like work without a formal publisher or sponsoring institution. Use the field `howpublished` to supply publishing information in free format, if applicable. The field `type` may be useful as well.

Required fields: `author`/`editor`, `title`, `year`/`date`

Optional fields: `subtitle`, `titleaddon`, `language`, `howpublished`, `type`, `note`, `location`, `eid`, `chapter`, `pages`, `pagetotal`, `addendum`, `pubstate`, `doi`, `eprint`, `eprintclass`, `eprinttype`, `url`, `urldate`

collection A single-volume collection with multiple, self-contained contributions by distinct authors which have their own title. The work as a whole has no overall author but it will usually have an editor.

Required fields: `editor`, `title`, `year`/`date`

Optional fields: `editora`, `editorb`, `editorc`, `translator`, `annotator`, `commentator`, `introduction`, `foreword`, `afterword`, `subtitle`, `titleaddon`, `maintitle`, `mainsubtitle`, `maintitleaddon`, `language`, `origlanguage`, `volume`, `part`, `edition`, `volumes`, `series`, `number`, `note`, `publisher`, `location`, `isbn`, `eid`, `chapter`, `pages`, `pagetotal`, `addendum`, `pubstate`, `doi`, `eprint`, `eprintclass`, `eprinttype`, `url`, `urldate`

mvcollection A multi-volume `@collection`. For backwards compatibility, multi-volume collections are also supported by the entry type `@collection`. However, it is advisable to make use of the dedicated entry type `@mvcollection`.

Required fields: `editor`, `title`, `year`/`date`

Optional fields: `editora`, `editorb`, `editorc`, `translator`, `annotator`, `commentator`, `introduction`, `foreword`, `afterword`, `subtitle`, `titleaddon`, `language`, `origlanguage`, `edition`, `volumes`, `series`, `number`, `note`, `publisher`, `location`, `isbn`, `pagetotal`, `addendum`, `pubstate`, `doi`, `eprint`, `eprintclass`, `eprinttype`, `url`, `urldate`

incollection A contribution to a collection which forms a self-contained unit with a distinct author and title. The `author` refers to the `title`, the `editor` to the `booktitle`, i. e., the title of the collection.

Required fields: author, title, editor, booktitle, year/date

Optional fields: editor, editora, editorb, editorc, translator, annotator, commentator, introduction, foreword, afterword, subtitle, titleaddon, maintitle, mainsubtitle, maintitleaddon, booksubtitle, booktitleaddon, language, origlanguage, volume, part, edition, volumes, series, number, note, publisher, location, isbn, eid, chapter, pages, addendum, pubstate, doi, eprint, eprintclass, eprinttype, url, urldate

suppcollection Supplemental material in a @collection. This type is similar to @suppbook but related to the @collection entry type. The standard styles will treat this entry type as an alias for @incollection.

dataset A data set or a similar collection of (mostly) raw data.

Required fields: author/editor, title, year/date

Optional fields: subtitle, titleaddon, language, edition, type, series, number, version, note, organization, publisher, location, addendum, pubstate, doi, eprint, eprintclass, eprinttype, url, urldate

manual Technical or other documentation, not necessarily in printed form. The author or editor is omissible in terms of § 2.3.2.

Required fields: author/editor, title, year/date

Optional fields: subtitle, titleaddon, language, edition, type, series, number, version, note, organization, publisher, location, isbn, eid, chapter, pages, pagetotal, addendum, pubstate, doi, eprint, eprintclass, eprinttype, url, urldate

misc A fallback type for entries which do not fit into any other category. Use the field howpublished to supply publishing information in free format, if applicable. The field type may be useful as well. author, editor, and year are omissible in terms of § 2.3.2.

Required fields: author/editor, title, year/date

Optional fields: subtitle, titleaddon, language, howpublished, type, version, note, organization, location, month, addendum, pubstate, doi, eprint, eprintclass, eprinttype, url, urldate

online An online resource. author, editor, and year are omissible in terms of § 2.3.2. This entry type is intended for sources such as web sites which are intrinsically online resources. Note that all entry types support the url field. For example, when adding an article from an online journal, it may be preferable to use the @article type and its url field.

Required fields: author/editor, title, year/date, doi/eprint/url

Optional fields: subtitle, titleaddon, language, version, note, organization, month, addendum, pubstate, eprintclass, eprinttype, urldate

patent A patent or patent request. The number or record token is given in the number field. Use the type field to specify the type and the location field to indicate the scope of the patent, if different from the scope implied by the type. Note that the location field is treated as a key list with this entry type, see § 2.2.1 for details.

Required fields: author, title, number, year/date

Optional fields: holder, subtitle, titleaddon, type, version, location, note, month, addendum, pubstate, doi, eprint, eprintclass, eprinttype, url, urldate

periodical An complete issue of a periodical, such as a special issue of a journal. The title of the periodical is given in the title field. If the issue has its own title in addition to the main title of the periodical, it goes in the issuetitle field. The editor is omissible in terms of § 2.3.2.

Required fields: editor, title, year/date

Optional fields: editora, editorb, editorc, subtitle, titleaddon, issuetitle, issuesubtitle, issuetitleaddon, language, series, volume, number, issue, month, note, issn, addendum, pubstate, doi, eprint, eprintclass, eprinttype, url, urldate

suppperiodical Supplemental material in a @periodical. This type is similar to @suppbook but related to the @periodical entry type. The role of this entry type may be more obvious if you bear in mind that the @article type could also be called @inperiodical. This type may be useful when referring to items such as regular columns, obituaries, letters to the editor, etc. which only have a generic title. Style guides may require such items to be formatted differently from articles in the strict sense of the word. The standard styles will treat this entry type as an alias for @article.

proceedings A single-volume conference proceedings. This type is very similar to @collection. It supports an optional organization field which holds the sponsoring institution. The editor is omissible in terms of § 2.3.2.

Required fields: title, year/date

Optional fields: editor, subtitle, titleaddon, maintitle, mainsubtitle, maintitleaddon, eventtitle, eventtitleaddon, eventdate, venue, language, volume, part, volumes, series, number, note, organization, publisher, location, month, isbn, eid, chapter, pages, pagetotal, addendum, pubstate, doi, eprint, eprintclass, eprinttype, url, urldate

mvproceedings A multi-volume @proceedings entry. For backwards compatibility, multi-volume proceedings are also supported by the entry type @proceedings. However, it is advisable to make use of the dedicated entry type @mvproceedings

Required fields: title, year/date

Optional fields: editor, subtitle, titleaddon, eventtitle, eventtitleaddon, eventdate, venue, language, volumes, series, number, note, organization, publisher, location, month, isbn, pagetotal, addendum, pubstate, doi, eprint, eprintclass, eprinttype, url, urldate

inproceedings An article in a conference proceedings. This type is similar to @incollection. It supports an optional organization field.

Required fields: author, title, booktitle, year/date

Optional fields: editor, subtitle, titleaddon, maintitle, mainsubtitle, maintitleaddon, booksubtitle, booktitleaddon, eventtitle, eventtitleaddon, eventdate, venue, language, volume, part, volumes, series, number, note, organization, publisher, location, month, isbn, eid, chapter, pages, addendum, pubstate, doi, eprint, eprintclass, eprinttype, url, urldate reference A single-volume work of reference such as an encyclopedia or a dictionary. This is a more specific variant of the generic `@collection` entry type. The standard styles will treat this entry type as an alias for `@collection`.

mvreference A multi-volume `@reference` entry. The standard styles will treat this entry type as an alias for `@mvcollection`. For backwards compatibility, multi-volume references are also supported by the entry type `@reference`. However, it is advisable to make use of the dedicated entry type `@mvreference`.

inreference An article in a work of reference. This is a more specific variant of the generic `@incollection` entry type. The standard styles will treat this entry type as an alias for `@incollection`.

report A technical report, research report, or white paper published by a university or some other institution. Use the `type` field to specify the type of report. The sponsoring institution goes in the `institution` field.

Required fields: `author`, `title`, `type`, `institution`, `year/date`

Optional fields: `subtitle`, `titleaddon`, `language`, `number`, `version`, `note`, `location`, `month`, `isrn`, `eid`, `chapter`, `pages`, `pagetotal`, `addendum`, `pubstate`, `doi`, `eprint`, `eprintclass`, `eprinttype`, `url`, `urldate`

set An entry set. This entry type is special, see § 3.14.5 for details.

software Computer software. The standard styles will treat this entry type as an alias for `@misc`.

thesis A thesis written for an educational institution to satisfy the requirements for a degree. Use the `type` field to specify the type of thesis.

Required fields: `author`, `title`, `type`, `institution`, `year/date`

Optional fields: `subtitle`, `titleaddon`, `language`, `note`, `location`, `month`, `isbn`, `eid`, `chapter`, `pages`, `pagetotal`, `addendum`, `pubstate`, `doi`, `eprint`, `eprintclass`, `eprinttype`, `url`, `urldate`

unpublished A work with an author and a title which has not been formally published, such as a manuscript or the script of a talk. Use the fields `howpublished` and `note` to supply additional information in free format, if applicable.

Required fields: `author`, `title`, `year/date`

Optional fields: `subtitle`, `titleaddon`, `type`, `eventtitle`, `eventtitleaddon`, `eventdate`, `venue`, `language`, `howpublished`, `note`, `location`, `isbn`, `month`, `addendum`, `pubstate`, `doi`, `eprint`, `eprintclass`, `eprinttype`, `url`, `urldate`

xdata This entry type is special. `@xdata` entries hold data which may be inherited by other entries using the `xdata` field. Entries of this type only serve as data containers; they may not be cited or added to the bibliography. See § 3.14.6 for details.

custom[a–f] Custom types for special bibliography styles. The standard styles defined no bibliography drivers for these types and will fall back to using the driver for `@misc`.

