# The Biblatex Package

**Programmable Bibliographies and Citations**

**Philipp Lehman** **Version 3.7**
**(with Philip Kime, Audrey Boruvka** **16/11/2016**
**and Joseph Wright)**

# Biblatex 宏包说明文档摘译

Zhenzhen Hu[1] Wenbo Sheng [2]

2016-02-08

译者按: 之所以摘译 biblatex，一是出于对 keep texing 的兴趣，二是考虑到中文资料里总体介绍 latex 的资料其实较多，反而是在一些专项部分中文资料较少，看宏包的英文说明当然没有问题，但有的文档内容长达几百页，一般用户真心没有精力去看，即便是找一些自己需要的功能也比较麻烦，所以考虑对参考文献的 biblatex 宏包文档进行摘译，是希望能在这一方面有所贡献。关于这一点 Wenbo 兄也很有同感，在几年前 biblatex 还是 2.x 版本的时候他就深入研究了 biblatex 并翻译了 1-4 节很多内容。我在 biblatex-gb7714-2015 样式宏包中提议翻译 biblatex 文档之后，我们决定合作来推进这个事情，尽管都只有部分业余时间可以利用，但我们认为只要有空就积累一点，那么终有完成的时候。

如同我在 biblatex-gb7714-2015 样式宏包说明文档中介绍的那样，biblatex 宏包具有很多强大功能比如参考文献表划分、文献集、样式定制、动态数据处理等等，在科技论文或书籍写作中特别有用 (尤其是在对参考文献著录和标注格式有特殊要求的情况下)。可以说，biblatex 是 latex 文档写作中参考文献问题的完整解决方案，也一定程度上代表了这一方面的未来趋势。总之，本项目的总体任务是完成 biblatex 宏包说明文档关键内容的摘译，希望能对使用 biblatex 和对参考文献样式有深度定制要求的用户有所帮助，当然也希望能使中文 latex 资料库更为全面和深入。需要说明的是，限于作者水平，其中难免存在一些错误和理解不到位的地方，欢迎批评指正，欢迎 @ 译者邮箱。最后感谢 CTEX 和 Latexstudio 论坛，感谢论坛上各位作者关于 biblatex 参考文献方面的工作分享和经验介绍。

# 目录



[1]Email:hzzmail@163.com
[2]Email:wbsheng88@foxmail.com

# 表格



# 中英文术语对照表

| | | |
|---|---|---|
| 参考文献标注样式 | citation style | 158 |
| 参考文献著录样式 | Bibliography Style | 140 |
| 域 | field | 10 |

| | | |
|---|---|---|
| 条目 | entry | 136 |
| 条目类型 | entry type | 10 |
| 样式 | style | 10 |

# 1 引言

这是关于 Biblatex 包的语法文档，使用范例文档参考文档[3]。快速开始，请浏览 §§ 1.1、2.1、2.2、2.3、3.1、3.3、3.7、3.8、3.12 节。

## 1.1 关于 Biblatex

Biblatex 包提供了一套与 LaTeX 配合使用的高级参考文献工具。它重新实现了 LaTeX 提供的参考文献功能。该包使用后端程序 Biber 来处理 BibTeX 格式的数据文件，并完成排序、标签生成和更多功能。参考文献的格式化完全由 TeX 宏指令控制。具备良好的 LaTeX 知识就足以设计新的参考文献著录样式和标注样式。

Biblatex 也支持参考文献表细分、在一个文档内包含多个参考文献表、以及域缩写等参考文献信息表。参考文献表可以根据主题进行分块或者分段。与参考文献著录样式类似，所有的标注引用命令也可以自由定义。

提供的功能还包括：文献数据的 Unicode 支持、自定义排序、不同排序方式的多参考文献表、自定义标签和动态数据修改等。Biber/Biblatex 的版本兼容性见 § 1.5.5 节。该包可完全实现本地化，可与 `babel` 和 `polyglossia` 宏包配合使用。该包支持的语言详见表 表 3。

## 1.2 许可



## 1.3 反馈

请使用 Github 的 Biblatex 项目页报告 bug 和提交所需功能[5]。在提出功能需求，请确保你已经彻底研究过本手册。如果你不想报告 bug 或者请求新功能，而只是需要帮助，可以考虑在 `comp.text.tex` 新闻组或者 TeX-LaTeX Stack Exchange 提交问题。[6]

## 1.4 致谢

The language modules of this package are made possible thanks to the following contributors: Augusto Ritter Stoffel, Mateus Araújo (Brazilian); Sebastià Vila-Marta (Catalan); Ivo Pletikosić (Croatian); Michal Hoftich (Czech); Jonas Nyrup (Danish);

[3] http://ctan.org/pkg/biblatex/doc/examples

[4] http://www.ctan.org/tex-archive/macros/latex/base/lppl.txt

[5] http://github.com/plk/biblatex

[6] http://tex.stackexchange.com/questions/tagged/biblatex

Johannes Wilm (Danish/Norwegian); Alexander van Loon, Pieter Belmans, Hendrik Maryns (Dutch); Hannu Väisänen, Janne Kujanpää (Finnish); Denis Bitouzé (French); Apostolos Syropoulos, Prokopis (Greek); Baldur Kristinsson (Icelandic); Enrico Gregorio, Andrea Marchitelli (Italian); Håkon Malmedal (Norwegian); Anastasia Kandulina, Yuriy Chernyshov (Polish); José Carlos Santos (Portuguese); Oleg Domanov (Russian); Tea Tušar and Bogdan Filipič (Slovene); Ignacio Fernández Galván (Spanish); Per Starbäck, Carl-Gustav Werner, Filip Åsblom (Swedish).

## 1.5 前提与必备

本节介绍所需资源和兼容性问题。

### 1.5.1 必须资源

如下资源是必须的，否则 Biblatex 无法正常工作。

ε-TEX Biblatex 宏包依赖于 ε-TEX 。很长时间以来，TEX 发行版就带有 ε-TEX ，并且近来主流的发行版都默认使用。Biblatex 宏包会检查是否在 ε-TEX 下运行。只需要像平常一样编译你的文档即可，基本上是可以运行的。如果你得到错误信息，尝试用 `elatex` 或 `pdfelatex` 分别代替 `latex` 或 `pdflatex` 来编译文档。

Biber Biber 是 Biblatex 默认的后端程序。你只需要 BIBTEX 或者 Biber 中的一个后端程序。TEXLive 中带有 Biber ，也可以从 SourceForge 得到。[7] Biber 使用 C 程序库 `btparse` 解析 BIBTEX 格式文件，这既为了兼容 BIBTEX 的解析规则，也用于修正一些常见问题。详见 Perl 的 `Text::BibTeX` 模块（module）的手册页。[8]

etoolbox 自动加载，提供 Biblatex 所需的通用编程工具，可以从 CTAN 下载。[9]

kvoptions 自动加载，用于内部选项处理。可以和 `oberdiek` 宏包集一起从 CTAN 下载。[10]

logreq 自动加载，它提供的前端可用于将机器可读信息写入辅助 log 文件，可以从 CTAN 下载。[11]

xstring 自动加载，提供了一些高级字符串处理宏。可以从 CTAN 下载。[12]

除了上述资源，Biblatex 还需要 `keyval`、`ifthen` 以及 `url` 等标准 LATEX 宏包。常见的 TEX 发行版中都会带有这些宏包，而且本宏包会自动加载。

### 1.5.2 推荐包

这一节所列出的宏包对于运行 Biblatex 不是必须的。不过，它们可以提供一些值得推荐的额外功能，或者加强已有的特征。宏包载入的顺序并不重要。

[7] http://biblatex-biber.sourceforge.net/
[8] http://search.cpan.org/~ambs/Text-BibTeX
[9] http://ctan.org/pkg/etoolbox
[10] http://ctan.org/pkg/kvoptions
[11] http://ctan.org/pkg/logreq/
[12] http://ctan.org/pkg/xstring/

babel/polyglossia `babel` 和 `polyglossia` 宏包提供了多语种排版的核心架构。如果你使用美式英语以外的语言写作，那么强烈推荐使用这两个宏包中的一个。你应当在 Biblatex 之前载入 `babel` 或 `polyglossia`，这样 Biblatex 宏包可以自动检测。

csquotes 如果使用该宏包，Biblatex 会使用它的引用语工具给相应标题加上语言相关的引号。如果没有，那么 Biblatex 会使用作为后备的美式英语的引号。当使用其它语言写作时，强烈推荐使用 `csquotes` 宏包。[13]

xpatch `xpatch` 宏包为 Biblatex 宏、驱动和格式指令扩展了 `etoolbox` 的一些补丁命令。[14]

### 1.5.3 兼容的包

Biblatex 宏包专门为本节所列出的文档类和宏包提供了兼容性代码。

hyperref `hyperref` 宏包将引用转化为超链接。详见 § 3.1.2.1 一节中的 `hyperref` 和 `backref` 宏包选项。当使用 `hyperref` 宏包时，最好在 Biblatex 之后载入。

showkeys `showkeys` 宏包会打印出文档中标注和文献表中条目的内部键值。宏包载入的顺序不重要。

memoir 使用 `memoir` 文档类会调整默认的参考文献标题，从而与该文档类默认的页面布局相协调。更多使用提示请参考 § 3.13.2 一节。

KOMA-Script 使用 `scrartcl`、`scrbook` 或 `scrreprt` 文档类中的任何一个都会调整默认的参考文献标题，从而与这些文档类默认的页面布局相协调。更多使用提示请参考 § 3.13.1 一节。

### 1.5.4 不兼容的包

本节列出了与 Biblatex 不兼容的宏包。Biblatex 从根本上重新实现了 LaTeX 的文献功能，因此很自然地与修改这些功能的所有宏包相冲突。这并不是 Biblatex 独有的——在列出的宏包中，出于同样的原因，有些宏包相互之间也是不兼容的。

babelbib `babelbib` 宏包为多语种文献提供了支持，这正是 Biblatex 的一个典型特点。使用 `langid` 域和宏包选项 `autolang` 即可实现类似的功能。请注意，当载入 `bable` 或 `polyglossia` 宏包时 Biblatex 会自动调整主文档的语言。如果想要在文献中每个条目里切换语言，你只需要以上提到的特性。具体细节请参考 §§ 2.2.3 和 3.1.2.1 以及 § 3.9 几节。

backref `backref` 宏包可以在参考文献中创建反向引用。类似的功能请参考 § 3.1.2.1 一节中的宏包选项 `hyperref` 和 `backref`。

bibtopic `bibtopic` 宏包支持根据主题、类型或者其它标准细分文献。对于按照主题细分文献，可以参考 § 3.7.6 一节的类型特征以及 § 3.7.2 一节中相应的。另外，你也可以使用 `keywords` 域结合 `keyword` 和 `notkeyword` 过滤器来实现相应功能，细节请参考 §§ 2.2.3 和 3.7.2。对于按照类型细分文献，可以使用 `type` 和 `nottype` 过滤器。相关例子请参考 § 3.12.4。

[13] http://ctan.org/pkg/csquotes/

[14] http://ctan.org/pkg/xpatch/

bibunits bibunits 宏包支持多个部分（例如每一章内）的参考文献。请参考 chapterbib。

chapterbib chapterbib 宏包支持多个部分的参考文献。使用 refsection 环境和 section 过滤器可以实现相应效果。此外，你也可能需要 refsegment 环境和 segment 过滤器。细节请参考 §§ 3.7.4、3.7.5、3.7.2。相关实例请参考 § 3.12.3。

cite cite 可以自动对引用编号进行排序，并且将连续的数字缩写为一个区间。它也可以配置引用中的标点符号。关于引用编号的排序和缩写，请参考 § 3.1.2.1 一节中的 sortcites 宏包选项和 § 3.3.1 一节中的 numeric-comp 引用样式。关于可配置的标点请参考 § 3.10。

citeref 另一个可以创建反向引用的宏包。参考 backref 条目。

inlinebib inlinebib 宏包用于脚注文献这种传统引用样式。相应的功能请参考 § 3.3.1 中详细的引用样式说明。

jurabib jurabib 宏包原本用于法学和司法文件（主要是德文）中的引用，它也为人文学科中的使用者提供了一些特性。在提供这些特征方面，jurabib 和 Biblatex 有一些类似之处，但是采用的手段是截然不同的。由于 jurabib 和 Biblatex 都是那种功能齐备的宏包，鉴于篇幅这里不再赘述它们的异同之处。

mcite mcite 提供了分组引用的支持，也就是说，不同条目可以指向同一处引用，并且在参考文献中作为同一条目列在一起。引用组依照被引用的条目定义，不过这只在未排序的参考文献中有效。Biblatex 宏包同样支持分组引用，在本手册中称之为“条目集”或“参考文献集”。细节请参考 §§ 3.12.5、3.7.11、3.8.10。

mciteplus mcite 宏包的一个加强版的重新实现，可以支持排序文献的分组。参考 mcite 宏包条目。

multibib multibib 宏包支持依照主题或其它标准细分文献。参考 bibtopic 宏包条目。

natbib natbib 宏包支持编号和作者—年份引用格式，以及 cite 宏包中的合并排序和压缩代码。它同样提供了一些额外的引用命令和几种设置选项。相应的功能请参考 § 3.3.1 中的 numeric 和 author-year 引用样式及其变种，§ 3.1.2.1中的 sortcites 宏包选项，§ 3.8 中的引用命令，以及 §§ 3.7.7、3.7.8、3.10 中讨论的工具。也可以参考 § 3.8.9。

splitbib splitbib 宏包支持按照主题细分文献。参考 bibtopic 宏包条目。

titlesec titlesec 宏包重新定义了一些用户水平的文档划分命令，例如 \chapter 或 \section。这种方法与 Biblatex 的 refsection 和 refsegment 选项设置引起的内部命令改动不兼容，具体描述在 § 3.1.2.1 一节。

ucs ucs 宏包提供 UTF-8 编码输入的支持。可以使用 inputenc 宏包的标准 utf8 模块或者 XeTeX 、LuaTeX 等支持 Unicode 的编译引擎来实现这一功能。

| Biber 版本 | Biblatex 版本 |
|---|---|
| 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 |

**Table 2: Biber/Biblatex 兼容性**

### 1.5.5 Biber/Biblatex 兼容性

Biber 的版本与 Biblatex 的版本有着紧密的联系。你需要二者正确的组合。如果发现来自于不兼容的 Biblatex 版本信息，Biber 会在处理过程中发出警告。表 2 展示了最近一些版本的兼容性状况。

# 2 数据库指南

本节描述 `blx-dm.def` 中定义的默认数据模型。该文件是宏包的一部分。该数据模型的定义由 § 4.5.4节中的宏实现。因此，可以重新定义 Biblatex 和 Biber 所用的数据模型，使得数据源可以包括新的条目类型和域（当然这需要样式文件支持）。数据模型规范还允许定义约束，使得数据源可以根据数据模型进行校验（使用 Biber 的 `--validate_datamodel` 选项）。若需要定制数据模型，请参考 `blx-dm.def` 文件和 § 4.5.4 节。

## 2.1 条目类型

本节介绍 Biblatex 默认数据模型支持的条目类型及每种条目类型支持的域。

### 2.1.1 常规类型

下面的列表说明了每种条目类型支持的域。注意，每种条目类型的域的使用是由参考文献样式决定的。因此，下面的列表有两个目的，一是说明有本包提供的标准样式支持的域，二是作为定制样式的模板。注意，所谓“必选”域并不是在所有情况下都严格必不可少的，详见 § 2.3.2 节。而标记“可选”的域技术上是可选的。通常来说，文献格式规则往往不仅需要“必选”域。默认的数据模型为一些数据域、ISBN 和 `gender` 等特殊域定义了一些约束。但这些约束仅用于校验这些域是否合乎数据模型（通过 Biber 的 `--validate_datamodel` 选项）。通用域如 `abstract`、`annotation`、`label` 和 `shorthand` 并不在下面的列表中，因为它们独立于条目类型；§ 2.2.3节讨论的特殊域同样也独立于条目类型，因此也不在下面的列表中。默认的支持类型见文件 `blx-dm.def`，内有 Biblatex 数据模型的完整规范。

article 指期刊、杂志、报纸或其他周期性刊物的文章。它是独立个体，有自己的标题。刊物名在 `journaltitle` 域中给出。如果在出版物标题外，期号也有自己的标题，那么在 `issuetitle` 域中给出。注意，`editor` 及相关域指的是期刊，而 `translator` 及其相关域则涉及到文章。

必选域： `author`, `title`, `journaltitle`, `year/date`

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

book 单卷本的书籍，有一名或多名作者，并且这些作者作为整体共享该著作。该条目类型也涵盖了传统 BibTeX 的 `@inbook` 类型，详见 § 2.3.1 节。

必选域： `author`, `title`, `year/date`

可选域： `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`, `chapter`, `pages`, `pagetotal`, `addendum`, `pubstate`, `doi`, `eprint`, `eprintclass`, `eprinttype`, `url`, `urldate`

mvbook 多卷本书籍。为了向后兼容，多卷书也可用 `@book` 条目类型。然而建议最好使用专用条目类型 `@mvbook`。

必选域： `author`, `title`, `year/date`

可选域： `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 书的一部分。它是一个独立的单元，有自己的标题。注意，该类型的定义不同于标准 BibTeX 给出的定义，见 § 2.3.1 节。

必选域： author, title, booktitle, year/date

可选域： 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, chapter, pages, addendum, pubstate, doi, eprint, eprintclass, eprinttype, url, urldate

bookinbook 类似于 @inbook，但用于原本已经单独出版的作品。典型的例子是在一位作者的作品集中再版的书籍。

suppbook @book （书）的补充材料，与 @inbook 条目类型很相近。@inbook 用于一本书中自己带有标题的部分，例如一本散文集中同一作者的单独一篇散文；而本条目用于诸如序言、导论、前言、后记等部分，通常只有一般性的标题。一些样式指南需要定制该类型的格式区别于 @inbook。不过标准样式则认为它是 @inbook 的别名。

booklet 类似于书籍，但没有正式的出版者或赞助机构。如果可以的话，使用howpublished域可以提供自由格式的出版信息。也可以用 type 域。

必选域： author/editor, title, year/date

可选域： subtitle, titleaddon, language, howpublished, type, note, location, chapter, pages, pagetotal, addendum, pubstate, doi, eprint, eprintclass, eprinttype, url, urldate

collection 单卷本的作品集，包括了一些有不同标题和作者的独立作品。作品集没有总体意义上的作者，但通常有一位编辑。

必选域： editor, title, year/date

可选域： 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, chapter, pages, pagetotal, addendum, pubstate, doi, eprint, eprintclass, eprinttype, url, urldate

mvcollection 多卷本作品集。为了向后兼容，也可用 @collection 条目类型。然而建议最好使用专用条目类型 @mvcollection。

必选域： editor, title, year/date

可选域： 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 作品集中的一篇作品，是一个独立的单元，有自己的标题。author 指的是 title 的作者，而 editor 指的是 booktitle（即文集的标题）的编者。

必选域： author, title, booktitle, year/date

可选域： 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, chapter, pages, addendum, pubstate, doi, eprint, eprintclass, eprinttype, url, urldate

suppcollection `@collection` 中的补充材料。类似于 `@suppbook` 之于 `@book`。标准样式将其视为 `@incollection` 的别名。

manual 技术或其它文档，不必是出版的形式。按照 § 2.3.2 一节，`author` 或者 `editor` 是可以省略的。

必选域： author/editor, title, year/date

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

misc 备选类型，用于无法归入任何其它类别的条目。适当的话，使用 `howpublished` 域，可以提供自由格式的出版信息。也可以使用 `type` 域。按照 § 2.3.2 节，`author`、`editor` 和 `year` 可以省略。

必选域： author/editor, title, year/date

可选域： subtitle, titleaddon, language, howpublished, type, version, note, organization, location, date, month, year, addendum, pubstate, doi, eprint, eprintclass, eprinttype, url, urldate

online 在线资源。按照 § 2.3.2 节，`author`，`editor` 和 `year` 可以省略。该类型用于网址等本质上的在线资源。注意：所有条目类型都支持 `url` 域。比如，当增加一篇来自在线期刊的文章时，应优先使用 `@article` 条目和它的 `url` 域。

必选域： author/editor, title, year/date, url

可选域： subtitle, titleaddon, language, version, note, organization, date, month, year, addendum, pubstate, urldate

patent 专利或专利申请。号码或记录号在 `number` 域中给出。`type` 域用于描述类型，`location` 域则用于描述专利范围，如果存在与 `type` 领域不同的情况。注意，`location` 在本条目类型中以键值列表的方式处理，详见 § 2.2.1 节。

必选域： author, title, number, year/date

可选域： holder, subtitle, titleaddon, type, version, location, note, date, month, year, addendum, pubstate, doi, eprint, eprintclass, eprinttype, url, urldate

periodical 周期性刊物中完整的一期，比如某一期刊的某一期特刊。标题在 `title` 域中给出。如果该期在期刊的主标题外有其自己的标题，那么由 `issuetitle` 域中给出。根据 § 2.3.2 节，`editor` 域可以省略。

必选域： editor, title, year/date

可选域： editora, editorb, editorc, subtitle, issuetitle, issuesubtitle, language, series, volume, number, issue, date, month, year, note, issn, addendum, pubstate, doi, eprint, eprintclass, eprinttype, url, urldate

suppperiodical @periodical 的补充材料，类似于 @suppbook 之于 @book。如果你意识到 @article 类型其实就是 @inperiodical，那么本条目类型的作用显而易见了。该类型应用于只有一般性题目的栏目，例如固定专栏、讣告、致编辑的信等。一些样式指南会严格定制该类型的格式区别于 @article。不过标准样式则认为它是 @article 的别名。

proceedings 单卷本的会议记录。这一类型与 @collection 非常相似。它支持可选的 organization 域用于给出主办机构。根据 § 2.3.2 节，editor 域可以省略。

必选域： title, year/date

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

mvproceedings 多卷 @proceedings 条目，类似于 @mvbook 之于 @book。

必选域： title, year/date

可选域： 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 会议集中的一篇文章，与 @incollection 类似。支持 organization 可选域。

必选域： author, title, booktitle, year/date

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

reference 单卷本的参考文献集，诸如百科全书或词典等。它是通用 @collection 条目的特殊变种。标准样式将其视为 @collection 的别名。

mvreference 多卷本的 @reference 条目。标准样式将其视为 @mvcollection 的别名。出于向后兼容性，也可以使用 @reference 条目。不过，还是建议使用专门的 @mvreference 条目类型。

inreference 参考文献集中的一篇文章，它是通用 @incollection 条目的特殊变种。标准样式将其视为 @incollection 的别名。

report 由大学或其它机构发行的技术报告、研究报告以及白皮书等。使用 type 域来确定报告的类型。主办机构由 institution 域给出。

必选域： author, title, type, institution, year/date

可选域： subtitle, titleaddon, language, number, version, note, location, month, isrn, chapter, pages, pagetotal, addendum, pubstate, doi, eprint, eprintclass, eprinttype, url, urldate

set 条目集，是一种特殊类型条目，详见 § 3.12.5 节。

thesis 为满足教育机构的学位要求而写的学位论文。使用 type 域确定学位论文类型。

必选域： author, title, type, institution, year/date

可选域： subtitle, titleaddon, language, note, location, month, isbn, chapter, pages, pagetotal, addendum, pubstate, doi, eprint, eprintclass, eprinttype, url, urldate

unpublished 有作者和标题但是没有正式出版的作品，例如手稿或演讲稿等。需要的话，可使用 howpublished 域和 note 域提供自由格式的附加信息。

必选域： author, title, year/date

可选域： subtitle, titleaddon, language, howpublished, note, location, isbn, date, month, year, addendum, pubstate, url, urldate

xdata 特殊类型，@xdata 条目处理的数据可以被其它条目用 xdata 域继承。这一条目类型只是作为数据容器，不可被引用或加入到参考文献中，详见 § 3.12.6 节。

custom[a–f] 用于特殊参考文献样式的自定义类型，标准样式中不使用。

### 2.1.2 类型别名

本节中列出的条目类型用于向后兼容传统的 BibTeX 样式。这些别名由后端在数据处理时一并处理，样式中仅能见到这些别名所指代的类型，而不是别名本身。所有未知的条目类型一般输出为 @misc 条目。

conference BibTeX 遗留的 @inproceedings 的别名。

electronic @online的别名。

mastersthesis 类似于 @thesis 不过 type 域是可选的，默认为‘Master’s thesis’ 。可使用 type 域重定义。

phdthesis 类似于 @thesis 不过 type 域是可选的，默认为‘PhD thesis’ 。可使用 type 域重定义。

techreport 类似于@report 不过 type 域是可选的，默认为 ‘technical report’。可使用 type 域重定义。

www @online 的别名，用于兼容 jurabib 宏包。

### 2.1.3 不支持的条目类型

本节中的条目类型类似于自定义类型 `@custom[a–f]`。即，标准样式不支持这些类型，若使用标准样式，将会以 `@misc` 条目类处理。

artwork 视觉艺术作品，例如绘画、雕塑和装饰艺术。

audio 录音品，典型的有音频 CD、DVD、录音磁带或类似媒介。参考 `@music` 类型。

bibnote 这一特殊条目类型并不像其它类型那样用于 `bib` 文件中。它主要提供给 `notes2bib` 等第三方宏包，用于将注记并入文献中。注记应该在 `note` 域中。请谨记，`@bibnote` 类型与 `\defbibnote` 命令毫无关系。`\defbibnote` 命令用来在参考文献的开始或末尾添加评论，而 `@bibnote` 类型是为那些提供尾注参考条目的宏包准备的。

commentary 与常规书籍地位不同的评注，如司法评论等。

image 图像、图画、摄影和类似媒介。

jurisdiction 法庭判决、法庭记录和类似物。

legislation 法律、法案、立法提案和类似物。

legal 协议等的法律文书。

letter 私人通信，例如信件、电子邮件、备忘录等。

movie 动画。参考 `@video` 类型。

music 音乐刻录，`@audio` 的一种具体变种形式。

performance 音乐或戏剧表演和其它一些表演艺术作品。这一条目类型指的是表演的事件，而不是录制，评论或付印的剧本。

review 一些其它工作的回顾总结。这是 `@article` 类型的一个具体变种。标准样式将其视为 `@article` 的一个别称。

software 电脑软件。

standard 由一个标准组织（例如国际标准组织）发布的国家或国际标准。

video 视听记录，典型的包括 DVD、VHS 录像带或其它类似媒介。参考 `@movie` 类型。

## 2.2 条目域

本节概述 Biblatex 默认数据模型支持的域。数据模型使用的数据类型的介绍请参考 § 2.2.1 小节，实际的域列表见 §§ 2.2.2 和 2.2.3 小节。

### 2.2.1 数据类型

在 bib 文件等数据源中，所有的文献数据都在域中指定。其中一些域，例如 author 和 editor，可以包括一个项目列表。在 BibTeX 文件格式中，这种列表结构通过关键词“and”来分隔列表中的每一项。Biblatex 宏包实现了三种不同的数据类型来处理文献数据：姓名列表（name list）、文本列表（literal list）和域（field）。此外，列表和域中还有一些子类型，以及一个内容类型（content type）用于从语义上区分那些无法根据数据类型进行区分的域（见 § 4.5.4 节）。这一节大致概括了本宏包所支持的数据类型。关于 BibTeX 文件格式域到 Biblatex 的数据类型的对应，请参考 §§ 2.2.2 和 2.2.3 等节。

**姓名列表（name list）** 根据分隔词 and 将其解析并划分成独立的项目。然后列表中的每一项进一步分成四个姓名成分：[15] 名（given name，默认值）、姓名前缀（name prefix, 如 von、van、of、da、de、della 等）、姓（family name），以及姓名后缀（name suffix, 如 junior、senior 等）。可以通过调整数据模型的定义来定制有效的姓名成分，见 § 4.2.3 节。在 bib 文件中，姓名列表可以用关键词“and others”来截断。典型的姓名列表是 author 和 editor。

在默认的数据模型中，姓名列表域会为每一个姓名列表自动创建相应的 \ifuse* 测试（见 § 4.6.2 节）。同时也自动创建了一个 ifuse* 选项用以控制姓名的标记和排序行为（见 § 3.1.3.1 节）。Biber 支持定制姓名成分组合，不过目前的定义与传统 BibTeX 支持的姓名成分相同：

- 姓（family name，即‘last’ 部分）
- 名（given name，即‘first’ 部分）
- 前缀（name prefix，即‘von’ 部分）
- 后缀（name suffix，即‘Jr’ 部分）

默认数据模型使用 \DeclareDatamodelConstant 命令（见 4.5.4 节）将支持的姓名成分定义成一个恒定列表。然而，由于姓名成分通常需要硬编码到文献驱动程序和后端处理程序中，因此，如果想支持额外的姓名成分，将其简单地添加到姓名成分列表中是远远不够的。关于如何定义和使用定制姓名成分的细节，可以参考示例文件 93-nameparts.tex。关于如何使用定制姓名成分来消除姓名歧义的信息，参见 § 4.11.4 节中的 \DeclareUniquenameTemplate 命令。

**文本列表（literal list）** 由分隔词 and 划分成独立的项目，但各项不再进一步细分。在 bib 文件中，文本列表可以用关键词“and others”来截断。其中又有两个子类型：

**（狭义的）文本列表（literal lists in the strict sense）** 按照如上所述进行处理。各独立的项目就简单如实打印。典型的文本列表是 publisher 和 location。

[15] 这是针对西方人名的划分。对于中文来说，姓名无需划分。当然中文名的拼音可以进行对应的划分。——译注

**关键字列表（key list）** 是文本列表的变种，可以包括可打印的数据和本地化的关键字。对于列表中每一项，首先测试它是否是已知的本地化关键字（本地化关键字的默认定义在 § 4.9.2 节中）。如果是，那么打印本地化的字符串；否则这些项就按本身打印出来。典型的关键字列表是 `language`。

**域（field）** 通常以整体打印。有如下多种子类型：

**文本域（literal field）** 会如实打印。典型的文本域是 `title` 和 `note`。

**范围域（range field）** 包含了一个或更多范围，其中所有的短划线都规范化用 `\bibrangedash` 命令取代。一个范围指的是一个非短划线部分后紧跟一个或多个短划线再紧跟一个非短划线部分（比如 `5–7`）。任意数目的连续短划线都只会产生一个表示范围的横线。典型的范围域是 `pages` 域。也可以参考 `\bibrangessep` 命令，它用于定制多重范围间的分隔符。如果不包括范围，那么范围域将被忽略并生成警告信息。可以使用 `\DeclareSourcemap` 命令在解析范围域之前对其进行整理（见 § 4.5.3 节）。

**整数域（integer field）** 包含的整数当打印时会转化为序数或者字符串。典型的例子是 `extrayear` 和 `volume` 域。这些域会按照数字进行排序。出于排序的目的，Biber 会尝试将非阿拉伯数字的表示（例如罗马数字）转成相应的整数。

**日期部分域（datepart field）** 处理未格式化的整数，当打印时会转化为序数或者字符串。典型的例子是 `month` 域。在数据模型中，对于每一个数据类型为 `date` 的域 X，会自动创建带有如下名称的日期部分域：

⟨*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`

其中，对于任何 `datatype=date` 的数据模型域，⟨*datetype*⟩ 是在 'date' 之前的字符串。例如，在默认数据模型中，日期域 `date` 对应的是'event', 'orig', 'url' 和空字符串 ''。

**日期域（date field）** 处理形如 `yyyy-mm-ddThh:nn[+-][hh[:nn]Z]` 格式的日期，或者格式为 `yyyy-mm-ddThh:nn[+-][hh[:nn]Z]/yyyy-mm-ddThh:nn[+-][hh[:nn]Z]` 的日期范围，或者其它 EDTF level 1 允许的格式，见 § 2.3.8 节。日期域的特殊之处在于，日期会被解析并分解成各个日期部分类型的成分。当数据模型中定义一个数据类型为 `date` 的域时，会自动定义并识别相应的 `datepart` 组件（见上文）。典型的例子是 `date` 域。

**抄录域（verbatim field）** 在抄录模式下处理，可以包含特殊字符。典型的抄录域是 `file` 和 `doi`。

**URI 域** 在抄录模式下处理，可以包含特殊字符。如果看起来不像其实质，也可以进行 URL 转义。（They are also URL-escaped if they don't look like they already are. ）典型的例子是 `url` 域。

**分隔值域（separated value field）** 被分隔的文本值列表。例子是 `keywords` 和 `options` 域。通过 `xsvsep` 选项可以将分隔符配置成任何 Perl 正则表达式，其默认值是通常 BibTeX 中的（西文）逗号或者逗号加空格。

**模式域（pattern field）** 是必须匹配某一特定模式的文本域。例子有 § 2.2.3 一节中的 `gender` 域。

**关键字域（key field）** 可以处理可打印的数据或本地化的关键字。首先测试是否是已知的本地化关键字键值（本地化关键字的默认定义在 § 4.9.2 一节中）。如果是，就打印本地化的字符串；否则，就按本身来打印。典型的例子是 `type` 域。