2.1.2 Type Aliases

The entry types listed in this section are provided for backwards compatibility with traditional BibTeX styles. These aliases are resolved by the backend as the data is processed. `biblatex` and the styles will see only the entry type the alias points to (the target), not the alias name (the source). In particular `biblatex`-side per-type operations like type-specific formatting and filtering only work for the target type, not the source type. This 'hard alias' is unlike the 'soft alias' relation in the previous subsection. The relevant mappings for the `bibtex` driver can be found in § A.1.

conference	A legacy alias for `@inproceedings`.
electronic	An alias for `@online`.
mastersthesis	Similar to `@thesis` except that the `type` field is optional and defaults to the localised term 'Master's thesis'. You may still use the `type` field to override that.
phdthesis	Similar to `@thesis` except that the `type` field is optional and defaults to the localised term 'PhD thesis'. You may still use the `type` field to override that.
techreport	Similar to `@report` except that the `type` field is optional and defaults to the localised term 'technical report'. You may still use the `type` field to override that.
www	An alias for `@online`, provided for `jurabib` compatibility.

2.1.3 Non-standard Types

The types in this section are similar to the custom types `@custom[a--f]`, i. e., the standard bibliography styles provide no bibliography drivers for these types. In the standard styles they will use the bibliography driver for `@misc` entries—exceptions to this rule are noted in the descriptions below. The types are known to the default data model and will be happily accepted by `biber`.

artwork	Works of the visual arts such as paintings, sculpture, and installations.
audio	Audio recordings, typically on audio CD, DVD, audio cassette, or similar media. See also `@music`.
bibnote	This special entry type is not meant to be used in the `bib` file like other types. It is provided for third-party packages like `notes2bib` which merge notes into the bibliography. The notes should go into the `note` field. Be advised that the `@bibnote` type is not related to the `\defbibnote` command in any way. `\defbibnote` is for adding comments at the beginning or the end of the bibliography, whereas the `@bibnote` type is meant for packages which render endnotes as bibliography entries.
commentary	Commentaries which have a status different from regular books, such as legal commentaries.
image	Images, pictures, photographs, and similar media.
jurisdiction	Court decisions, court recordings, and similar things.
legislation	Laws, bills, legislative proposals, and similar things.
legal	Legal documents such as treaties.
letter	Personal correspondence such as letters, emails, memoranda, etc.

movie	Motion pictures. See also `@video`.
music	Musical recordings. This is a more specific variant of `@audio`.
performance	Musical and theatrical performances as well as other works of the performing arts. This type refers to the event as opposed to a recording, a score, or a printed play.
review	Reviews of some other work. This is a more specific variant of the `@article` type. The standard styles will treat this entry type as an alias for `@article`.
standard	National and international standards issued by a standards body such as the International Organization for Standardization.
video	Audiovisual recordings, typically on DVD, VHS cassette, or similar media. See also `@movie`.

2.2 Entry Fields

This section gives an overview of the fields supported by the `biblatex` default data model. See § 2.2.1 for an introduction to the data types used by the data model specification and §§ 2.2.2 and 2.2.3 for the actual field listings.

2.2.1 Data Types

In datasources such as a `bib` file, all bibliographic data is specified in fields. Some of those fields, for example `author` and `editor`, may contain a list of items. This list structure is implemented by the BibTeX file format via the keyword '`and`', which is used to separate the individual items in the list. The `biblatex` package implements three distinct data types to handle bibliographic data: name lists, literal lists, and fields. There are also several list and field subtypes and a content type which can be used to semantically distinguish fields which are otherwise not distinguishable on the basis of only their datatype (see § 4.5.4). This section gives an overview of the data types supported by this package. See §§ 2.2.2 and 2.2.3 for information about the mapping of the BibTeX file format fields to `biblatex`'s data types.

Name lists are parsed and split up into the individual items at the `and` delimiter. Each item in the list is then dissected into the name part components: by default the given name, the name prefix (von, van, of, da, de, della, …), the family name, and the name suffix (junior, senior, …). The valid name parts can be customised by changing the datamodel definition described in § 4.2.3. Name lists may be truncated in the `bib` file with the keyword '`and others`'. Typical examples of name lists are `author` and `editor`.

Name list fields automatically have an `\ifuse*` test created as per the name lists in the default data model (see § 4.6.2). They also automatically have an `ifuse*` option created which controls labelling and sorting behaviour with the name (see § 3.1.3.1). `biber` supports a customisable set of name parts but currently this is defined to be the same set of parts as supported by traditional BibTeX:

- Family name (also known as 'last' part)
- Given name (also known as 'first' part)
- Name prefix (also known as 'von' part)
- Name suffix (also known as 'Jr' part)

The supported list of name parts is defined as a constant list in the default data model using the `\DeclareDatamodelConstant` command (see 4.5.4). However, it is not enough to simply add to this list in order to add support for another name part as name parts typically have to be hard coded into bibliography drivers and the backend processing. See the example file `93-nameparts.tex` for details on how to define and use custom name parts. Also see `\DeclareUniquenameTemplate` in § 4.11.4 for information on how to customise name disambiguation using custom name parts.

Literal lists are parsed and split up into the individual items at the `and` delimiter but not dissected further. Literal lists may be truncated in the `bib` file with the keyword '`and others`'. There are two subtypes:

- **Literal lists** in the strict sense are handled as described above. The individual items are simply printed as is. Typical examples of such literal lists are `publisher` and `location`.
- **Key lists** are a variant of literal lists which may hold printable data or localisation keys. For each item in the list, styles should perform a test to determine whether it is a known localisation key (the localisation keys defined by default are listed in § 4.9.2). If so, the localised string should be printed. If not, the item should be printed as is. The standard styles are set up to exhibit this behaviour for all key lists listed below. New key lists do not automatically perform this test, it has to be implemented explicitly via the list format. A typical example of a key list is `language`.

Fields are usually printed as a whole. There are several subtypes:

- **Literal fields** are printed as is. Typical examples of literal fields are `title` and `note`.
- **Range fields** consist of one or more ranges where all dashes are normalized and replaced by the command `\bibrangedash`. A range is something optionally followed by one or more dashes optionally followed by some non-dash (e.g. `5--7`). Any number of consecutive dashes will only yield a single range dash. A typical example of a range field is the `pages` field. See also the `\bibrangessep` command which can be used to customise the separator between multiple ranges. Range fields will be skipped and will generate a warning if they do not consist of one or more ranges. You can normalise messy range fields before they are parsed using `\DeclareSourcemap` (see § 4.5.3).
- **Integer fields** hold integers which may be converted to ordinals or strings as they are printed. A typical example is the `extradate` or `volume` field. Such fields are sorted as integers. `biber` makes a (quite serious) effort to map non-arabic representations (roman numerals for example) to integers for sorting purposes. See the `noroman` option which can be used to suppress roman numeral parsing. This can help in cases where there is an ambiguity between parsing as roman numerals or alphanumeric (e.g. 'C'), see § 3.1.2.3.
- **Datepart fields** hold unformatted integers which may be converted to ordinals or strings as they are printed. A typical example is the `month` field. For every field of datatype `date` in the datamodel, datepart fields are automatically created with the following names: `<datetype>year`, `<datetype>endyear`, `<datetype>month`,

<datetype>endmonth, <datetype>day, <datetype>endday, <datetype>hour, <datetype>endhour, <datetype>minute, <datetype>endminute, <datetype>second, <datetype>endsecond, <datetype>timezone, <datetype>endtimezone. <datetype> is the string preceding 'date' for any datamodel field of datatype=date. For example, in the default datamodel, 'event', 'orig', 'url' and the empty string '' for the date field date.

Date fields hold a date specification in yyyy-mm-ddThh:nn[+-][hh[:nn]Z] format or a date range in yyyy-mm-ddThh:nn[+-][hh[:nn]Z]/yyyy-mm-ddThh:nn[+-][hh[:nn]Z] format and other formats permitted by ISO8601-2 Clause 4, level 1, see § 2.3.8. Date fields are special in that the date is parsed and split up into its datepart type components. The datepart components (see above) are automatically defined and recognised when a field of datatype date is defined in the datamodel. A typical example is the date field.

Verbatim fields are processed in verbatim mode and may contain special characters. Typical examples of verbatim fields are file and doi.

URI fields are processed in verbatim mode and may contain special characters. They are also URL-escaped if they don't look like they already are. The typical example of a uri field is url.

Separated value fields A separated list of literal values. Examples are the keywords and options fields. The separator can be configured to be any Perl regular expression via the xsvsep option which defaults to the usual BibTeX comma surrounded by optional whitespace.

Pattern fields A literal field which must match a particular pattern. An example is the gender field from § 2.2.3.

Key fields May hold printable data or localisation keys. Styles should perform a test to determine whether the value of the field is a known localisation key (the localisation keys defined by default are listed in § 4.9.2). If so, the localised string should be printed. If not, the value should be printed as is. The standard styles are set up to handle all key fields listed below in that way. New key fields do not automatically perform the test, it has to be enabled explicitly in the field format. A typical example is the type field.

Code fields Holds TeX code.

2.2.2 Data Fields

The fields listed in this section are the regular ones holding printable data in the default data model. The name on the left is the default data model name of the field as used by biblatex and its backend. The biblatex data type is given to the right of the name. See § 2.2.1 for explanation of the various data types.