**代码域（code field）** 处理 TeX 代码。

### 2.2.2 数据域

本节所列的域是在默认数据模型中处理可打印数据的常规域。左边的名称是域的默认数据模型名，会在 Biblatex 和后端使用。右边则是相应的 Biblatex 数据类型。不同数据类型的解释请参考 § 2.2.1 节。

一些域标记为“label”域，这说明当打印文献列表时（在 § 3.7.3 节的意义下）这些域通常用于标签缩写。Biblatex 会自动创建支持这些域的宏，详见 § 3.7.3。

abstract 域 (文本)

该域用来记录 `bib` 文件中的摘要，在某些特别的文献样式会打印出来。但在所有的标准文献样式中都不会使用。

addendum 域 (文本)

在条目末尾打印的杂项文献数据。它与 `note` 域类似，而不同之处是在文献条目末尾打印。

afterword 列表 (姓名)

后记的作者。如果与 `editor` 或 `translator` 相同，那么标准样式在参考文献中就会自动把这些域关联起来。参考 `introduction` 域和 `foreword` 域。

annotation 域 (文本)

该域在实现带注释的文献样式时很有用。所有的标准文献样式都不使用。请注意，该域与 `annotator` 域毫无关系，后者是释文（被引用著作的一部分）的作者。

annotator 列表 (姓名)

释文的作者。如果与 `editor` 或 `translator` 相同，那么标准样式在参考文献中就会自动把这些域关联起来。参考 `commentator` 域。

author 列表 (姓名)

`title` 域的作者。

authortype 域 (关键字)

作者的类型。该域会影响介绍作者的字符串。标准文献样式不使用。

bookauthor 列表 (姓名)

`booktitle` 域的作者。

bookpagination 域 (关键字)

如果该作品是另一件作品的一部分，该域就是被附作品的分页格式。也就是说，`bookpagination` 相对于 `pagination` 正如同 `booktitle` 相对于 `title`。该域的值会影响 `pages` 和 `pagetotal` 域的格式。关键字应当是简单的形式。可能的关键字包括 `page`、`column`、`line`、`verse`、`section` 和 `paragraph` 等。参考 `pagination` 域以及 § 2.3.10 节。

booksubtitle 域 (文本)

`booktitle` 的副标题。如果 `subtitle` 域指的是一个更大出版物中的一部分作品的副标题，那么该域则给出了整个作品的副标题。参考 `subtitle`。

booktitle 域 (文本)

如果 `title` 域指的是一个更大出版物中的一部分工作的标题，那么该域则给出了整个作品的标题。参考 `title`。

booktitleaddon 域 (文本)

`booktitle` 的附语，会用不同的字体打印。

chapter 域 (文本)

作品的章节或其它单元。

commentator 列表 (姓名)

作品评论的作者。请注意，该域用于那种带评论的作品版本，即，在作者之外还有一位评论者。如果作品是独立的评论，那么评论者应该在 `author` 域中给出。如果评论者与 `editor` 或 `translator` 相同，那么标准样式就会在文献中自动将这些域关联起来。参考 `annotator` 域。

date 域 (日期)

出版日期。参考 `month` 和 `year` 域以及 § 2.3.8 节。

doi 域 (抄录)

作品的数字对象标识符（Digital Object Identifier, DOI）。

edition 域 (整数或文本)

出版物的版次。这必须是整数而不是序数。不要用 `edition={First}` 或 `edition={1st}`，而要用 `edition={1}`。文献样式会将其转为跟语言相关的序数。也可以用文本字符串表示版次，例如“Third, revised and expanded edition”。

editor 列表 (姓名)

`title`、`booktitle` 或者 `maintitle` 的编辑，这取决于条目类型。如果不是“`editor`”的话，使用 `editortype` 域来确定具体的角色。更多提示参考 § 2.3.6 节。

editora 列表 (姓名)

次要编辑，执行汇集、编校等不同编辑任务。使用 `editoratype` 域来确定具体的角色。更多提示参考 § 2.3.6 节。

editorb 列表 (姓名)

另一位执行不同任务的次要编辑。使用 `editorbtype` 域来确定具体的角色。更多提示参考 § 2.3.6 节。

editorc 列表 (姓名)

另一位执行不同编辑任务的次要编辑。使用 `editorctype` 域来确定具体的角色。更多提示参考 § 2.3.6 节。

editortype 域 (关键字)

`editor` 执行的编辑任务类型。默认支持的任务包括 `editor`、`compiler`、`founder`、`continuator`, `redactor`、`reviser` 和 `collaborator`。默认值是“`editor`”，此时该域可以省略。更多提示参考 § 2.3.6 节。

editoratype 域 (关键字)

类似于 `editortype` 但对应的是 `editora` 域。更多提示参考 § 2.3.6 节。

editorbtype 域 (关键字)

类似于 `editortype` 但对应的是 `editorb` 域。更多提示参考 § 2.3.6 节。

editorctype 域 (关键字)

类似于 `editortype` 但对应的是 `editorc` 域。更多提示参考 § 2.3.6 节。

eid 域 (文本)

`@article` 的电子标识符（electronic identifier）。

entrysubtype 域 (文本)

该域用于确定一个条目类型的子类型。它不会在标准样式中使用，但可用于支持细粒化条目类型的文献样式。

eprint 域 (抄录)

在线出版物的电子标识符。它大致相当于 DOI，但针对于某个档案、资源库、服务或系统。参考 § 3.12.7 一节以及 `eprinttype` 和 `eprintclass` 域。

eprintclass 域 (文本)

由 eprinttype 域指明的资源额外信息。它可以是档案的一部分、标识服务的路径、某个排序的分类等等。参考 § 3.12.7 一节以及 eprint 和 eprinttype 域。

eprinttype 域 (文本)

eprint 标识符的类型，例如 eprint 所指的档案、资源库、服务或系统的名称。参考 § 3.12.7 一节以及 eprint 和 eprintclass 域。

eventdate 域 (日期)

会议、研讨会或其它在 @proceedings 和 @inproceedings 条目中事件的发生日期。该域还可以用于在 § 2.1.3 一节所列的定制类型。参考 eventtitle 和 venue 域以及 § 2.3.8 一节。

eventtitle 域 (文本)

会议、研讨会或其它在 @proceedings 和 @inproceedings 条目中事件的标题。该域还可以用于在 § 2.1.3 一节所列的定制类型。请注意，该域处理事件的主标题。诸如“Proceedings of the Fifth XYZ Conference”之类的信息会归入 titleaddon 或 booktitleaddon 域。参考 eventdate 和 venue 域。

eventtitleaddon 域 (文本)

eventtitle 域的附语。例如可以用于已知事件的首字母缩写词。

file 域 (抄录)

某个作品的 PDF 或其它版本的本地链接。标准文献样式中不使用。

foreword 列表 (姓名)

作品前言的作者。如果前言的作者与 editor 或 translator 相同，那么标准样式就会在文献中自动将其与这些域关联起来。参考 introduction 和 afterword 域。

holder 列表 (名称)

@patent 的持有者（如果与 author 不同的话）。注意，共同持有者需要各自放到额外的花括号里，参考 § 2.3.3 一节。该域可以用于 § 2.1.3 一节所列的定制类型中。

howpublished 域 (文本)

不适合任何常见类型的非常规出版物的出版公告。

indextitle 域 (文本)

在索引中用于取代常规 title 域的标题。如果你有一个带有“An Introduction to ...”之类标题的条目，并且想索引为“Introduction to ..., An”，那么就可以使用该域。样式作者需要注意，如果 indextitle 没有定义，那么 Biblatex 会自动将 title 域的值复制给 indextitle。

institution 列表 (文本)

大学或其它研究机构的名字，这取决于条目类型。传统的 BibTeX 使用 `school` 域来表示。本宏包也支持 `school`，但只作为本域的别名。参考 §§ 2.2.5 和 2.3.4。

introduction 列表 (姓名)

作品导论的作者。如果导论的作者与 `editor` 或 `translator` 相同，那么标准样式就会在文献中自动将这些域关联起来。参考 `foreword` 和 `afterword` 域。

isan 域 (文本)

音像作品的视听数码国际标准（International Standard Audiovisual Number, ISAN）。不会在标准文献样式中使用。

isbn 域 (文本)

书籍的国际标准书号（International Standard Book Number, ISBN）。

ismn 域 (文本)

乐谱等的发行音乐作品的国际标准印刷音乐作品编码（International Standard Music Number, ISMN）。

isrn 域 (文本)

技术报告的国际标准技术报告编码（International Standard Technical Report Number, ISRN）。

issn 域 (文本)

连续出版物的国际标准连续出版物号（International Standard Serial Number, ISSN）。

issue 域 (文本)

期刊的卷数。该域适用的期刊特点是，每一卷由“Spring”或“Summer”等名称而不是由月份或数字确定。由于 `issue` 的位置与 `month` 和 `number` 类似，该域也可用于双重卷数或其它特殊场合[16]。参考 `month` 和 `number` 域以及 § 2.3.9 一节。

issuesubtitle 域 (文本)

期刊或其它连续出版物中某一卷的副标题。

issuetitle 域 (文本)

期刊或其它连续出版物中某一卷的标题。

iswc 域 (文本)

音乐作品的国际标准音乐作品编码（International Standard Work Code, ISWC）。标准文献样式中不使用。

[16] 例如增刊、特刊等。——译注

journalsubtitle 域 (文本)

期刊、报纸或其它连续出版物的副标题。

journaltitle 域 (文本)

期刊、报纸或其它连续出版物的标题。

label 域 (文本)

如果缺失生成常规标签所需的某一数据，那么该域就是替代常规标签而被引用样式所用的指定文本。例如，当*作者-年份*引用样式要生成条目的引用，但作者或年份缺失，那么它就会使用后备的 `label`。详情请参考 § 2.3.2 节。请注意，与 `shorthand` 域相反，`label` 只是作为后备而使用。同样参考 `shorthand`。

language 列表 (关键字)

作品的语言。语言可以按字面或者本地化关键字确定。如果使用本地化关键字，那么前缀 `lang` 将省略。参考 `origlanguage` 域并比较 § 2.2.3 节中的 `langid`。

library 域 (文本)

该域可用于记录图书馆名称或书架号码等信息。某些特殊的文献样式可能需要打印出来。但在标准文献样式中不使用。

location 列表 (文本)

出版地，即 `publisher` 或 `institution` （取决于条目类型）的所在地。传统 BibTeX 使用 `address` 域，在这里作为别名也被支持。参考 §§ 2.2.5 和 2.3.4 几节。在 `@patent` 条目里，该列表表示专利范围。该文本列表可用于 § 2.1.3 中的定制类型。

mainsubtitle 域 (文本)

对应于 `maintitle` 的副标题。参考 `subtitle` 域。

maintitle 域 (文本)

多卷本书籍（例如著作集）的主标题。如果 `title` 或 `booktitle` 域指的是多卷本中某一卷的标题，那么该域则给出了全集的标题。

maintitleaddon 域 (文本)

`maintitle` 的附言，会用不同的字体打印。

month 域 (日期部分)

出版月份。必须是整数，而不能是序数或字符。例如使用 `month={1}` 而不是 `month={January}`。文献样式会在需要时将它转换为语言相关的字符串或序数。参考 `date` 以及 §§ 2.3.9 和 2.3.8。

nameaddon 域 (文本)

参考文献中立即在作者名之后输出的插入语。标准文献样式中不使用。该域可用于添加别名或笔名（或者给出原名，如果作者的化名更熟知的话）。

note 域 (文本)

不可归类于其它域的杂项文献数据。`note` 域可以用于记录自由格式的文献数据。典型的 `note` 域包括一些出版信息，例如“Reprint of the edition London 1831”。参考 `addendum`。

number 域 (整数)

期刊的期数或者 `series` 丛书中某本书的卷数/期数。参考 `issue` 以及 §§ 2.3.7 和 2.3.9。在 `@patent` 条目中，这是专利或专利申请的号码或记录标识。应该是整数，但实际上不必是阿拉伯数字的形式，因为 Biber 为了排序会自动将罗马数字或者阿拉伯数码转成整数。

organization 列表 (文本)

出版 `@manual` 或者 `@online` 资源以及赞助会议的组织。参考 § 2.3.4 节。

origdate 域 (日期)

如果作品是译作、重印或其它类似情况，该域指的是原始版次的出版日期。在标准文献样式中不使用。参考 `date` 域。

origlanguage 域 (关键字)

如果作品是译作，该域指的是原作的语言。参考 `language` 域。

origlocation 列表 (文本)

如果作品是译作、重印或其它类似情况，该域指的是原始版次的 `location`。标准文献样式不使用。参考 `location` 域和 § 2.3.4 节。

origpublisher 列表 (文本)

如果作品是译作、重印或其它类似情况，该域指的是原始版次的 `publisher`。在标准文献样式中不使用。参考 `publisher` 域和 § 2.3.4 节。

origtitle 域 (文本)

如果作品是译作，该域指的是原作的 `title`。标准文献样式不使用。参考 `title` 域。

pages 域 (范围)

一个或多个页码数或页码范围。如果这项作品是其它出版作品的一部分，例如期刊或文集中的文章，该域指的是在那项作品中的相关页码范围。它也可以用于指明著作中某一特定部分（例如一本书中的一章）。

pagetotal 域 (文本)

作品的总页码数。

pagination 域 (关键字)

作品的分页格式。该域的值或影响引用命令的 ⟨*postnote*⟩ 选项的格式。该域应当以单数的形式给出。可能的关键字包括 `page`、`column`、`line`、`verse`、`section` 和 `paragraph`。参考 `bookpagination` 域以及 §§ 2.3.10 和 3.13.3 节。

part 域 (文本)

部分卷的编号。该域只用于书籍而不能用于期刊。它可以用于一个逻辑卷册包括两个或更多实际卷册的情形。此时逻辑卷册的编号由 `volume` 给出，而这一卷的每一部分的编号由 `part` 给出。参考 `volume` 域。

publisher 列表 (文本)

出版者的名字。参考 § 2.3.4 一节。

pubstate 域 (关键字)

作品的出版状态，例如“in press”。已知的出版状态请参考 § 4.9.2.11 一节。

reprinttitle 域 (文本)

作品重印时的标题。标准样式中不使用。

series 域 (文本)

丛书的名称，例如“Studies in ...”，或者期刊系列的编号。系列出版的丛书通常带有编号。其编号或者卷数由 `number` 域给出。请注意，`@article` 条目类型也使用 `series` 域，但是以一种特别的方式处理。参考 § 2.3.7 一节。

shortauthor 列表 (姓名) Label field

作者名的缩写形式。该域主要用于集体作者的缩写形式。参考 § 2.3.3 一节。

shorteditor 列表 (姓名) Label field

编辑名的缩写形式。该域主要用于集体编辑的缩写形式。参考 § 2.3.3 一节。

shorthand 域 (文本) Label field

替代通常的标签而被引用样式使用的指定域。如果有定义，那么它会覆盖默认的标签。参考 `label` 域。

shorthandintro 域 (文本)

本宏包附带一些的引用样式会使用比较冗长的引用格式，例如，在第一次引用时会使用诸如“henceforth cited as [shorthand]”的短语来声明 shorthand。如果 `shorthandintro` 域有定义，它将覆盖标准的声明短语。请注意，使用的备选短语必须包含 shorthand。

shortjournal 域 (文本) Label field

`journaltitle` 的缩写版本或其首字母缩略语。标准文献样式中不会使用。

shortseries 域 (文本) Label field

`series` 的缩写版本或其首字母缩略语。标准文献样式中不会使用。

shorttitle 域 (文本) Label field

缩略形式的标题。该域通常不会包括在参考文献列表中。它可用于 `author-title` 格式的引用。如果有该域的话，`author-title` 引用样式使用该域来替代 `title` 域。

subtitle 域 (文本)

作品的副标题。

title 域 (文本)

作品的标题。

titleaddon 域 (文本)

`title` 的附文，会用不同字体打印。

translator 列表 (名称)

`title` 或 `booktitle` 的译者，具体取决于条目类型。如果译者与 `editor` 相同，标准样式会在文献中自动将这些域关联起来。

type 域 (关键字)

`manual`、`patent`、`report` 或 `thesis` 的类型。该域可用于 § 2.1.3 节的定制类型。

url 域 (uri)

在线出版物的 URL。如果它不是 URL-转义的（没有“%”字符），那么会根据 RFC 3987 [17] 将其 URI-转义，也就是说，即使 Unicode 字符也会正确转义。

urldate 域 (日期)

`url` 域中网址的获取日期。参考 § 2.3.8 一节。

venue 域 (文本)

`@proceedings` 和 `@inproceedings` 条目中的会议、研讨会或其它事件的地点。该域可用于 § 2.1.3 一节所列的定制类型。请注意，`location` 列表指的是出版地点，因此对应于 `publisher` 和 `institution` 列表。而会议事件的会场地点则由 `venue` 域给出。参考 `eventdate` 和 `eventtitle` 域。

version 域 (文本)

软件、手册等作品的修订次数。

[17] 参考https://tools.ietf.org/html/rfc3987——译注

volume 域 (整数)

多卷本或连续出版物中作品的卷数。应当是整数，但不必是阿拉伯数字的形式。这是因为 Biber 为了排序会将罗马数字和阿拉伯数码自动转成整数。参考 `part` 域。

volumes 域 (整数)

多卷本著作的总卷数。根据文献条目类型，该域对应于 `title` 或 `maintitle` 域。应当是整数，但不必是阿拉伯数字的形式。这是因为 Biber 为了排序会将罗马数字和阿拉伯数码自动转成整数。

year 域 (文本)

出版年份。不过使用 `date` 域更好些，因为它也和普通年份兼容。参考 § 2.3.8 节。

### 2.2.3 特殊域

本节中的域不包括可打印的数据，而是有其它用途。在默认数据模型中可用于所有条目类型。

crossref 域 (条目关键字)

该域提供的条目关键字可用于交叉引用。带有 `crossref` 域的子条目可以从由 `crossref` 域指定的父条目继承数据。如果引用某个父条目的子条目数量达到一个阈值，该父条目就会自动添加到参考文献中，即使它没有显式地被正文引用。该阈值可以由 § 3.1.2.1 节中的 `mincrossrefs` 宏包选项设置。样式作者请注意，在 Biblatex 层面上子条目的 `crossref` 域是否有定义取决于父条目是否可用。如果父条目可用，那么子条目的 `crossref` 域将被定义。反之，子条目仍然可以继承父条目的数据但是其 `crossref` 域是未定义的。父条目是否被添加到文献中（间接地由于阈值或者直接被显式地引用）对于该域的定义并不重要。参考本节中的 `xref` 域以及 § 2.4.1 一节。

entryset 域 (分隔值)

该域用于指明条目集。详见 § 3.12.5 节。该域在后端过程中被处理，不出现在 `.bbl` 中。

execute 域 (代码)

包含了任意 TEX 代码的特殊域，这些代码会在获取相应的条目数据时被执行。这对处理特殊情况很有用。概念上，该域可以类比于 § 4.10.6 节中的钩子命令 `\AtEveryBibitem`、`\AtEveryLositem` 和 `\AtEveryCitekey`，但不同之处在于该域可以基于 `bib` 文件中的每一条目进行逐条定义。该域中的任何代码都会在这些钩子命令后立即自动执行。

gender 域 (匹配 `sf`、`sm`、`sn`、`pf`、`pm`、`pn`、`pp` 其中之一的模式)

作者或编辑（如果没有作者的话）的词性。支持以下以下标识符：`sf`（阴性单数，单独的女性名）, `sm`（阳性单数，单独的男性名），`sn`（中性单数，单独的中性名），

pf（阴性复数，多个女性名），pm（阳性复数，多个男性名），pn（中性复数，多个中性名），pp（复数，不同词性名的混合）。这一信息只在特殊的文献和引用样式以及某些语言中是需要的。例如，某一引用样式会用拉丁语“idem”等词汇来代替反复出现的作者名字。如果按照英语或法语的习惯使用这一拉丁词汇，那么就没有必要确定词性。然而在德语出版物中，这样的词汇通常用德语给出，此时就依赖于词性。

ids 域 (条目关键字的分隔值列表)

主要引用关键字的别称。一个条目可以通过别称引用，Biblatex 会将引用视为它使用了原本的引用关键字。借助于该域，用户可以在改变引用关键字的同时仍然可以使用带有旧引用关键字条目的其它文件。该域在后端过程中被处理，不出现在 .bbl 中。

indexsorttitle 域 (文本)

排序索引使用的标题。与 indextitle 域不同，该域只用于排序。而索引中打印出来的标题是 indextitle 或title 域。当标题中含有与索引排序相冲突的特殊字符或命令时，该域会很有用。考虑如下例子：

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

文献样式作者请注意，当 indexsorttitle 没有定义时，Biblatex 会自动将 indextitle 或 title 域的值复制给该域。

keywords 域 (分隔值)

关键词的分隔值列表。这些关键词用于文献过滤（filter），参考 §§ 3.7.2 和 3.12.4 节，通常不会打印出来。请注意，使用默认的分隔符（西文逗号）时，分隔符左右的空格会被忽略。

langid 域 (标识符)

文献条目的语种标识。出于向后兼容性另提供了别称 hyphenation。标识符必须是 babel/polyglossia 宏包中的语言名称。该信息用于文献中切换断词模式和本地化字符串。请注意，语言名称是大小写敏感的。目前本宏包支持的语言在表 3 中给出。需要注意的是，babel 宏包将标识符 english 当作 british 或 american 的别称，具体取决于 babel 的版本。而 Biblatex 宏包总是将其当作 american 的别称。为了避免可能的混淆，最好使用语言标识符 american 和 british（babel）或者语言选项来确定语言变种（polyglossia，使用 langidopts 域）。比较 § 2.2.2 一节中的 language 域。

langidopts 域 (文本)

对于 polyglossia 的用户，该域会允许每个条目有自己的语言选项。当使用本宏包的选项 autolang=langname 时，它们将被传递到 polyglossia 的语言切换机制。例如，以下的域

| 语言 | 地区/方言 | 标识符 |
|---|---|---|
| 加泰罗尼亚语 | 西班牙、法国、安道尔、意大利 | `catalan` |
| 克罗地亚语 | 克罗地亚、波黑、塞尔维亚 | `croatian` |
| 捷克语 | 捷克共和国 | `czech` |
| 丹麦语 | 丹麦 | `danish` |
| 荷兰语 | 荷兰 | `dutch` |
| 英语 | 美国 | `american`, `USenglish`, `english` |
| | 英国 | `british`, `UKenglish` |
| | 加拿大 | `canadian` |
| | 澳大利亚 | `australian` |
| | 新西兰 | `newzealand` |
| 芬兰语 | 芬兰 | `finnish` |
| 法语 | 法国、加拿大 | `french` |
| 德语 | 德国 | `german` |
| | 奥地利 | `austrian` |
| | 瑞士 | `swissgerman` |
| 德语（新正字法） | 德国 | `ngerman` |
| | 奥地利 | `naustrian` |
| | 瑞士 | `nswissgerman` |
| 希腊语 | 希腊 | `greek` |
| 意大利语 | 意大利 | `italian` |
| 挪威语 | 挪威 | `norwegian`, `norsk`, `nynorsk` |
| 波兰语 | 波兰 | `polish` |
| 葡萄牙语 | 巴西 | `brazil` |
| | 葡萄牙 | `portuguese`, `portuges` |
| 俄语 | 俄罗斯 | `russian` |
| 斯洛伐克语 | 斯洛伐克 | `slovak` |
| 斯洛文尼亚语 | 斯洛文尼亚 | `slovene` |
| 西班牙语 | 西班牙 | `spanish` |
| 瑞典语 | 瑞典 | `swedish` |

**Table 3: 支持的语种**

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

会将文献条目置于如下代码块中

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

options 域 (分隔的 ⟨*key*⟩=⟨*value*⟩ 选项)

⟨*key*⟩=⟨*value*⟩ 形式的条目选项分隔值列表。该域用于设置每一条目的选项，详见 § 3.1.3 一节。请注意，引用和文献样式会定义额外的条目选项。

presort 域 (字符串)

用于修正文献排列顺序的特殊域。该域是文献排列时排序程序考虑的第一个项目，因此可用于将文献条目分组。这在使用文献过滤创建文献细分时很有用。更多细节请参考 § 3.5 以及 § 4.5.6 节。该域在后端过程中被处理，不出现在 `.bbl` 中。

related 域 (分隔值)

与本条目相关的其它条目的引用关键字。相互关系由 `relatedtype` 域来确定。进一步细节请参考 § 3.4 一节。

relatedoptions 域 (分隔值)

为相关条目设置基于类型的选项。请注意，这不会设置相关条目本身的选项，而只会作用到用作数据源的父条目的临时副本上。

relatedtype 域 (标识符)

标识符，为列在 `related` 域中的关键字列表指定相互关系类型。该标识符是本地化字符串，会打印在相关条目列表的数据之前。该域也用于为相关条目指明类型相关的格式指令和参考文献宏。更多信息参考 § 3.4 节。

relatedstring 域 (文本)

用于覆盖 `relatedtype` 确定的参考文献字符串。更多信息参考 § 3.4 节。

sortkey 域 (文本)

用来修正文献排序的域。该域可以认为是最主要的排序键值。当存在时，Biblatex 会在排序时使用该域，并且忽略除了 `presort` 域之外的所有信息。详见 § 3.5 节。该域在后端过程中被处理，不出现在 `.bbl` 中。

sortname 列表 (姓名)

用于修正文献排序的姓名或姓名列表。当存在时，该列会在文献排序时取代 author 或 editor 域。详见 § 3.5 节。该域在后端过程中被处理，不出现在 .bbl 中。

sortshorthand 域 (文本)

与 sortkey 类似但用于缩略语列表中。当存在时，Biblatex 会在缩略语列表排序中用该域取代 shorthand 域。当 shorthand 域含有带格式命令（如 \emph 或 \textbf）的缩略语时，该域是很有用的。该域在后端过程中被处理，不出现在 .bbl 中。

sorttitle 域 (文本)

用于修正文献排序的域。当存在时，该域会在文献排序时取代 title 域。如果一个条目带有“An Introduction to...”这样的标题，并且你想让它按字母“I”而不是“A”排序，那么 sorttitle 域就会派上用场。这时，你就可以在 sorttitle 域中填上“Introduction to...”。详见 § 3.5 节。该域在后端过程中被处理，不出现在 .bbl 中。

sortyear 域 (整数)

用于修正文献排序的域。当存在时，该域会在文献排序时取代 year 域。详见 § 3.5 节。该域在后端过程中被处理，不出现在 .bbl 中。

xdata 域 (条目关键字的分隔值列表)

该域从一个或更多 @xdata 条目中继承数据。从概念上讲，xdata 域与 crossref 和 xref 域相关：crossref 创建一个父/子的逻辑关系并继承数据；xref 创建一个不继承数据的父/子逻辑关系；而 xdata 则继承数据却不创建关系。xdata 的值可以是一个单独的条目关键字或者条目关键字的分隔值列表。详见 § 3.12.6 节。该域在后端过程中被处理，不出现在 .bbl 中。

xref 域 (条目关键字)

该域可用于代替交叉引用机制。它与 crossref 域的不同之处在于，子条目不会从其 xref 域所列的父条目中继承数据。如果引用某个父条目的子条目数量达到一个阈值，该父条目就会自动添加到文献中，即使它并没有显式地被引用。该阈值可以由 § 3.1.2.1 节的 mincrossrefs 宏包选项设置。文献样式作者需要注意，在 Biblatex 层面上，子条目的 xref 域是否有定义取决于父条目是否可用。如果父条目可用，那么子条目的 xref 域将被定义。反之，其 xref 域是未定义的。父条目是否被添加到文献中（间接地由于阈值或者直接被显式地引用）对于域的定义并不重要。参考本节中的 crossref 域以及 § 2.4.1 节。

### 2.2.4 可定制域

本节中的域用于特定的参考文献样式，标准样式不使用。

name[a–c] 列表 (姓名)

特殊文献样式的定制列表。标准文献样式不使用。

name[a–c]type 域 (关键字)

类似于 `authortype` 和 `editortype` 域，不过对应的是 `name[a–c]` 域。标准文献样式不使用。

list[a–f] 列表 (文本)

特殊文献样式的定制列表。标准文献样式不使用。

user[a–f] 域 (文本)

特殊文献样式的定制列表。标准文献样式不使用。

verb[a–c] 域 (文本)

类似于以上的定制域，不过这些是抄录域。标准文献样式不使用。

### 2.2.5 域的别名

本节列出的别名用于向后兼容传统 BibTeX 以及其它基于传统 BibTeX 样式的应用。请注意，当 `bib` 文件被处理时就立即解析这些别名。因此所有的文献和引用样式必须使用它们所指的域的名称，而不能是别名。而在 `bib` 文件中，既可以使用别名也可以使用原名，但不能同时使用。

address 列表 (文本)

`location` 的别名，用于 BibTeX 相容性。传统的 BibTeX 使用这一稍微有些误导性的域 `address` 来表示出版地点，即出版者的所在地，而 Biblatex 使用更一般的 `location` 域。参考 §§ 2.2.2 和 2.3.4 节。

annote 域 (文本)

`annotation` 的别名，用于 `jurabib` 宏包兼容性。参考 § 2.2.2 节。

archiveprefix 域 (文本)

`eprinttype` 的别名，用于 arXiv 的兼容性。参考 §§ 2.2.2 和 3.12.7 节。

journal 域 (文本)

`journaltitle` 的别名，用于 BibTeX 兼容性。参考 § 2.2.2 节。

key 域 (文本)

`sortkey` 的别名，用于 BibTeX 兼容性。参考 § 2.2.3 节。

pdf 域 (抄录)

`file` 的别名，用于 JabRef 兼容性。参考 § 2.2.2 节。

primaryclass 域 (literal)

An alias for `eprintclass`, provided for arXiv compatibility. See §§ 2.2.2 和 3.12.7.

primaryclass　域 (文本)

eprintclass 的别名，用于 arXiv 的兼容性。参考 §§ 2.2.2 和 3.12.7 节。

school　列表 (文本)

institution 的别名，用于 BibTeX 兼容性。institution 域用于传统 BibTeX 中的技术报告，而 school 域处理与之相关的研究机构。在这两种情况下，Biblatex 宏包都会使用一般的域 institution。参考 §§ 2.2.2 和 2.3.4。

## 2.3 使用注意事项

对于熟悉 BibTeX 的用户来说，本宏包支持的绝大部分条目类型和域都是很直观的。然而，且不说本宏包额外新增的类型和域，一些很常见的类型和域的处理方式也需要进一步解释一下。本宏包考虑到包含一些相容性代码，用于处理那些由传统 BibTeX 样式产生的 bib 文件。但不幸的是，自动处理所有留传下来的文件是不可能的，因为 Biblatex 的数据模型域传统的 BibTeX 有少许不同。因此，为了在本宏包下能正确运行，这样的 bib 文件也许需要稍作修改。大体上，下列事项与传统的 BibTeX 样式不同：

- @inbook 条目类型。详见 §§ 2.1.1 和 2.3.1 节。
- institution、organization、publisher 域以及相应的别名 address 和 school。详见 §§ 2.2.2、2.2.5、2.3.4 节。
- 一些标题类型的处理。详见 § 2.3.5 节。
- series 域。详见 §§ 2.2.2 和 2.3.7 节。
- year 和 month 域。详见 §§ 2.2.2、2.3.8、2.3.9 节。
- edition 域。详见 § 2.2.2 节。
- key 域。详见 § 2.3.2 节。

jurabib 宏包的用户请注意，shortauthor 域被 Biblatex 视作姓名列表，详见 § 2.3.3 节。

### 2.3.1 @inbook 条目类型

@inbook 条目类型只用于那些书籍中有自己标题的独立部分。它与 @book 的关系正如同 @incollection 与 @collection 的关系。参考 § 2.3.5 中的例子。如果你想要指的是书的某一章节，使用 book 类型并添加 chapter 或 pages 域即可。究竟参考文献是否可以引用章节是有争议的，因为章并不是文献实体。

### 2.3.2 缺失和可忽略数据

在 § 2.1.1 节中标记为“required”的域并不一定在所有情况下都是严格要求的。大部分只含 title 域的条目类型对于本宏包所带的文献样式也可以过得去。就参考文献而言，匿名出版的书籍、没有明确编辑的周期出版物、或者没有明确

作者的软件手册都应当不会有问题。但是，引用样式也许会有不同的要求。例如，author-year 引用格式就明确要求 author/editor 域和 year 域。

一般来说可以使用 label 域代替引用要求的任意缺失数据。label 域的使用方式取决于引用样式。如果 author/editor 域或 year 域缺失，那么本宏包所带的 author-year 引用样式会将 label 域作为后备信息。另一方面，数字样式根本不会用到这些，因为数字格式与可用数据无关。此外，author-title 样式也会忽略这些，因为 title 域足够生成惟一的引用，并且标题几乎在所有情形中都是存在的。label 域也可以用于覆盖自动生成的 labelalpha 域中的非数值部分，这用于按字母排序的引用样式。详见 § 4.2.4 节。

请注意，当 author 和 editor 域都缺失时，传统的 BibTeX 样式支持 key 域用于依字母排序。Biblatex 宏包将 key 视为 sortkey 的别名。此外，它还提供了非常细致化的排序控制，详见 §§ 2.2.3 和 3.5 节。natbab 宏包使用 key 域作为后备的引用标签。使用 label 域来代替它。

### 2.3.3 集体作者和编者

集体作者和编辑分别在 author 和 editor 域中给出。请注意，他们必须再用花括号括起来，以防被认为是个人姓名进而被分解成姓名成分。如果你想在引用时给出简称或首字母缩写的形式，请使用 shortauthor 域。

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

默认的引用样式会在所有的引用里使用短名称而在参考文献中打印全名。对于集体编辑，则使用 editor 和 shorteditor 域。由于这些域都被视作姓名列表，因此，只要把所有的合作者和单位用花括号括起来，就可以将个人姓名和集体名称混合起来使用。

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

从 jurabib 宏包切换到 Biblatex 宏包的用户需要注意，shortauthor 域被视作姓名列表。

### 2.3.4 文本列表

按照 § 2.2 节，institution、organization、publisher 和 location 等域是文本列表。origlocation、origpublisher，以及作为别名的 address 和 school 域也是如此。所有的这些域都可以包含一个由关键词“and”分隔的项目列表。如果它们本身带有“and”文本，那么必须用括号括起来。

```
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},
```

请注意以上例子中作为文本和作为列表分隔符的“and”之间的区别。你也可以把整个名称用括号括起来：

```
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}},
```

对于没有为在 Biblatex 宏包中使用而更新的旧文件，如果它们的域中不含“and”文本，那么仍然可以运行。然而，请注意此时你会缺失那些关于文本列表的新特性，例如可配置的格式和自动截词。

### 2.3.5 标题

以下例子展示了如何处理不同类型的标题。首先是一个作为整体的五卷本作品：

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

多卷本作品的每一卷通常有自己的标题。假设全集的第四卷是莎士比亚的十四行诗，并且我们要单独引用该卷：

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

如果单卷没有标题，我们在 `title` 域中使用主标题，并标明卷数：

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

在下个例子里，我们引用一卷的一部分，但是该部分自成一个独立作品且有自己的标题。相应的卷也有一个标题，并且整个作品有一个主标题：

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

假设全集的第一卷是翻版时由一位著名学者写的随笔。这不是通常意义上编辑写的简介，而是一份独立的作品。全集同样另有一位编辑：

```
@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},
  ...
```

更多例子请参考 § 2.3.7 节。

### 2.3.6 编辑角色

`editor` 域（包括 `editor`、`editora`、`editorb`、`editorc`）中编辑作用的类型可以由相应的 `editor...type` 域确定。默认支持以下的角色。缺省值是“`editor`”，此时 `editortype` 域可以省略。

editor 主要编辑。这是最普遍的编辑角色，也是默认值。

compiler 类似于 `editor` 但使用场合是编辑主要进行编纂工作。

founder 诸如“全集”或长期的法律评论等连续出版物或综合出版项目的创刊者。

continuator 继续创刊编辑（`founder`）工作的编辑，不过随后由现任编辑（`editor`）所接替。

redactor 从事编修工作的次要编辑。

reviser 从事校订工作的次要编辑。

collaborator 次要编辑或者主编的顾问。

例如，如果编辑的任务是编纂的话，你可以在相应的 editortype 域中指明：

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

除主编之外可以有次要编辑：

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

连续出版物或长期出版项目通常有不同阶段的编辑。例如，在现任编辑之外还可以有一位创刊编辑：

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

请注意，只有 editor 在引用和文献排列中起作用。如果一个条目特地引用创刊编辑（并且据此在文献中排列），那么创刊者在 editor 域中给出，而现任编辑则移动到 editor... 域中：

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

可以通过初始化和定义新的本地化关键字来增加更多的角色，关键字的名称对应于 editor...type 域中的标识符。详见 §§ 3.9 和 4.9.1 节。

### 2.3.7 出版物和期刊系列

在传统的 BibTeX 样式中，series 域既用于多卷本作品的主标题，也用于出版物的系列，也就是说，同一位出版者的关系较松散的一系列书籍，主要关于一个

大致的方向或者同一个研究领域。这种用法是模糊不清的。因此，本宏包引入了 maintitle 域来表示多卷本作品，而 series 只用于出版物系列。此时该系列中的一本书的卷号由 number 域给出：

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

@article 条目类型也使用 series 域，但是使用方式比较特殊。首先，会执行一个测试来确定该域的值是否是整数。如果是的话，它会以序数的形式打印；反之，会执行另一个测试来确定它是否是本地化关键字。如果是的话，会打印本地化字符串；反之则按照本身打印。考虑下面这个以带有数字系列的期刊的例子：

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

该条目会打印成“*Journal Name.* 3rd ser. 15.7 (1995)”。一些期刊会使用“旧系列”和“新系列”等指定名而不是一个数字。这样的指定名也可以由 series 域给出，或者是一个文本字符串，或者是一个本地化关键字。考虑如下这个使用本地化关键字 newseries 的例子：

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

该条目会打印成“*Journal Name.* New ser. 9 (1998)”。默认定义的本地化关键字列表请参考 § 4.9.2 节。

### 2.3.8 日期和时间规范

日期域遵循扩展日期时间格式[18]（Extended Date/Time Format, EDTF）标准 0 和标准 1，例如默认数据模型的日期域 date、origdate、eventdate 和 urldate。此

[18] https://www.loc.gov/standards/datetime/pre-submission.html

| 日期规格 | 日期格式（例） | |
|---|---|---|
| | 短格式/12 小时格式 | 长格式/24 小时格式 |
| 1850 | 1850 | 1850 |
| 1997/ | 1997– | 1997– |
| /1997 | –1997 | –1997 |
| 1997/unknown | 1997– | 1997– |
| 1997/* | 1997– | 1997– |
| unknown/1997 | –1997 | –1997 |
| */1997 | –1997 | –1997 |
| 1997/open | 1997– | 1997– |
| open/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 |

**Table 4: 日期规范**

外也支持 ISO8601-2 的当前草案[19] 的第 4.5 节中的开放式范围规范。相比于有些混乱的ISO8601V2004允许格式，EDTF 是其中一个更严格的子集，更适用于文献数据。除了 EDTF 空日期范围标记外，还通过给定范围分隔符并省略结束或开始日期的方式（例如 YYYY/、/YYYY）指明无末端或无开端的日期范围。表 4 列出了一些有效的日期规范以及由 Biblatex 自动生成的日期格式。日期格式与语言有关，因此会自动调整。如果条目中没有 date 域，Biblatex 还会考虑 year 和 month 域。不过这仅仅出于对传统 BibTeX 的向后兼容性，并不鼓励使用。因为显式的 year 和 month 域不能解析为日期的元信息标记，只能原样使用。样式作者需要注意，date 或 origdate 等日期域只在 bib 文件中有效。随着 bib 文件的处理，所有的日期都被解析并分为各个日期部件。通过 § 4.2.4.3 节讨论的特殊域，日期和时间部件可以为样式所用。更多信息请参考该节和156 页的表 11 节。

EDTF 日期是天文学日期，其中第“0”年是存在的。当输出公元前年代（BCE/BC era）的日期时（见下面的 dateera 选项），请注意它们通常要早一年，因为公元前年代没有第 0 年（第 0 年就是公元前 1 年）。该转换是自动完成的，见表 6 中的例子。

如同默认日期域，日期域的名称必须以字符串“date”结尾。当在数据模型中添加新的日期域时（见 § 4.5.4 节）必须注意这一点。Biblatex 在读入日期模型后会检查所有的日期域，如果发现有日期域不遵循这一命名约定就会报错并退出。

EDTF 通过负日期格式支持公元前（before common era, BCE/BC）日期，此外还支持“近似”（circa）和不确定的日期。这样的日期格式设置可以检测的内部标记，进而可以插入合适的本地化标记（例如 circa 或 beforecommonera）。另外，未定日期（EDTF 5.2.2）会自动展开成合适的日期范围，随之另有一个⟨*datetype*⟩dateunspecified 域指明未定数据的间隔尺寸。文献样式可以使用该信息构造合适的日期格式，但标准样式不会使用。42 页的表 5 列出了允许的 EDTF 未定日期格式、范围扩展和 ⟨*datetype*⟩dateunspecified 域的值（§ 4.2.4.1 节）。

[19] http://www.loc.gov/standards/datetime/iso-tc154-wg5_n0039_iso_wd_8601-2_2016-02-16.pdf

| 日期规范 | 扩展范围 | 元信息 |
|---|---|---|
| 199u | 1990/1999 | yearindecade |
| 19uu | 1900/1999 | yearincentury |
| 1999-uu | 1999-01/1999-12 | monthinyear |
| 1999-01-uu | 1999-01-01/1999-01-31 | dayinmonth |
| 1999-uu-uu | 1999-01-01/1999-12-31 | dayinyear |

**Table 5: EDTF 5.2.2 未定日期解析**

表 6 展示了使用恰当的测试和格式化的格式。参考 § 4.6.2 节的日期元信息测试以及 § 4.9.2.21 节的本地化字符串。关于相应测试和使用本地化字符串的完整的例子另参考 `96-dates.tex` 示例文件。

在标准样式以及没有定制内部宏 `\mkdaterange*` 的定制样式中，'circa'、不确定信息和年代信息的输出由 § 3.1.2.1 节中的宏包选项 `datecirca`、`dateuncertain`、`dateera` 和 `dateeraauto` 控制。43 页中的 表 6列出了使用这些选项的例子。

### 2.3.9 月份和期刊的卷号

`month` 是整数域。文献样式按照要求将月份转化成不同语言的字符串。出于向后兼容性，你也可以在 `month` 域中使用以下的三字母缩写形式：`jan`、`feb`、`mar`、`apr`、`may`、`jun`、`jul`、`aug`、`sep`、`oct`、`nov`、`dec`。请注意，这些缩写词是 BibTeX 字符串，不能带有任何括号或引号。即，不要用 `month={jan}` 或 `month="jan"`，而直接使用 `month=jan`。像 `month={8/9}` 这样的月份是不可识别的，此时请使用 `date` 域来表示日期范围。季刊通常由“Spring”或“Summer”等标识符确定，这些标识符应在 `issue` 域中给出。在 `@article` 条目中，`issue` 域的位置与 `month` 域类似，并且会覆盖后者。

### 2.3.10 标记页码

当在条目的 `pages` 域中或引用命令的 ⟨*postnote*⟩ 选项中指明页码或页码范围时，通常习惯让 Biblatex 自动添加“p.”或“pp.”等前缀，而这也确实是本宏包的默认设置。然而，一些作品或许使用不同的页码标记格式，或者不是按页码而是按韵节或者行号引用。此时 `pagination` 和 `bookpagination` 就可以起作用了。例如考虑如下条目：

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

`bookpagination` 域会影响文献列表中 `pages` 和 `pagetotal` 的格式。由于 `page` 是默认的，因此在上面这个例子中该域可以省略。此时页码范围的格式是“pp. 53–65”。假设引用该作品时习惯使用韵节号而不是页码数。这可以通过 `pagination` 域反映

| 日期规范 | 格式化日期（例） | |
|---|---|---|
| | 输出格式 | 输出格式注记 |
| 0000 | 1 BC | dateera=christian 打印本地化字符串 beforechrist |
| -0876 | 877 BCE | dateera=secular 打印本地化字符串 beforecommonera |
| -0877/-0866 | 878 BC–867 BC | 使用 \ifdateera 测试和本地化字符串 beforechrist |
| 0768 | 0768 CE | dateeraauto 设置为 1000，并使用本地化字符串 commonera |
| -0343-02 | 344-02 BCE | |
| 0343-02-03 | 343-02-03 CE | 以及 dateeraauto=400 |
| 0343-02-03 | 343-02-02 CE | 以及 dateeraauto=400 和 julian |
| 1723~ | circa 1723 | 使用 \ifdatecirca 测试 |
| 1723? | 1723? | 使用 \ifdateuncertain 测试 |
| 1723?~ | circa 1723? | 使用 \ifdateuncertain 和 \ifdatecirca 测试 |
| 2004-22 | 2004 | 另外，season 设置为本地化字符串 ‘summer’ |
| 2004-24 | 2004 | 另外，season 设置为本地化字符串 ‘winter’ |

**Table 6: 增强的日期规范**

出来，从而影响任何引用命令的 ⟨*postnote*⟩ 选项格式。如引用命令 `\cite[17]{key}` 的随后注记格式就会是“v. 17”。若设置 pagination 域为 section，那么就会产生“§ 17”。进一步的使用说明请参考 § 3.13.3 节。

pagination 和 bookpagination 都是关键字域。如果关键字是已定义的，本宏包会尝试使用它们的值作为本地化关键字的值。在 bib 文件中要使用关键字名称的单数形式，复数形式会自动形成的。预定义的关键字有 page、column、line、verse、section 和 paragraph，其中 page 是默认值。在使用 pagination 和 bookpagination 时，字符串“none”有特殊意义，它将取消相应条目的前缀。如果对于某一条目的页码标记格式没有预定义的本地化关键字，你可以直接添加它们。参考 § 3.9 节中的 `\NewBibliographyString` 和 `\DefineBibliographyStrings` 命令。你需要定义两个本地化字符串来对应额外的页码标记格式：单数形式（本地化关键字对应于 pagination 域的值）和复数形式（本地化关键字必须是单数形式加上字母“s”）。具体例子可以参考 § 4.9.2 节的预定义关键字。

## 2.4 提示与警告

本节提供了一些关于本宏包数据层面上的额外提示，另外也说明了一些常见问题。

### 2.4.1 交叉引用

Biber 的一大特色就是带有灵活数据继承规则的交叉引用机制并且高度可定制化。因此不再需要从父条目复制域或者向子条目中添加空白域，而可以用很自然的方式指定条目：

```
@Book{book,
  author        = {Author},
  title         = {Booktitle},
  subtitle      = {Booksubtitle},
  publisher     = {Publisher},
```

```
  location      = {Location},
  date          = {1995},
}
@InBook{inbook,
  crossref      = {book},
  title         = {Title},
  pages         = {5--25},
}
```

父条目的 `title` 和 `subtitle` 会分别复制给子条目的 `booktitle` 和 `booksubtitle`。父条目的 `author` 会成为子条目的 `bookauthor`，并且由于子条目没有提供 `author` 域，它也会复制给子条目的 `author` 域。继承数据之后的子条目会大致如下：

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

默认的映射法则设置列表请参考附录 B。请注意，所有这一切都是可以定制的。关于如何配置 Biber 的交叉引用机制请参考 § 4.5.11 以及 § 2.2.3 节。

#### 2.4.1.1 xref 域

除了 `crossref` 域之外，Biblatex 也支持一种简化的交叉引用机制。而该机制则基于 `xref` 域。如果你想在两个相关条目间创建父/子关系，但就数据而言又希望保持它们的独立性，那么该域会很有用。`xref` 域与 `crossref` 的不同之处在于子条目不会从父条目继承任何数据。如果一个父条目被若干子条目引用，那么它将被 Biblatex 自动添加到参考文献中。相关子条目数量的阈值由 § 3.1.2.1 节的 `mincrossrefs` 宏包选项所控制。参考 § 2.2.3 一节。

### 2.4.2 排序和编码问题

Biber 处理 Ascii、Latin 1 等 8 比特编码以及 UTF-8 。它的特色在于真正的 Unicode 支持，并且能够在实时运行中以一种鲁棒的方式重新编码 `bib` 数据。对于排序，Biber 使用 Perl 实现 Unicode 排序算法（Unicode Collation Algorithm，UCA），该算法在 Unicode 技术标准 #10 中有介绍[20]。基于 Unicode 通用区域数据库（Common Locale Data Repository，CLDR）的排序裁剪也是支持的[21] 。

支持 Unicode 不仅仅意味着处理 UTF-8 输入。Unicode 是一个复杂的标准，不仅仅涵盖了它最著名的的部分——Unicode 字符编码和 UTF-8 等传输编码。它同样

[20] http://unicode.org/reports/tr10/

[21] http://cldr.unicode.org/

对字符串排序等方面做了标准化，这对不同语言中的排序是必要的。例如，使用 Unicode 排序算法（UCA），Biber 可以处理字符“ß”，而不需要任何人工干预。你需要针对本地化排序所做的就是指定本地化设置：

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

如果通过 babel 或者 polyglossia 等宏包将德语设置为主文档语言，设置方式为：

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

这会使得 Biblatex 将 babel/polyglossia 主文档语言传递过来并映射为合适的默认本地化语言。Biber 不会尝试从操作环境中获取本地化信息，因为这会使得文本处理取决于文档外，而这有悖于 TEX 可移植性的精神。由于 babel/polyglossia 实际上与文档的环境有关，因此这也是有意义的。请注意，这对于 Latin 9 等 8 比特编码也是有效的，也就是说，你可以利用基于 Unicode 的排序，即使你没有使用 UTF-8 输入。关于如何正确地指定输入和数据编码请参考 § 2.4.2.1。

**2.4.2.1 指定编码** 当在 bib 中使用非 Ascii 编码时，重要的一点是要意识到，什么是 Biblatex 能做的，而什么是需要人工调整的。本宏包照顾 LATEX 一方，也就是说，只要 bibencoding 宏包选项设置正确，本宏包确保从 bbl 文件导入的数据能被正确地解释执行。所有的这一切都会自动处理，除了在某些情况下设置 bibencoding 选项外，不需要其它步骤。以下是一些典型的使用场景以及文件导言区中的相关部分：

- tex 和 bib 文件中都使用 Ascii 记法，使用 pdfTEX 或传统的 TEX 编译：

  ```
  \usepackage{biblatex}
  ```

- tex 中使用 Latin 1 编码（ISO-8859-1），bib 文件中使用 Ascii 记法，用 pdfTEX 或传统的 TEX 编译：

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

- tex 和 bib 文件中都使用 Latin 9 编码（ISO-8859-15），用 pdfTEX 或传统的 TEX 编译：

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

  由于 bibencoding=auto 是默认设置，因此该选项可以省略。以下的设置具有相同效果：

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

- tex 文件中使用 UTF-8 编码，bib 文件中使用 Latin 1（ISO-8859-1）编码，用 pdfTEX 或传统的 TEX 编译：

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

在原生 UTF-8 模式下使用 XETEX 或 LuaTEX 编译的相同场景：

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

Biber 可以处理 Ascii 记法、Latin 1 等 8 比特编码以及UTF-8 。它也能在运行中实时重新编码 bib （取代受限宏层面的重新编码是 Biblatex 的特性）。如果你正确指定 bib 文件的编码，这一特性就会在需要时自动执行。除了以上讨论的场景外，Biber 还能够处理以下情况：

- 直接的 UTF-8 工作流，即，在 tex 和 bib 文件中都使用 UTF-8 编码并使用 pdfTEX 或传统的 TEX 编译：

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

由于 bibencoding=auto 是默认设置，因此该选项可以省略：

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

在原生 UTF-8 模式下使用 XETEX 或 LuaTEX 编译的相同场景：

```
\usepackage{biblatex}
```

- 甚至可以在 tex 文件使用 8 比特编码，而在 bib 文件中使用 UTF-8 编码，只要 bib 文件中的所有字符都能被所选择的 8 比特编码覆盖：

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

当对 UTF-8 编码使用传统的 TEX 或 pdfTEX 时，可能需要一些变通处理，因为 inputenc 的 utf8 模块并不能覆盖所有的 Unicode。粗略地讲，它只覆盖了西欧字符的 Unicode 范围。当载入带有 utf8 选项的 inputenc 宏包时，Biblatex 通常会指示 Biber 将 bib 数据重新编码为 UTF-8 。如果 bib 文件中的字符超出了 inputenc 支持的 Unicode 范围这可能会导致一些 inputenc 的错误，。

- 如果你受到这个问题的影响，尝试设置 safeinputenc 选项：

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

  如果该选项被启用，Biblatex 会忽略 inputenc 的 utf8 选项而使用 Ascii。Biber 随后会尝试将 bib 数据转化为 Ascii 记法。例如，它将 Ş 转化为 \k{S}。该选项类似于设置 texencoding=ascii 但是只影响这一特定场合（带有 UTF-8 的 inputenc/inputenx 宏包）。这一处理利用了这一事实：Unicode 和UTF-8 传输编码都与 Ascii 向后兼容。

如果 bib 文件中的数据主要是 Ascii 不过含有一些会导致问题的字符串（例如一些作者的名字），那么这一变通处理办法也是可行的。不过需要注意的是，它不会奇迹般地让传统的 TEX 或 pdfTEX 支持 Unicode。如果偶尔遇到一些奇特字符不被 inputenc 支持，当使用重音命令（例如用 \d{S} 取代 Ş）时，这些字符也可以被 TEX 处理。然而，如果你需要完全的 Unicode 支持，请转向 XETEX 或 LuaTEX 。

inputenc 不能处理某些 UTF-8 字符的典型错误是：

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

但也可以不那么明显，如：

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

# 3 用户使用手册

本部分介绍了 Biblatex 宏包的用户接口。该使用指南涵盖了要使用 Biblatex 自带的标准样式所需的所有信息。无论怎样都应该首先阅读该使用指南。如果你想编写你自己的引用和文献样式，请继续阅读随后的作者指南。

## 3.1 宏包选项

所有的宏包选项都以 ⟨*key*⟩=⟨*value*⟩ 记法给出。对于所有的布尔型键值，true 是可以省略的。例如，给出不带值的 sortcites 等价于 sortcites=true。

### 3.1.1 载入时选项

以下的选项必须在 Biblatex 载入时使用，即作为 `\usepackage` 的可选项。

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

指定数据库后端程序。支持以下后端：

`biber` Biber，Biblatex 的默认后端程序，支持 Ascii、8 比特编码、UTF-8、实时重新编码、本地化定制排序，以及许多其它特性。本地化定制排序，大小写敏感排序和大小写优先排序分别由选项 `sortlocale`、`sortcase` 和 `sortupper` 控制。

`bibtex` 遗留下来的 BibTeX。传统的 BibTeX 只支持 Ascii 编码，并且排序总是大小写不敏感的。

`bibtex8` `bibtex8` 是 BibTeX 的 8 比特实现，支持 Ascii 和 Latin 1 等 8 比特编码。

使用 BibTeX 作为后端详见 § ?? 节。

`style=`⟨*file*⟩ default: `numeric`

加载文献样式文件 ⟨*file*⟩`.bbx` 和引用样式文件 ⟨*file*⟩`.cbx`。关于标准样式的概览请参考 § 3.3 节。

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

加载文献样式文件 ⟨*file*⟩`.bbx`。关于标准文献样式的概览见 § 3.3.2 节。

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

加载引用样式文件 ⟨*file*⟩`.cbx`。关于标准引用样式的概览见 § 3.3.1 节。

`natbib=true, false` default: `false`

加载兼容性模块，以提供 `natbib` 宏包引用命令的同名替代命令。详见 § 3.8.9 节。

`mcite=true, false` default: `false`

加载引用模块，以提供 `mcite`/类 `mciteplus` 的引用命令。详见 § 3.8.10 节。

### 3.1.2 导言区选项

**3.1.2.1 一般选项** 下列选项既可以作为 `\usepackage` 的可选项，也可以在配置文件和导言区中使用。右边列出的是本宏包的默认值。请注意，文献和引用样式可以在载入时修改默认设置，详见 § 3.3 节。

`sorting=nty, nyt, nyvt, anyt, anyvt, ynt, ydnt, none, debug, ⟨name⟩` default: `nty`

参考文献的排序方式。除非另加说明，文献条目总是按升序排列。有以下预设可选值。

`nty` 按照姓名、标题、年份排序。

| | |
|---|---|
| nyt | 按照姓名、年份、标题排序。 |
| nyvt | 按照姓名、年份、卷数、标题排序。 |
| anyt | 按照标签的字母顺序、姓名、年份、标题排序。 |
| anyvt | 按照标签的字母顺序、姓名、年份、卷数、标题排序。 |
| ynt | 按照年份、姓名、标题排序。 |
| ydnt | 按照年份（降序）、姓名、标题排序。 |
| none | 不进行排序。所有的条目按照引用顺序处理。 |
| debug | 按照条目键值排列。该选项只用于程序调试。 |
| ⟨*name*⟩ | 使用随 `\DeclareSortingScheme`（§ 4.5.6）定义的 ⟨*name*⟩。 |

只有与打印相应标签的文献样式相结合，使用“字母顺序”排序格式才有意义。请注意，一些文献样式会将宏包选项从宏包的默认值（`nty`）初始化为另外一个值。详见 § 3.3.2 节。关于以上排序选项的深度解读以及排序过程中涉及的域，请参考 § 3.5 节。关于如何调整预定义格式或者定义新格式也可参考 § 4.5.6。

`sortcase`=`true`, `false` default: `true`

文献和缩略语列表的排序是否是大小写敏感的。

`sortupper`=`true`, `false` default: `true`

对应于 Biber 的 `--sortupper` 命令行选项。当该选项激活时，文献会按照“大写先于小写”的顺序排列。关闭该选项则使用“小写先于大写”的顺序。

`sortlocale`=`auto`, ⟨*locale*⟩

该选项设置全局的排序区域语言。如果排序格式没有使用 `\printbibliography` 命令的 ⟨*locale*⟩ 选项，那么就会继承该区域语言设置。如果该选项值设置为 `auto`，那么需要使用 `babel`/`polyglossia` 宏包并设置主文档语言标识符，否则将设置为 `en_US`。Biber 会将 `babel`/`polyglossia` 语言标识符映射为有意义的本地化标识符（参考 Biber 文档）。因此，你可以指定常规的本地化标识符，例如 `de_DE_phonebook`、`es_ES`；另外如果对 Biber 的语言映射满意的话，还可以指定支持的 `babel`/`polyglossia` 语言标识符。

`related`=`true`, `false` default: `true`

是否使用相关条目中的信息。参考 § 3.4 节。

`sortcites`=`true`, `false` default: `false`

当多个条目关键字传给引用命令时，是否对引用进行排序。如果该选项激活，就会根据当前的文献排序格式（见 § 3.7.10 节）对引用进行排序。该特征对所有的引用样式都有效。

`maxnames`=⟨*integer*⟩ default: 3

影响所有名称列表（`author`、`editor` 等）的阈值。如果某个列表超过了该阈值，即，它包括的姓名数量超过 ⟨*integer*⟩，那么就会根据 `minnames` 选项的设置进行自动截断。`maxnames` 是设置 `maxbibnames` 和 `maxcitenames` 的主选项。

`minnames`=⟨*integer*⟩ default: 1

影响所有名称列表（`author`、`editor` 等）的限制值。如果某个列表包含的姓名数量超过 ⟨*integer*⟩，那么就会根据 `minnames` 选项的设置自动截断。⟨*minnames*⟩ 的值必须小于或等于 ⟨*maxnames*⟩。`minnames` 是设置 `minbibnames` 和 `mincitenames` 的主选项。

`maxbibnames`=⟨*integer*⟩ default: ⟨*maxnames*⟩

类似于 `maxnames` 但只影响参考文献。

`minbibnames`=⟨*integer*⟩ default: ⟨*minnames*⟩

类似于 `minnames` 但只影响参考文献。

`maxcitenames`=⟨*integer*⟩ default: ⟨*maxnames*⟩

类似于 `maxnames` 但只影响正文中的引用。

`mincitenames`=⟨*integer*⟩ default: ⟨*minnames*⟩

类似于 `minnames` 但只影响正文中的引用。

`maxitems`=⟨*integer*⟩ default: 3

类似于 `maxnames` 但影响所有的文本列表（`publisher`、`location`等）。

`minitems`=⟨*integer*⟩ default: 1

类似于 `minnames` 但影响所有的文本列表（`publisher`、`location`等）。

`autocite`=`plain`, `inline`, `footnote`, `superscript`, ...