Some fields are marked as 'label' fields which means that they are often used as abbreviation labels when printing bibliography lists in the sense of section § 3.8.3. biblatex automatically creates supporting macros for such fields. See § 3.8.3.

abstract field (literal)

This field is intended for recording abstracts in a bib file, to be printed by a special bibliography style. It is not used by all standard bibliography styles.

addendum field (literal)

Miscellaneous bibliographic data to be printed at the end of the entry. This is similar to the `note` field except that it is printed at the end of the bibliography entry.

afterword list (name)

The author(s) of an afterword to the work. If the author of the afterword is identical to the `editor` and/or `translator`, the standard styles will automatically concatenate these fields in the bibliography. See also `introduction` and `foreword`.

annotation field (literal)

This field may be useful when implementing a style for annotated bibliographies. It is not used by all standard bibliography styles. Note that this field is completely unrelated to `annotator`. The `annotator` is the author of annotations which are part of the work cited.

annotator list (name)

The author(s) of annotations to the work. If the annotator is identical to the `editor` and/or `translator`, the standard styles will automatically concatenate these fields in the bibliography. See also `commentator`.

author list (name)

The author(s) of the `title`.

authortype field (key)

The type of author. This field will affect the string (if any) used to introduce the author.

bookauthor list (name)

The author(s) of the `booktitle`.

bookpagination field (key)

If the work is published as part of another one, this is the pagination scheme of the enclosing work, i. e., `bookpagination` relates to `pagination` like `booktitle` to `title`. The value of this field will affect the formatting of the `pages` and `pagetotal` fields. The key should be given in the singular form. Possible keys are `page`, `column`, `line`, `verse`, `section`, and `paragraph`. See also `pagination` as well as § 2.3.12.

booksubtitle field (literal)

The subtitle related to the `booktitle`. If the `subtitle` field refers to a work which is part of a larger publication, a possible subtitle of the main work is given in this field. See also `subtitle`.

booktitle field (literal)

If the `title` field indicates the title of a work which is part of a larger publication, the title of the main work is given in this field. See also `title`.

booktitleaddon field (literal)

An annex to the `booktitle`, to be printed in a different font.

chapter field (literal)

A chapter or section or any other unit of a work.

commentator list (name)

The author(s) of a commentary to the work. Note that this field is intended for commented editions which have a commentator in addition to the author. If the work is a stand-alone commentary, the commentator should be given in the `author` field. If the commentator is identical to the `editor` and/or `translator`, the standard styles will automatically concatenate these fields in the bibliography. See also `annotator`.

date field (date)

The publication date. See also `month` and `year` as well as §§ 2.3.8 and 2.3.9.

doi field (verbatim)

The Digital Object Identifier of the work.

edition field (integer or literal)

The edition of a printed publication. This must be an integer, not an ordinal. Don't say `edition={First}` or `edition={1st}` but `edition={1}`. The bibliography style converts this to a language dependent ordinal. It is also possible to give the edition as a literal string, for example "Third, revised and expanded edition".

editor list (name)

The editor(s) of the `title`, `booktitle`, or `maintitle`, depending on the entry type. Use the `editortype` field to specify the role if it is different from '`editor`'. See § 2.3.6 for further hints.

editora list (name)

A secondary editor performing a different editorial role, such as compiling, redacting, etc. Use the `editoratype` field to specify the role. See § 2.3.6 for further hints.

editorb list (name)

Another secondary editor performing a different role. Use the `editorbtype` field to specify the role. See § 2.3.6 for further hints.

editorc list (name)

Another secondary editor performing a different role. Use the `editorctype` field to specify the role. See § 2.3.6 for further hints.

editortype field (key)

The type of editorial role performed by the `editor`. Roles supported by default are `editor`, `compiler`, `founder`, `continuator`, `redactor`, `reviser`, `collaborator`, `organizer`. The role '`editor`' is the default. In this case, the field is omissible. See § 2.3.6 for further hints.

editoratype field (key)

Similar to `editortype` but referring to the `editora` field. See § 2.3.6 for further hints.

editorbtype field (key)

Similar to `editortype` but referring to the `editorb` field. See § 2.3.6 for further hints.

editorctype field (key)

Similar to `editortype` but referring to the `editorc` field. See § 2.3.6 for further hints.

eid field (literal)

The electronic identifier of an `@article` or chapter-like section of a larger work often called 'article number', 'paper number' or the like. This field may replace the `pages` field for journals deviating from the classic pagination scheme of printed journals by only enumerating articles or papers and not pages.

Not to be confused with `number`, which for `@articles` subdivides the `volume`.

entrysubtype field (literal)

This field, which is not used by the standard styles, may be used to specify a subtype of an entry type. This may be useful for bibliography styles which support a finer-grained set of entry types.

eprint field (verbatim)

The electronic identifier of an online publication. This is roughly comparable to a DOI but specific to a certain archive, repository, service, or system. See § 3.14.7 for details. Also see `eprinttype` and `eprintclass`.

eprintclass field (literal)

Additional information related to the resource indicated by the `eprinttype` field. This could be a section of an archive, a path indicating a service, a classification of some sort, etc. See § 3.14.7 for details. Also see `eprint` and `eprinttype`.

eprinttype field (literal)

The type of `eprint` identifier, e. g., the name of the archive, repository, service, or system the `eprint` field refers to. See § 3.14.7 for details. Also see `eprint` and `eprintclass`.

eventdate field (date)

The date of a conference, a symposium, or some other event in `@proceedings` and `@inproceedings` entries. This field may also be useful for the custom types listed in § 2.1.3. See also `eventtitle` and `venue` as well as § 2.3.8.

eventtitle field (literal)

The title of a conference, a symposium, or some other event in `@proceedings` and `@inproceedings` entries. This field may also be useful for the custom types listed in § 2.1.3. Note that this field holds the plain title of the event. Things like "Proceedings of the Fifth XYZ Conference" go into the `titleaddon` or `booktitleaddon` field, respectively. See also `eventdate` and `venue`.

eventtitleaddon field (literal)

An annex to the `eventtitle` field. Can be used for known event acronyms, for example.

file	field (verbatim) A local link to a PDF or other version of the work. Not used by the standard bibliography styles.
foreword	list (name) The author(s) of a foreword to the work. If the author of the foreword is identical to the `editor` and/or `translator`, the standard styles will automatically concatenate these fields in the bibliography. See also `introduction` and `afterword`.
holder	list (name) The holder(s) of a `@patent`, if different from the `author`. Note that corporate holders need to be wrapped in an additional set of braces, see § 2.3.3 for details. This list may also be useful for the custom types listed in § 2.1.3.
howpublished	field (literal) A publication notice for unusual publications which do not fit into any of the common categories.
indextitle	field (literal) A title to use for indexing instead of the regular `title` field. This field may be useful if you have an entry with a title like "An Introduction to ..." and want that indexed as "Introduction to ..., An". Style authors should note that `biblatex` automatically copies the value of the `title` field to `indextitle` if the latter field is undefined.
institution	list (literal) The name of a university or some other institution, depending on the entry type. Traditional BibTeX uses the field name `school` for theses, which is supported as an alias. See also §§ 2.2.5 and 2.3.4.
introduction	list (name) The author(s) of an introduction to the work. If the author of the introduction is identical to the `editor` and/or `translator`, the standard styles will automatically concatenate these fields in the bibliography. See also `foreword` and `afterword`.
isan	field (literal) The International Standard Audiovisual Number of an audiovisual work. Not used by the standard bibliography styles.
isbn	field (literal) The International Standard Book Number of a book.
ismn	field (literal) The International Standard Music Number for printed music such as musical scores. Not used by the standard bibliography styles.
isrn	field (literal) The International Standard Technical Report Number of a technical report.

issn field (literal)

The International Standard Serial Number of a periodical.

issue field (literal)

The issue of a journal. This field is intended for journals whose individual issues are identified by a designation such as 'Spring' or 'Summer' rather than the month or a number. The placement of `issue` is similar to `month` and `number`. Integer ranges and short designators are better written to the `number` field. See also `month`, `number` and §§ 2.3.10 and 2.3.11.

issuesubtitle field (literal)

The subtitle of a specific issue of a journal or other periodical.

issuetitle field (literal)

The title of a specific issue of a journal or other periodical.

issuetitleaddon field (literal)

An annex to the `issuetitle`, to be printed in a different font.

iswc field (literal)

The International Standard Work Code of a musical work. Not used by the standard bibliography styles.

journalsubtitle field (literal)

The subtitle of a journal, a newspaper, or some other periodical.

journaltitle field (literal)

The name of a journal, a newspaper, or some other periodical.

journaltitleaddon field (literal)

An annex to the `journaltitle`, to be printed in a different font.

label field (literal)

A designation to be used by the citation style as a substitute for the regular label if any data required to generate the regular label is missing. For example, when an author-year citation style is generating a citation for an entry which is missing the author or the year, it may fall back to `label`. See § 2.3.2 for details. Note that, in contrast to `shorthand`, `label` is only used as a fallback. See also `shorthand`.

language list (key)

The language(s) of the work. Languages may be specified literally or as localisation keys (see § 4.9.2, especially § 4.9.2.18). If localisation keys are used, the prefix `lang` is omissible: both `langenglish` and `english` can be used. If the `clearlang` option is set, the content of this field may be cleared if it matches the `babel`/`polyglossia` language of the document (or the language specified explicitly with the `language` option), see § 3.1.2.1. See also `origlanguage` and compare `langid` in § 2.2.3.

library field (literal)

This field may be useful to record information such as a library name and a call number. This may be printed by a special bibliography style if desired. Not used by the standard bibliography styles.

location list (literal)

The place(s) of publication, i. e., the location of the `publisher` or `institution`, depending on the entry type. Traditional BibTeX uses the field name `address`, which is supported as an alias. See also §§ 2.2.5 and 2.3.4. With `@patent` entries, this list indicates the scope of a patent. This list may also be useful for the custom types listed in § 2.1.3.

mainsubtitle field (literal)

The subtitle related to the `maintitle`. See also `subtitle`.

maintitle field (literal)

The main title of a multi-volume book, such as *Collected Works*. If the `title` or `booktitle` field indicates the title of a single volume which is part of multi-volume book, the title of the complete work is given in this field.

maintitleaddon field (literal)

An annex to the `maintitle`, to be printed in a different font.

month field (literal)

The publication month. This must be an integer, not an ordinal or a string. Don't say `month={January}` but `month={1}`. The bibliography style converts this to a language dependent string or ordinal where required. This field is a literal field only when given explicitly in the data (for plain BibTeX compatibility for example). It is however better to use the `date` field as this supports many more features. See §§ 2.3.8 and 2.3.9.

nameaddon field (literal)

An addon to be printed immediately after the author name in the bibliography. Not used by the standard bibliography styles.

note field (literal)

Miscellaneous bibliographic data which does not fit into any other field. The `note` field may be used to record bibliographic data in a free format. Publication facts such as "Reprint of the edition London 1831" are typical candidates for the `note` field. See also `addendum`.

number field (literal)

The number of a journal or the volume/number of a book in a `series`. See also `issue` as well as §§ 2.3.7, 2.3.10, 2.3.11. With `@patent` entries, this is the number or record token of a patent or patent request. Normally this field will be an integer or an integer range, but it may also be a short designator that is not entirely numeric such as "S1", "Suppl. 2", "3es". In these cases the output should be scrutinised carefully.