该选项控制 § 3.8.4 节中讨论的 `\autocite` 命令的行为。`plain` 选项使得 `\autocite` 相当于 `\cite`；`inline` 选项使得它相当于 `\parencite`；`footnote` 选项使得它相当于 `\footcite`；`superscript` 选项使得它相当于 `\supercite`。选项 `plain`、`inline` 和 `footnote` 总是可行的，而 `superscript` 只能用于本宏包所带的数值引用样式中。引用样式也可以定义其它选项值。该选项的默认设置取决于所选的引用样式，参考 § 3.3.1 节。

`autopunct`=`true`, `false` default: `true`

该选项控制引用命令是否先扫描标点。详见 § 3.8 节和 § 4.7.5 节中的命令 `\DeclareAutoPunctuation`。

language=autobib, autocite, auto, ⟨*language*⟩ default: autobib

该选项控制多语种支持功能。当其设置为 autobib、autocite 或 auto 时，Biblatex 会尝试从 babel/polyglossia 宏包中获取文档的主语言（如果babel/polyglossia 宏包不可用则设置为后备的英语）。也可以手动选择文档的语言，此时选择的语言会覆盖条目的 langid 域，并且仍需要使用 autolang 选项选择语言切换环境以确定如何处理手动选择语言的切换方式。关于所支持的语言和相应的标识符请参考表 3。autobib 使用条目中的 langid 域和 autolang 选项确定的语言环境为条目切换语言。autocite 使用条目中的 langid 域和 autolang 选项确定的语言环境为引用切换语言。而 auto 是同时设置 autobib 和 autocite 的缩略语。

clearlang=true, false default: true

如果激活该选项，Biblatex 会自动清除那些与文档的 babel/polyglossia 语言（或者由 language 选项显式确定的语言）相匹配的所有条目的 language 域，以便略去冗余的语言设定。该特性所需的语言映射由 § 4.9.1 节的 \DeclareRedundantLanguages 命令提供。

autolang=none, hyphen, other, other*, langname default: none

如果载入了 babel/polyglossia 宏包并且文献条目包含 langid 域（见 § 2.2.3 节），那么该选项可以控制使用哪种 babel 语言环境[22]。请注意，当载入 babel/polyglossia 宏包时 Biblatex 会自动适应主文档的语言。在多语言文档中，只要涉及到引用和文献的默认语言，本宏包也会连续地调整以适应当前语言。切换语言的效果取决于该选项选择的语言环境，可用的选择有：

- none　关闭该特性，也就是不使用任何语言环境。
- hyphen　将条目装入 hyphenrules 环境中。如果可用的话，会为条目的 hyphenation 域指定的语言导入断词模式。
- other　将条目装入 otherlanguage 环境中。这将导入特定语言的断词模式，激活 babel/polyglossia 和 Biblatex 为相应语言提供的所有额外定义，并翻译“editor”和“volume”等键项。这些额外定义包括日期格式、序数和其它类似项目的本地化。
- other*　将条目装入 otherlanguage* 环境中。请注意，此时 Biblatex 将 otherlanguage* 环境视为 otherlanguage 环境但其它宏包不会。
- langname　只用于 polyglossia 宏包。将条目装入 ⟨*languagename*⟩ 环境中。该选项值对 polyglossia 用户的好处是注意了 langidopts 域，这样可以为每一条目增加语言选项（类似于选择语言变种）。当使用 babel 时，该选项值与 other 选项值相同。

block=none, space, par, nbpar, ragged default: none

该选项控制块（block）之间的额外空白，即文献条目的更大的分段。可用的选项值有：

[22] polyglossia 宏包也可以理解 babel 语言环境，因此本选项可以同时控制 babel/polyglossia 语言环境。

none　不添加任何东西。

space　在块之间添加额外的水平间距，类似于标准 LaTeX 文档类的默认行为。

par　每一块都另起一段，类似于标准 LaTeX 文档类的 openbib 选项。

nbpar　类似于 par 选项但是不允许在块的边界和条目内部分页。

ragged　插入一个小的负惩罚项以鼓励在块的边界处断行并设置左对齐。

也可以直接重新定义 \newblockpunct 命令实现不同的效果，见 § 3.10.1 节。更多信息见 § 4.7.1 节。

notetype=foot+end, footonly, endonly　default: foot+end

该选项控制 § 4.10.4 节中 \mkbibfootnote、\mkbibendnote 和类似封装的行为。可用的选项值有：

foot+end　同时支持脚注和尾注，即，\mkbibfootnote 会生成脚注而 \mkbibendnote 会生成尾注。

footonly　强制脚注，即，\mkbibendnote 也生成脚注。

endonly　强制尾注，即，\mkbibfootnote 也生成尾注。

hyperref=true, false, auto　default: auto

是否将引用和向后引用转化为可点击的超链接。这一特性需要 hyperref 宏包。它同样需要引用样式的支持。本宏包所带的所有标准样式都支持超链接。hyperref=auto 会自动检测 hyperref 宏包是否载入。

backref=true, false　default: false

是否在文献中打印出反向引用。反向引用是一组用于标明引用相应文献的页码数。如果在文档中有 refsection 环境，反向引用在每个引用章节中是局部的。严格地讲，该选项只控制 Biblatex 是否收集所需引用的数据。该特性也需要文献样式的支持。本宏包所带的所有标准样式都支持该特性。

backrefstyle=none, three, two, two+, three+, all+　default: three

该选项控制反向引用中的连续页码的格式。可用样式有：

none　不启用该特性，即，不压缩页码列表。

three　将任意连续三页或更多页缩写为页码范围，例如，“1, 2, 11, 12, 13, 21, 22, 23, 24”会压缩成“1, 2, 11–13, 21–24”。

two　将任意连续两页或更多页缩写成页码范围，例如上面的例子会变成“1–2, 11–13, 21–24”。

two+　概念类似于 two，但是连续两页的打印格式为开始页和本地化字符串 sequens，例如上面的例子会变成“1 sq., 11–13, 21–24”。

three+　概念类似于 two+，但是连续三页的打印格式为开始页可本地化字符串 sequentes，例如上面的例子会变成“1 sq., 11 sqq., 21–24”。

all+ 概念类似于 three+，但是任意连续页码将打印成末端不封闭的形式，例如上面的例子会变成“1 sq., 11 sqq., 21 sqq.”。

所有这些样式都同时支持阿拉伯和罗马数字。为了避免可能的歧义，不同数字集在生成数字范围时不会混在一起，例如，“iii, iv, v, 6, 7, 8”将缩写为“iii–v, 6–8”。

backrefsetstyle=setonly, memonly, setormem, setandmem, memandset, setplusmem default: setonly

该选项控制怎样处理指向 @set 条目及其成员的反向引用。可用选项有：

setonly 所有的反向引用都添加到 @set 条目中。而其成员的 pageref 列表为空。

memonly 条目集成员的引用添加到各自成员中。@set 条目的引用添加到所有成员中。@set 条目的 pageref 列表为空。

setormem @set 条目的引用添加到 @set 条目中。其成员的引用添加到各自成员中。

setandmem @set 条目的引用添加到 @set 条目中。其成员的引用添加到各自成员和 @set 条目中。

memandset @set 条目的引用添加到 @set 条目和所有成员中。其成员的引用添加到各自成员中。

setplusmem @set 条目的引用添加到 @set 条目和所有成员中。其成员的引用添加到各自成员和 @set 条目中。

indexing=true, false, cite, bib default: false

该选项控制文献和引用中的索引。更准确地说，它影响 § 4.6.2 节的 \ifciteindex 和 \ifbibindex 命令。该选项可以在全局、基于每一类型或每一条目设置。可用的选择有：

true 全局激活索引。

false 全局不激活索引。

cite 只在引用中激活索引。

bib 只在文献中激活索引。

该特性需要引用样式的支持。本宏包所带的所有标准样式都支持引用和文献条目中的索引。请注意，为了得到索引仍然需要用 \makeindex 命令全局激活索引模式。

loadfiles=true, false default: false

该选项控制是否载入通过 \printfile 命令所需的外部文件。另参考 § 3.12.8 和 § 4.4.1 节中的 \printfile 命令。请注意，出于性能原因该特性默认不激活。

refsection=none, part, chapter, section, subsection default: none

该选项自动在文档分段处（例如一章或一节）开始一个新的参考文献分节。这等价于 \newrefsection 命令，参考 § 3.7.4 节。可用的文档分段如下：

none　不启用该特性。

part　在每个 \part 命令处开始一个参考文献分节。

chapter　在每个 \chapter 命令处开始一个参考文献分节。

section　在每个 \section 命令处开始一个参考文献分节。

subsection　在每个 \subsection 命令处开始一个参考文献分节。

这些命令对应带星号的版本不会开始新的参考文献分节。

refsegment=none, part, chapter, section, subsection　default: none

类似于 refsection 选项，但是开始一个新的参考文献片段。这等价于 \newrefsegment 命令，详见 § 3.7.5 节。当两个选项都使用时，请注意该选项只能应用到比 refsection 选项应用的更低水平的文档分段，同时，嵌套的参考文献片段相对于所附的参考文献分节来讲是局部的。

citereset=none, part, chapter, section, subsection　default: none

该选项在文档分段处（例如一章或一节）自动执行 § 3.8.8 节介绍的 \citereset 命令。可用的文档分段有：

none　不启用该特性。

part　在每个 \part 命令后执行重置。

chapter　在每个 \chapter 命令后执行重置。

section　在每个 \section 命令后执行重置。

subsection　在每个 \subsection 命令后执行重置。

这些命令对应带星号的版本不会引起重置。

abbreviate=true, false　default: true

是否在引用和文献中使用较长或者缩写的字符串。该选项会影响本地化模块。如果启用该选项，“editor”等键值项会缩写，反之则会全部写出。

date=year, short, long, terse, comp, ymd, edtf　default: comp

该选项控制日期规范的基本格式。有以下选择：

year　只使用年份，例如：

2010

2010–2012

short　使用短格式和详细的日期范围，例如：

01/01/2010

21/01/2010–30/01/2010

01/21/2010–01/30/2010

`long` 使用长格式和详细的日期范围，例如：

1st January 2010

21st January 2010–30th January 2010

January 21, 2010–January 30, 2010

`terse` 使用短格式和紧凑的日期范围，例如：

21–30/01/2010

01/21–01/30/2010

`comp` 使用长格式和紧凑的日期范围，例如：

21st–30th January 2010

January 21–30, 2010

`edtf` 使用严格的扩展日期/时间格式[23] (`yyyy-mm-dd`)，例如：

2010-01-01

2010-01-21/2010-01-30

`ymd` 不同于严格的 EDTF 格式，使用可以被其它选项修改的年-月-日格式，例如：

2010-1-1

2010-1-21/2010-1-30

注意，`edtf` 格式会强制开启 `dateera=astronomical`, `datezeros=true`, `timezeros=true`, `seconds=true`, ⟨*datetype*⟩`time=24h` 以及 `julian=false` 等键值。

从以上例子可以看出，实际的日期格式是与语言相关的。请注意，在所有长格式中月份名称与 `abbreviate` 宏包选项相对应。所有短格式中月和日的首位零可以另外由 `datezeros` 宏包选项控制。所有短格式中时分秒的首位零可以另外由 `timezeros` 宏包选项控制。如果要输出时刻，那么秒和时区的打印分别由 `seconds` 和 `timezones` 选项控制。

`julian` 和 `gregorianstart` 选项可以用于控制何时输出儒略历日期。

`labeldate`=`year`, `short`, `long`, `terse`, `comp`, `ymd`, `edtf` default: `year`

类似于 `date` 选项但是控制由 `\DeclareLabeldate` 选择的日期域的格式。

⟨*datetype*⟩`date`=`year`, `short`, `long`, `terse`, `comp`, `ymd`, `edtf` default: `comp`

类似于 `date` 选项但是控制数据模型中 ⟨*datetype*⟩`date` 域的格式。

`alldates`=`year`, `short`, `long`, `terse`, `comp`, `edtf`

将数据模型中所有日期的选项设置为相同值。默认数据模型中的日期域为 `date`、`origdate`、`eventdate` 和 `urldate`。

[23] https://www.loc.gov/standards/datetime/pre-submission.html

julian=true, false default: false

该选项控制是否将 gregorianstart 选项指定日期之前的日期自动转换为儒略历。改变的日期在 \ifdatejulian 和 \if⟨*datetype*⟩datejulian 测试下会返回“true”（见 § 4.6.2 节）。请谨记，只包含年份的日期不会转换为儒略历，例如“1565”，这是因为没有月日的日期在儒略历表示下会引起混乱[24] 例如，在“1565”的例子中，在公历（格里高利历）“1565 年 1 月 10 日”之后的日期是儒略历“1565”年，而之前的日期是儒略历“1564”年。

gregorianstart=⟨*YYYY-MM-DD*⟩

该选项控制在哪一日期之前的日期可以转换为儒略历。选项值有严格的字符串格式：4 位年份、2 位月份和日期，并且由短划线分隔（或者具有“Dash”属性的任何有效 Unicode 字符）。默认值是“1582-10-15”，即标准公历（格里高利历）的颁布日期。只有 julian 设置为“true”时本选项才起作用。

datezeros=true, false default: true

该选项控制 short 和 terse 日期成分当没有覆盖特定格式时打印是否首位补零。

timezeros=true, false default: true

该选项控制没有覆盖特定格式时，时刻成分打印是否首位补零。

timezones=true, false default: false

该选项控制打印时刻时是否输出时区。

seconds=true, false default: false

该选项控制打印时刻时是否输出秒。

dateabbrev=true, false default: true

该选项控制 long 和 comp 日期格式打印时带有完整还是缩写的月份名。该选项类似于一般的 abbreviate 选项但是只针对日期格式。

datecirca=true, false default: false

该选项控制关于日期是否输出“circa”信息。如果设置为 true，那么日期将由 \datecircaprint 宏的展开来引导，见 § 3.10.1 节。

dateuncertain=true, false default: false

该选项控制是否输出日期的不确定信息。如果设置为 true，那么日期将由 \dateuncertainprint 宏的展开来引导，并由 \enddateuncertainprint 宏结束，见 § 3.10.1 节。

[24] 缺失时刻的日期也有可能出现这一问题，不过对于文献作品问题不大。

`dateera`=`astronomical`, `secular`, `christian` default: `astronomical`

该选项控制如何打印日期纪元信息。选项值“astronomical”使用 `\dateeraprintpre` 命令在起止日期之前打印纪元信息。而选项值“secular”和“christian”使用 `\dateeraprint` 命令在起止日期之后打印纪元信息。缺省情况下，使用“astronomical”效果是在公元前日期之前使用负号，而使用“‘secular’”或“‘christian’”的效果是在公元前日期之后加上“BCE”或“BC”等相关的本地化字符串。见 § 3.10.1 节的有关评论以及 § 4.9.2.21 节的本地化字符串。

`dateeraauto`=⟨*integer*⟩ default: `0`

该选项设置天文学年份，使得在之前的年份自动添加纪元本地化字符串。只有当 `dateera` 设置为“secular”或者“christian”时本选项才起作用。

`time`=`12h`, `24h`, `24hcomp` default: `24h`

该选项控制时刻规范的基本格式。可用的选择有：

`24h` 24 小时格式，例如：

14:03:23

14:3:23

14:03:23+05:00

14:03:23Z

14:21:23–14:23:45

14:23:23–14:23:45

`24hcomp` 带有缩写范围的 24 小时格式，例如：

14:21–23 （小时相同）

14:23:23–45 （小时和分钟相同）

`12h` 带有本地化上下午标识符的 12 小时格式，例如：

2:34 PM

2:34 PM–3:50 PM

从以上例子可以看出，实际的时刻格式是与语言相关的。请注意，如果与指定的区域不同，那么上下午（AM/PM）字符串对应与 `abbreviate` 宏包选项。24 小时格式首位补零的话可以单独用 `timezeros` 宏包选项控制。此外与语言相关并可以单独定制的还有时刻成分之间的分隔符`\bibtimesep`、`\bibtzminsep`，以及时刻与时区之间的分隔符 `\bibtimezonesep`，见 § 3.10.2 节。还有一些全局的宏包选项可以控制是否打印秒和时区（分别是 `seconds` 和 `timezones`，见 § 3.1.2.1 节）。如果有时区的话，要么是字符 ‘Z’，要么是表示正负偏移量的数值。默认样式不打印时刻信息。定制样式可以使用 `\print`⟨*datetype*⟩`time` 命令打印时刻，见 § 4.4.1 节。

`labeltime`=`12h`, `24h`, `24hcomp` default: `24h`

类似于 `time` 选项但是控制由 `\DeclareLabeldate` 选择的域中得到的时刻部分的格式。

⟨*datetype*⟩`time`=12h, 24h, 24hcomp default: 24h

类似于 `time` 选项但是控制数据模型中 ⟨*datetype*⟩`date` 域中得到的时刻部分格式。

`alltimes`=12h, 24h, 24hcomp

为数据模型中所有的时刻设置相同的 `labeltime` 和 ⟨*datetype*⟩`time` 值。默认数据模型中支持时刻部分的日期域有 `date`、`origdate`、`eventdate` 和 `urldate`。

`dateusetime`=true, false default: false

确定在日期域的日期成分后是否打印时刻成分。日期和时刻成分的分隔符是 `\bibdatetimesep`，见 § 3.10.2 节。如果使用紧凑的日期格式（见 § 3.1.2.1 节），那么该选项则不起作用，否则会引起混乱。

`labeldateusetime`=true, false default: false

类似于 `dateusetime` 选项，但是控制是否打印 `\DeclareLabeldate` 选择的域中的时刻成分。

⟨*datetype*⟩`dateusetime`=true, false default: false

类似于 `dateusetime` 选项，但是控制是否打印数据模型中 ⟨*datetype*⟩`date` 域的时刻成分。

`alldatesusetime`=true, false default: false

为数据模型中所有的 ⟨*datetype*⟩`date` 域设置 `labeldateusetime` 和 ⟨*datetype*⟩`dateusetime` 选项值。

`defernumbers`=true, false default: false

与标准 LaTeX 不同，本宏包生成的数字标签一般在文档正文的一开始就分配给引用列表全体。如果该选项被激活，数字标签（也就是 § 4.2.4 中讨论的 `labelnumber` 域）在参考文献第一次打印出某条目时就被分配。进一步解释见 § 3.13.5 节。该选项在后端将数据导出到 `bbl` 文件后仍然需要两次 LaTeX 运行（在由分页变化等要求的编译之外）。需要注意的一件重要的事是，如果你在文档中改变了该选项值（或者那些依赖于本选项的选项值，例如与`\printbibliography` 宏相关的选项，§ 3.7.2 节），那么很可能需要删除当前的 `aux` 文件然后重新运行 LaTeX 来获得正确的数字编号，见 § 4.1 节。

`punctfont`=true, false default: false

该选项激活一个可选机制，用来处理不同字体打印的域（例如斜体的标题）之后的单位标点。详见 § 4.7.1 节中的 `\setpunctfont`。

`arxiv`=abs, ps, pdf, format default: abs

arXiv 链接的路径选择。如果启用超链接支持，该选项会控制 arXiv 的 `eprint` 域会指向该文件的哪个版本。以下是可用的选择：

`abs` 链接到摘要页面。

`ps` 链接到 PostScript 版本。

`pdf` 链接到 PDF 版本。

`format` 链接到格式选择页面。

关于 arXiv 和电子出版信息的支持详见 § 3.12.7 节。

`texencoding=auto`, ⟨*encoding*⟩ default: `auto`

指定 `tex` 文件的编码。该选项影响从后端转向 Biblatex 的数据。该选项对应于 Biber 的 `--output-encoding` 选项。可用的选择有：

`auto` 尝试自动识别输入的编码。如果有 `inputenc/inputenx/luainputenc` 等宏包，Biblatex 会从宏包中获取主要编码。否则，当探测到 XeTeX 或 LuaTeX 引擎时设定为 UTF-8 编码，其余情况设为 Ascii。

⟨*encoding*⟩ 显式指定编码为 ⟨*encoding*⟩。少数情况下自动检测失败，或你出于某种原因想强制为某个编码，那么此时可以使用该选项。

请注意如果 `bibencoding = auto`，那么设置 `texencoding =` ⟨*encoding*⟩ 也会影响 `bibencoding` 选项。

`bibencoding=auto`, ⟨*encoding*⟩ default: `auto`

指定 `bib` 文件的编码。该选项对应于 Biber 程序的 `--output-encoding` 选项。可用选择有：

`auto` 当工作流是透明时，即，当 `bib` 文件的编码与 `tex` 文件的编码相同时使用该选项。

⟨*encoding*⟩ 如果 `bib` 文件的编码与 `tex` 不同，你需要显式指定编码。

默认情况下，Biblatex 假定 `tex` 和 `bib` 文件使用相同的编码（`bibencoding=auto`）。

`safeinputenc=true, false` default: `false`

如果启动该选项，Biblatex 会在载入 `inputenc/inputenx` 宏包并且输入代码是 UTF-8 时自动加入 `texencoding=ascii`，也就是说，它会忽略任何基于宏的 UTF-8 支持而只是用 Ascii。然后 Biber 会尝试将 `bib` 文件中的任意非 Ascii 数据转化为 Ascii。例如，它会将 Ș 转化为 `\d{S}`。关于为什么需要启用该选项的原因，请参考 § 2.4.2.1 节。

`bibwarn=true, false` default: `true`

默认情况下，Biblatex 会报告后端产生的关于 `bib` 文件数据的警告，就像 LaTeX 警告一样。使用该选项会取消此警告。

`mincrossrefs`=⟨*integer*⟩ default: 2

当需要后端运行时设置交叉引用的最小数目为 ⟨*integer*⟩[25]。该选项也影响 `xref` 域的处理。详见 § 2.2.3 节以及 § 2.4.1 节对该域的描述。

`minxrefs`=⟨*integer*⟩ default: 2

类似于 `mincrossrefs` 但针对于 `xref` 域。

**3.1.2.2 特定样式选项** 下列选项由标准样式（而不是宏包内核）提供。技术上讲，它们和 § 3.1.2.1 中的选项一样也是导言区选项。

`isbn`=`true`, `false` default: `true`

该选项控制是否打印 `isbn`/`issn`/`isrn` 等域。

`url`=`true`, `false` default: `true`

该选项控制是否打印 `url` 域并获取日期。该选项只影响 `url` 信息是可选的那些条目类型。而 `@online` 条目的 `url` 域总是打印出来的。

`doi`=`true`, `false` default: `true`

该选项控制是否打印 `doi` 域。

`eprint`=`true`, `false` default: `true`

该选项控制是否打印 `eprint` 信息。

**3.1.2.3 内部选项** 下列导言区选项的默认设置由文献和引用样式控制。除了 `pagetracker` 和 ⟨*name*⟩`inits` 是你可能想调整的之外，一般没有必要显式地设置。

`pagetracker`=`true`, `false`, `page`, `spread` default: `false`

该选项控制 § 4.6.2 节的 `\ifsamepage` 和 `\iffirstonpage` 测试所需的页码跟踪器。可用选择有：

| | |
|---|---|
| `true` | 在自动模式下激活。该选项在 LaTeX 处于双面模式时类似于 `spread`，否则类似于 `page`。 |
| `false` | 不激活跟踪器。 |
| `page` | 在页面模式下激活。此时跟踪器基于每一页。 |
| `spread` | 在跨页模式下激活。此时跟踪器基于每一页面（双页）。 |

注意所有的浮动体都禁用该跟踪器，见 § 4.11.5 节。

`citecounter`=`true`, `false`, `context` default: `false`

该选项控制 § 4.6.2 节的 `citecounter` 所需的引用计数器。可用的选择有：

[25] 如果一个条目被 `bib` 文件中的其它条目引用的数目达到这个阈值，它就会载入到参考文献中，即使没有显式地被引用。这是 BibTeX 格式的标准特性，并不是 Biblatex 特有的。更多信息见 § 2.2.3 节中关于 `crossref` 域的描述。

`true` 在全局模式下启用引用计数器。

`false` 禁用引用计数器。

`context` 在内容相关模式下启用引用计数器。此时，脚注和正文中的引用将独立计数。

`citetracker=true, false, context, strict, constrict` default: `false`

该选项控制 § 4.6.2 节的 `\ifciteseen` 和 `\ifentryseen` 测试所需的引用跟踪器。可用选择有：

`true` 在全局模式下启用跟踪器。

`false` 禁用跟踪器。

`context` 在内容相关模式下启用跟踪器。此时，脚注和正文中的引用将独立跟踪。

`strict` 在严格模式下启用跟踪器。此时，跟踪器只考虑独立的引用，即，引用命令只传递单个的条目键。

`constrict` 该模式是 `context` 和 `strict` 的结合。

注意所有的浮动体都禁用该跟踪器，见 § 4.11.5 节。

`ibidtracker=true, false, context, strict, constrict` default: `false`

该选项控制 § 4.6.2 节的 `\ifciteibid` 测试所需的“如前所述”（ibidem）跟踪器。可用的选择有：

`true` 在全局模式下启用跟踪器。

`false` 禁用跟踪器。

`context` 在内容相关模式下启用跟踪器。此时，脚注和正文中的引用将独立跟踪。

`strict` 在严格模式下启用跟踪器。此时将不考虑那些模糊不清的参考文献。如果当前引用（包含“ibidem”）或者之前引用（“ibidem”的指向）包含多个文献时，就认为是模糊不清的。[26]

`constrict` 该模式是 `context` 和 `strict` 的结合。它也保持对脚注数量的跟踪，不过检测脚注中含义不清的文献时比 `strict` 更加严格。除了 `strict` 选项的条件外，只有当前引用和之前引用都在同一个或者连续的脚注中时，脚注中的文献才认为是含义清晰的。

注意所有的浮动体都禁用该跟踪器，见 § 4.11.5 节。

`opcittracker=true, false, context, strict, constrict` default: `false`

该选项控制 § 4.6.2 节的 `\ifopcit` 测试所需的“opcit”跟踪器。该特性类似于“ibidem”跟踪器，不同之处在于它跟踪的是基于某一作者或编辑的引用，即，如果

[26] 例如，假设一开始的引用是“Jones, *Title*; Williams, *Title*”，而接下来的是“ibidem”。从技术角度来看，“ibidem”指向“Williams”是相对清晰的，因为这是之前引用命令处理的最后文献。然而对于用户而言，“ibidem”也可以同时指向这两个标题，因此含义不清。严格模式就避免这种含义不清的文献。

引用项目与之前项目的作者或编辑相同，那么 `\ifopcit` 的结果为 `true`。可用选择有：

| | |
|---|---|
| `true` | 在全局模式下启用该跟踪器。 |
| `false` | 禁用该跟踪器。 |
| `context` | 在内容相关模式下启用该跟踪器。此时，脚注和正文中的引用会独立跟踪。 |
| `strict` | 在严格模式下启用该跟踪器。此时将不跟踪那些含义不清的引用。详见 `ibidtracker=strict` 选项。 |
| `constrict` | 该模式是 `context` 和 `strict` 的结合。详见 `ibidtracker=constrict` 选项的解释。 |

注意所有的浮动体都禁用该跟踪器，见 § 4.11.5 节。

`loccittracker`=`true`, `false`, `context`, `strict`, `constrict` default: `false`

该选项控制 § 4.6.2 节的 `\ifloccit` 测试所需的“loccit”跟踪器。该特性类似于“opcit”跟踪器，不同之处在于它也会检查 ⟨*postnote*⟩ 选项是否匹配，即，如果引用项目与之前引用指向的页数相同，那么 `\ifloccit` 的结果为 `true`。可用选择有：

| | |
|---|---|
| `true` | 在全局模式下启用该跟踪器。 |
| `false` | 禁用该跟踪器。 |
| `context` | 在内容相关模式下启用该跟踪器。此时，脚注和正文中的引用会独立跟踪。 |
| `strict` | 在严格模式下启用该跟踪器。此时将不跟踪那些含义不清的引用。详见 `ibidtracker=strict` 选项。此外，该模式也会检查 ⟨*postnote*⟩ 选项是否是数值型的（基于 § 4.6.2 节的 `\ifnumerals` 命令）。 |
| `constrict` | 该模式是 `context` 和 `strict` 的结合。详见 `ibidtracker=constrict` 选项的解释。此外，该模式也会检查 ⟨*postnote*⟩ 选项是否是数值型的（基于 § 4.6.2 节的 `\ifnumerals` 命令）。 |

注意所有的浮动体都禁用该跟踪器，见 § 4.11.5 节。

`idemtracker`=`true`, `false`, `context`, `strict`, `constrict` default: `false`

该选项控制 § 4.6.2 节的 `\ifciteidem` 测试所需的“idem”跟踪器。可用选择有：

| | |
|---|---|
| `true` | 在全局模式下启用该跟踪器。 |
| `false` | 禁用该跟踪器。 |
| `context` | 在内容相关模式下启用该跟踪器。此时，脚注和正文中的引用会独立跟踪。 |
| `strict` | 该选项是 `true` 的别名。提供该选项只是为了和其它跟踪器保持一致。由于“idem”不会像“ibidem”或“op. cit.”那样引起歧义，因此不必使用 `strict` 跟踪模式。 |

constrict　该模式类似于 context，只有一个额外条件：对于脚注中的引用，只有当前引用和之前引用位于同一个或相连的脚注中才会认为是含义明确的。

注意所有的浮动体都禁用该跟踪器，见 § 4.11.5 节。

parentracker=true, false　default: true

该选项控制括号跟踪器，用于对嵌套的圆括号和方括号的跟踪。所得信息用于 § 3.8.5 节的 \parentext 和 \brackettext 命令、§ 4.10.4 节的 \mkbibparens 和 \mkbibbrackets 命令，以及同样来自于 § 4.10.4 节的 \bibopenparen, \bibcloseparen, \bibopenbracket, \bibclosebracket 等命令。

maxparens=⟨*integer*⟩　default: 3

圆括号和方括号嵌套的最大层级。如果嵌套深度大于该值则会报错。

⟨*namepart*⟩inits=true, false　default: false

启用该选项时所有的 ⟨*namepart*⟩ 姓名部分都会用首字母表示。该选项会影响 § 4.6.2 节的 \if⟨*namepart*⟩inits 测试。在数据模型中有效的姓名部分由 \DeclareDatamodelConstant 命令定义，见 § 4.2.3 节。

terseinits=true, false　default: false

该选项控制 Biblatex 生成的首字母格式。当启用时，首字母缩写将采用没有点号和空格的紧凑形式。例如 Donald Ervin Knuth 的首字母缩写默认是“D. E.”，而在此选项启用时会是“DE”。该选项会影响 § 4.6.2 中的 \ifterseinits 测试。该选项会重新定义一些控制首字母格式的宏。详见 § 3.13.4 节。

labelalpha=true, false　default: false

是否提供特殊的域 labelalpha 和 extraalpha，详见 § 4.2.4 节。该选项可以基于每一条目设置。另见 maxalphanames 和 minalphanames 选项。表 8 总结了各种 extra* 歧义消除的计数器以及它们所跟踪的选项。

maxalphanames=⟨*integer*⟩　default: 3

类似于 maxnames 但用于定制 labelalpha 域的格式。

minalphanames=⟨*integer*⟩　default: 1

类似于 minnames 但用于定制 labelalpha 域的格式。

labelnumber=true, false　default: false

是否提供特殊域 labelnumber，详见 § 4.2.4 节。该选项可基于每一类型而设置。

labeltitle=true, false　default: false

是否提供特殊域extratitle，详见 § 4.2.4 节。请注意，特殊域 labeltitle 总是提供的，而该选项控制是否利用 labeltitle 生成 extratitle 信息。该选项也可基于每一类型而设置。表 8 总结了各种 extra* 消除歧义的计数器以及所跟踪的选项。

| 选项 | 测试 | 跟踪的域 |
|---|---|---|
| singletitle | \ifsingletitle | labelname |
| uniquetitle | \ifuniquetitle | labeltitle |
| uniquebaretitle | \ifuniquebaretitle | labeltitle （当 labelname 为空时） |
| uniquework | \ifuniquework | labelname+labeltitle |

**Table 7: 惟一性选项**

labeltitleyear=true, false default: false

是否提供特殊域 extratitleyear，详见 § 4.2.4 节。请注意，特殊域 labeltitle 总是提供的，而该选项控制是否利用 labeltitle 生成 extratitleyear 信息。该选项也可基于每一类型而设置。表 8 总结了各种 extra* 消除歧义的计数器以及所跟踪的选项。

labeldateparts=true, false default: false

是否提供特殊域 labelyear, labelmonth, labelday, labelendyear, labelendmonth, labelendday, labelhour, labelendhour, labelminute, labelendminute, labelsecond, labelendsecond, labelseason, labelendseason, labeltimezone, labelendtimeone 以及 extrayear，详见 § 4.2.4 节。该选项也可基于每一类型而设置。表 8 总结了各种 extra* 消除歧义的计数器以及所跟踪的选项。

singletitle=true, false default: false

是否提供 \ifsingletitle 测试所需的数据，详见 § 4.6.2 节。关于该测试中数据的控制因素详见 表 7。该选项也可基于每一类型而设置。

uniquetitle=true, false default: false

是否提供 \ifuniquetitle 测试所需的数据，详见 § 4.6.2 节。关于该测试中数据的控制因素详见 表 7。该选项也可基于每一类型而设置。

uniquebaretitle=true, false default: false

是否提供 \ifuniquebaretitle 测试所需的数据，详见 § 4.6.2 节。关于该测试中数据的控制因素详见 表 7。该选项也可基于每一类型而设置。

uniquework=true, false default: false

是否提供 \ifuniquework 测试所需的数据，详见 § 4.6.2 节。关于该测试中数据的控制因素详见 表 7。该选项也可基于每一类型而设置。

uniqueprimaryauthor=true, false default: false

是否提供 \ifuniqueprimaryauthor 测试所需的数据，详见 § 4.6.2 节。

uniquename=true, false, init, full, allinit, allfull, mininit, minfull default: false

是否更新 uniquename 计数器，详见 § 4.6.2 节。该特性会消除 labelname 列表中各个姓名的歧义。该选项也可基于每一类型而设置。可用的选择有：

| | |
|---|---|
| true | full 的别称。 |
| false | 禁用该特性。 |
| init | 只使用首字母消除歧义。 |
| full | 根据要求使用首字母或全名消除歧义。 |
| allinit | 类似于 init，但是会对 labelname 列表中所有姓名消除歧义，即便超出了 maxnames/minnames/uniquelist 选项。 |
| allfull | 类似于 full，但是会对 labelname 列表中所有姓名消除歧义，即便超出了 maxnames/minnames/uniquelist 选项。 |
| mininit | init 的变种，只对列表中有同一姓（last name）的姓名消除歧义。 |
| minfull | full 的变种，只对列表中有同一姓（last name）的姓名消除歧义。 |

请注意，uniquename 选项也会影响 uniquelist 选项、\ifsingletitle 测试，以及 extrayear 域。更多细节和实例见 § 4.11.4 节。

uniquelist=true, false, minyear　　default: false

是否更新 uniquelist 计数器，详见 § 4.6.2 节。如果 labelname 列表在 maxnames/minnames 截断后含义不清，那么该特性会消除 labelname 列表中的歧义。本质上，该选项会覆盖基于每一域的 maxnames/minnames 设置。该选项也可基于每一类型而设置。可用的选择有：

| | |
|---|---|
| true | 消除 labelname 列表的歧义。 |
| false | 禁用该特性。 |
| minyear | 只有当被截断的列表与带有相同 labelyear 的另一列表相同时才会消除 labelname 列表的歧义。该操作模式适用于作者–年份样式中，如果要求 labeldateparts=true 的场合。 |

请注意，uniquelist 选项也会影响 \ifsingletitle 测试和 extrayear 域。更多细节和实例见 § 4.11.4 节。

### 3.1.3 条目选项

条目选项是控制参考文献数据条目处理的包选项。可以在以下不同尺度上设置。

**3.1.3.1 导言区/类型/条目选项** 下列选项可以基于类型或条目在 options 域中设置。此外还可以在 \usepackage 的可选项以及配置文件和导言区中使用。这可用于全局改变默认样式。

useauthor=true, false　　default: true

是否在标签和排序中使用 author 域。如果一个条目包含 author 域但出于某种原因通常不以作者引用，该选项是很有用的。设置 useauthor=false 并不意味着 author 被完全忽略了，而只是在标签和排序中不使用。该条目将按照 editor 或 title 域

| 选项 | 激活域 | 激活计数器 | 计数器跟踪 |
|---|---|---|---|
| labelalpha | labelalpha | extraalpha | label |
| labeldateparts | labelyear | extrayear | labelname+ |
| | labelmonth | | labelyear |
| | labelday | | |
| | labelendyear | | |
| | labelendmonth | | |
| | labelendday | | |
| | labelhour | | |
| | labelminute | | |
| | labelsecond | | |
| | labelendhour | | |
| | labelendminute | | |
| | labelendsecond | | |
| | labelseason | | |
| | labelendseason | | |
| | labeltimezone | | |
| | labelendtimezone | | |
| labeltitle | — | extratitle | labelname+labeltitle |
| labeltitleyear | — | extratitleyear | labeltitle+labelyear |

**Table 8: 歧义消除计数器**

的字母顺序排列。在标准样式中，此时 author 会在标题后打印。参考 § 3.5 节。该选项可以基于每一类型设置。

useeditor=true, false default: true

在标签和排序中是否用 editor 域来代替缺失的 author 域。如果一个条目包含 editor 域但是通常不以作者引用，那么该选项是很有用的。设置 useeditor=false 并不意味着 editor 被完全忽略了，而只是在标签和排序中不用 editor 来代替缺失的 author。该条目会以 title 的字母顺序排列。在标准样式中，此时 editor 会在标题之后打印。参考 § 3.5 节。该选项可以基于每一类型或条目而设置。

usetranslator=true, false default: false

在标签和排序中是否用 translator 代替缺失的 author/editor。设置 usetranslator = true 并不意味着 translator 会覆盖 author/editor，而只是当 author/editor 缺失或者 useauthor 和 useeditor 选项设置为 false 时作为后备。也就是说，如果要按译者而不是作者引用一本书，你需要设置如下选项：

```
@Book{...,
  options    = {useauthor=false,usetranslator=true},
  author     = {...},
  translator = {...},
  ...
```

该选项也可以基于每一类型或条目而设置。在标准样式中，translator 默认在标题之后打印，参考 § 3.5 节。

use⟨*name*⟩=true, false　　default: true

按照 useauthor、useeditor 和 usetranslator，数据模型中定义的所有的姓名列表都有一个选项用于控制自动定义的排序和标签行为。此时会自动创建全局、基于类型和基于条目选项而调用的 use⟨*name*⟩。

useprefix=true, false　　default: false

是否在以下几种情况下考虑默认数据模型中的姓名前缀部分（von、van、of,、da、de、della 等）：

- 在引用中打印姓
- 排序
- 生成标签的某些类型
- 生成姓名惟一性信息。
- 参考文献的格式方面

如果激活该选项，Biblatex 总是在姓氏前面加上该前缀。例如 Ludwig van Beethoven 将引作“van Beethoven”并按照“Van Beethoven, Ludwig”排序，而如果未激活该选项（默认情况），将引作“Beethoven”并按照“Beethoven, Ludwig van”排序。该选项也可基于每一类型设置。当使用 biblatexml 数据源以及 Biber 支持的 BibTeX 扩展名格式时，还可以基于每一姓名列表或姓名而设置。

indexing=true, false, cite, bib

indexing 选项也可以基于每一类型或条目设置。详见 § 3.1.2.1 节。

#### 3.1.3.2 类型/条目选项

下列选项只能基于类型或条目在 options 域中设置，而不能全局设置。

skipbib=true, false　　default: false

如果激活该选项，该条目在参考文献中将被排除在外但仍然可以引用。该选项也可以基于每一类型设置。

skipbiblist=true, false　　default: false

如果激活该选项，该选项将在文献列表中被排除在外，但仍然包含在参考文献中并可以被 shorthand 引用。该选项也可以基于每一类型设置。

skiplab=true, false　　default: false

如果激活该选项，Biblatex 不会给该条目分配标签。正常操作不需要该选项。要小心使用。当激活时，Biblatex 不能保证相应条目有唯一的引用，而且那些需要标签的引用样式可能不能为该条目创建有效的引用。该选项也可以基于每一类型设置。

dataonly=true, false　　default: false

设置该选项等价于设置 uniquename = false、uniquelist = false、skipbib、skipbiblist 和 skiplab。正常操作不需要该选项。要小心使用。该选项也可以基于每一类型设置。

#### 3.1.3.3 条目选项

下列选项只能基于条目在 options 域中而不能全局或基于类型设置。

labelnamefield=⟨*fieldname*⟩

指定搜索 labelname 时首先考虑的域。本质上，只在该条目中该域会放到 \DeclareLabelname 创建的搜索列表之前。

labeltitlefield=⟨*fieldname*⟩

指定搜索 labeltitle 时首先考虑的域。本质上，只在该条目中该域会放到 \DeclareLabeltitle 创建的搜索列表之前。

### 3.1.4 遗留选项

下面的遗留选项可以全局地在 \documentclass 的可选项中使用，也可以局部地在 \usepackage 的可选项中使用：

openbib 该选项用于向后兼容标准 LaTeX 文档类。openbib 类似于 block=par。 Deprecated

## 3.2 全局定制

除了编写新的引用和文献样式，本宏包中还有很多定制样式的方法。定制通常在导言区中进行，但也可以在配置文件中进行以便长期使用。配置文件也可以用于将宏包选项从默认值初始化为不同的值。

### 3.2.1 配置文件

当可用时，本宏包会导入配置文件 biblatex.cfg。该文件会在宏包的末尾，紧跟在引用和文献样式导入之后立即被读取。

### 3.2.2 设置宏包选项

§ 3.1.1 节中的实时载入宏包选项必须在 \usepackage 的可选项中给出。§ 3.1.2 节中的宏包选项同样可以在导言区中给出。以下命令用于执行选项：

\ExecuteBibliographyOptions[⟨*entrytype, ...*⟩]{⟨*key=value, ...*⟩}

该命令也可以在配置文件中使用，以修改宏包选项的默认设置。某些选项还可以基于每一条目而设置。此时，可选的 ⟨*entrytype*⟩ 选项用来确定条目类型。⟨*entrytype*⟩ 选项可以是逗号分隔的值列表。

## 3.3 标准样式

本节简要地描述了本宏包所带的所有文献和引用样式。如果你想自己写样式文件，请参考 § 4 节。

### 3.3.1 标注样式

本宏包所带的引用样式实现了一些常见的引用格式。所有的标准样式都支持 shorthand 域，并且支持超链接和索引。

numeric 该样式实现与 LaTeX 的标准文献工具类似的数值式引用格式。它应当与某种能在参考文献中打印出相应标签的数值式文献样式一起使用，并用于文内引用。该样式在宏包载入时设置如下的宏包选项：autocite=inline、labelnumber=true。该样式还额外提供了一个导言区选项 subentry，这会影响条目集的处理。如果该选项被禁止，指向条目集中某一成员的引用会指向整个的条目集。如果该选项被激活，该样式会支持类似于“[5c]”这样指向条目集中的子条目的引用（这个例子是第三个条目）。详见样式例子。

numeric-comp numeric 样式紧凑形式的变种，会将两个以上的连续数字打印成一个区间。该样式类似于 cite 宏包和数值模式中的 natbib 宏包的 sort&compress 选项。例如，“[8, 3, 1, 7, 2]”会变成“[1–3, 7, 8]”。它用于文中引用。该样式在宏包载入时设置如下的宏包选项：autocite=inline、sortcites=true、labelnumber=true。它也提供了 subentry 选项。

numeric-verb numeric 样式详细形式的变种。不同之处在于对一组引用的处理，并且只当不同的条目键值传递给单个引用命令时才会显示差异。例如，“[2, 5, 6]”会变成“[2]; [5]; [6]”。它用于文中引用。该样式在宏包载入时设置如下的宏包选项：autocite=inline、labelnumber=true。它也提供了 subentry 选项。

alphabetic 该样式实现的字母顺序引用格式类似于传统 BibTeX 的 alpha.bst 样式。字母标签某种程度上类似于紧凑的作者–年份样式，但是使用的方式类似于数字引用格式。例如，“Jones 1995”会是“[Jon95]”；而“Jones and Williams 1986”会缩写为“[JW86]”。该样式应当与一种字母顺序文献样式一起使用，从而可以在参考文献中打印出相应的标签。它用于文内引用。该样式在载入时设置如下的宏包选项：autocite=inline、labelalpha=true。

alphabetic-verb alphabetic 样式详细格式的变种。不同之处在于对一组引用的处理，并且只当不同的条目键值传递给单个引用命令时才会显示差异。例如“[Doe92; Doe95; Jon98]”会变成“[Doe92]; [Doe95]; [Jon98]”。它用于文内引用。该样式在载入时设置如下的宏包选项：autocite=inline、labelalpha=true。

authoryear 该样式实现了作者–年份引用格式。如果参考文献中包含多个由同一作者同一年份发表的作品，那么年份后会附加一个字母用以区分。例如该样式会打印出“Doe 1995a; Doe 1995b; Jones 1998”这样的引用。该样式应当与一种作者–年份文献样式一起使用，从而可以在参考文献中打印出相应的标签。它起初用于文内引用，但也可以用在脚注中。该样式在载入时设置如下的宏包选项：autocite=inline、labeldate=true、uniquename=full、uniquelist=true。

authoryear-comp authoryear 样式紧凑格式的变种。如果传递给单个引用命令的一列文献作者相同，那么该作者只会打印一次。如果它们年份也相同，那么年份也只会打印一次。例如，“Doe 1995b; Doe 1992; Jones 1998; Doe 1995a”在该样式下会变成“Doe 1992,

1995a,b; Jones 1998”。它起初用于文中引用，但也可以用在脚注中。该样式在载入时设置如下的宏包选项：autocite=inline、sortcites=true、labeldate=true、uniquename=full、uniquelist=true。

authoryear-ibid　authoryear 样式的变种，会用缩略语 *ibidem* 替代重复的引用，除非该引用在当前页或跨页是第一次出现，或者 *ibidem* 在宏包选项 ibidtracker=constrict 的意义下表意不清。该样式在载入时设置如下的宏包选项：autocite=inline、labeldate=true、uniquename=full、uniquelist=true、ibidtracker=constrict、pagetracker=true。该样式还额外提供了一个导言区选项 ibidpage。详见样式例子。

authoryear-icomp　一个结合了 authoryear-comp 和 authoryear-ibid 的样式。该样式在载入时设置如下的宏包选项：autocite=inline、labeldate=true、uniquename=full、uniquelist=true、ibidtracker=constrict、pagetracker=true、sortcites=true。该样式还额外提供了一个导言区选项 ibidpage。详见样式例子。

authortitle　该样式实现了一个简单的作者–标题引用格式。如果可用的话，它会使用 shorttitle 域。它用于脚注中给出的引用。该样式在载入时设置如下的宏包选项：autocite=footnote、uniquename=full、uniquelist=true。

authortitle-comp　authortitle 样式紧凑格式的变种。如果传递给单个引用命令的一列文献作者相同，那么该作者只会打印一次。例如，“Doe, *First title*; Doe, *Second title*”在此样式下会变成“Doe, *First title*, *Second title*”。它用于脚注中给出的引用。该样式在载入时设置如下的宏包选项：autocite=footnote、sortcites=true、uniquename=full、uniquelist=true。

authortitle-ibid　authortitle 样式的变种，会用缩略语 *ibidem* 替代重复的引用，除非该引用在当前页或跨页是第一次出现，或者 *ibidem* 在宏包选项 ibidtracker=constrict 的意义下表意不清。它用于脚注中给出的引用。该样式在载入时设置如下的宏包选项：autocite=footnote、uniquename=full、uniquelist=true、ibidtracker=constrict、pagetracker=true。该样式还额外提供了一个导言区选项 ibidpage。详见样式例子。

authortitle-icomp　结合了 authortitle-comp 和 authortitle-ibid 特性的样式。该样式在载入时设置如下的宏包选项：autocite = footnote、uniquename = full、uniquelist = true、ibidtracker=constrict、pagetracker=true、sortcites=true。该样式还额外提供了一个导言区选项 ibidpage。详见样式例子。

authortitle-terse　authortitle 样式简明格式的变种，如果文献中包含多个相应作者/编辑的作品，那么只会打印出标题。如果可用的话，该样式会使用 shorttitle 域。它在文中引用和脚注中引用都适用。该样式在载入时设置如下的宏包选项：autocite=inline、singletitle=true、uniquename=full、uniquelist=true。

authortitle-tcomp　结合了 authortitle-comp 和 authortitle-terse 特性的样式。如果可用的话，该样式会使用 shorttitle 域。它在文内引用和脚注中引用都适用。该样式在载入时设置如下的宏包选项：autocite=inline、sortcites=true、singletitle=true、uniquename=full、uniquelist=true。

authortitle-ticomp 结合了 authortitle-icomp 和 authortitle-terse 特性的样式。换句话说就是带有 *ibidem* 特性的 authortitle-tcomp 样式变种。它在文中引用和脚注中引用都适用。该样式在载入时设置如下的宏包选项：autocite=inline、ibidtracker=constrict、pagetracker=true、sortcites=true、singletitle=true、uniquename=full、uniquelist=true。该样式还额外提供了一个导言区选项 ibidpage。详见样式例子。

verbose 详细的引用样式，在第一次引用某条目时会打印出类似于参考文献那样的长引用格式，并且在之后打印出短格式。如果可用的话，shorttitle 域会用在所有的短格式中。如果 shorthand 域有定义，该 shorthand 会在第一次引用时被引入并在之后作为短格式被使用。由于在第一次引用时提供了所有的文献数据，因此该样式的使用不需要参考文献和 shorthand 列表。它用于脚注中给出的引用。该样式在载入时设置如下的宏包选项：autocite=footnote、citetracker=context。该样式还额外提供了一个导言区选项 citepages。详见样式例子。

verbose-ibid verbose 样式的变种，会用缩略语 *ibidem* 替代重复的引用，除非该引用在当前页或跨页是第一次出现，或者 *ibidem* 在宏包选项 ibidtracker=strict 的意义下表意不清。它用于脚注中给出的引用。该样式在载入时设置如下的宏包选项：autocite=footnote、citetracker=context、ibidtracker=constrict、pagetracker=true。该样式还额外提供了导言区选项 ibidpage 和 citepages。详见样式例子。

verbose-note 该样式与 verbose 样式类似，会在第一次引用某条目时打印出类似于参考文献那样的长格式，并且在之后打印出短格式。与 verbose 样式不同的是，短格式会指向带有长格式的脚注。如果文献包含了多个同一作者/编辑的作品，该短格式会带有标题。如果可用的话，所有的短格式会使用 shorttitle 域。如果 shorthand 域有定义，它会被 verbose 样式处理。由于在第一次引用时提供了所有的文献数据，因此该样式的使用不需要参考文献和 shorthand 列表。该样式仅仅用于脚注中给出的引用。该样式在载入时设置如下的宏包选项：autocite=footnote、citetracker=context、singletitle=true。该样式还额外提供了导言区选项 pageref 和 citepages。详见样式例子。

verbose-inote verbose-note 样式的变种，会用缩略语 *ibidem* 替代重复的引用，除非该引用在当前页或跨页是第一次出现，或者 *ibidem* 在宏包选项 ibidtracker=strict 的意义下表意不清。该样式仅仅用于脚注中给出的引用。该样式在载入时设置如下的宏包选项：autocite=footnote、citetracker=context、ibidtracker=constrict、singletitle=true、pagetracker=true。该样式还额外提供了导言区选项 ibidpage、pageref 和 citepages。详见样式例子。

verbose-trad1 该样式实现了传统的引用格式。与 verbose 样式类似，它会在第一次引用某条目时打印出类似于参考文献那样的长格式，并且在之后打印出短格式。此外，它在重复的引用中使用学术性缩略语 *ibidem*、*idem*、*op. cit.* 和 *loc. cit.* 来代替重复的作者、标题、页码数。如果 shorthand 域有定义，那么会在第一次引用时被引入并在之后作为短格式被使用。由于在第一次引用时提供了所有的文献数据，因此该样式的使用不需要参考文献和 shorthand 列表。它用于脚注中给出的引用。该样式在载入时设置如下的宏包选项：autocite=footnote、citetracker=

context、ibidtracker=constrict、idemtracker=constrict、opcittracker=context、loccittracker=context。该样式还额外提供了导言区选项 ibidpage、strict 和 citepages。详见样式例子。

verbose-trad2 另外一种传统引用格式。它同样类似于 verbose 样式但是在重复的引用中使用 *ibidem* 和 *idem* 等学术性缩略语。与 verbose-trad1 样式不同的是，*op. cit.* 缩略语的逻辑有所不同，并且不使用 *loc. cit.*。事实上它更类似于 verbose-ibid 和 verbose-inote 而不是 verbose-trad1。该样式在载入时设置如下的宏包选项：autocite=footnote、citetracker=context、ibidtracker=constrict、idemtracker=constrict。该样式还额外提供了导言区选项 ibidpage、strict 和 citepages。详见样式例子。

verbose-trad3 仍然是一种传统的引用格式。它类似于 verbose-trad2 样式，但是使用缩略语 *ibidem* 和 *op. cit.* 的方式稍有不同。该样式在载入时设置如下的宏包选项：autocite=footnote、citetracker=context、ibidtracker=constrict、loccittracker=constrict。该样式也额外提供了导言区选项 strict 和 citepages。详见样式例子。

reading 一个同名的文献样式所带的引用样式，会载入 authortitle 样式。

下列样式具有特殊目的，不用于文档的最终版本。

draft 在引用中使用条目键的草稿样式。该样式在载入时设置如下的宏包选项：autocite=plain。

debug 该样式会打印出条目键而不是标签。它只用于调试，在载入时设置如下的宏包选项：autocite=plain。

### 3.3.2 参考文献样式

本宏包所带的所有文献样式对于每一文献条目都使用相同的基本格式。不同之处仅仅在于参考文献中打印的标签种类和文献列表的总体格式。每一个引用样式都有一个对应的文献样式。请注意，一些文献样式仅仅载入了另外更一般的样式，因此这里没有提及。例如，文献样式 authortitle-comp 会载入 authortitle 样式。

numeric 该样式打印出类似于 LaTeX 标准文献功能的数值标签。它应与数值引用样式结合使用。请注意，shorthand 域会覆盖默认标签。该样式在载入时设置如下的宏包选项：labelnumber=true。该样式还额外提供了一个导言区选项 subentry，这会影响条目集的处理。如果该选项被激活，条目集中的所有成员都会用一个字母标记，这可用于集成员的引用而不是整个条目集。详见样式例子。

alphabetic 该样式打印的字母顺序标签类似于传统 BibTeX 的 alpha.bst 样式。它应与字母顺序引用样式结合使用。请注意，shorthand 域会覆盖默认标签。该样式在载入时设置如下的宏包选项：labelalpha=true、sorting=anyt。

authoryear 该样式不同于其它样式之处在于，发表日期不是在条目的末尾而是在作者/编辑之后。它应与一个作者–年份引用样式结合使用。重复的作者和编辑名会用短横线代替，除非该条目是当前页或跨页的第一个。该样式额外提供了导言区选项 dashed

来控制该特征。此外还额外提供了导言区选项 mergedate。详见样式例子。该样式在载入时设置如下的宏包选项：labeldate=true、sorting=nyt、pagetracker=true、mergedate=true。

authortitle 该样式不会打印出任何标签。它应与一个作者–年份引用样式结合使用。重复的作者和编辑名会用短横线代替，除非该条目是当前页或跨页的第一个。该样式额外提供了一个导言区选项 dashed 来控制该特征。详见样式例子。该样式在载入时设置如下的宏包选项：pagetracker=true。

verbose 该样式类似于 authortitle 样式。该样式额外提供了一个导言区选项 dashed。详见样式例子。该样式在载入时设置如下的宏包选项：pagetracker=true。

reading 这一特殊的文献样式是为个人阅读列表、带有注释的文献和类似应用而设计的。它选择性地在参考文献中包含 annotation、abstract、library 和 file 等域。如果需要的话，它还会在参考文献中添加不同种类的短标题。该样式还额外提供了导言区选项 entryhead、entrykey、annotation、abstract、library 和 file 来控制是否在参考文献中打印相应的项目。详见样式例子。见 § 3.12.8 节。该样式在载入时设置如下的宏包选项：loadfiles=true、entryhead=true、entrykey=true、annotation=true、abstract=true、library=true、file=true。

下列样式具有特殊目的，不用于文档的最终版本。

draft 草稿样式会在参考文献中包含条目键。文献会按照条目键排序。该样式在载入时设置如下的宏包选项：sorting=debug。

debug 该样式会以表格形式打印出所有的文献数据。它只用于调试，在载入时设置如下的宏包选项：sorting=debug。

## 3.4 关联条目

几乎所有的文献样式都需要作者去确定条目之间的某些关系类型，例如“Reprint of”、“Reprinted in”等等。当然不可能通过提供数据域来覆盖所有的关系，为此，Biblatex 通过使用条目域 related、relatedtype 和 relatedstring 提供了一种一般性的机制。被关联的条目不需要被引用，本身也不会出现在参考文献中（当然，除非它自己另外单独被引用），而是作为数据源被拷贝一份副本。relatedtype 域需要确定在相关联条目的信息前打印的本地化字符串，例如“Orig. Pub. as”。relatedstring 域可以用于覆盖那些通过 relatedtype 确定的字符串。一些例子如下：

```
@Book{key1,
  ...
  related     = {key2},
  relatedtype = {reprintof},
  ...
}
```

```
@Book{key2,
  ...
}
```

这里我们指定条目 key1 是条目 key2 的重印本。在 Book 条目的文献驱动里，当为条目 key1 而调用 \usebibmacro{related} 时：

- 如果本地化字符串“reprintof”有定义，那么将以 relatedstring:reprintof 格式打印出来。如果该格式指令没有定义，这些字符串将以 relatedstring:default 格式打印。
- 如果宏 related:reprintof 有定义，那么将用于确定条目 key2 包含的信息的格式，否则将使用宏 related:default。
- 如果 related:reprintof 格式有定义，那么将用于确定本地化字符串和数据的格式；如果该格式没有定义，将使用 related 格式。

也支持串联或者循环关系：

```
@Book{key1,
  ...
  related     = {key2},
  relatedtype = {reprintof},
  ...
}

@Book{key2,
  ...
  related     = {key3},
  relatedtype = {translationof},
  ...
}

@Book{key3,
  ...
  related     = {key2},
  relatedtype = {translatedas},
  ...
}
```

也可以实现同一条目的多重关系：

```
@MVBook{key1,
  ...
  related     = {key2,key3},
```

```
  relatedtype = {multivolume},
  ...
}

@Book{key2,
  ...
}

@Book{key3,
  ...
}
```

请注意，多重关联条目列表中的顺序是很重要的。多重关联条目的数据将按照该域中所列的顺序打印。关于该特征背后的机制请参考 § 4.5.1 节。可以通过 § 3.1.2.1 中的宏包选项 `related` 来关闭该特征。

可以使用 `relatedoptions` 域来设置关联条目数据克隆的选项。如果你需要覆盖默认设置的关于所有关联条目克隆的 `dataonly` 选项，那么该域是很有用的。例如，如果你想在文档中展示一些相关联克隆体的名称，同时想要它们不与其它条目的名称相混淆，但是正常情况下这不会发生的，因为相关联克隆体由基于每一条目的 `dataonly` 选项设置，这反过来又设置了 `uniquename=false` 和 `uniquelist=false`。此时，你只需设置 `relatedoptions` 为 `skiplab, skipbib`。

## 3.5 排序选项

本宏包支持多种文献排序格式。排序格式由 § 3.1.2.1 节中的 `sorting` 宏包选项确定。除了常规的数据域之外，还有一些特殊域也可用于优化文献排序。附录 C.1 和 C.2 大致概述了 Biblatex 支持的字母顺序排序格式。而年代顺序排序格式则列在 附录 C.3 中。以下依次是这些格式的一些解释。

在排序过程中首先要考虑的事项总是条目的 `presort` 域。如果该域没有定义，Biblatex 会使用缺省值“`mm`”作为预排序字符串。其次考虑的是 `sortkey` 域。如果该域有定义，它将作为主要的排序关键字。此时除了 `presort` 域，将不考虑其它信息。如果 `sortkey` 域没有定义，排序将使用姓名信息。本宏包将依次尝试使用 `sortname`、`author`、`editor` 和 `translator` 等域。考虑哪些域也取决于 `useauthor`、`useeditor` 和 `usetranslator` 选项的设置。如果这三个选项都没有启用，那么 `sortname` 也将被忽略。请注意，所有的名称域都与 `maxnames` 和 `minnames` 有关。如果没有名称域是合适的，或者由于它们没有定义、或者由于 `use`⟨*name*⟩ 域都未启用，那么 Biblatex 将采用 `sorttitle` 和 `title` 作为最后的备选。余下考虑的诸项依次是：`sortyear` 域（如果给出的话），否则考虑 `year` 域的前四个数字；`sorttitle` 域（如果给出的话），否则考虑 `title` 域；`volume` 域。请注意，附录 C.2 展示的排序格式包括了额外一项：`labelalpha` 域是“alphabetic”文献样式所使用的标签。严格地讲，用于排序的字符串是 `labelalpha` + `extraalpha`。附录 C.2 中的排序格式只可以与字母顺序样式联合使用。

附录 C.3 展示的年代排序格式同样使用域 presort 和 sortkey（如果有定义的话）。其次考虑的是 sortyear 或者 year 域，这当然取决于是否可用。ynt 格式将从该域中提取前四个数字。如果这两个域都没有定义，那么将使用后备值 9999。这意味着没有年份的条目都会移动到列表末尾。ydnt 格式从概念上也是类似的，不过是用降序排列年份。与 ynt 格式一样，后备值是 9999。余下考虑的项与上面讨论的字母排序格式类似。请注意，ydnt 排序格式只对日期按照降序排列。其它项仍和平常一样按照升序排列。

通常来说不需要使用 sortkey、sortname 或 sorttitle 等特殊域。Biblatex 宏包通过使用条目常规域的数据就很容易得到所需的排列顺序。只有当你想手动修改文献排序或者所需的数据缺失时，你才需要使用这些特殊域。关于特殊域的可能用法请参考 § 2.2.3 节中的描述。

## 3.6 数据注解

理想状态下，文献数据文件中不应当有格式信息。然而，有时只有通过这种有争议的做法才能实现想要的结果。数据注解（data annotations）就是一种解决该问题的方法。通过允许用户在文献数据源中添加某种语义信息（而不是排版标记），使得文献样式可以在标记时使用该信息。例如，如果想要按照如下规则高亮某些作品中的姓名：学生作者在文献中用上标星号表示，而通讯作者用粗体表示；那么，可以尝试如下方法：

```
@MISC{Article1,
  AUTHOR = {Last1\textsuperscript{*}, First1 and \textbf{Last2}, \textbf{
    ↪ First2} and Last3, First3}
}
```

这一做法有一些问题。首先，它会打断 BibTeX 脆弱的姓名解析程序指令，可能根本不能编译。其次，数据与标记的混合是硬编码的：其它样式不易共享和使用该数据。当然，在样式或者文件中使用 Biblatex 内部指令可能实现该格式，但是这一做法比较复杂而且不可靠，很多用户不愿意使用。

为了处理这些问题，Biblatex 提供了一般性的数据注解功能，使得可以向数据域、数据域列表中的项目（例如姓名），以及某些项目的一部分（例如姓、名等姓名部分）中附加逗号分隔列表作为注解。此外还提供了一些宏来检查可以用于格式指令的注解。

数据注解有三种“尺度”，按照特性增加的顺序依次为：

- field—用于数据源条目中的顶层域
- item—用于数据源条目中列表域中的项目
- part—用于数据源条目中列表域中项目的一部分

BibTeX 和 biblatexml 数据源都支持数据注解。

在 biblatexml 数据源中添加数据注解是很容易的，因为可以通过简单的 XML 属性来指定。继续上面的例子，我们有：

```
<bltx:entries xmlns:bltx="http://biblatex-biber.sourceforge.net/
    ↪ biblatexml">
  <bltx:entry id="test" entrytype="misc">
    <bltx:names type="author">
      <bltx:name>
        <bltx:namepart type="given" initial="F">First1</bltx:namepart>
        <bltx:namepart type="family" initial="L" annotation="student">
    ↪ Last1</bltx:namepart>
      </bltx:name>
      <bltx:name annotation="corresponding">
        <bltx:namepart type="given" initial="F">First2</bltx:namepart>
        <bltx:namepart type="family" initial="L">Last2</bltx:namepart>
      </bltx:name>
      <bltx:name>
        <bltx:namepart type="given" initial="F">First3</bltx:namepart>
        <bltx:namepart type="family" initial="L">Last3</bltx:namepart>
      </bltx:name>
    </bltx:names>
  </bltx:entry>
</bltx:entries>
```

这里，向数据项目中添加注解的方式是很显然的。而在 BibTeX 数据源中，注解的格式就没有那么直观了：

```
@MISC{ann1,
  AUTHOR    = {Last1, First1 and Last2, First2 and Last3, First3},
  AUTHOR+an = {1:family=student;2=corresponding},
}
```

这里域姓名后缀 `+an` 可以由用户定义[27]，用于标记某个数据域为去掉后缀的域的注解。BibTeX 注解域的格式如下：

```
<annotationspecs> ::= <annotationspec> [ ";" <annotationspec> ]
<annotationspec>  ::= [ <itemcount> [ ":" <part> ] ] "=" <annotations>
<annotations>     ::= <annotation> [ "," <annotation> ]
<annotation>      ::= (string)
```

也就是说，多个特性之间由分号分隔。每一特性是一个等号后跟一个逗号分隔的注解关键字列表。为了为列表中某一项作注解，需要将该列表项的编号放在等号前面（列表从 1 开始编号）。如果需要为列表项的某一部分做注解，需要将该部分名放在编号之后，并且前接一个冒号。姓名部分的名称在数据模型中有定义，见 § 4.2.3 节。以下是一些例子：

[27] 见 Biber 的 `–annotation-marker` 选项。

```
AUTHOR      = {Last1, First1 and Last2, First2 and Last3, First3},
AUTHOR+an   = {3:given=annotation1, annotation2},
TITLE       = {A title},
TITLE+an    = {=a title annotation, another title annotation},
LANGUAGE    = {english and french},
LANGUAGE+an = {1=annotation3; 2=annotation4}
}
```

为了在文献格式中获取注解信息，这里提供了三个宏，分别对应与相应的注解尺度：

\iffieldannotation{⟨*annotation*⟩}{⟨*true*⟩}{⟨*false*⟩}

如果当前数据域有注解，那么执行 ⟨*true*⟩，否则为 false。

\ifitemannotation{⟨*annotation*⟩}{⟨*true*⟩}{⟨*false*⟩}

如果当前数据域的当前项目有注解，那么执行 ⟨*true*⟩，否则为 false。

\ifpartannotation{⟨*part*⟩}{⟨*annotation*⟩}{⟨*true*⟩}{⟨*false*⟩}

如果当前数据域中当前项目中名为 ⟨*part*⟩ 的部分有注解，那么执行 ⟨*true*⟩，否则为 false。

这些宏的使用场合与 `\currentfield`, `\currentlist` 和 `\currentname` 等命令相同（见 § 4.4.2 节），即，在格式指令内部。它们自动确定当前被处理的数据域的名称，以及能够确定列表域中当前项目的 `listcount` 值。姓名部分等项目部分需要显式地指明。下面的例子可以用于姓名格式指令，说明如何使用注解信息来解决本节之前提出的问题：在所有注解为“student”的姓之后加上星号：

```
  \ifpartannotation{family}{student}
    {\textsuperscript{*}}
    {}%
```

将标记为“corresponding”的姓名列表项中的姓和名加粗：

```
\renewcommand*{\mkbibnamegiven}[1]{%
  \ifitemannotation{corresponding}
    {\textbf{#1}}
    {#1}}

\renewcommand*{\mkbibnamefamily}[1]{%
  \ifitemannotation{corresponding}
    {\textbf{#1}}
    {#1}}
```

## 3.7 参考文献命令

### 3.7.1 数据源

`\addbibresource`[⟨*options*⟩]{⟨*resource*⟩}

将 ⟨*resource*⟩ 添加到默认资源列表中，例如 `.bib` 文件。该命令只能在导言区中使用。它取代了过时的 `\bibliography` 命令。请注意，文件名包括扩展名，所以不要省略文件名中的 `.bib` 扩展名。另外要注意的是，⟨*resource*⟩ 只能是一个单独的数据源。添加更多的资源需要多次调用 `\addbibresource` 命令，例如：

```
\addbibresource{bibfile1.bib}
\addbibresource{bibfile2.bib}
\addbibresource[location=remote]{http://www.citeulike.org/bibtex/group
    ↪ /9517}
\addbibresource[location=remote,label=lan]{ftp://192.168.1.57/~user/file.
    ↪ bib}
```

由于 ⟨*resource*⟩ 字符串的读取类似于抄录模式，因此它可以包含任意的字符。唯一的限制是其中任何的花括号必须左右匹配。可用的 ⟨*options*⟩ 如下：

`label`=⟨*identifier*⟩

给该数据源分配一个标签。⟨*identifier*⟩ 可以用于在 `refsection` 环境的可选参数中以取代该数据源的全名（见 § 3.7.4 节）。

`location`=⟨*location*⟩ default: `local`

数据源的地址。⟨*location*⟩ 可以是 `local` 或者 `remote`，分别对应本地数据和在线 URL 数据。远程资源需要 Biber 程序。支持 HTTP 和 FTP 协议。远程的 URL 必须是 `bib` 文件的合法路径全称或者是返回 `bib` 文件的 URL。