Since `number` is—maybe counterintuitively given its name—a literal field, sorting templates will not treat its contents as integers, but as literal strings, which means that "11" may sort between "1" and "2". If integer sorting is desired, the field can be declared an integer field in a custom data model (see § 4.5.4). But then the sorting of non-integer values is not well defined.

The 'article number' or 'paper number', which can be used instead of—or along with—a page range to pinpoint a specific article within another work, goes into the `eid` field.

organization list (literal)

The organization(s) that published a `@manual` or an `@online` resource, or sponsored a conference. See also § 2.3.4.

origdate field (date)

If the work is a translation, a reprint, or something similar, the publication date of the original edition. Not used by the standard bibliography styles. See also `date`.

origlanguage list (key)

If the work is a translation, the language(s) of the original work. See also `language`.

origlocation list (literal)

If the work is a translation, a reprint, or something similar, the `location` of the original edition. Not used by the standard bibliography styles. See also `location` and § 2.3.4.

origpublisher list (literal)

If the work is a translation, a reprint, or something similar, the `publisher` of the original edition. Not used by the standard bibliography styles. See also `publisher` and § 2.3.4.

origtitle field (literal)

If the work is a translation, the `title` of the original work. Not used by the standard bibliography styles. See also `title`.

pages field (range)

One or more page numbers or page ranges. If the work is published as part of another one, such as an article in a journal or a collection, this field holds the relevant page range in that other work. It may also be used to limit the reference to a specific part of a work (a chapter in a book, for example). For papers in electronic journals with a non-classical pagination setup the `eid` field may be more suitable.

pagetotal field (literal)

The total number of pages of the work.

pagination field (key)

The pagination of the work. The value of this field will affect the formatting the ⟨*postnote*⟩ argument to a citation command. The key should be given in the singular form. Possible keys are `page`, `column`, `line`, `verse`, `section`, and `paragraph`. See also `bookpagination` as well as §§ 2.3.12 and 3.15.3.

part field (literal)

The number of a partial volume. This field applies to books only, not to journals. It may be used when a logical volume consists of two or more physical ones. In this case the number of the logical volume goes in the `volume` field and the number of the part of that volume in the `part` field. See also `volume`.

publisher list (literal)

The name(s) of the publisher(s). See also § 2.3.4.

pubstate field (key)

The publication state of the work, e. g., 'in press'. See § 4.9.2.11 for known publication states.

reprinttitle field (literal)

The title of a reprint of the work. Not used by the standard styles.

series field (literal)

The name of a publication series, such as "Studies in …", or the number of a journal series. Books in a publication series are usually numbered. The number or volume of a book in a series is given in the `number` field. Note that the `@article` entry type makes use of the `series` field as well, but handles it in a special way. See § 2.3.7 for details.

shortauthor list (name) Label field

The author(s) of the work, given in an abbreviated form. This field is mainly intended for abbreviated forms of corporate authors, see § 2.3.3 for details.

shorteditor list (name) Label field

The editor(s) of the work, given in an abbreviated form. This field is mainly intended for abbreviated forms of corporate editors, see § 2.3.3 for details.

shorthand field (literal) Label field

A special designation to be used by the citation style instead of the usual label. If defined, it overrides the default label. See also `label`.

shorthandintro field (literal)

The verbose citation styles which comes with this package use a phrase like "henceforth cited as [shorthand]" to introduce shorthands on the first citation. If the `shorthandintro` field is defined, it overrides the standard phrase. Note that the alternative phrase must include the shorthand.

shortjournal field (literal) Label field

A short version or an acronym of the `journaltitle`. Not used by the standard bibliography styles.

shortseries field (literal) Label field

A short version or an acronym of the `series` field. Not used by the standard bibliography styles.

shorttitle field (literal) Label field

The title in an abridged form. This field is usually not included in the bibliography. It is intended for citations in author-title format. If present, the author-title citation styles use this field instead of `title`.

subtitle field (literal)

The subtitle of the work.

title field (literal)

The title of the work.

titleaddon field (literal)

An annex to the `title`, to be printed in a different font.

translator list (name)

The translator(s) of the `title` or `booktitle`, depending on the entry type. If the translator is identical to the `editor`, the standard styles will automatically concatenate these fields in the bibliography.

type field (key)

The type of a `manual`, `patent`, `report`, or `thesis`. This field may also be useful for the custom types listed in § 2.1.3.

url field (uri)

The URL of an online publication. If it is not URL-escaped (no '%' chars) it will be URI-escaped according to RFC 3987, that is, even Unicode chars will be correctly escaped.

urldate field (date)

The access date of the address specified in the `url` field. See also § 2.3.8.

venue field (literal)

The location of a conference, a symposium, or some other event in `@proceedings` and `@inproceedings` entries. This field may also be useful for the custom types listed in § 2.1.3. Note that the `location` list holds the place of publication. It therefore corresponds to the `publisher` and `institution` lists. The location of the event is given in the `venue` field. See also `eventdate` and `eventtitle`.

version field (literal)

The revision number of a piece of software, a manual, etc.

volume field (integer)

The volume of a multi-volume book or a periodical. It is expected to be an integer, not necessarily in arabic numerals since `biber` will automatically convert from roman numerals or arabic letter to integers internally for sorting purposes. See also `part`. See the `noroman` option which can be used to suppress roman numeral parsing. This can help in cases where there is an ambiguity between parsing as roman numerals or alphanumeric (e.g. 'C'), see § 3.1.2.3.

volumes field (integer)

The total number of volumes of a multi-volume work. Depending on the entry type, this field refers to `title` or `maintitle`. It is expected to be an integer, not necessarily in arabic numerals since `biber` will automatically convert from roman numerals or arabic letter to integers internally for sorting purposes. See the `noroman` option which can be used to suppress roman numeral parsing. This can help in cases where there is an ambiguity between parsing as roman numerals or alphanumeric (e.g. 'C'), see § 3.1.2.3.

year field (literal)

The year of publication. This field is a literal field only when given explicitly in the data (for plain BibTeX compatibility for example). It is however better to use the `date` field as this is compatible with plain years too and supports many more features. See §§ 2.3.8 and 2.3.9.

2.2.3 Special Fields

The fields listed in this section do not hold printable data but serve a different purpose. They apply to all entry types in the default data model.

crossref field (entry key)

This field holds an entry key for the cross-referencing feature. Child entries with a `crossref` field inherit data from the parent entry specified in the `crossref` field. If the number of child entries referencing a specific parent entry hits a certain threshold, the parent entry is automatically added to the bibliography even if it has not been cited explicitly. The threshold is settable with the `mincrossrefs` package option from § 3.1.2.1. Style authors should note that whether or not the `crossref` fields of the child entries are defined on the `biblatex` level depends on the availability of the parent entry. If the parent entry is available, the `crossref` fields of the child entries will be defined. If not, the child entries still inherit the data from the parent entry but their `crossref` fields will be undefined. Whether the parent entry is added to the bibliography implicitly because of the threshold or explicitly because it has been cited does not matter. See also the `xref` field in this section as well as § 2.4.1.

entryset field (separated values)

This field is specific to entry sets. See § 3.14.5 for details. This field is consumed by the backend processing and does not appear in the `.bbl`.

execute field (code)

A special field which holds arbitrary TeX code to be executed whenever the data of the respective entry is accessed. This may be useful to handle special cases. Conceptually, this field is comparable to the hooks `\AtEveryBibitem`, `\AtEveryLositem`, and `\AtEveryCitekey` from § 4.10.6, except that it is definable on a per-entry basis in the `bib` file. Any code in this field is executed automatically immediately after these hooks.

gender field (Pattern matching one of: `sf`, `sm`, `sn`, `pf`, `pm`, `pn`, `pp`)

The gender of the author or the gender of the editor, if there is no author. The following identifiers are supported: `sf` (feminine singular, a single female name), sm (masculine singular, a single male name), sn (neuter singular, a single neuter name), pf (feminine plural, a list of female names), pm (masculine plural, a list of male names), pn (neuter plural, a list of neuter names), pp (plural, a mixed gender list of names). This information is only required by special bibliography and citation styles and only in certain languages. For example, a citation style may replace recurrent author names with a term such as 'idem'. If the Latin word is used, as is custom in English and French, there is no need to specify the gender. In German publications, however, such key terms are usually given in German and in this case they are gender-sensitive.

langid field (identifier)

The language id of the bibliography entry. The alias hyphenation is provided for backwards compatibility. The identifier must be a language name known to the babel/polyglossia packages. This information may be used to switch hyphenation patterns and localise strings in the bibliography. Note that the language names are case sensitive. The languages currently supported by this package are given in table 2. Note that babel treats the identifier english as an alias for british or american, depending on the babel version. The biblatex package always treats it as an alias for american. It is preferable to use the language identifiers american and british (babel) or a language specific option to specify a language variant (polyglossia, using the langidopts field) to avoid any possible confusion. Compare language in § 2.2.2.

langidopts field (literal)