`type`=⟨*type*⟩ default: `file`

资源的类型。目前唯一支持的类型是 `file`。

`datatype`=⟨*datatype*⟩ default: `bibtex`

资源的数据类型（格式）。目前支持以下格式：

`bibtex` BibTeX 格式。

`biblatexml` 针对 Biblatex 的实验性 XML 格式。见 § D。

`\addglobalbib`[⟨*options*⟩]{⟨*resource*⟩}

该命令不同于 `\addbibresource` 之处在于将 ⟨*resource*⟩ 添加到全局数据源列表中。不过，只有当文档中有参考文献章节并且使用 `refsection` 环境的可选参数（见 § 3.7.4 节）作为确定代替默认资源列表的备选资源时，考虑默认数据源和全局数据源的不同才是有意义的。任何全局资源将被添加到所有的参考文件章节中。

`\addsectionbib`[⟨*options*⟩]{⟨*resource*⟩}

该命令与 `\addbibresource` 的不同之处在于，会记录数据源的 ⟨*options*⟩ 但是 ⟨*resource*⟩ 没有添加到任何数据源列表中。有该需求的场合是 (1) 该数据源仅仅用于 `refsection` 环境的可选参数中（§ 3.7.4 节）；(2) 该数据源需要不同于默认设置的选项。此时，`\addsectionbib` 会在导言区中设置合适的 ⟨*options*⟩，从而在其使用前声明 ⟨*resource*⟩。`label` 选项可以用于分配给该资源一个简短的名称。

`\bibliography`{⟨*bibfile, ...*⟩} Deprecated

添加文献资源的过时命令，仅处于向后兼容性而支持。类似 `\addbibresource`，该命令只能在导言区中使用，并将资源添加到默认资源列表中。它的选项是逗号分隔的 `bib` 文件列表。文件名中的 `.bib` 扩展名可以省略。也可以通过多次调用该命令来添加更多文件。该命令已过时，请考虑使用 `\addbibresource` 来取代。

### 3.7.2 参考文献

`\printbibliography`[⟨*key=value, ...*⟩]

该命令可以打印出参考文献。它的可选参数是以 ⟨*key*⟩=⟨*value*⟩ 形式给出的一列选项。可用的选项如下：

`env`=⟨*name*⟩ default: `bibliography/shorthands`

可以用 `\defbibenvironment` 定义的环境来控制参考文献和 shorthands 列表的高层次布局。该选项选择了一个环境。⟨*name*⟩ 对应于用 `\defbibenvironment` 定义环境时的标识符。缺省状态下，`\printbibliography` 命令使用标识符 `bibliography`；而 `\printshorthands` 使用 `shorthands`。另见 §§ ?? 和 3.7.7 节。

`heading`=⟨*name*⟩ default: `bibliography/shorthands`

参考文献和 shorthand 列表通常有一个章标题或者节标题。该选项选择由 `\defbibheading` 定义的标题名 ⟨*name*⟩。缺省状态下，`\printbibliography` 命令使用标题名 `bibliography`；而 `\printshorthands` 使用 `shorthands`。另见 §§ ?? 和 3.7.7 节。

`title`=⟨*text*⟩

如果标题定义支持的话，该选项覆盖由 `heading` 选项提供的缺省标题名。详见 § 3.7.7 节。

`prenote`=⟨*name*⟩

前注是打印在标题之后、文献列表之前的任意文本片段。该选项选择由 `\defbibnote` 所定义的前注 ⟨*name*⟩。缺省状态下不打印任何前注。该注记使用标准正文字体。它不受 `\bibsetup` 和 `\bibfont` 的影响但可以包含自己的字体声明。详见 § 3.7.8 节。

postnote=⟨*name*⟩

后注是打印在参考文献列表之后的任意文本片段。该选项选择由 `\defbibnote` 所定义的后注 ⟨*name*⟩。缺省状态下不打印任何后注。该注记使用标准正文字体。它不受 `\bibsetup` 和 `\bibfont` 的影响但可以包含自己的字体声明。详见 § 3.7.8。

section=⟨*integer*⟩ default: `current section`

只打印在第 ⟨*integer*⟩ 文节中引用的条目。该参考文献节从 1 开始编号。所有在 `refsection` 环境外给出的引用标记为第零节。详见 § 3.7.4 和 § 3.12.3 节中的使用例子。

segment=⟨*integer*⟩ default: `0`

只打印在第 ⟨*integer*⟩ 文献段中引用的条目。参考文献段从 1 开始编号。所有在 `refsection` 环境外给出的引用标记为第零段。详见 § 3.7.4 和 § 3.12.3 节中的使用例子。请注意，一节内部的片段是在该节中局部编号的，故而需要的片段是被查询（或者当前激活的）节的第 n 段。

type=⟨*entrytype*⟩

只打印类型为 ⟨*entrytype*⟩ 的条目。

nottype=⟨*entrytype*⟩

只打印类型不为 ⟨*entrytype*⟩ 的条目。该选项可以使用多次。

subtype=⟨*subtype*⟩

只打印域 `entrysubtype` 定义为 ⟨*subtype*⟩ 的条目。

notsubtype=⟨*subtype*⟩

只打印域 `entrysubtype` 没有定义或者不为 ⟨*subtype*⟩ 的条目。该选项可以使用多次。

keyword=⟨*keyword*⟩

只打印域 `keywords` 包括 ⟨*keyword*⟩ 的条目。该选项可以使用多次。

notkeyword=⟨*keyword*⟩

只打印域 `keywords` 不包括 ⟨*keyword*⟩ 的条目。该选项可以使用多次。

category=⟨*category*⟩

只打印属于 ⟨*category*⟩ 类型的条目。该选项可以使用多次。

notcategory=⟨*category*⟩

只打印不属于 ⟨*category*⟩ 类型的条目。该选项可以使用多次。

filter=⟨*name*⟩

使用由 \defbibfilter 定义的 filter ⟨*name*⟩ 来过滤条目。详见 § 3.7.9 节。

check=⟨*name*⟩

使用由 \defbibcheck 定义的 check ⟨*name*⟩ 来过滤条目。详见 § 3.7.9 节。

resetnumbers=⟨*true,false,number*⟩

该选项只用于数值引用/参考文献样式，并且要求 § 3.1.2.1 中的 defernumbers 选项全局启用。如果启用的话，它将重新设置分配给相应文献中条目的数值标签，即，编号会重新从 1 开始。此外还可以传递数值给该选项以重置编号为给定的数值，例如 resetnumbers=10，这样可以改进整个文档中编号的连续性。请小心使用本选项，因为在手动重新设置下，Biblatex 不能保证标签是全局唯一的。

omitnumbers=true, false

该选项只用于数值引用/参考文献样式，并且要求 § 3.1.2.1 中的 defernumbers 选项全局启用。如果启用的话，Biblatex 不会为相应文献中的条目分配数值标签。当数值型子文献和其它不同格式（例如作者-标题或者作者-年份）的子文献相混合时，这是很有用的。

\bibbysection[⟨*key=value, ...*⟩]

该命令会自动遍历所有的参考文献节。这等价于为每一节给出一个 \printbibliography 命令，不过会有额外好处：自动跳过不含参考文献的节。请注意，\bibbysection 一开始寻找第 1 节中的文献。它会忽略 refsection 外给出的文献，因为它们被分配给第零节。使用例子请参考 § 3.12.3 节。选项可以是由 \printbibliography 支持的一个子集。有效选项是 env、heading、prenote 和 postnote。当前文献内容排序格式会应用在所有的节中（见 § 3.7.10 节）。

\bibbysegment[⟨*key=value, ...*⟩]

该命令会自动遍历所有的参考文献段。这等价于为当前 refsection 的每一段给出一个 \printbibliography 命令，不过会有额外好处：自动跳过不含参考文献的片段。请注意，\bibbysection 一开始寻找第 1 段中的文献。它会忽略 refsection 外给出的文献，因为它们被分配给第 0 段。使用例子请参考 § 3.12.3。选项可以是由 \printbibliography 支持的一个子集。有效选项是 env、heading、prenote 和 postnote。当前文献内容排序格式会用于所有的段中（见 § 3.7.10 节）。

\bibbycategory[⟨*key=value, ...*⟩]

该命令遍历所有的文献类型。这等价于为每一类型给出一个 \printbibliography 命令，不过会有额外好处：自动跳过空类型。类型按照声明的顺序处理。示例见 § 3.12.3 节。选项可以是由 \printbibliography 支持的一个子集。有效选项是 env、heading、prenote 和 postnote。请注意，heading 对于该命令是无效的。当前类型的名字会自动作为标题名。这等价于传递 heading=⟨*category*⟩ 给 \printbibliography，并且意味着对于每一类型都必须有一个匹配的标题定义。当前文献内容排序格式会用于所有的类型中（见 § 3.7.10 节）。

`\printbibheading`[⟨*key=value, ...*⟩]

该命令打印出由 `\defbibheading` 定义的参考文献标题。它有一个可选项，是用 ⟨*key*⟩=⟨*value*⟩ 记号给出的选项列表。选项是 `\printbibliography` 支持的一个小子集。有效选项是 `heading` 和 `title`。缺省情况下，该命令使用标题 `bibliography`。详见 § 3.7.7 节。实例也可见 §§ 3.12.3 和 3.12.4 节。

如果想要在参考文献中使用非全局排序格式的另外一种排序格式，使用 § 3.7.10 节提供的文献内容切换命令。

### 3.7.3 参考文献列表

Biblatex 除了可以打印常规参考文献之外，还能根据文献数据打印任意文献信息列表，例如，与特定条目或者期刊标题缩写有关的速记缩写列表。

文献列表与常规参考文献不同的是，使用同一文献驱动打印所有条目，而不是根据条目类型使用特定于条目的驱动。

`\printbiblist`[⟨*key=value, ...*⟩]{⟨*biblistname*⟩}

该命令用于打印文献列表。其可选项是 ⟨*key*⟩=⟨*value*⟩ 形式的一列选项。除了 `resetnumbers` 和 `omitnumbers`，`\printbibliography` 命令（见 § 3.7.2 节）支持的其它选项在这里都是有效的。如果文档中有任何 `refsection` 环境，那么文献列表只针对于这些环境，详见 § 3.7.4 节。默认情况下该命令使用标题 `biblist`，详见 § 3.7.7 节。

必选项 ⟨*biblistname*⟩ 是文献列表的标题，用于确定如下项目：

- 用于打印列表条目的默认文献驱动。
- 使用 `\DeclareBiblistFilter` 声明的默认 filter（见 § 4.5.7 节），用于过滤 Biber 返回的条目。
- 使用 `\defbibcheck` 命令声明的默认 check（见 § 3.7.9 节），用于后置处理列表条目。
- 默认使用的 bib 环境。
- 默认使用的排序格式名称。

在列表的排序方面，默认使用与该文献列表同名的排序格式（如果存在的话）。只有当未定义时才会切换到备选的当前内容排序格式（见 § 3.7.10 节）。

最常用的文献列表是关于某些条目的速记列表，出于向后兼容性专门有一个别名 `\printshorthands[…]`，定义如下：

```
\printbiblist[...]{shorthand}
```

Biblatex 自动支持默认数据模型中标记为“Label fields”的数据域（见 § 2.2.2 节）。这些域已经自动为其定义了如下项目：

- 默认的 bib 环境（见 § 3.7.7 节）。

- 文献列表 filter（见 § 4.5.7 节）
- 一些支持的格式和长度（见 § 4.10.5 和 § 4.10.4 节）。

因此，打印带有这些域的文献列表只需要很少的设置。例如，想要打印出期刊标题缩写列表，只需要将如下一小段代码放在导言区中：

```
\DeclareBibliographyDriver{shortjournal}{%
  \printfield{journaltitle}}
```

然后在正文中想要打印列表的地方使用如下代码：

```
\printbiblist[title={Journal Shorthands}]{shortjournal}
```

由于默认数据模型将 `shortjournal` 定义为“标签域”，因此在这个例子中：

- 使用自动创建的“shortjournal”bib 环境。
- 使用自动创建的“shortjournal”文献列表 filter，返回 `.bbl` 文件中只带有 `shortjournal` 域的条目。
- 使用定义的“shortjournal”文献驱动来打印条目。
- 使用默认的“biblist”标题，但是这里用“Journal Shorthands”来代替。
- 如果没有名为 `shortjournal` 的格式，那么使用当前文献内容排序格式。

很多情况下想要根据列表中标签域进行排序。由于根据列表名可以自动获取排序格式，因此此时可以简单地使用如下代码：

```
\DeclareSortingScheme{shortjournal}{
  \sort{
        \field{shortjournal}
  }
}
```

自然地，`\printbiblist` 命令的选项以及环境、filters 等的定义可以覆盖所有的默认设置。因此通过这种方法可以从文献数据中打印任意类型的文献列表，并且包含各式信息。

文献列表通常用于打印各类 shorthand 列表。如果多个条目有相同的 shorthand 就会导致重复的条目。例如，如果有几篇论文在同一期刊上，那么期刊缩写列表中就会出现重复条目。不过，这样的列表会自动获取与列表同名的 `\bibcheck`，进而定义相应的 check 来删除重复项目。如果使用 `shortjournal` 域来定义打印所有期刊缩写的列表，那么需要定义如下的 `\bibcheck`：

```
\defbibcheck{shortjournal}{%
   \iffieldundef{shortjournal}{\skipentry}{%
```

```
\iffieldundef{journal}{\skipentry}{%
 \ifcsdef{\strfield{shortjournal}=\strfield{journal}}
   {\skipentry}
   {\savefieldcs{journal}{\strfield{shortjournal}=\strfield{journal
↪ }}}}}}
```

### 3.7.4 参考文献分节

在文档中，refsection 环境用于标记参考文献分节。该环境主要用于在文档的每一章、节或其它部分中实现各自独立的参考文献和 shorthand 列表。在一个文献分节内部，所有引用文献分配的标签都局部在该环境中。技术上，尽管文献分节通常在每一章或每一节中使用，但它们与 \chapter 和 \section 等文档划分是完全独立的。关于自动实现这一功能请参考 § 3.1.2.1 中的 refsection 宏包选项。使用例子也可以参见 § 3.12.3。

\begin{refsection}[⟨*resource, ...*⟩]

\end{refsection}

可选项是特定于该参考文献分节的逗号分隔资源列表。如果省略了该选项，参考文献节会使用缺省的数据源列表，由导言区的 \addbibresource 指定。如果提供了该选项，它会替代缺省的资源列表。不过，由 \addglobalbib 指定的全局文献资源总是包含在内的。refsection 环境不可以相互嵌套，但是可以在 refsection 环境内使用 refsegment 环境来进一步分段。当打印参考文献时，使用 \printbibliography 的 section 选项来选择节；同样地当打印文献列表时使用 \printbiblist 对应的选项。参考文献分节从 1 开始编号。当前节的编号也被写入副本文件中。所有在 refsection 环境外给出的引用都归到第 0 节中。如果在 refsection 内部使用 \printbibliography 环境，它会自动选择当前节。此时不需要 section 选项。这也适用于 \printbiblist。

\newrefsection[⟨*resource, ...*⟩]

该命令类似于 refsection 环境，不同之处在于它是单独命令而不是一个环境。它会自动结束之前的文献分节（如果有的话）并立即开始新的一节。请注意，文档中由最后一个 \newrefsection 开始的文献节会延续到文档的最后。如果你想提前终止的话可以使用 \endrefsection。

### 3.7.5 参考文献分段

在文档中，refsegment 环境用来标记参考文献片段。该环境用于实现在文档的每一章、节或其它部分中将全局的参考文献分成片段。技术上，尽管文献分段通常在每一章或每一节中使用，但它们与 \chapter 和 \section 等文档划分是完全独立的。关于自动实现这一功能请参考 § 3.1.2.1 中的 refsegment 宏包选项。使用例子也可以参见 § 3.12.3。

\begin{refsegment}

\end{refsegment}

regsection 与 refsegment 环境的不同之处在于，前者创建局部于该环境的标签而后者仅为 \printbibliography 命令的 segment filter 提供目标而不影响标签。在整个文档中它们是唯一确定的。refsegment 环境不可以嵌套，但是你可以将其与 refsection 环境结合使用来将文献节细分为段。此时，这些文献分段是局部于被包含的 refsection 环境的。当打印参考文献时，使用 \printbibliography 的 segment 选项来选择文献分段。在一节内，文献段从 1 开始编号，并且当前段的编号会被写入到一个副本文件中。所有在 refsegment 环境之外的引用都归到第 0 段。与 refsection 环境相反，当 \printbibliography 在一个 refsegment 环境内使用时，当前文献分段并不自动选定。

\newrefsegment 该命令类似于 refsegment 环境，不同之处在于它是单独命令而不是一个环境。它会自动结束之前的文献分段（如果有的话）并立即开始新的一段。请注意，由最后一个 \newrefsegment 开始的文献分段会延续到文档结束。如果你想提前终止的话可以使用 \endrefsegment。

### 3.7.6 参考文献分类

参考文献分类允许将参考文献针对不同主题或不同文献类型分成若干部分，例如分成主要文献和次要文献。使用例子参见 § 3.12.4 节。

\DeclareBibliographyCategory{⟨*category*⟩}

声明一个新的 ⟨*category*⟩，可以和 \addtocategory 以及 \printbibliography 的 category、notcategory filter 结合使用。该命令在导言区中使用。

\addtocategory{⟨*category*⟩}{⟨*key*⟩}

将 ⟨*key*⟩ 关键字分配给 ⟨*category*⟩ 类，可以和 \addtocategory 以及 \printbibliography 的 category、notcategory filter 结合使用。该命令可以在导言区和正文中使用。⟨*key*⟩ 可以是一个单独条目关键字或者逗号分隔的键值列表。该分配是全局的。

### 3.7.7 参考文献标题与环境

\defbibenvironment{⟨*name*⟩}{⟨*begin code*⟩}{⟨*end code*⟩}{⟨*item code*⟩}

该命令定义参考文献环境。其中 ⟨*name*⟩ 是标识符，当选择该环境时会传递给 \printbibliography 和 \printshorthands 的 env 选项。⟨*begin code*⟩ 是该环境开始时执行的 LaTeX 代码；而 ⟨*end code*⟩ 在该环境结束时执行；⟨*item code*⟩ 是在参考文献或者 shorthand 列表的每一条目开始时执行的代码。如下是基于 LaTeX 标准 list 环境定义的例子。

```
\defbibenvironment{bibliography}
  {\list{}
```

```
    {\setlength{\leftmargin}{\bibhang}%
     \setlength{\itemindent}{-\leftmargin}%
     \setlength{\itemsep}{\bibitemsep}%
     \setlength{\parsep}{\bibparsep}}}
  {\endlist}
  {\item}
```

如上述例子所示，`\defbibenvironment` 的使用大体类似于 `\newenvironment`，不同之处在于有一个额外的必选项 ⟨*item code*⟩。

`\defbibheading`{⟨*name*⟩}[⟨*title*⟩]{⟨*code*⟩}

该命令定义参考文献标题。其中 ⟨*name*⟩ 是标识符，当选择该标题时会传递给 `\printbibliography` 和 `\printshorthands` 的 `env` 选项。⟨*code*⟩ 是能生成完整标题的 LaTeX 代码，包括页眉和目录中的条目（如果必要的话）。如果 `\printbibliography` 或 `\printshorthands` 带有 `title` 选项，那么 `title` 将作为 `#1` 传递给标题定义；否则由可选的 ⟨*title*⟩ 确定的标题将作为 `#1` 传递给标题定义。⟨*title*⟩ 选项通常是 `\bibname`、`\refname` 或者 `\biblistname` （见 § 4.9.2.1 节）。如果在导言区中改变文档标题时，那么之后通常需要该命令。如下是一个简单标题定义的例子：

```
\defbibheading{bibliography}[\bibname]{%
  \chapter*{#1}%
  \markboth{#1}{#1}}
```

以下预定义的标题与 `\printbibliography` 和 `\printbibheading` 结合使用：

`bibliography`

如果没有给出 `heading` 选项，那么这是 `\printbibliography` 使用的默认标题。缺省定义取决于文档类。如果文类提供 `\chapter` 命令，那么该标题就类似于标准 LaTeX 的 `book` 文类的参考文献标题，即使用 `\chapter*` 来创建不带编号的章，并且不包含在目录中。如果没有 `\chapter` 命令，那么它将类似于标准 LaTeX 的 `article` 文类的参考文献标题，即使用 `\section*` 来创建不带编号的节，并且不包含在目录中。标题中使用的字符串也取决于文档类。`book` 文档类使用本地化字符串 `bibliography`，在其它文档类中则是 `references`（见 § 4.9.2 节）。关于文档类的提示也可以见 §§ 3.13.1 和 3.13.2 节。

`subbibliography`

类似于 `bibliography`，但是标题格式低一级。即，在 `book` 文档类中使用 `\section*` 而不是 `\chapter*`，其它情况使用 `\subsection*` 而不是 `\section*`。

`bibintoc`

类似于 `bibliography` 但是在目录中添加条目。

`subbibintoc`

类似于 `subbibliography` 但是在目录中添加条目。

`bibnumbered`

类似于 `bibliography` 但是使用 `\chapter` 或 `\section` 来创建带编号的条目，同时也添加到目录中。

`subbibnumbered`

类似于 `bibliography` 但是使用 `\section` 或 `\subsection` 来创建带编号的条目，同时也添加到目录中。

`none`

空白的标题定义，用来取消标题。

以下预定义的标题与 `\printshorthands` 结合使用：

`biblist`

如果没有给出 `heading` 选项，那么这是 `\printbiblist` 使用的缺省标题。类似于上面的 `bibliography`，不过是使用本地化字符串 `shorthands` 而不是 `bibliography` 或 `references` （见 § 4.9.2 节）。关于文档类的提示另见 §§ 3.13.1 和 3.13.2 节。

`biblistintoc`

类似于 `shorthands` 但是在目录中添加条目。

`biblistnumbered`

类似于上面的 `biblist` 但是使用 `\chapter` 或 `\section` 来创建带编号的标题，同时也添加到目录中。

### 3.7.8 参考文献注记

`\defbibnote{`⟨*name*⟩`}{`⟨*text*⟩`}`

定义名为 ⟨*name*⟩ 的参考文献注记，通过 `\printbibliography` 和 `\printbiblist` 的 `prenote` 和 `postnote` 选项使用。⟨*text*⟩ 可以是任意文本片段，通常包含若干段落和字体声明。另见 § 3.13.6 节。

### 3.7.9 参考文献过滤和检查

`\defbibfilter{`⟨*name*⟩`}{`⟨*expression*⟩`}`

定义一个可定制的文献过滤 ⟨*name*⟩，可以通过 `\printbibliography` 的 `filter` 选项使用。⟨*expression*⟩ 是复合测试，基于逻辑运算符 `and`、`or`、`not`，组运算符 `(...)`，以及以下的基本测试：

`segment=`⟨*integer*⟩

匹配所有在参考文献分段 ⟨*integer*⟩ 中引用的条目。

`type=`⟨*entrytype*⟩

匹配所有类型为 ⟨*entrytype*⟩ 的条目。

`subtype=`⟨*subtype*⟩

匹配所有 `entrysubtype` 域为 ⟨*subtype*⟩ 的条目。

`keyword=`⟨*keyword*⟩

匹配所有 `keywords` 域包含 ⟨*keyword*⟩ 的条目。如果 ⟨*keyword*⟩ 包含空格，那么需要用括号括起来。

`category=`⟨*category*⟩

匹配所有由 `\addtocategory` 归入 ⟨*category*⟩ 类的条目。

如下是一个 filter 表达式的例子：

```
\defbibfilter{example}{%
  ( type=book or type=inbook )
  and keyword=abc
  and not keyword={x y z}
}
```

该 filter 匹配的条目规则是，条目类型是 `@book` 或 `@inbook`，`keywords` 域包含关键词“`abc`”但不包含“`x y z`”。从以上例子可以看出，所有的元素由空白分开（空格、制表符或者换行）。等号周围没有空白。逻辑运算使用 `etoolbox` 宏包的 `\ifboolexpr` 执行。关于该语法详见 `etoolbox` 手册。Biblatex 旧版本中使用的 `ifthen` 宏包的 `\ifthenelse` 语法这里仍然支持。如下是相同的测试，使用 `ifthen` 样式的语法：

```
\defbibfilter{example}{%
  \( \type{book} \or \type{inbook} \)
  \and \keyword{abc}
  \and \not \keyword{x y z}
}
```

请注意，定制的 filter 对于所在的参考文献分节是局部的。使用 `\printbibliography` 的 `section` filter 来选择不同的分节。这在定制 filter 中是不可能的。

`\defbibcheck{`⟨*name*⟩`}{`⟨*code*⟩`}`

定义了可定制的参考文献 check ⟨*name*⟩，可以通过 `\printbibliography` 的 `check` 选项使用。`\defbibcheck` 从概念上类似于 `\defbibfilter` 不过更加低层。与高层次

表达式不同，⟨*code*⟩ 是 LaTeX 代码，更像是驱动定义中使用的代码，可以执行任意测试来决定是否打印某个给定的条目。当执行 ⟨*code*⟩ 时，相应条目的文献数据是可用的。在 ⟨*code*⟩ 中使用 `\skipentry` 命令会跳过当前条目。例如，下面的 check 只会输出带有 `abstract` 域的条目：

```
\defbibcheck{abstract}{%
  \iffieldundef{abstract}{\skipentry}{}}
...
\printbibliography[check=abstract]
```

下面的 check 会排除所有在 2000 年之前出版的条目：

```
\defbibcheck{recent}{%
  \iffieldint{year}
    {\ifnumless{\thefield{year}}{2000}
       {\skipentry}
       {}}
    {\skipentry}}
```

更多细节请参见作者指南，特别是 §§ 4.6.2 和 4.6.3 节。

### 3.7.10 著录文境

参考文献列表中文献的引用和打印都处于某个*著录文境*（context）内。对于某一条目，著录文境决定了实际用于引用或者提供文献信息的数据。一个著录文境包括以下信息[28]：

- 排序格式
- 构建姓名排序关键字的格式
- 使用字母或数值标签的引用格式的前缀字符串

著录文境具有双重意义。首先，会用于设置影响打印参考文献的选项；其次，设置的选项还可以影响引用命令打印的数据。前一应用场景是很常见的，例如，打印多个带有不同排序格式的参考文献表。

```
\usepackage[sorting=nyt]{biblatex}
\begin{document}
\cite{one}
\cite{two}
\printbibliography
\newrefcontext[sorting=ydnt]
\printbibliography
```

[28] 设计“著录文境”这一概念的目的在于，使其在未来具有可扩展性。

这里我们打印两个参考文献表。其中一个带有默认的“nyt”排序格式，另一个则使用“ydnt”排序格式。

为了说明著录文境的第二种类型应用，我们必须意识到这一点：条目的实际数据可以基于不同的著录文境而变化。在以下的情况中这一点尤其明显：由后端生成的 `extra*` 域（例如 `extrayear`）与条目在排序之后的顺序有关，这样出来的结果就是预期的“a, b, c”的顺序。这就表明，条目的数据在不同排序格式下可以不一样。如果文档中包含多个带有不同排序格式的参考文献列表，那么 `.bbl` 文件中可能出现多个排序列表，它们带有同一条目但是其数据不同（例如 `extrayear` 的值可以不同）。著录文境的目的就在于将这些事项封装在一个语境内部，这样 Biblatex 就可以使用正确的条目数据。以下的例子展示了使用与全局排序格式不同的另一格式打印参考文献列表，使得同一条目的 `extra*` 域在不同排序列表中是不同的：

```
\usepackage[sorting=nyt,style=authoryear]{biblatex}
\DeclareSortingScheme{yntd}{
  \sort{
    \field[strside=left,strwidth=4]{sortyear}
    \field[strside=left,strwidth=4]{year}
    \literal{9999}
  }
  \sort{
    \field{sortname}
    \field{author}
    \field{editor}
  }
  \sort[direction=descending]{
    \field{sorttitle}
    \field{title}
  }
}
\begin{document}
\cite{one}
\cite{two}
\printbibliography
\newrefcontext[sorting=yntd]
\cite{one}
\cite{two}
\printbibliography
```

这里，第二次使用引用命令和 `\printbibliography` 命令时会使用在定制“yntd”排序格式的著录文境中的数据，这与默认的“nyt”格式相关联的数据可能会不相同。也就是说，对于同一条目，不同著录文境中的引用标签（在使用 `extrayear` 的 `authoryear` 样式中）可以不同，这样对其引用就可以不一致。

引用文境可以使用 \DeclareRefcontext 命令进行声明，然后通过文境的名称进行使用，见以下说明。

默认情况下，用于引用的数据来自于打印该条目的最后一个参考文献列表所在的引用文境。例如：

```
\DeclareRefcontext{ap}{labelprefix=A}
\begin{document}

\cite{book, article, misc}

\printbibliography[type=book]

\newrefcontext{ap}
\printbibliography[type=article]

\newrefcontext[sorting=ydnt]
\printbibliography[type=misc]

\end{document}
```

这个例子同时展示了引用文境的声明和使用。在该例子中假设条目类型就是条目的键名，文献引用就对应与用户通常预料到的默认场景。

- 条目 book 的引用会从全局引用文境中提取数据，因为打印该条目的最后的参考文献列表位于全局引用文境中。
- 条目 article 的引用会从带有 labelprefix=A 的引用文境中提取数据，因此引用时会带有前缀“A”。
- 条目 misc 的引用会从带有 sorting=ydnt 的引用文境中提取数据。

有这样一种情况，条目在多个参考文献列表中并且有不同的形式或者可能带有不同的标签（例如，带有不同 labelprefix 值的数字格式）。此时需要告诉 Biblatex 希望从哪个引用文境中提取引用信息。如上所述，这可以通过显式地将引用置于引用文境中而实现。但是在大文档中这种方式会很繁重，因此提供了将引用以程序化的方式分配到引用文境的功能，见下面的 \assignrefcontext* 宏命令。

\DeclareRefcontext{⟨*name*⟩}{⟨*key=value, ...*⟩}

声明一个名称为 ⟨*name*⟩ 的引用文境。⟨*key=value*⟩ 选项定义该文境的属性。所有的文境属性都是可选的，缺省为全局设置。有效的选项为：

sorting=⟨*name*⟩

指定由之前的 \DeclareSortingScheme 命令定义的排序格式。对于在该文境内引用命令中的条目，该格式用于确定检索和打印的数据。

sortingnamekeyscheme=⟨*name*⟩

指定由之前的 \DeclareSortingNamekeyScheme 命令定义的排序姓名关键字格式。该格式用于为文境内的姓名构建排序关键字。

labelprefix=⟨*string*⟩

该选项只用于数字型引用和文献样式，需要全局开启 § 3.1.2.1 节中的 defernumbers 选项。设置改选项也会为该文境范围内的任意 \printbibliography 启用 resetnumbers 选项（除非 resetnumbers 被用户指定的值覆盖）。该选项将 ⟨*string*⟩ 作为前缀分配到该引用文境中的所有条目。例如，如果 ⟨*string*⟩ 是 A，那么打印出来的数值标签就会形如 [A1], [A2], [A3] 等。特别适用的场合是，将参考文献列表划分成带有不同前缀的子列表。⟨*string*⟩ 可以用于所有有关条目中 labelprefix 域中的样式。详见 § 4.2.4.2 节。

\begin{refcontext}[⟨*key=value, ...*⟩]{⟨*name*⟩}

\end{refcontext}

将引用文境封装在一个环境内。可能的 ⟨*key*⟩=⟨*value*⟩ 可选项和 \DeclareRefcontext 中的相同，并且覆盖名为 ⟨*name*⟩ 的引用文境的选项。⟨*name*⟩ 也可以省略成 {}，甚至空的括号也可以省略[29]

refcontext 环境不可以相互嵌套，如果这样的话 Biblatex 会报错。

\newrefcontext[⟨*key=value, ...*⟩]{⟨*name*⟩}

该命令类似于 refcontext 环境，不同之处在于这是单独的命令而不是环境。它会自动结束任何之前以 \newrefcontext 开始的引用文境片段（如果有的话），并立即开启新的引用文境。注意，文档中最后的 \newrefcontext 命令开启的引用文境会一直持续到文档的最后。如果想要提前终止，那么需要使用 \endrefcontext 命令。

在文档的开始，总会有一个全局的文境，其中为每一引用文境选项进行了全局设置。这里的例子总结了引用文境的不同设置：

```
\usepackage[sorting=nty]{biblatex}

\DeclareRefcontext{testrc}{sorting=nyt}

% Global reference context:
%   sorting=nty
%   sortingnamekeyscheme=global
%   labelprefix=

\begin{document}
```