For polyglossia users, allows per-entry language specific options. The literal value of this field is passed to polyglossia's language switching facility when using the package option autolang=langname. For example, the fields:

```
langid         = {english},
langidopts     = {variant=british},
```

would wrap the bibliography entry in:

```
\english[variant=british]
...
\endenglish
```

ids field (separated list of entrykeys)

Citation key aliases for the main citation key. An entry may be cited by any of its aliases and biblatex will treat the citation as if it had used the primary citation key. This is to aid users who change their citation keys but have legacy documents which use older keys for the same entry. This field is consumed by the backend processing and does not appear in the .bbl.

indexsorttitle field (literal)

The title used when sorting the index. In contrast to indextitle, this field is used for sorting only. The printed title in the index is the indextitle or the title field. This field may be useful if the title contains special characters or commands which interfere with the sorting of the index. Consider this example:

Table 2: Supported Languages

Language	Region/Dialect	Identifiers
Basque	France, Spain	`basque`
Bulgarian	Bulgaria	`bulgarian`
Catalan	Spain, France, Andorra, Italy	`catalan`
Croatian	Croatia, Bosnia and Herzegovina, Serbia	`croatian`
Czech	Czech Republic	`czech`
Danish	Denmark	`danish`
Dutch	Netherlands	`dutch`
English	USA	`american`, `USenglish`, `english`
	United Kingdom	`british`, `UKenglish`
	Canada	`canadian`
	Australia	`australian`
	New Zealand	`newzealand`
Estonian	Estonia	`estonian`
Finnish	Finland	`finnish`
French	France, Canada	`french`
German	Germany	`german`
	Austria	`austrian`
	Switzerland	`swissgerman`
German (new)	Germany	`ngerman`
	Austria	`naustrian`
	Switzerland	`nswissgerman`
Greek	Greece	`greek`
Hungarian	Hungary	`magyar`, `hungarian`
Icelandic	Iceland	`icelandic`
Italian	Italy	`italian`
Latvian	Latvia	`latvian`
Lithuanian	Lithuania	`lithuanian`
Marathi	India	`marathi`
Norwegian (Bokmål)	Norway	`norsk`
Norwegian (Nynorsk)	Norway	`nynorsk`
Polish	Poland	`polish`
Portuguese	Brazil	`brazil`
	Portugal	`portuguese`, `portuges`
Romanian	Romania	`romanian`
Russian	Russia	`russian`
Serbian (Latin)	Serbia	`serbian`
Serbian (Cyrillic)	Serbia	`serbianc`
Slovak	Slovakia	`slovak`
Slovene	Slovenia	`slovene`, `slovenian`
Spanish	Spain	`spanish`
Swedish	Sweden	`swedish`
Turkish	Turkey	`turkish`
Ukrainian	Ukraine	`ukrainian`

```
title          = {The \LaTeX\ Companion},
indextitle     = {\LaTeX\ Companion, The},
indexsorttitle = {LATEX Companion},
```

Style authors should note that `biblatex` automatically copies the value of either the `indextitle` or the `title` field to `indexsorttitle` if the latter field is undefined.

keywords field (separated values)

A separated list of keywords. These keywords are intended for the bibliography filters (see §§ 3.8.2 and 3.14.4), they are usually not printed. Note that with the default separator (comma), spaces around the separator are ignored.

options field (separated ⟨*key*⟩=⟨*value*⟩ options)

A separated list of entry options in ⟨*key*⟩=⟨*value*⟩ notation. This field is used to set options on a per-entry basis. See § 3.1.3 for details. Note that citation and bibliography styles may define additional entry options.

presort field (string)

A special field used to modify the sorting order of the bibliography. This field is the first item the sorting routine considers when sorting the bibliography, hence it may be used to arrange the entries in groups. This may be useful when creating subdivided bibliographies with the bibliography filters. Please refer to § 3.6 for further details. Also see § 4.5.6. This field is consumed by the backend processing and does not appear in the `.bbl`.

related field (separated values)

Citation keys of other entries which have a relationship to this entry. The relationship is specified by the `relatedtype` field. Please refer to § 3.5 for further details.

relatedoptions field (separated values)

Per-type options to set for a related entry. Note that this does not set the options on the related entry itself, only the `dataonly` clone which is used as a datasource for the parent entry.

relatedtype field (identifier)

An identifier which specified the type of relationship for the keys listed in the `related` field. The identifier is a localised bibliography string printed before the data from the related entry list. It is also used to identify type-specific formatting directives and bibliography macros for the related entries. Please refer to § 3.5 for further details.

relatedstring field (literal)

A field used to override the bibliography string specified by `relatedtype`. Please refer to § 3.5 for further details.

sortkey field (literal)

A field used to modify the sorting order of the bibliography. Think of this field as the master sort key. If present, `biblatex` uses this field during sorting and ignores everything else, except for the `presort` field. Please refer to § 3.6 for further details. This field is consumed by the backend processing and does not appear in the `.bbl`.

sortname list (name)

A name or a list of names used to modify the sorting order of the bibliography. If present, this list is used instead of `author` or `editor` when sorting the bibliography. Please refer to § 3.6 for further details. This field is consumed by the backend processing and does not appear in the `.bbl`.

sortshorthand field (literal)

Similar to `sortkey` but used in the list of shorthands. If present, `biblatex` uses this field instead of `shorthand` when sorting the list of shorthands. This is useful if the `shorthand` field holds shorthands with formatting commands such as `\emph` or `\textbf`. This field is consumed by the backend processing and does not appear in the `.bbl`.

sorttitle field (literal)

A field used to modify the sorting order of the bibliography. If present, this field is used instead of the `title` field when sorting the bibliography. The `sorttitle` field may come in handy if you have an entry with a title like "An Introduction to..." and want that alphabetized under 'I' rather than 'A'. In this case, you could put "Introduction to..." in the `sorttitle` field. Please refer to § 3.6 for further details. This field is consumed by the backend processing and does not appear in the `.bbl`.

sortyear field (integer)

A field used to modify the sorting order of the bibliography. In the default sorting templates, if this field is present, it is used instead of the `year` field when sorting the bibliography. Please refer to § 3.6 for further details. This field is consumed by the backend processing and does not appear in the `.bbl`.

xdata field (separated list of entrykeys)

This field inherits data from one or more `@xdata` entries. Conceptually, the `xdata` field is related to `crossref` and `xref`: `crossref` establishes a logical parent/child relation and inherits data; `xref` establishes as logical parent/child relation without inheriting data; `xdata` inherits data without establishing a relation. The value of the `xdata` may be a single entry key or a separated list of keys. See § 3.14.6 for further details. This field is consumed by the backend processing and does not appear in the `.bbl`.

xref field (entry key)

This field is an alternative cross-referencing mechanism. It differs from `crossref` in that the child entry will not inherit any data from the parent entry specified in the `xref` field. If the number of child entries referencing a specific parent entry hits a certain threshold, the parent entry is automatically added to the bibliography even if it has not been cited explicitly. The threshold is settable with the `minxrefs` package option from § 3.1.2.1. Style authors should note that whether or not the `xref` fields of the child entries are defined on the `biblatex` level depends on the availability of the parent entry. If the parent entry is available, the `xref` fields of the child entries will be defined. If not, their `xref` fields will be undefined. Whether the parent entry is added to the bibliography implicitly because of the threshold or explicitly because it has been cited does not matter. See also the `crossref` field in this section as well as § 2.4.1.

2.2.4 Custom Fields

The fields listed in this section are intended for special bibliography styles. They are not used by the standard bibliography styles.

name[a–c] list (name)

Custom lists for special bibliography styles. Not used by the standard bibliography styles.

name[a–c]type field (key)

Similar to `authortype` and `editortype` but referring to the fields `name[a--c]`. Not used by the standard bibliography styles.

list[a–f] list (literal)

Custom lists for special bibliography styles. Not used by the standard bibliography styles.

user[a–f] field (literal)

Custom fields for special bibliography styles. Not used by the standard bibliography styles.

verb[a–c] field (verbatim)

Similar to the custom fields above except that these are verbatim fields. Not used by the standard bibliography styles.

2.2.5 Field Aliases

The aliases listed in this section are provided for backwards compatibility with traditional BibTeX and other applications based on traditional BibTeX styles. Note that these aliases are immediately resolved as the `bib` file is processed. All bibliography and citation styles must use the names of the fields they point to, not the alias. In `bib` files, you may use either the alias or the field name but not both at the same time.

address list (literal)

An alias for `location`, provided for BibTeX compatibility. Traditional BibTeX uses the slightly misleading field name `address` for the place of publication, i. e., the location of the publisher, while `biblatex` uses the generic field name `location`. See §§ 2.2.2 and 2.3.4.

annote field (literal)

An alias for `annotation`, provided for `jurabib` compatibility. See § 2.2.2.

archiveprefix field (literal)

An alias for `eprinttype`, provided for arXiv compatibility. See §§ 2.2.2 and 3.14.7.

journal field (literal)

An alias for `journaltitle`, provided for BibTeX compatibility. See § 2.2.2.

key field (literal)

An alias for `sortkey`, provided for BibTeX compatibility. See § 2.2.3.

pdf field (verbatim)

An alias for `file`, provided for JabRef compatibility. See § 2.2.2.

primaryclass field (literal)

An alias for `eprintclass`, provided for arXiv compatibility. See §§ 2.2.2 and 3.14.7.

school list (literal)

An alias for `institution`, provided for BibTeX compatibility. The `institution` field is used by traditional BibTeX for technical reports whereas the `school` field holds the institution associated with theses. The `biblatex` package employs the generic field name `institution` in both cases. See §§ 2.2.2 and 2.3.4.