[29] 这种有点怪异的句法是出于对 Biblatex <3.5 的向后兼容性。

```
\begin{refcontext}{testrc}
% reference context:
%   sorting=nyt
%   sortingnamekeyscheme=global
%   labelprefix=
\end{refcontext}

\begin{refcontext}[labelprefix=A]{testrc}
% reference context:
%   sorting=nyt
%   sortingnamekeyscheme=global
%   labelprefix=A
\end{refcontext}

\begin{refcontext}[sorting=ydnt,labelprefix=A]
% reference context:
%   sorting=ydnt
%   sortingnamekeyscheme=global
%   labelprefix=A
\end{refcontext}

\newrefcontext}[labelprefix=B]
% reference context:
%   sorting=nty
%   sortingnamekeyscheme=global
%   labelprefix=B
\endrefcontext

\newrefcontext}[sorting=ynt,labelprefix=C]{testrc}
% reference context:
%   sorting=ynt
%   sortingnamekeyscheme=global
%   labelprefix=C
\endrefcontext
```

\assignrefcontextkeyws[⟨*key=value, ...*⟩]{⟨*keyword1,keyword2, ...*⟩}
\assignrefcontextkeyws*[⟨*key=value, ...*⟩]{⟨*keyword1,keyword2, ...*⟩}
\assignrefcontextcats[⟨*key=value, ...*⟩]{⟨*category1, category2, ...*⟩}
\assignrefcontextcats*[⟨*key=value, ...*⟩]{⟨*category1, category2, ...*⟩}
\assignrefcontextentries[⟨*key=value, ...*⟩]{⟨*entrykey1, entrykey2, ...*⟩}
\assignrefcontextentries*[⟨*key=value, ...*⟩]{⟨*entrykey1, entrykey2, ...*⟩}
\assignrefcontextentries[⟨*key=value, ...*⟩]{⟨***⟩}
\assignrefcontextentries*[⟨*key=value, ...*⟩]{⟨***⟩}

当默认行为不充分时，这些命令会自动将引用置于引用文境中。默认行为是指，从最后打印条目的参考文献列表所在的引用文境中提取引用数据。对于没有在参考文献列表中打印但是以某种方式使用的引用，默认会从文档一开始建立的全局引用文境中提取数据。为了覆盖这一行为，可以手动将引用命令放置在 `refcontext` 环境内，但是这样容易出错并且很繁琐。除此之外，可以登记一个关于 ⟨*keywords*⟩、⟨*categories*⟩ 和 ⟨*entrykeys*⟩ 的逗号分隔列表。这样，任何带有指定关键字的条目、任何指定类别的条目（见 § 3.12.4 节）、任何指定引用关键字的条目都会分别从由 ⟨*refcontext key/values*⟩ 指定的特定引用文境中提取数据，并按照对应的 `refcontext` 环境选项进行解析。这样的引用文境自动分配方式特定于当前的参考文献分节。你可以在任意的这些命令中指定相同的引用键，但需要注意的是，分配方式按照 ⟨*keywords*⟩, ⟨*categories*⟩, ⟨*entrykeys*⟩ 的顺序，后面的规范会覆盖之前的规范。`\assignrefcontextentries` 命令可以接受单个的星号作为选项以代替一列条目键，这样可以将某一参考文献分节中的所有条目键都分配给某个引用文境，而不必显式列出。例如：

```
\assignrefcontextentries[labelprefix=A]{key2}
\cite{key1}
\begin{refcontext}[labelprefix=B]
\cite{key2}
\end{refcontext}
```

这里 `key2` 的引用数据会从引用文境 `labelprefix=A` 中提取，而不是 `labelprefix=B`。即，标签的前缀是“A”不是“B”。带星号的版本不会覆盖局部的引用文境，也就是：

```
\assignrefcontextentries*[labelprefix=A]{key2}
\cite{key1}
\begin{refcontext}[labelprefix=B]
\cite{key2}
\end{refcontext}
```

`key2` 的引用数据会从 `labelprefix=B` 引用文境中提取。注意，这些命令大部分情况下都不必使用，除非多个参考文献表中有相同的文献引用，并且 Biblatex 按照

默认设置不知道引用应该指向哪一个文献列表。详见文件 94-labelprefix.tex 中的例子。

### 3.7.11 动态条目集

除了 @set 条目类型之外，Biblatex 也支持基于文档或参考文献分节定义的动态条目集。下面的命令定义了 ⟨*key*⟩ 集合，可以用在导言区或正文中：

\defbibentryset{⟨*key*⟩}{⟨*key1,key2,key3, ...*⟩}

⟨*key*⟩ 是集合的条目键，像其它条目键一样用于指向该集合。⟨*key*⟩ 必须是唯一的，并且不能与其它条目键名冲突。第二个选项是组成该集合的逗号分隔条目键列表。\defbibentryset 也蕴含了与 \nocite 命令的等价性，即，所有声明的集合也都添加到参考文献表中。当多次声明相同集合时，只有第一次调用的 \defbibentryset 会定义该集合。接下来的对于相同 ⟨*key*⟩ 的定义将被忽略并如同 \nocite⟨*key*⟩ 一样处理。在正文中定义的动态条目集如果包含在 refsection 环境中，那么是局部的。否则它们会归到第 0 文献分节中。在导言区中定义的动态条目集也归到第 0 文献分节中。详见 § 3.12.5 节。

## 3.8 引用命令 Citation Commands

All citation commands generally take one mandatory and two optional arguments. The ⟨*prenote*⟩ is text to be printed at the beginning of the citation. This is usually a notice such as 'see' or 'compare'. The ⟨*postnote*⟩ is text to be printed at the very end of the citation. This is usually a page number. If only one of these arguments is given, it is taken as a postnote. If you want to specify a prenote but no postnote, you need to leave the second optional argument empty, as in \cite[see][]{key}. The ⟨*key*⟩ argument to all citation commands is mandatory. This is the entry key or a comma-separated list of keys corresponding to the entry keys in the bib file. In sum, all basic citations commands listed further down have the following syntax:

\command[⟨*prenote*⟩][⟨*postnote*⟩]{⟨*keys*⟩}⟨*punctuation*⟩

If the autopunct package option from § 3.1.2.1 is enabled, they will scan ahead for any ⟨*punctuation*⟩ immediately following their last argument. This is useful to avoid spurious punctuation marks after citations. This feature is configured with \DeclareAutoPunctuation, see § 4.7.5 for details.

### 3.8.1 标准命令 Standard Commands

The following commands are defined by the citation style. Citation styles may provide any arbitrary number of specialized commands, but these are the standard commands typically provided by general-purpose styles.

\cite[⟨prenote⟩][⟨postnote⟩]{⟨key⟩}
\Cite[⟨prenote⟩][⟨postnote⟩]{⟨key⟩}

These are the bare citation commands. They print the citation without any additions such as parentheses. The numeric and alphabetic styles still wrap the label in square brackets since the reference may be ambiguous otherwise. \Cite is similar to \cite but capitalizes the name prefix of the first name in the citation if the useprefix option is enabled, provided that there is a name prefix and the citation style prints any name at all.

\parencite[⟨prenote⟩][⟨postnote⟩]{⟨key⟩}
\Parencite[⟨prenote⟩][⟨postnote⟩]{⟨key⟩}

These commands use a format similar to \cite but enclose the entire citation in parentheses. The numeric and alphabetic styles use square brackets instead. \Parencite is similar to \parencite but capitalizes the name prefix of the first name in the citation if the useprefix option is enabled, provided that there is a name prefix and the citation style prints any name at all.

\footcite[⟨prenote⟩][⟨postnote⟩]{⟨key⟩}
\footcitetext[⟨prenote⟩][⟨postnote⟩]{⟨key⟩}

These command use a format similar to \cite but put the entire citation in a footnote and add a period at the end. In the footnote, they automatically capitalize the name prefix of the first name if the useprefix option is enabled, provided that there is a name prefix and the citation style prints any name at all. \footcitetext differs from \footcite in that it uses \footnotetext instead of \footnote.

### 3.8.2 样式相关命令 Style-specific Commands

The following additional citation commands are only provided by some of the citation styles which ship with this package.

\textcite[⟨prenote⟩][⟨postnote⟩]{⟨key⟩}
\Textcite[⟨prenote⟩][⟨postnote⟩]{⟨key⟩}

These citation commands are provided by all styles that ship with this package. They are intended for use in the flow of text, replacing the subject of a sentence. They print the authors or editors followed by a citation label which is enclosed in parentheses. Depending on the citation style, the label may be a number, the year of publication, an abridged version of the title, or something else. The numeric and alphabetic styles use square brackets instead of parentheses. In the verbose styles, the label is provided in a footnote. Trailing punctuation is moved between the author or editor names and the footnote mark. \Textcite is similar to \textcite but capitalizes the name prefix of the first name in the citation if the useprefix option is enabled, provided that there is a name prefix.

\smartcite[⟨prenote⟩][⟨postnote⟩]{⟨key⟩}

\Smartcite[⟨prenote⟩][⟨postnote⟩]{⟨key⟩}

Like \parencite in a footnote and like \footcite in the body.

\cite*[⟨prenote⟩][⟨postnote⟩]{⟨key⟩}

This command is provided by all author-year and author-title styles. It is similar to the regular \cite command but merely prints the year or the title, respectively.

\parencite*[⟨prenote⟩][⟨postnote⟩]{⟨key⟩}

This command is provided by all author-year and author-title styles. It is similar to the regular \parencite command but merely prints the year or the title, respectively.

\supercite{⟨key⟩}

This command, which is only provided by the numeric styles, prints numeric citations as superscripts without brackets. It uses \supercitedelim instead of \multicitedelim as citation delimiter. Note that any ⟨*prenote*⟩ and ⟨*postnote*⟩ arguments are ignored. If they are given, \supercite will discard them and issue a warning message.

### 3.8.3 有限标注表 Qualified Citation Lists

This package supports a class of special citation commands called 'multicite' commands. The point of these commands is that their argument is a list of citations where each item forms a fully qualified citation with a pre- and/or postnote. This is particularly useful with parenthetical citations and citations given in footnotes. It is also possible to assign a pre- and/or postnote to the entire list. The multicite commands are built on top of backend commands like \parencite and \footcite. The citation style provides a multicite definition with \DeclareMultiCiteCommand (see § 4.3.1). The following example illustrates the syntax of multicite commands:

```
\parencites[35]{key1}[88--120]{key2}[23]{key3}
```

The format of the arguments is similar to that of the regular citation commands, except that only one citation command is given. If only one optional argument is given for an item in the list, it is taken as a postnote. If you want to specify a prenote but no postnote, you need to leave the second optional argument of the respective item empty:

```
\parencites[35]{key1}[chapter 2 in][]{key2}[23]{key3}
```

In addition to that, the entire citation list may also have a pre- and/or postnote. The syntax of these global notes differs from other optional arguments in that they are given in parentheses rather than the usual brackets:

```
\parencites(and chapter 3)[35]{key1}[78]{key2}[23]{key3}
\parencites(Compare)()[35]{key1}[78]{key2}[23]{key3}
\parencites(See)(and the introduction)[35]{key1}[78]{key2}[23]{key3}
```

Note that the multicite commands keep on scanning for arguments until they encounter a token that is not the start of an optional or mandatory argument. If a left brace or bracket follows a multicite command, you need to mask it by adding `\relax` or a control space (a backslash followed by a space) after the last valid argument. This will cause the scanner to stop.

```
\parencites[35]{key1}[78]{key2}\relax[...]
\parencites[35]{key1}[78]{key2}\␣{...}
```

By default, this package provides the following multicite commands which correspond to regular commands from §§ 3.8.1 和 3.8.2:

`\cites`(⟨*multiprenote*⟩)(⟨*multipostnote*⟩)[⟨*prenote*⟩][⟨*postnote*⟩]{⟨*key*⟩}...[⟨*prenote*⟩][⟨*postnote*⟩]{⟨*key*⟩}
`\Cites`(⟨*multiprenote*⟩)(⟨*multipostnote*⟩)[⟨*prenote*⟩][⟨*postnote*⟩]{⟨*key*⟩}...[⟨*prenote*⟩][⟨*postnote*⟩]{⟨*key*⟩}

The multicite version of `\cite` and `\Cite`, respectively.

`\parencites`(⟨*multiprenote*⟩)(⟨*multipostnote*⟩)[⟨*prenote*⟩][⟨*postnote*⟩]{⟨*key*⟩}...[⟨*prenote*⟩][⟨*postnote*⟩]{⟨*key*⟩}
`\Parencites`(⟨*multiprenote*⟩)(⟨*multipostnote*⟩)[⟨*prenote*⟩][⟨*postnote*⟩]{⟨*key*⟩}...[⟨*prenote*⟩][⟨*postnote*⟩]{⟨*key*⟩}

The multicite version of `\parencite` and `\Parencite`, respectively.

`\footcites`(⟨*multiprenote*⟩)(⟨*multipostnote*⟩)[⟨*prenote*⟩][⟨*postnote*⟩]{⟨*key*⟩}...[⟨*prenote*⟩][⟨*postnote*⟩]{⟨*key*⟩}
`\footcitetexts`(⟨*multiprenote*⟩)(⟨*multipostnote*⟩)[⟨*prenote*⟩][⟨*postnote*⟩]{⟨*key*⟩}...[⟨*prenote*⟩][⟨*postnote*⟩]{⟨*key*⟩}

The multicite version of `\footcite` and `\footcitetext`, respectively.

`\smartcites`(⟨*multiprenote*⟩)(⟨*multipostnote*⟩)[⟨*prenote*⟩][⟨*postnote*⟩]{⟨*key*⟩}...[⟨*prenote*⟩][⟨*postnote*⟩]{⟨*key*⟩}
`\Smartcites`(⟨*multiprenote*⟩)(⟨*multipostnote*⟩)[⟨*prenote*⟩][⟨*postnote*⟩]{⟨*key*⟩}...[⟨*prenote*⟩][⟨*postnote*⟩]{⟨*key*⟩}

The multicite version of `\smartcite` and `\Smartcite`, respectively.

`\textcites`(⟨*multiprenote*⟩)(⟨*multipostnote*⟩)[⟨*prenote*⟩][⟨*postnote*⟩]{⟨*key*⟩}...[⟨*prenote*⟩][⟨*postnote*⟩]{⟨*key*⟩}
`\Textcites`(⟨*multiprenote*⟩)(⟨*multipostnote*⟩)[⟨*prenote*⟩][⟨*postnote*⟩]{⟨*key*⟩}...[⟨*prenote*⟩][⟨*postnote*⟩]{⟨*key*⟩}

The multicite version of `\textcite` and `\Textcite`, respectively.

`\supercites`(⟨*multiprenote*⟩)(⟨*multipostnote*⟩)[⟨*prenote*⟩][⟨*postnote*⟩]{⟨*key*⟩}...[⟨*prenote*⟩][⟨*postnote*⟩]{⟨*key*⟩}

The multicite version of `\supercite`. This command is only provided by the numeric styles.

### 3.8.4 与样式无关的命令 Style-independent Commands

Sometimes it is desirable to give the citations in the source file in a format that is not tied to a specific citation style and can be modified globally in the preamble. The format of the citations is easily changed by loading a different citation style. However,

when using commands such as \parencite or \footcite, the way the citations are integrated with the text is still effectively hard-coded. The idea behind the \autocite command is to provide higher-level citation markup which makes global switching from inline citations to citations given in footnotes (or as superscripts) possible. The \autocite command is built on top of backend commands like \parencite and \footcite. The citation style provides an \autocite definition with \DeclareAutoCiteCommand (see § 4.3.1). This definition may be activated with the autocite package option from § 3.1.2.1. The citation style will usually initialize this package option to a value which is suitable for the style, see § 3.3.1 for details. Note that there are certain limits to high-level citation markup. For example, inline author-year citation schemes often integrate citations so tightly with the text that it is virtually impossible to automatically convert them to footnotes. The \autocite command is only applicable in cases in which you would normally use \parencite or \footcite (or \supercite, with a numeric style). The citations should be given at the end of a sentence or a partial sentence, immediately preceding the terminal punctuation mark, and they should not be a part of the sentence in a grammatical sense (like \textcite, for example).

\autocite[⟨*prenote*⟩][⟨*postnote*⟩]{⟨*key*⟩}
\Autocite[⟨*prenote*⟩][⟨*postnote*⟩]{⟨*key*⟩}

In contrast to other citation commands, the \autocite command does not only scan ahead for punctuation marks following its last argument to avoid double punctuation marks, it actually moves them around if required. For example, with autocite=footnote, a trailing punctuation mark will be moved such that the footnote mark is printed after the punctuation. \Autocite is similar to \autocite but capitalizes the name prefix of the first name in the citation if the useprefix option is enabled, provided that there is a name prefix and the citation style prints any name at all.

\autocite*[⟨*prenote*⟩][⟨*postnote*⟩]{⟨*key*⟩}
\Autocite*[⟨*prenote*⟩][⟨*postnote*⟩]{⟨*key*⟩}

The starred variants of \autocite do not behave differently from the regular ones. The asterisk is simply passed on to the backend command. For example, if \autocite is configured to use \parencite, then \autocite* will execute \parencite*.

\autocites(⟨*multiprenote*⟩)(⟨*multipostnote*⟩)[⟨*prenote*⟩][⟨*postnote*⟩]{⟨*key*⟩}...[⟨*prenote*⟩][⟨*postnote*⟩]{⟨*key*⟩}
\Autocites(⟨*multiprenote*⟩)(⟨*multipostnote*⟩)[⟨*prenote*⟩][⟨*postnote*⟩]{⟨*key*⟩}...[⟨*prenote*⟩][⟨*postnote*⟩]{⟨*key*⟩}

This is the multicite version of \autocite. It also detects and moves punctuation if required. Note that there is no starred variant. \Autocites is similar to \autocites but capitalizes the name prefix of the first name in the citation if the useprefix option is enabled, provided that there is a name prefix and the citation style prints any name at all.

### 3.8.5 文本命令 Text Commands

The following commands are provided by the core of Biblatex. They are intended for use in the flow of text. Note that all text commands are excluded from citation tracking.

\citeauthor[⟨*prenote*⟩][⟨*postnote*⟩]{⟨*key*⟩}
\citeauthor*[⟨*prenote*⟩][⟨*postnote*⟩]{⟨*key*⟩}
\Citeauthor[⟨*prenote*⟩][⟨*postnote*⟩]{⟨*key*⟩}
\Citeauthor*[⟨*prenote*⟩][⟨*postnote*⟩]{⟨*key*⟩}

These commands print the authors. Strictly speaking, it prints the `labelname` list, which may be the `author`, the `editor`, or the `translator`. `\Citeauthor` is similar to `\citeauthor` but capitalizes the name prefix of the first name in the citation if the `useprefix` option is enabled, provided that there is a name prefix. The starred variants effectively force maxcitenames to 1 for just this command on so only print the first name in the labelname list (potentially followed by the "et al" string if there are more names). This allows more natural textual flow when refering to a paper in the singular when otherwise `\citeauthor` would generate a (naturally plural) list of names.

\citetitle[⟨*prenote*⟩][⟨*postnote*⟩]{⟨*key*⟩}
\citetitle*[⟨*prenote*⟩][⟨*postnote*⟩]{⟨*key*⟩}

This command prints the title. It will use the abridged title in the `shorttitle` field, if available. Otherwise it falls back to the full title found in the `title` field. The starred variant always prints the full title.

\citeyear[⟨*prenote*⟩][⟨*postnote*⟩]{⟨*key*⟩}
\citeyear*[⟨*prenote*⟩][⟨*postnote*⟩]{⟨*key*⟩}

This command prints the year (`year` field or year component of `date`). The starred variant includes the `extrayear` information, if any.

\citedate[⟨*prenote*⟩][⟨*postnote*⟩]{⟨*key*⟩}
\citedate*[⟨*prenote*⟩][⟨*postnote*⟩]{⟨*key*⟩}

This command prints the full date (`date` or `year`). The starred variant includes the `extrayear` information, if any.

\citeurl[⟨*prenote*⟩][⟨*postnote*⟩]{⟨*key*⟩}

This command prints the `url` field.

\parentext{⟨*text*⟩}

This command wraps the ⟨*text*⟩ in context sensitive parentheses.

\brackettext{⟨*text*⟩}

This command wraps the ⟨*text*⟩ in context sensitive brackets.

### 3.8.6 特殊命令 Special Commands

The following special commands are also provided by the core of Biblatex.

\nocite{⟨*key*⟩}
\nocite{*}

This command is similar to the standard LaTeX \nocite command. It adds the ⟨*key*⟩ to the bibliography without printing a citation. If the ⟨*key*⟩ is an asterisk, all entries available in the bib file are added to the bibliography. Like all other citation commands, \nocite commands in the document body are local to the enclosing refsection environment, if any. In contrast to standard LaTeX, \nocite may also be used in the document preamble. In this case, the references are assigned to reference section 0.

\fullcite[⟨*prenote*⟩][⟨*postnote*⟩]{⟨*key*⟩}

This command uses the bibliography driver for the respective entry type to create a full citation similar to the bibliography entry. It is thus related to the bibliography style rather than the citation style.

\footfullcite[⟨*prenote*⟩][⟨*postnote*⟩]{⟨*key*⟩}

Similar to \fullcite but puts the entire citation in a footnote and adds a period at the end.

\volcite[⟨*prenote*⟩]{⟨*volume*⟩}[⟨*page*⟩]{⟨*key*⟩}
\Volcite[⟨*prenote*⟩]{⟨*volume*⟩}[⟨*page*⟩]{⟨*key*⟩}

These commands are similar to \cite and \Cite but intended for references to multivolume works which are cited by volume and page number. Instead of the ⟨*postnote*⟩, they take a mandatory ⟨*volume*⟩ and an optional ⟨*page*⟩ argument. Since they merely compose the postnote and pass it to the \cite command provided by the citation style as a ⟨*postnote*⟩ argument, these commands are style independent. The format of the volume portion is controlled by the field formatting directive volcitevolume, the format of the page/text portion is controlled by the field formatting directive volcitepages (§ 4.10.4). The delimiter printed between the volume portion and the page/text portion may be modified by redefining the macro \volcitedelim (§ 4.10.1).

\volcites(⟨*multiprenote*⟩)(⟨*multipostnote*⟩)[⟨*prenote*⟩]{⟨*volume*⟩}[⟨*page*⟩]{⟨*key*⟩}
...[⟨*prenote*⟩]{⟨*volume*⟩}[⟨*page*⟩]{⟨*key*⟩}
\Volcites(⟨*multiprenote*⟩)(⟨*multipostnote*⟩)[⟨*prenote*⟩]{⟨*volume*⟩}[⟨*page*⟩]{⟨*key*⟩}
...[⟨*prenote*⟩]{⟨*volume*⟩}[⟨*page*⟩]{⟨*key*⟩}

The multicite version of \volcite and \Volcite, respectively.

\pvolcite[⟨prenote⟩]{⟨volume⟩}[⟨page⟩]{⟨key⟩}
\Pvolcite[⟨prenote⟩]{⟨volume⟩}[⟨page⟩]{⟨key⟩}

Similar to \volcite but based on \parencite.

\pvolcites(⟨multiprenote⟩)(⟨multipostnote⟩)[⟨prenote⟩]{⟨volume⟩}[⟨page⟩]{⟨key⟩}
...[⟨prenote⟩]{⟨volume⟩}[⟨page⟩]{⟨key⟩}
\Pvolcites(⟨multiprenote⟩)(⟨multipostnote⟩)[⟨prenote⟩]{⟨volume⟩}[⟨page⟩]{⟨key⟩}
...[⟨prenote⟩]{⟨volume⟩}[⟨page⟩]{⟨key⟩}

The multicite version of \pvolcite and \Pvolcite, respectively.

\fvolcite[⟨prenote⟩]{⟨volume⟩}[⟨page⟩]{⟨key⟩}
\ftvolcite[⟨prenote⟩]{⟨volume⟩}[⟨page⟩]{⟨key⟩}

Similar to \volcite but based on \footcite and \footcitetext, respectively.

\fvolcites(⟨multiprenote⟩)(⟨multipostnote⟩)[⟨prenote⟩]{⟨volume⟩}[⟨page⟩]{⟨key⟩}
...[⟨prenote⟩]{⟨volume⟩}[⟨page⟩]{⟨key⟩}
\Fvolcites(⟨multiprenote⟩)(⟨multipostnote⟩)[⟨prenote⟩]{⟨volume⟩}[⟨page⟩]{⟨key⟩}
...[⟨prenote⟩]{⟨volume⟩}[⟨page⟩]{⟨key⟩}

The multicite version of \fvolcite and \Fvolcite, respectively.

\svolcite[⟨prenote⟩]{⟨volume⟩}[⟨page⟩]{⟨key⟩}
\Svolcite[⟨prenote⟩]{⟨volume⟩}[⟨page⟩]{⟨key⟩}

Similar to \volcite but based on \smartcite.

\svolcites(⟨multiprenote⟩)(⟨multipostnote⟩)[⟨prenote⟩]{⟨volume⟩}[⟨page⟩]{⟨key⟩}
...[⟨prenote⟩]{⟨volume⟩}[⟨page⟩]{⟨key⟩}
\Svolcites(⟨multiprenote⟩)(⟨multipostnote⟩)[⟨prenote⟩]{⟨volume⟩}[⟨page⟩]{⟨key⟩}
...[⟨prenote⟩]{⟨volume⟩}[⟨page⟩]{⟨key⟩}

The multicite version of \svolcite and \Svolcite, respectively.

\tvolcite[⟨prenote⟩]{⟨volume⟩}[⟨page⟩]{⟨key⟩}
\Tvolcite[⟨prenote⟩]{⟨volume⟩}[⟨page⟩]{⟨key⟩}

Similar to \volcite but based on \textcite.

\tvolcites(⟨multiprenote⟩)(⟨multipostnote⟩)[⟨prenote⟩]{⟨volume⟩}[⟨page⟩]{⟨key⟩}
...[⟨prenote⟩]{⟨volume⟩}[⟨page⟩]{⟨key⟩}
\Tvolcites(⟨multiprenote⟩)(⟨multipostnote⟩)[⟨prenote⟩]{⟨volume⟩}[⟨page⟩]{⟨key⟩}
...[⟨prenote⟩]{⟨volume⟩}[⟨page⟩]{⟨key⟩}

The multicite version of \tvolcite and \Tvolcite, respectively.

\avolcite[⟨prenote⟩]{⟨volume⟩}[⟨page⟩]{⟨key⟩}
\Avolcite[⟨prenote⟩]{⟨volume⟩}[⟨page⟩]{⟨key⟩}

Similar to \volcite but based on \autocite.

\avolcites(⟨multiprenote⟩)(⟨multipostnote⟩)[⟨prenote⟩]{⟨volume⟩}[⟨page⟩]{⟨key⟩}
...[⟨prenote⟩]{⟨volume⟩}[⟨page⟩]{⟨key⟩}
\Avolcites(⟨multiprenote⟩)(⟨multipostnote⟩)[⟨prenote⟩]{⟨volume⟩}[⟨page⟩]{⟨key⟩}
...[⟨prenote⟩]{⟨volume⟩}[⟨page⟩]{⟨key⟩}

The multicite version of \avolcite and \Avolcite, respectively.

\notecite[⟨prenote⟩][⟨postnote⟩]{⟨key⟩}
\Notecite[⟨prenote⟩][⟨postnote⟩]{⟨key⟩}

These commands print the ⟨*prenote*⟩ and ⟨*postnote*⟩ arguments but no citation. Instead, a \nocite command is issued for every ⟨*key*⟩. This may be useful for authors who incorporate implicit citations in their writing, only giving information not mentioned before in the running text, but who still want to take advantage of the automatic ⟨*postnote*⟩ formatting and the implicit \nocite function. This is a generic, style-independent citation command. Special citation styles may provide smarter facilities for the same purpose. The capitalized version forces capitalization (note that this is only applicable if the note starts with a command which is sensitive to Biblatex's punctuation tracker).

\pnotecite[⟨prenote⟩][⟨postnote⟩]{⟨key⟩}
\Pnotecite[⟨prenote⟩][⟨postnote⟩]{⟨key⟩}

Similar to \notecite but the notes are printed in parentheses.

\fnotecite[⟨prenote⟩][⟨postnote⟩]{⟨key⟩}

Similar to \notecite but the notes are printed in a footnote.

### 3.8.7 底层命令 Low-level Commands

The following commands are also provided by the core of Biblatex. They grant access to all lists and fields at a lower level.

\citename[⟨prenote⟩][⟨postnote⟩]{⟨key⟩}[⟨format⟩]{⟨name list⟩}

The ⟨*format*⟩ is a formatting directive defined with \DeclareNameFormat. Formatting directives are discussed in § 4.4.2. If this optional argument is omitted, this command falls back to the format citename. The last argument is the name of a ⟨*name list*⟩, in the sense explained in § 2.2.

`\citelist`[⟨*prenote*⟩][⟨*postnote*⟩]{⟨*key*⟩}[⟨*format*⟩]{⟨*literal list*⟩}

The ⟨*format*⟩ is a formatting directive defined with `\DeclareListFormat`. Formatting directives are discussed in § 4.4.2. If this optional argument is omitted, this command falls back to the format `citelist`. The last argument is the name of a ⟨*literal list*⟩, in the sense explained in § 2.2.

`\citefield`[⟨*prenote*⟩][⟨*postnote*⟩]{⟨*key*⟩}[⟨*format*⟩]{⟨*field*⟩}

The ⟨*format*⟩ is a formatting directive defined with `\DeclareFieldFormat`. Formatting directives are discussed in § 4.4.2. If this optional argument is omitted, this command falls back to the format `citefield`. The last argument is the name of a ⟨*field*⟩, in the sense explained in § 2.2.

### 3.8.8 其它命令 Miscellaneous Commands

The commands in this section are little helpers related to citations.

`\citereset` This command resets the citation style. This may be useful if the style replaces repeated citations with abbreviations like *ibidem*, *idem*, *op. cit.*, etc. and you want to force a full citation at the beginning of a new chapter, section, or some other location. The command executes a style specific initialization hook defined with the `\InitializeCitationStyle` command from § 4.3.1. It also resets the internal citation trackers of this package. The reset will affect the `\ifciteseen`, `\ifentryseen`, `\ifciteibid`, and `\ifciteidem` tests discussed in § 4.6.2. When used inside a `refsection` environment, the reset of the citation tracker is local to the current `refsection` environment. Also see the `citereset` package option in § 3.1.2.1.

`\citereset*` Similar to `\citereset` but only executes the style's initialization hook, without resetting the internal citation trackers.

`\mancite` Use this command to mark manually inserted citations if you mix automatically generated and manual citations. This is particularly useful if the citation style replaces repeated citations by an abbreviation like *ibidem* which may get ambiguous or misleading otherwise. Always use `\mancite` in the same context as the manual citation, e. g., if the citation is given in a footnote, include `\mancite` in the footnote. The `\mancite` command executes a style specific reset hook defined with the `\OnManualCitation` command from § 4.3.1. It also resets the internal 'ibidem' and 'idem' trackers of this package. The reset will affect the `\ifciteibid` and `\ifciteidem` tests discussed in § 4.6.2.

`\pno` This command forces a single page prefix in the ⟨*postnote*⟩ argument to a citation command. See § 3.13.3 for further details and usage instructions. Note that this command is only available locally in citations and the bibliography.

`\ppno` Similar to `\pno` but forces a range prefix. See § 3.13.3 for further details and usage instructions. Note that this command is only available locally in citations and the bibliography.

\nopp Similar to \pno but suppresses all prefixes. See § 3.13.3 for further details and usage instructions. Note that this command is only available locally in citations and the bibliography.

\psq In the ⟨*postnote*⟩ argument to a citation command, this command indicates a range of two pages where only the starting page is given. See § 3.13.3 for further details and usage instructions. The suffix printed is the localization string `sequens`, see § 4.9.2. The spacing inserted between the suffix and the page number may be modified by redefining the macro \sqspace. The default is an unbreakable interword space. Note that this command is only available locally in citations and the bibliography.

\psqq Similar to \psq but indicates an open-ended page range. See § 3.13.3 for further details and usage instructions. The suffix printed is the localization string `sequentes`, see § 4.9.2. This command is only available locally in citations and the bibliography.

\RN{⟨*integer*⟩}

This command prints an integer as an uppercase Roman numeral. The formatting applied to the numeral may be modified by redefining the macro \RNfont.

\Rn{⟨*integer*⟩}

Similar to \RN but prints a lowercase Roman numeral. The formatting applied to the numeral may be modified by redefining the macro \Rnfont.

### 3.8.9 与natbib兼容的命令 natbib Compatibility Commands

The natbib package option loads a natbib compatibility module. The module defines aliases for the citation commands provided by the natbib package. This includes aliases for the core citation commands \citet and \citep as well as the variants \citealt and \citealp. The starred variants of these commands, which print the full author list, are also supported. The \cite command, which is handled in a particular way by natbib, is not treated in a special way. The text commands (\citeauthor, \citeyear, etc.) are also supported, as are all commands which capitalize the name prefix (\Citet, \Citep, \Citeauthor, etc.). Aliasing with \defcitealias, \citetalias, and \citepalias is possible as well. Note that the compatibility commands will not emulate the citation format of the natbib package. They merely alias natbib's commands to functionally equivalent facilities of the Biblatex package. The citation format depends on the main citation style. However, the compatibility style will adapt \nameyeardelim to match the default style of the natbib package.