2.3 Usage Notes

The entry types and fields supported by this package should for the most part be intuitive to use for anyone familiar with BibTeX. However, apart from the additional types and fields provided by this package, some of the familiar ones are handled in a way which is in need of explanation. This package includes some compatibility code for `bib` files which were generated with a traditional BibTeX style in mind. Unfortunately, it is not possible to handle all legacy files automatically because `biblatex`'s data model is slightly different from traditional BibTeX. Therefore, such `bib` files will most likely require editing in order to work properly with this package. In sum, the following items are different from traditional BibTeX styles:

- The entry type `@inbook`. See §§ 2.1.1 and 2.3.1 for details.
- The fields `institution`, `organization`, and `publisher` as well as the aliases `address` and `school`. See §§ 2.2.2, 2.2.5, 2.3.4 for details.
- The handling of certain types of titles. See § 2.3.5 for details.
- The field `series`. See §§ 2.2.2 and 2.3.7 for details.
- The fields `year` and `month`. See §§ 2.2.2, 2.3.8, 2.3.9, 2.3.10 for details.
- The field `edition`. See § 2.2.2 for details.
- The field `key`. See § 2.3.2 for details.

Users of the `jurabib` package should note that the `shortauthor` field is treated as a name list by `biblatex`, see § 2.3.3 for details.

2.3.1 The Entry Type `@inbook`

Use the `@inbook` entry type for a self-contained part of a book with its own title only. It relates to `@book` just like `@incollection` relates to `@collection`. See § 2.3.5 for examples. If you want to refer to a chapter or section of a book, simply use the `book` type and add a `chapter` and/or `pages` field. Whether a bibliography should at all include references to chapters or sections is controversial because a chapter is not a bibliographic entity.

2.3.2 Missing and Omissible Data

The fields marked as 'required' in § 2.1.1 are not strictly required in all cases. The bibliography styles which come with this package can get by with as little as a title field for most entry types. A book published anonymously, a periodical without an explicit editor, or a software manual without an explicit author should pose no problem as far as the bibliography is concerned. Citation styles, however, may have different requirements. For example, an author-year citation scheme obviously requires an author/editor and a year field.

You may generally use the label field to provide a substitute for any missing data required for citations. How the label field is employed depends on the citation style. The author-year citation styles which come with this package use the label field as a fallback if either the author/editor or the year is missing. The numeric styles, on the other hand, do not use it at all since the numeric scheme is independent of the available data. The author-title styles ignore it as well, because the bare title is usually sufficient to form a unique citation and a title is expected to be available in any case. The label field may also be used to override the non-numeric portion of the automatically generated labelalpha field used by alphabetic citation styles. See § 4.2.4 for details.

Note that traditional BibTeX styles support a key field which is used for alphabetizing if both author and editor are missing. The biblatex package treats key as an alias for sortkey. In addition to that, it offers very fine-grained sorting controls, see §§ 2.2.3 and 3.6 for details. The natbib package employs the key field as a fallback label for citations. Use the label field instead.

2.3.3 Corporate Authors and Editors

Corporate authors and editors are given in the author or editor field, respectively. Note that they must be wrapped in an extra pair of curly braces to prevent data parsing from treating them as personal names which are to be dissected into their components. Use the shortauthor field if you want to give an abbreviated form of the name or an acronym for use in citations.

```
author       = {{National Aeronautics and Space Administration}},
shortauthor  = {NASA},
```

The default citation styles will use the short name in all citations while the full name is printed in the bibliography. For corporate editors, use the corresponding fields editor and shorteditor. Since all of these fields are treated as name lists, it is possible to mix personal names and corporate names, provided that the names of all corporations and institutions are wrapped in braces.

```
editor       = {{National Aeronautics and Space Administration}
                and Doe, John},
shorteditor  = {NASA and Doe, John},
```

Users switching from the jurabib package to biblatex should note that the shortauthor field is treated as a name list.

2.3.4 Literal Lists

The fields institution, organization, publisher, and location are literal lists in terms of § 2.2. This also applies to origlocation, origpublisher and to the field aliases address and school. All of these fields may contain a list of items separated by the keyword 'and'. If they contain a literal 'and', it must be wrapped in braces.

```
publisher    = {William Reid {and} Company},
institution  = {Office of Information Management {and} Communications
    ↪ },
organization = {American Society for Photogrammetry {and} Remote
    ↪ Sensing
                and
                American Congress on Surveying {and} Mapping},
```

Note the difference between a literal '{and}' and the list separator 'and' in the above examples. You may also wrap the entire name in braces:

```
publisher    = {{William Reid and Company}},
institution  = {{Office of Information Management and Communications}
    ↪ },
organization = {{American Society for Photogrammetry and Remote
    ↪ Sensing}
                and
                {American Congress on Surveying and Mapping}},
```

Legacy files which have not been updated for use with biblatex will still work if these fields do not contain a literal 'and'. However, note that you will miss out on the additional features of literal lists in this case, such as configurable formatting and automatic truncation.

2.3.5 Titles

The following examples demonstrate how to handle different types of titles. Let's start with a five-volume work which is referred to as a whole:

```
@MvBook{works,
  author     = {Shakespeare, William},
  title      = {Collected Works},
  volumes    = {5},
  ...
```

The individual volumes of a multi-volume work usually have a title of their own. Suppose the fourth volume of the *Collected Works* includes Shakespeare's sonnets and we are referring to this volume only:

```
@Book{works:4,
  author     = {Shakespeare, William},
  maintitle  = {Collected Works},
  title      = {Sonnets},
  volume     = {4},
  ...
```

If the individual volumes do not have a title, we put the main title in the title field and include a volume number:

```
@Book{works:4,
  author     = {Shakespeare, William},
  title      = {Collected Works},
  volume     = {4},
  ...
```

In the next example, we are referring to a part of a volume, but this part is a self-contained work with its own title. The respective volume also has a title and there is still the main title of the entire edition:

```
@InBook{lear,
  author     = {Shakespeare, William},
  bookauthor = {Shakespeare, William},
  maintitle  = {Collected Works},
  booktitle  = {Tragedies},
  title      = {King Lear},
  volume     = {1},
  pages      = {53-159},
  ...
```

Suppose the first volume of the *Collected Works* includes a reprinted essay by a well-known scholar. This is not the usual introduction by the editor but a self-contained work. The *Collected Works* also have a separate editor:

```
@InBook{stage,
  author     = {Expert, Edward},
  title      = {Shakespeare and the Elizabethan Stage},
  bookauthor = {Shakespeare, William},
  editor     = {Bookmaker, Bernard},
  maintitle  = {Collected Works},
  booktitle  = {Tragedies},
  volume     = {1},
  pages      = {7-49},
  ...
```

See § 2.3.7 for further examples.

2.3.6 Editorial Roles

The type of editorial role performed by an editor in one of the `editor` fields (i. e., `editor`, `editora`, `editorb`, `editorc`) may be specified in the corresponding `editor...type` field. The following roles are supported by default. The role '`editor`' is the default. In this case, the `editortype` field is omissible.

editor The main editor. This is the most generic editorial role and the default value.

compiler Similar to `editor` but used if the task of the editor is mainly compiling.

founder The founding editor of a periodical or a comprehensive publication project such as a 'Collected Works' edition or a long-running legal commentary.

continuator An editor who continued the work of the founding editor (`founder`) but was subsequently replaced by the current editor (`editor`).

redactor	A secondary editor whose task is redacting the work.
reviser	A secondary editor whose task is revising the work.
collaborator	A secondary editor or a consultant to the editor.
organizer	Similar to `editor` but used if the task of the editor is mainly organizing.

For example, if the task of the editor is compiling, you may indicate that in the corresponding `editortype` field:

```
@Collection{...,
  editor      = {Editor, Edward},
  editortype  = {compiler},
  ...
```

There may also be secondary editors in addition to the main editor:

```
@Book{...,
  author      = {...},
  editor      = {Editor, Edward},
  editora     = {Redactor, Randolph},
  editoratype = {redactor},
  editorb     = {Consultant, Conrad},
  editorbtype = {collaborator},
  ...
```

Periodicals or long-running publication projects may see several generations of editors. For example, there may be a founding editor in addition to the current editor:

```
@Book{...,
  author      = {...},
  editor      = {Editor, Edward},
  editora     = {Founder, Frederic},
  editoratype = {founder},
  ...
```

Note that only the `editor` is considered in citations and when sorting the bibliography. If an entry is typically cited by the founding editor (and sorted accordingly in the bibliography), the founder goes into the `editor` field and the current editor moves to one of the `editor...` fields:

```
@Collection{...,
  editor      = {Founder, Frederic},
  editortype  = {founder},
  editora     = {Editor, Edward},
  ...
```

You may add more roles by initializing and defining a new localisation key whose name corresponds to the identifier in the `editor...type` field. See §§ 3.10 and 4.9.1 for details.

2.3.7 Publication and Journal Series

The `series` field is used by traditional BibTeX styles both for the main title of a multi-volume work and for a publication series, i. e., a loosely related sequence of books by the same publisher which deal with the same general topic or belong to the same field of research. This may be ambiguous. This package introduces a `maintitle` field for multi-volume works and employs `series` for publication series only. The volume or number of a book in the series goes in the `number` field in this case:

```
@Book{...,
  author        = {Expert, Edward},
  title         = {Shakespeare and the Elizabethan Age},
  series        = {Studies in English Literature and Drama},
  number        = {57},
  ...
```

The `@article` entry type makes use of the `series` field as well, but handles it in a special way. First, a test is performed to determine whether the value of the field is an integer. If so, it will be printed as an ordinal. If not, another test is performed to determine whether it is a localisation key. If so, the localised string is printed. If not, the value is printed as is. Consider the following example of a journal published in numbered series:

```
@Article{...,
  journal         = {Journal Name},
  series          = {3},
  volume          = {15},
  number          = {7},
  year            = {1995},
  ...
```

This entry will be printed as "*Journal Name*. 3rd ser. 15.7 (1995)". Some journals use designations such as "old series" and "new series" instead of a number. Such designations may be given in the `series` field as well, either as a literal string or as a localisation key. Consider the following example which makes use of the localisation key `newseries`:

```
@Article{...,
  journal         = {Journal Name},
  series          = {newseries},
  volume          = {9},
  year            = {1998},
  ...
```

This entry will be printed as "*Journal Name*. New ser. 9 (1998)". See § 4.9.2 for a list of localisation keys defined by default.

2.3.8 Date and Time Specifications

Date fields such as the default data model dates `date`, `origdate`, `eventdate`, and `urldate` adhere to ISO8601-2 Extended Format specification level 1. In addition to the ISO8601-2 empty date range markers, you may also specify an open ended/start

Table 3: Date Specifications

Date Specification	Formatted Date (Examples)	
	Short/12-hour Format	Long/24-hour Format
`1850`	1850	1850
`1997/`	1997–	1997–
`/1997`	–1997	–1997
`1997/..`	1997–	1997–
`../1997`	–1997	–1997
`1967-02`	02/1967	February 1967
`2009-01-31`	31/01/2009	31st January 2009
`1988/1992`	1988–1992	1988–1992
`2002-01/2002-02`	01/2002–02/2002	January 2002–February 2002
`1995-03-30/1995-04-05`	30/03/1995–05/04/1995	30th March 1995–5th April 1995
`2004-04-05T14:34:00`	05/04/2004 2:34 PM	5th April 2004 14:34:00

date range by giving the range separator and omitting the end/start date (e. g., `YYYY/`, `/YYYY`). See table 3 for some examples of valid date specifications and the formatted dates automatically generated by `biblatex`. The formatted date is language specific and will be adapted automatically. If there is no `date` field in an entry, `biblatex` will also consider the fields `year` and `month` for backwards compatibility with traditional BibTeX but this is not encouraged as explicit `year` and `month` are not parsed for date meta-information markers or times and are used as-is. Style authors should note that date fields like `date` or `origdate` are only available in the `bib` file. All dates are parsed and dissected into their components as the `bib` file is processed. The date and time components are made available to styles by way of the special fields discussed in § 4.2.4.3. See this section and table 10 on page 173 for further information.

ISO8601-2 Extended Format dates are astronomical dates in which year '0' exists. When outputting dates in BCE or BC era (see the `dateera` option below), note that they will typically be one year earlier since BCE/BC era do not have a year 0 (year 0 is 1 BCE/BC). This conversion is automatic. See examples in table 5.

Date field names *must* end with the string 'date', as with the default date fields. Bear this in mind when adding new date fields to the datamodel (see § 4.5.4). `biblatex` will check all date fields after reading the data model and will exit with an error if it finds a date field which does not adhere to this naming convention.

ISO8601-2 supports dates before common era (BCE/BC) by way of a negative date format and supports 'approximate' (circa) and uncertain dates. Such date formats set internal markers which can be tested for so that appropriate localised markers (such as `circa` or `beforecommonera`) can be inserted. Also supported are 'unspecified' dates (ISO8601-2 4.3) which are automatically expanded into appropriate data ranges accompanied by a field <`datetype`>`dateunspecified` which details the granularity of the unspecified data. Styles may use this information to format such dates appropriately but the standard styles do not do this. See table 4 on page 40 for the allowed ISO8601-2 'unspecified' formats, their range expansions and <`datetype`>`dateunspecified` values (see § 4.2.4.1).

Table 5 shows formats which use appropriate tests and formatting. See the date meta-information tests in § 4.6.2 and the localisation strings in § 4.9.2.21. See also the `96-dates.tex` example file for complete examples of the tests and localisation strings use.

The output of 'circa', uncertainty and era information in standard styles (or custom styles not customising the internal `\mkdaterange*` macros) is controlled by the

Table 4: ISO8601-2 4.3 Unspecified Date Parsing

Date Specification	Expanded Range	Meta-information
`199X`	1990/1999	yearindecade
`19XX`	1900/1999	yearincentury
`1999-XX`	1999-01/1999-12	monthinyear
`1999-01-XX`	1999-01-01/1999-01-31	dayinmonth
`1999-XX-XX`	1999-01-01/1999-12-31	dayinyear

package options `datecirca`, `dateuncertain`, `dateera` and `dateeraauto` (see § 3.1.2.1). See table 5 on page 41 for examples which assumes these options are all used.

2.3.9 Year, Month and Date

The fields `year` and `month` are still supported by `biblatex`, but the full set of date features (day and time precision, ranges, ...) can only be used with `date`. It is therefore recommended to prefer `date` over `year` and `month` unless backwards compatibility of the `bib` file with classical BibTeX is required.

2.3.10 Months and Journal Issues

The `month` field is an integer field. The bibliography style converts the month to a language-dependent string as required. For backwards compatibility, you may also use the following three-letter abbreviations in the `month` field: `jan`, `feb`, `mar`, `apr`, `may`, `jun`, `jul`, `aug`, `sep`, `oct`, `nov`, `dec`. Note that these abbreviations are BibTeX strings which must be given without any braces or quotes. When using them, don't say `month={jan}` or `month="jan"` but `month=jan`. It is not possible to specify a month such as `month={8/9}`. Use the `date` field for date ranges instead. Quarterly journals are typically identified by a designation such as 'Spring' or 'Summer' which should be given in the `issue` field. The placement of the `issue` field in `@article` entries is similar to and overrides the `month` field.

2.3.11 Journal Numbers and Issues

The terms 'number', 'issue' and even 'issue number' are often used synonymously by journals to refer to the subdvision of a `volume`. The fact that `biblatex`'s data model has fields of both names can sometimes lead to confusion about which field should be used. First and foremost the word that the journal uses for the subdivsion of a `volume` should be of minor importance, what matters is the role in the data model. As a rule of thumb `number` is the right field in most circumstances. In the standard styles `number` modifies `volume`, whereas `issue` modifies the date (year) of the entry. Numeric identifiers and short designators that are not necessarily (entirely) numeric such as 'A', 'S1', 'C2', 'Suppl. 3', '4es' would go into the `number` field, because they usually modify the `volume`. The output of—especially longer—non-numeric input for `number` should be checked since it could potentially look odd with some styles. The field `issue` can be used for designations such as 'Spring', 'Winter' or 'Michaelmas term' if that is commonly used to refer to the journal.

The 'article number' or 'paper number', which can be used instead of—or along with—a page range to pinpoint a specific article within another work, goes into the `eid` field, whose placement in the standard styles is similar to the analogous `pages` field.

Table 5: Enhanced Date Specifications

Date Specification	Formatted Date (Examples)	
	Output Format	Output Format Notes
`0000`	1 BC	`dateera=christian` prints `beforechrist` localisation
`-0876`	877 BCE	`dateera=secular` prints `beforecommonera` localisation string
`-0877/-0866`	878 BC–867 BC	using `\ifdateera` test and `beforechrist` localisation string
`0768`	0768 CE	using `dateeraauto` set to '1000' and `commonera` localisation string
`-0343-02`	344-02 BCE	
`0343-02-03`	343-02-03 CE	with `dateeraauto=400`
`0343-02-03`	343-02-02 CE	with `dateeraauto=400` and `julian`
`1723~`	circa 1723	using `\ifdatecirca` test
`1723?`	1723?	using `\ifdateuncertain` test
`1723%`	circa 1723?	using `\ifdateuncertain` and `\ifdatecirca` tests
`2004-22`	2004	also, `yeardivision` is set to the localisation string 'summer'
`2004-24`	2004	also, `yeardivision` is set to the localisation string 'winter'

2.3.12 Pagination

When specifying a page or page range, either in the `pages` field of an entry or in the ⟨*postnote*⟩ argument to a citation command, it is convenient to have `biblatex` add prefixes like 'p.' or 'pp.' automatically and this is indeed what this package does by default. However, some works may use a different pagination scheme or may not be cited by page but rather by verse or line number. This is when the `pagination` and `bookpagination` fields come into play. As an example, consider the following entry:

```
@InBook{key,
  title          = {...},
  pagination     = {verse},
  booktitle      = {...},
  bookpagination = {page},
  pages          = {53--65},
  ...
```

The `bookpagination` field affects the formatting of the `pages` and `pagetotal` fields in the list of references. Since `page` is the default, this field is omissible in the above example. In this case, the page range will be formatted as 'pp. 53–65'. Suppose that, when quoting from this work, it is customary to use verse numbers rather than page numbers in citations. This is reflected by the `pagination` field, which affects the formatting of the ⟨*postnote*⟩ argument to any citation command. With a citation like `\cite[17]{key}`, the postnote will be formatted as 'v. 17'. Setting the `pagination` field to `section` would yield '§ 17'. See § 3.15.3 for further usage instructions.

The `pagination` and `bookpagination` fields are key fields. This package will try to use their value as a localisation key, provided that the key is defined. Always use the singular form of the key name in `bib` files, the plural is formed automatically. The keys `page`, `column`, `line`, `verse`, `section`, and `paragraph` are predefined, with `page` being the default. The string '`none`' has a special meaning when used in a `pagination` or `bookpagination` field. It suppresses the prefix for the respective entry. If there are no predefined localisation keys for the pagination scheme required by a certain entry, you can simply add them. See the commands `\NewBibliographyString` and \DefineBibliographyStrings in § 3.10. You need to define two localisation strings for each additional pagination scheme: the singular form (whose localisation key corresponds to the value of the pagination field) and the plural form (whose localisation key must be the singular plus the letter 's'). See the predefined keys in § 4.9.2 for examples.

2.4 Hints and Caveats

This section provides some additional hints concerning the data interface of this package. It also addresses some common problems.

2.4.1 Cross-referencing

biber features a highly customizable cross-referencing mechanism with flexible data inheritance rules. Duplicating certain fields in the parent entry or adding empty fields to the child entry is no longer required. Entries are specified in a natural way:

```
@Book{book,
  author        = {Author},
  title         = {Booktitle},
  subtitle      = {Booksubtitle},
  publisher     = {Publisher},
  location      = {Location},
  date          = {1995},
}
@InBook{inbook,
  crossref      = {book},
  title         = {Title},
  pages         = {5--25},
}
```

The title field of the parent will be copied to the booktitle field of the child, the subtitle becomes the booksubtitle. The author of the parent becomes the bookauthor of the child and, since the child does not provide an author field, it is also duplicated as the author of the child. After data inheritance, the child entry is similar to this:

```
author          = {Author},
bookauthor      = {Author},
title           = {Title},
booktitle       = {Booktitle},
booksubtitle    = {Booksubtitle},
publisher       = {Publisher},
location        = {Location},
date            = {1995},
pages           = {5--25},
```

See appendix B for a list of mapping rules set up by default. Note that all of this is customizable. See § 4.5.12 on how to configure biber's cross-referencing mechanism. See also § 2.2.3.

2.4.1.1 The xref field

In addition to the crossref field, biblatex supports a simplified cross-referencing mechanism based on the xref field. This is useful if you want to establish a parent/child relation between two associated entries but prefer to keep them independent as far as the data is concerned. The xref field differs from crossref in that the child entry will not inherit any data from the parent. If the parent is referenced by a certain number of child entries, biblatex will automatically add it to the bibliography. The threshold is controlled by the minxrefs package option from § 3.1.2.1.u See also § 2.2.3.

2.4.2 Sorting and Encoding Issues

biber handles US-ASCII, 8-bit encodings such as Latin 1, and UTF-8. It features true Unicode support and is capable of reencoding the bib data on the fly in a robust way. For sorting, biber uses a Perl implementation of the Unicode Collation Algorithm (UCA), as outlined in Unicode Technical Standard #10.[13] Collation tailoring based on the Unicode Common Locale Data Repository (CLDR) is also supported.[14]

Supporting Unicode implies much more than handling UTF-8 input. Unicode is a complex standard covering more than its most well-known parts, the Unicode character encoding and transport encodings such as UTF-8. It also standardizes aspects such as string collation, which is required for language-sensitive sorting. For example, by using the Unicode Collation Algorithm, biber can handle the character 'ß' without any manual intervention. All you need to do to get localised sorting is specify the locale:

```
\usepackage[sortlocale=de]{biblatex}
```

or if you are using German as the main document language via babel or polyglossia:

```
\usepackage[sortlocale=auto]{biblatex}
```

This will make biblatex pass the babel/polyglossia main document language as the locale which biber will map into a suitable default locale. biber will not try to get locale information from its environment as this makes document processing dependent on something not in the document which is against TeX's spirit of reproducibility. This also makes sense since babel/polyglossia are in fact the relevant environment for a document. Note that this will also work with 8-bit encodings such as Latin 9, i. e., you can take advantage of Unicode-based sorting even though you are not using UTF-8 input. See § 2.4.2.1 on how to specify input and data encodings properly.

2.4.2.1 Specifying Encodings

When using a non-US-ASCII encoding in the bib file, it is important to understand what biblatex can do for you and what may require manual intervention. The package takes care of the LaTeX side, i. e., it ensures that the data imported from the bbl file is interpreted correctly, provided that the bibencoding package option (or the datasource specific override for this, see § 3.8.1) is set properly. All of this is handled

[13] https://unicode.org/reports/tr10/
[14] http://cldr.unicode.org/

automatically and no further steps, apart from setting the `bibencoding` option in certain cases (namely when the encoding of the `bib` file differs from the encoding of the `tex` file), are required provided that you set up your document encoding (i. e., load `inputenc` or related packages if required) *before* `biblatex` is loaded. Here are a few typical usage scenarios along with the relevant lines from the document preamble:

- US-ASCII notation in both the `tex` and the `bib` file with pdfTeX or traditional TeX:

  ```
  \usepackage{biblatex}
  ```

- Latin 1 encoding (ISO-8859-1) in the `tex` file, US-ASCII notation in the `bib` file with pdfTeX or traditional TeX:

  ```
  \usepackage[latin1]{inputenc}
  \usepackage[bibencoding=ascii]{biblatex}
  ```

- Latin 9 encoding (ISO-8859-15) in both the `tex` and the `bib` file with pdfTeX or traditional:

  ```
  \usepackage[latin9]{inputenc}
  \usepackage[bibencoding=auto]{biblatex}
  ```

  Since `bibencoding=auto` is the default setting, the option is omissible. The following setup will have the same effect:

  ```
  \usepackage[latin9]{inputenc}
  \usepackage{biblatex}
  ```

- UTF-8 encoding in the `tex` file, Latin 1 (ISO-8859-1) in the `bib` file with pdfTeX or traditional TeX:

  ```
  \usepackage[utf8]{inputenc}
  \usepackage[bibencoding=latin1]{biblatex}
  ```

  The same scenario with LaTeX release 2018-04-01 or above, XeTeX or LuaTeX in native UTF-8 mode:

  ```
  \usepackage[bibencoding=latin1]{biblatex}
  ```

`biber` can handle US-ASCII notation, 8-bit encodings such as Latin 1, and UTF-8. It is also capable of reencoding the `bib` data on the fly (replacing the limited macro-level reencoding feature of `biblatex`). This will happen automatically if required, provided that you specify the encoding of the `bib` files properly. In addition to the scenarios discussed above, `biber` can also handle the following cases:

- Transparent UTF-8 workflow, i. e., UTF-8 encoding in both the `tex` and the `bib` file with pdfTeX or traditional TeX:

```
\usepackage[utf8]{inputenc}
\usepackage[bibencoding=auto]{biblatex}
```

Since `bibencoding=auto` is the default setting, the option is omissible:

```
\usepackage[utf8]{inputenc}
\usepackage{biblatex}
```

The same scenario with XeTeX or LuaTeX in native UTF-8 mode:

```
\usepackage{biblatex}
```

- It is even possible to combine an 8-bit encoded `tex` file with UTF-8 encoding in the `bib` file, provided that all characters in the `bib` file are also covered by the selected 8-bit encoding:

```
\usepackage[latin1]{inputenc}
\usepackage[bibencoding=utf8]{biblatex}
```

Some workarounds may be required when using traditional TeX or pdfTeX with UTF-8 encoding because `inputenc`'s `utf8` module does not cover all of Unicode. Roughly speaking, it only covers the Western European Unicode range. When loading `inputenc` with the `utf8` option, `biblatex` will normally instruct `biber` to reencode the `bib` data to UTF-8. This may lead to `inputenc` errors if some of the characters in the `bib` file are outside the limited Unicode range supported by `inputenc`.

- If you are affected by this problem, try setting the `safeinputenc` option:

```
\usepackage[utf8]{inputenc}
\usepackage[safeinputenc]{biblatex}
```

If this option is enabled, `biblatex` will ignore `inputenc`'s `utf8` option and use US-ASCII. `biber` will then try to convert the `bib` data to US-ASCII notation. For example, it will convert Ş to `\k{S}`. This option is similar to setting `texencoding=ascii` but will only take effect in this specific scenario (`inputenc`/`inputenx` with UTF-8). This workaround takes advantage of the fact that both Unicode and the UTF-8 transport encoding are backwards compatible with US-ASCII.

This solution may be acceptable as a workaround if the data in the `bib` file is mostly US-ASCII anyway, with only a few strings, such as some authors' names, causing problems. However, keep in mind that it will not magically make traditional TeX or pdfTeX support Unicode. It may help if the occasional odd character is not supported by `inputenc`, but may still be processed by TeX when using an accent command (e. g., `\d{S}` instead of Ṣ). If you need full Unicode support, however, switch to XeTeX or LuaTeX.

Typical errors when `inputenc` cannot handle a certain UTF-8 character are:

```
! Package inputenc Error: Unicode char <char> (U+<codepoint>)
(inputenc)                not set up for use with LaTeX.
```

but also less obvious things like:

```
! Argument of \UTFviii@three@octets has an extra }.
```

3 User Guide

This part of the manual documents the user interface of the `biblatex` package. The user guide covers everything you need to know in order to use `biblatex` with the default styles that come with this package. You should read the user guide first in any case. If you want to write your own citation and/or bibliography styles, continue with the author guide afterwards.

3.1 Package Options

All package options are given in ⟨*key*⟩`=`⟨*value*⟩ notation. The value `true` is omissible with all boolean keys. For example, giving `sortcites` without a value is equivalent to `sortcites=true`.

3.1.1 Load-time Options

The following options must be used as `biblatex` is loaded, i. e., in the optional argument to `\usepackage`.

`backend=bibtex, bibtex8, biber` default: `biber`

Specifies the database backend. The following backends are supported:

`biber`	`biber`, the default backend of `biblatex`, supports US-ASCII, 8-bit encodings, UTF-8, on-the-fly reencoding, locale-specific sorting, and many other features. Locale-specific sorting, case-sensitive sorting, and upper/lowercase precedence are controlled by the options `sortlocale`, `sortcase`, and `sortupper`, respectively.
`bibtex`	Legacy BibTeX. Traditional BibTeX supports US-ASCII encoding only. Sorting is always case-insensitive.
`bibtex8`	`bibtex8`, the 8-bit implementation of BibTeX, supports US-ASCII and 8-bit encodings such as Latin 1.

See § 3.16 for details of using BibTeX as a backend.

`style=`⟨*file*⟩ default: `numeric`

Loads the bibliography style ⟨*file*⟩`.bbx` and the citation style ⟨*file*⟩`.cbx`. See § 3.3 for an overview of the standard styles.

`bibstyle=`⟨*file*⟩ default: `numeric`

Loads the bibliography style ⟨*file*⟩`.bbx`. See § 3.3.2 for an overview of the standard bibliography styles.

`citestyle=`⟨*file*⟩ default: `numeric`

Loads the citation style ⟨*file*⟩`.cbx`. See § 3.3.1 for an overview of the standard citation styles.