### 3.8.10 类似mcite给的标注命令

Biber only

The mcite package option loads a special citation module which provides mcite/mciteplus-like citation commands. Strictly speaking, what the module provides are wrappers for the commands of the main citation style. For example, the following command:

| Standard Command | mcite-like Command |
|---|---|
| \cite | \mcite |
| \Cite | \Mcite |
| \parencite | \mparencite |
| \Parencite | \Mparencite |
| \footcite | \mfootcite |
| \footcitetext | \mfootcitetext |
| \textcite | \mtextcite |
| \Textcite | \Mtextcite |
| \supercite | \msupercite |

**Table 9: mcite-like commands**

```
\mcite{key1,setA,*keyA1,*keyA2,*keyA3,key2,setB,*keyB1,*keyB2,*keyB3}
```

is essentially equivalent to this:

```
\defbibentryset{setA}{keyA1,keyA2,keyA3}%
\defbibentryset{setB}{keyB1,keyB2,keyB3}%
\cite{key1,setA,key2,setB}
```

The \mcite command will work with any style since the \cite backend command is controlled by the main citation style as usual. The mcite module provides wrappers for the standard commands in §§ 3.8.1 和 3.8.2. See 表 10 for an overview. Pre and postnotes as well as starred variants of all commands are also supported. The parameters will be passed to the backend command. For example:

```
\mcite*[pre][post]{setA,*keyA1,*keyA2,*keyA3}
```

will execute:

```
\defbibentryset{setA}{keyA1,keyA2,keyA3}%
\cite*[pre][post]{setA}
```

Note that the mcite module is not a compatibility module. It provides commands which are very similar but not identical in syntax and function to mcite's commands. When migrating from mcite/mciteplus to Biblatex, legacy files must be updated. With mcite, the first member of the citation group is also the identifier of the group as a whole. Borrowing an example from the mcite manual, this group:

```
\cite{glashow,*salam,*weinberg}
```

consists of three entries and the entry key of the first one also serves as identifier of the entire group. In contrast to that, a Biblatex entry set is an entity in its own right. Therefore, it requires a unique entry key which is assigned to the set as it is defined:

| Input | Output | Comment |
|---|---|---|
| `\mcite{set1,*glashow,*salam,*weinberg}` | [1] | Defining and citing the set |
| `\mcite{set1}` | [1] | Subsequent citation of the set |
| `\cite{set1}` | [1] | Regular `\cite` works as usual |
| `\mcite{set1,*glashow,*salam,*weinberg}` | [1] | Redundant, but permissible |
| `\mcite{glashow}` | [1a] | Citing a set member |
| `\cite{weinberg}` | [1c] | Regular `\cite` works as well |

**Table 10: `mcite`-like syntax (sample output with `style=numeric` and `subentry` option)**

```
\mcite{set1,*glashow,*salam,*weinberg}
```

Once defined, an entry set is handled like any regular entry in a `bib` file. When using one of the `numeric` styles which ship with `biblatex` and activating its `subentry` option, it is even possible to refer to set members. See 表 10 for some examples. Restating the original definition of the set is redundant, but permissible. In contrast to `mciteplus`, however, restating a part of the original definition is invalid. Use the entry key of the set instead.

## 3.9 本地化命令 Localization Commands

The Biblatex package provides translations for key terms such as 'edition' or 'volume' as well as definitions for language specific features such as the date format and ordinals. These definitions, which are loaded automatically, may be modified or extended in the document preamble or the configuration file with the commands introduced in this section.

`\DefineBibliographyStrings{`⟨*language*⟩`}{`⟨*definitions*⟩`}`

This command is used to define localization strings. The ⟨*language*⟩ must be a language name known to the `babel`/`polyglossia` packages, i. e., one of the identifiers listed in 表 3 on page 31. The ⟨*definitions*⟩ are ⟨*key*⟩=⟨*value*⟩ pairs which assign an expression to an identifier:

```
\DefineBibliographyStrings{american}{%
  bibliography  = {Bibliography},
  shorthands    = {Abbreviations},
  editor        = {editor},
  editors       = {editors},
}
```

A complete list of all keys supported by default is given is § 4.9.2. Note that all expressions should be capitalized as they usually are when used in the middle of a sentence. The Biblatex package will automatically capitalize the first word when required at the

beginning of a sentence. Expressions intended for use in headings should be capitalized in a way that is suitable for titling. In contrast to `\DeclareBibliographyStrings`, `\DefineBibliographyStrings` overrides both the full and the abbreviated version of the string. See § 4.9.1 for further details.

`\DefineBibliographyExtras{`⟨*language*⟩`}{`⟨*code*⟩`}`

This command is used to adapt language specific features such as the date format and ordinals. The ⟨*language*⟩ must be a language name known to the `babel`/`polyglossia` packages. The ⟨*code*⟩, which may be arbitrary LaTeX code, will usually consist of redefinitions of the formatting commands from § 3.10.2.

`\UndefineBibliographyExtras{`⟨*language*⟩`}{`⟨*code*⟩`}`

This command is used to restore the original definition of any commands modified with `\DefineBibliographyExtras`. If a redefined command is included in § 3.10.2, there is no need to restore its previous definition since these commands are adapted by all language modules anyway.

`\DefineHyphenationExceptions{`⟨*language*⟩`}{`⟨*text*⟩`}`

This is a LaTeX frontend to TeX's `\hyphenation` command which defines hyphenation exceptions. The ⟨*language*⟩ must be a language name known to the `babel`/`polyglossia` packages. The ⟨*text*⟩ is a whitespace-separated list of words. Hyphenation points are marked with a dash:

```
\DefineHyphenationExceptions{american}{%
  hy-phen-ation ex-cep-tion
}
```

`\NewBibliographyString{`⟨*key*⟩`}`

This command declares new localization strings, i. e., it initializes a new ⟨*key*⟩ to be used in the ⟨*definitions*⟩ of `\DefineBibliographyStrings`. The ⟨*key*⟩ argument may also be a comma-separated list of key names. The keys listed in § 4.9.2 are defined by default.

## 3.10 格式命令 Formatting Commands

The commands and facilities presented in this section may be used to adapt the format of citations and the bibliography.

### 3.10.1 一般命令和钩子 Generic Commands and Hooks

The commands in this section may be redefined with `\renewcommand` in the document preamble. Note that all commands starting with `\mk`… take one argument. All of these commands are defined in `biblatex.def`.

`\bibsetup` Arbitrary code to be executed at the beginning of the bibliography, intended for commands which affect the layout of the bibliography.

`\bibfont` Arbitrary code setting the font used in the bibliography. This is very similar to `\bibsetup` but intended for switching fonts.

`\citesetup` Arbitrary code to be executed at the beginning of each citation command.

`\newblockpunct` The separator inserted between 'blocks' in the sense explained in § 4.7.1. The default definition is controlled by the package option `block` (see § 3.1.2.1).

`\newunitpunct` The separator inserted between 'units' in the sense explained in § 4.7.1. This will usually be a period or a comma plus an interword space. The default definition is a period and a space.

`\finentrypunct` The punctuation printed at the very end of every bibliography entry, usually a period. The default definition is a period.

`\entrysetpunct` The punctuation printed between bibliography subentries of an entry set. The default definition is a semicolon and a space. Biber only

`\bibnamedelima` This delimiter controls the spacing between the elements which make up a name part. It is inserted automatically after the first name element if the element is less than three characters long and before the last element. The default definition is an interword space penalized by the value of the `highnamepenalty` counter (§ 3.10.3). Please refer to § 3.13.4 for further details. Biber only

`\bibnamedelimb` This delimiter is inserted between the elements which make up a name part where `\bibnamedelima` does not apply. The default definition is an interword space penalized by the value of the `lownamepenalty` counter (§ 3.10.3). Please refer to § 3.13.4 for further details. Biber only

`\bibnamedelimc` This delimiter controls the spacing between name parts. It is inserted between the name prefix and the last name if `useprefix=true`. The default definition is an interword space penalized by the value of the `highnamepenalty` counter (§ 3.10.3). Please refer to § 3.13.4 for further details.

`\bibnamedelimd` This delimiter is inserted between all name parts where `\bibnamedelimc` does not apply. The default definition is an interword space penalized by the value of the `lownamepenalty` counter (§ 3.10.3). Please refer to § 3.13.4 for further details.

`\bibnamedelimi` This delimiter replaces `\bibnamedelima/b` after initials. Note that this only applies to initials given as such in the `bib` file, not to the initials automatically generated by Biblatex which use their own set of delimiters. Biber only

`\bibinitperiod` The punctuation inserted after initials unless `\bibinithyphendelim` applies. The default definition is a period (`\adddot`). Please refer to § 3.13.4 for further details. Biber only

\bibinitdelim The spacing inserted between multiple initials unless \bibinithyphendelim applies. The default definition is an unbreakable interword space. Please refer to § 3.13.4 for further details. Biber only

\bibinithyphendelim The punctuation inserted between the initials of hyphenated name parts, replacing \bibinitperiod and \bibinitdelim. The default definition is a period followed by an unbreakable hyphen. Please refer to § 3.13.4 for further details. Biber only

\bibindexnamedelima Replaces \bibnamedelima in the index.

\bibindexnamedelimb Replaces \bibnamedelimb in the index.

\bibindexnamedelimc Replaces \bibnamedelimc in the index.

\bibindexnamedelimd Replaces \bibnamedelimd in the index.

\bibindexnamedelimi Replaces \bibnamedelimi in the index.

\bibindexinitperiod Replaces \bibinitperiod in the index.

\bibindexinitdelim Replaces \bibinitdelim in the index.

\bibindexinithyphendelim Replaces \bibinithyphendelim in the index.

\revsdnamepunct The punctuation to be printed between the first and last name parts when a name is reversed. Here is an example showing a name with the default comma as \revsdnamedelim:

```
Jones, Edward
```

This command should be used with \bibnamedelimd as a reversed-name separator in formatting directives for name lists. Please refer to § 3.13.4 for further details.

\bibnamedash The dash to be used as a replacement for recurrent authors or editors in the bibliography. The default is an 'em' or an 'en' dash, depending on the indentation of the list of references.

\labelnamepunct The separator printed after the name used for alphabetizing in the bibliography (author or editor, if the author field is undefined). With the default styles, this separator replaces \newunitpunct at this location. The default definition is \newunitpunct, i. e., it is not handled differently from regular unit punctuation.

\subtitlepunct The separator printed between the fields title and subtitle, booktitle and booksubtitle, as well as maintitle and mainsubtitle. With the default styles, this separator replaces \newunitpunct at this location. The default definition is \newunitpunct, i. e., it is not handled differently from regular unit punctuation.

`\intitlepunct` The separator between the word "in" and the following title in entry types such as `@article`, `@inbook`, `@incollection`, etc. The default definition is a colon plus an interword space (e. g.,"Article, in: *Journal*" or "Title, in: *Book*"). Note that this is the separator string, not only the punctuation mark. If you don't want a colon after "in", `\intitlepunct` should still insert a space.

`\bibpagespunct` The separator printed before the `pages` field. The default is a comma plus an interword space.

`\bibpagerefpunct` The separator printed before the `pageref` field. The default is an interword space.

`\multinamedelim` The delimiter printed between multiple items in a name list like `author` or `editor` if there are more than two names in the list. The default is a comma plus an interword space. See `\finalnamedelim` for an example.[30]

`\finalnamedelim` The delimiter printed instead of `\multinamedelim` before the final name in a name list. The default is the localized term 'and', separated by interword spaces. Here is an example:

```
Michel Goossens, Frank Mittelbach and Alexander Samarin
Edward Jones and Joe Williams
```

The comma in the first example is the `\multinamedelim` whereas the string 'and' in both examples is the `\finalnamedelim`. See also `\finalandcomma` in § 3.10.2.

`\revsdnamedelim` An extra delimiter printed after the first name in a name list if the first name is reversed. The default is an empty string, i. e., no extra delimiter will be printed. Here is an example showing a name list with a comma as `\revsdnamedelim`:

```
Jones, Edward, and Joe Williams
```

In this example, the comma after 'Edward' is the `\revsdnamedelim` whereas the string 'and' is the `\finalnamedelim`, printed in addition to the former.

`\andothersdelim` The delimiter printed before the localization string '`andothers`' if a name list like `author` or `editor` is truncated. The default is an interword space.

`\multilistdelim` The delimiter printed between multiple items in a literal list like `publisher` or `location` if there are more than two items in the list. The default is a comma plus an interword space. See `\multinamedelim` for further explanation.

`\finallistdelim` The delimiter printed instead of `\multilistdelim` before the final item in a literal list. The default is the localized term 'and', separated by interword spaces. See `\finalnamedelim` for further explanation.

[30] Note that `\multinamedelim` is not used at all if there are only two names in the list. In this case, the default styles use the `\finalnamedelim`.

`\andmoredelim` The delimiter printed before the localization string 'andmore' if a literal list like publisher or location is truncated. The default is an interword space.

`\multicitedelim` The delimiter printed between citations if multiple entry keys are passed to a single citation command. The default is a semicolon plus an interword space.

`\supercitedelim` Similar to `\multicitedelim`, but used by the `\supercite` command only. The default is a comma.

`\compcitedelim` Similar to `\multicitedelim`, but used by certain citation styles when 'compressing' multiple citations. The default definition is a comma plus an interword space.

`\textcitedelim` Similar to `\multicitedelim`, but used by `\textcite` and related commands (§ 3.8.2). The default is a comma plus an interword space. The standard styles modify this provisional definition to ensure that the delimiter before the final citation is the localized term 'and', separated by interword spaces. See also `\finalandcomma` and `\finalandsemicolon` in § 3.10.2.

`\nametitledelim` The delimiter printed between the author/editor and the title by author-title and some verbose citation styles. The default definition is a comma plus an interword space.

`\nameyeardelim` The delimiter printed between the author/editor and the year by author-year citation styles. The default definition is an interword space.

`\labelalphaothers` A string to be appended to the non-numeric portion of the `labelalpha` field (i. e., the field holding the citation label used by alphabetic citation styles) if the number of authors/editors exceeds the `maxalphanames` threshold or the `author/editor` list was truncated in the `bib` file with the keyword '`and others`'. This will typically be a single character such as a plus sign or an asterisk. The default is a plus sign. This command may also be redefined to an empty string to disable this feature. In any case, it must be redefined in the preamble.

`\sortalphaothers` Similar to `\labelalphaothers` but used in the sorting process. Setting it to a different value is advisable if the latter contains formatting commands, for example: Biber only

```
\renewcommand*{\labelalphaothers}{\textbf{+}}
\renewcommand*{\sortalphaothers}{+}
```

If `\sortalphaothers` is not redefined, it defaults to `\labelalphaothers`.

`\prenotedelim` The delimiter printed after the ⟨*prenote*⟩ argument of a citation command. See § 3.8 for details. The default is an interword space.

`\postnotedelim` The delimiter printed before the ⟨*postnote*⟩ argument of a citation command. See § 3.8 for details. The default is a comma plus an interword space.

`\mkbibnamelast{`⟨*text*⟩`}` This command, which takes one argument, is used to format the last name of all authors, editors, translators, etc.

\mkbibnamefirst{⟨*text*⟩}Similar to \mkbibnamelast, but intended for the first name.

\mkbibnameprefix{⟨*text*⟩}Similar to \mkbibnamelast, but intended for the name prefix.

\mkbibnameaffix{⟨*text*⟩}Similar to \mkbibnamelast, but intended for the name affix.

\relatedpunct The separator between the relatedtype bibliography localization string and the data from the first related entry. Here is an example with \relatedpunct set to a dash:

```
A. Smith. Title. 2000, (Orig. pub. as-Origtitle)
```

\relateddelim The separator between the data of multiple related entries. The default definition is an optional dot plus linebreak. Here is an example where volumes A-E are related entries of the 5 volume main work:

```
Donald E. Knuth. Computers & Typesetting. 5 vols. Reading, Mass.: Addison
    ↪ -
Wesley, 1984-1986.
Vol. A: The TEXbook. 1984.
Vol. B: TEX: The Program. 1986.
Vol. C: The METAFONTbook. By. 1986.
Vol. D: METAFONT: The Program. 1986.
Vol. E: Computer Modern Typefaces. 1986.
```

### 3.10.2 语言相关命令 Language-specific Commands

The commands in this section are language specific. When redefining them, you need to wrap the new definition in a \DeclareBibliographyExtras command (in an .lbx file) or a \DefineBibliographyExtras command (user documents), see § 3.9 for details. Note that all commands starting with \mk… take one or more arguments.

\bibrangedash The language specific dash to be used for ranges of numbers.

\bibrangessep Biber only

The language specific separator to be used between multiple ranges.

\bibdatedash The language specific dash to be used for date ranges.

\mkbibdatelong Takes the names of three field as arguments which correspond to three date components (in the order year/month/day) and uses their values to print the date in the language specific long date format.

\mkbibdateshort Similar to \mkbibdatelong but using the language specific short date format.

\finalandcomma Prints the comma to be inserted before the final 'and' in a list, if applicable in the respective language. Here is an example:

```
Michel Goossens, Frank Mittelbach, and Alexander Samarin
```

\finalandcomma is the comma before the word 'and'. See also \multinamedelim, \finalnamedelim, \textcitedelim, and \revsdnamedelim in § 3.10.1.

\finalandsemicolon Prints the semicolon to be inserted before the final 'and' in a list of lists, if applicable in the respective language. Here is an example:

```
Goossens, Mittelbach, and Samarin; Bertram and Wenworth; and Knuth
```

\finalandsemicolon is the semicolon before the word 'and'. See also \textcitedelim in § 3.10.1.

\mkbibordinal{⟨*integer*⟩}

This command, which takes an integer as its argument, prints an ordinal number.

\mkbibmascord{⟨*integer*⟩}

Similar to \mkbibordinal, but prints a masculine ordinal, if applicable in the respective language.

\mkbibfemord{⟨*integer*⟩}

Similar to \mkbibordinal, but prints a feminine ordinal, if applicable in the respective language.

\mkbibneutord{⟨*integer*⟩}

Similar to \mkbibordinal, but prints a neuter ordinal, if applicable in the respective language.

\mkbibordedition{⟨*integer*⟩}

Similar to \mkbibordinal, but intended for use with the term 'edition'.

\mkbibordseries{⟨*integer*⟩}

Similar to \mkbibordinal, but intended for use with the term 'series'.

### 3.10.3 尺寸和计数器 Lengths and Counters

The length registers and counters in this section may be changed in the document preamble with \setlength and \setcounter, respectively.

\bibhang The hanging indentation of the bibliography, if applicable. This length is initialized to \parindent at load-time.

\biblabelsep The horizontal space between entries and their corresponding labels in the bibliography. This only applies to bibliography styles which print labels, such as the `numeric` and `alphabetic` styles. This length is initialized to twice the value of `\labelsep` at load-time.

\bibitemsep The vertical space between the individual entries in the bibliography. This length is initialized to `\itemsep` at load-time. Note that `\bibitemsep`, `\bibnamesep`, and `\bibinitsep` obey the rules for `\addvspace`, that is, when vertical space introduced by any of these commands immediately follows on from space introduced by another of them, the resulting total space is equal to the largest of them.

\bibnamesep Vertical space to be inserted between two entries in the bibliography whenever an entry starts with a name which is different from the initial name of the previous entry. The default value is zero. Setting this length to a positive value greater than `\bibitemsep` will group the bibliography by author/editor name. Note that `\bibitemsep`, `\bibnamesep`, and `\bibinitsep` obey the rules for `\addvspace`, that is, when vertical space introduced by any of these commands immediately follows on from space introduced by another of them, the resulting total space is equal to the largest of them.

\bibinitsep Vertical space to be inserted between two entries in the bibliography whenever an entry starts with a letter which is different from the initial letter of the previous entry. The default value is zero. Setting this length to a positive value greater than `\bibitemsep` will group the bibliography alphabetically. Note that `\bibitemsep`, `\bibnamesep`, and `\bibinitsep` obey the rules for `\addvspace`, that is, when vertical space introduced by any of these commands immediately follows on from space introduced by another of them, the resulting total space is equal to the largest of them.

\bibparsep The vertical space between paragraphs within an entry in the bibliography. The default value is zero.

abbrvpenalty This counter, which is used by the localization modules, holds the penalty used in short or abbreviated localization strings. For example, a linebreak in expressions such as "et al." or "ed. by" is unfortunate, but should still be possible to prevent overfull boxes. This counter is initialized to `\hyphenpenalty` at load-time. The idea is making TeX treat the whole expression as if it were a single, hyphenatable word as far as line-breaking is concerned. If you dislike such linebreaks, use a higher value. If you do not mind them at all, set this counter to zero. If you want to suppress them unconditionally, set it to 'infinite' (10 000 or higher).[31]

[31] The default values assigned to `abbrvpenalty`, `lownamepenalty`, and `highnamepenalty` are deliberately very low to prevent overfull boxes. This implies that you will hardly notice any effect on line-breaking if the text is set justified. If you set these counters to 10 000 to suppress the respective breakpoints, you will notice their effect but you may also be confronted with overfull boxes. Keep in mind that line-breaking in the bibliography is often more difficult than in the body text and that you can not resort to rephrasing a sentence. In some cases it may be preferable to set the entire bibliography `\raggedright` to prevent suboptimal linebreaks. In this case, even the fairly low default penalties will make a visible difference.

highnamepenalty This counter holds a penalty affecting line-breaking in names. Please refer to §§ 3.13.4 和 3.10.1 for explanation. The counter is initialized to `\hyphenpenalty` at load-time. Use a higher value if you dislike the respective linebreaks. If you do not mind them at all, set this counter to zero. If you prefer the traditional BibTeX behavior (no linebreaks at `highnamepenalty` breakpoints), set it to 'infinite' (10 000 or higher).

lownamepenalty Similar to `highnamepenalty`. Please refer to §§ 3.13.4 和 3.10.1 for explanation. The counter is initialized to half the `\hyphenpenalty` at load-time. Use a higher value if you dislike the respective linebreaks. If you do not mind them at all, set this counter to zero.

### 3.10.4 多用途命令 All-purpose Commands

The commands in this section are all-purpose text commands which are generally available, not only in citations and the bibliography.

\bibellipsis An ellipsis symbol with brackets: '[...]'.

\noligature Disables ligatures at this position and adds some space. Use this command to break up standard ligatures like 'fi' and 'fl'. It is similar to the `"|` shorthand provided by some language modules of the `babel`/`polyglossia` packages.

\hyphenate A conditional hyphen. In contrast to the standard `\-` command, this one allows hyphenation in the rest of the word. It is similar to the `"-` shorthand provided by some language modules of the `babel`/`polyglossia` packages.

\hyphen An explicit, breakable hyphen intended for compound words. In contrast to a literal '-', this command allows hyphenation in the rest of the word. It is similar to the `"=` shorthand provided by some language modules of the `babel`/`polyglossia` packages.

\nbhyphen An explicit, non-breakable hyphen intended for compound words. In contrast to a literal '-', this command does not permit line breaks at the hyphen but still allows hyphenation in the rest of the word. It is similar to the `"~` shorthand provided by some language modules of the `babel`/`polyglossia` packages.

\nohyphenation A generic switch which suppresses hyphenation locally. Its scope should normally be confined to a group.

\textnohyphenation{⟨*text*⟩}

Similar to `\nohyphenation` but restricted to the ⟨*text*⟩ argument.

\mknumalph{⟨*integer*⟩}

Takes an integer in the range 1–702 as its argument and converts it to a string as follows: 1=a, ..., 26=z, 27=aa, ..., 702=zz. This is intended for use in formatting directives for the `extrayear` and `extraalpha` fields.

`\mkbibacro{`⟨*text*⟩`}`

Generic command which typesets an acronym using the small caps variant of the current font, if available, and as-is otherwise. The acronym should be given in uppercase letters.

`\autocap{`⟨*character*⟩`}`

Automatically converts the ⟨*character*⟩ to its uppercase form if Biblatex's punctuation tracker would capitalize a localization string at the current location. This command is robust. It is useful for conditional capitalization of certain strings in an entry. Note that the ⟨*character*⟩ argument is a single character given in lowercase. For example:

```
\autocap{s}pecial issue
```

will yield 'Special issue' or 'special issue', as appropriate. If the string to be capitalized starts with an inflected character given in Ascii notation, include the accent command in the ⟨*character*⟩ argument as follows:

```
\autocap{\'e}dition sp\'eciale
```

This will yield 'Édition spéciale' or 'édition spéciale'. If the string to be capitalized starts with a command which prints a character, such as `\ae` or `\oe`, simply put the command in the ⟨*character*⟩ argument:

```
\autocap{\oe}uvres
```

This will yield 'Œuvres' or 'œuvres'.

## 3.11 语言相关注意点 Language-specific Notes

The facilities discussed in this section are specific to certain localization modules.

### 3.11.1 美语 American

The American localization module uses `\uspunctuation` from § 4.7.5 to enable 'American-style' punctuation. If this feature is enabled, all trailing commas and periods after `\mkbibquote` will be moved inside the quotes. If you want to disable this feature, use `\stdpunctuation` as follows:

```
\DefineBibliographyExtras{american}{%
  \stdpunctuation
}
```

By default, the 'American punctuation' feature is enabled by the `american` localization module only. The above code is only required if you want American localization without American punctuation. Since standard punctuation is the package default, it would be redundant with any other language.

It is highly advisable to always specify `american`, `british`, `australian`, etc. rather than `english` when loading the `babel`/`polyglossia` packages to avoid any possible confusion. Older versions of the `babel` package used to treat `english` as an alias for `british`; more recent ones treat it as an alias for `american`. The biblatex package essentially treats `english` as an alias for `american`, except for the above feature which is only enabled if `american` is requested explicitly.

### 3.11.2 西班牙语 Spanish

Handling the word 'and' is more difficult in Spanish than in the other languages supported by this package because it may be 'y' or 'e', depending on the initial sound of the following word. Therefore, the Spanish localization module does not use the localization string '`and`' but a special internal 'smart and' command. The behavior of this command is controlled by the `smartand` counter.

`smartand` This counter controls the behavior of the internal 'smart and' command. When set to 1, it prints 'y' or 'e', depending on the context. When set to 2, it always prints 'y'. When set to 3, it always prints 'e'. When set to 0, the 'smart and' feature is disabled. This counter is initialized to 1 at load-time and may be changed in the preamble. Note that setting this counter to a positive value implies that the Spanish localization module ignores `\finalnamedelim` and `\finallistdelim`.

`\forceE` Use this command in `bib` files if Biblatex gets the 'and' before a certain name wrong. As its name suggests, it will enforce 'e'. This command must be used in a special way to prevent confusing BibTeX. Here is an example:

```
author = {Edward Jones and Eoin Maguire},
author = {Edward Jones and {\forceE{E}}oin Maguire},
```

Note that the initial letter of the respective name component is given as an argument to `\forceE` and that the entire construct is wrapped in an additional pair of curly braces.

`\forceY` Similar to `\forceE` but enforces 'y'.

### 3.11.3 希腊语 Greek

The Greek localization module requires UTF-8 support. It will not work with any other encoding. Generally speaking, the Biblatex package is compatible with the `inputenc` package and with XeLaTeX. The `ucs` package will not work. Since `inputenc`'s standard `utf8` module is missing glyph mappings for Greek, this leaves Greek users with XeLaTeX. Note that you may need to load additional packages which set up Greek

fonts. As a rule of thumb, a setup which works for regular Greek documents should also work with Biblatex. However, there is one fundamental limitation. As of this writing, Biblatex has no support for switching scripts. Greek titles in the bibliography should work fine, provided that you use Biber as a backend, but English and other titles in the bibliography may be rendered in Greek letters. If you need multi-script bibliographies, using XeLaTeX is the only sensible choice.

#### 3.11.4 俄语 Russian

Like the Greek localization module, the Russian module also requires UTF-8 support. It will not work with any other encoding.

### 3.12 用法注意点 Usage Notes

The following sections give a basic overview of the Biblatex package and discuss some typical usage scenarios.

#### 3.12.1 概述 Overview

Using the Biblatex package is slightly different from using traditional BibTeX styles and related packages. Before we get to specific usage scenarios, we will therefore have a look at the structure of a typical document first:

```
\documentclass{...}
\usepackage[...]{biblatex}
\addbibresource{bibfile.bib}
\begin{document}
\cite{...}
...
\printbibliography
\end{document}
```

With traditional BibTeX, the `\bibliography` command serves two purposes. It marks the location of the bibliography and it also specifies the `bib` file(s). The file extension is omitted. With Biblatex, resources are specified in the preamble with `\addbibresource` using the full name with `.bib` suffix. The bibliography is printed using the `\printbibliography` command which may be used multiple times (see § 3.7 for details). The document body may contain any number of citation commands (§ 3.8). Processing this example file requires that a certain procedure be followed. Suppose our example file is called `example.tex` and our bibliographic data is in `bibfile.bib`. The procedure, then, is as follows:

##### 3.12.1.1 Biber

1. Run `latex` on `example.tex`. If the file contains any citations, Biblatex will request the respective data from Biber by writing commands to the auxiliary file `example.bcf`.

2. Run `biber` on `example.bcf`. Biber will retrieve the data from `bibfile.bib` and write it to the auxiliary file `example.bbl` in a format which can be processed by Biblatex.

3. Run `latex` on `example.tex`. Biblatex will read the data from `example.bbl` and print all citations as well as the bibliography.

#### 3.12.1.2 BibTeX

1. Run `latex` on `example.tex`. If the file contains any citations, Biblatex will request the respective data from BibTeX by writing commands to the auxiliary file `example.aux`.

2. Run `bibtex` on `example.aux`. BibTeX will retrieve the data from `bibfile.bib` and write it to the auxiliary file `example.bbl` in a format which can be processed by Biblatex.

3. Run `latex` on `example.tex`. Biblatex will read the data from `example.bbl` and print all citations as well as the bibliography.

Whenever a reference to a work which has not been cited before is added, this procedure must be repeated. This is also the case if the last reference to a work which has been cited before is removed because some citation labels may change in this case. In contrast to traditional BibTeX, there is normally no need to run `latex` twice after running the backend as far as the handling of bibliographic data is concerned.[32]

BibTeX only

Note that when using BibTeX as the backend this only applies to the most basic case. Using the `xref` field or the `entryset` field may require an additional LaTeX/BibTeX/LaTeX cycle. Some other facilities provided by Biblatex may also require an additional `latex` run to get certain references and the page tracking right. In this case, the usual warning messages such as "There were undefined references" and "Label(s) may have changed. Rerun to get cross-references right" will be printed.

### 3.12.2 辅助文件 Auxiliary Files

**3.12.2.1 Biber** The Biblatex package uses one auxiliary `bcf` file only. Even if there are citation commands in a file included via `\include`, you only need to run Biber on the main `bcf` file. All information Biber needs is in the `bcf` file, including information about all refsections if using multiple `refsection` environments (see § 3.12.3).

[32] That is, unless the `defernumbers` package option is enabled. See § 4.1

**3.12.2.2 BibTeX** By default, the Biblatex package uses the main aux file only. Even if there are citation commands in a file included via \include, which has its own aux file, you only need to run BibTeX on the main aux file. If you are using refsection environments in a document (see § 3.12.3) Biblatex will create one additional aux file for every refsection environment. In this case, you also need to run bibtex on each additional aux file. The name of the additional aux files is the base name of the main input file with the string -blx and a running number appended at the end. The Biblatex package issues a warning listing the files which require an additional BibTeX run. With the basic example presented in § 3.12.1, it would issue the following warning:

```
Package biblatex Warning: Please (re)run BibTeX on the file(s):
(biblatex)                example.aux
(biblatex)                and rerun LaTeX afterwards.
```

If the input file contained three refsection environments, the warning would read as follows:

```
Package biblatex Warning: Please (re)run BibTeX on the file(s):
(biblatex)                example1-blx.aux
(biblatex)                example2-blx.aux
(biblatex)                example3-blx.aux
(biblatex)                and rerun LaTeX afterwards.
```

Apart from these aux files, Biblatex uses an additional bib file with the same suffix to pass certain control parameters to BibTeX. In the example above, this file would be named example-blx.bib. In the event of a file name conflict, you can change the suffix by redefining the macro \blxauxsuffix in the document preamble. When using Biber, Biblatex writes a control file named example.bcf and ignores \blxauxsuffix. There is also no auxiliary bib file in this case.

Note that Biblatex will not overwrite any files it did not create. All auxiliary files created automatically by this package start with a special signature line. Before overwriting a file (excluding the main aux file, which is managed by LaTeX), Biblatex inspects the first line of the file to make sure that there is no file name conflict. If the file in question is missing the signature line, Biblatex will immediately issue an error message and abort before opening the output stream. In this case you should delete any spurious files accidentally left in the working directory. If the error persists, there may be a file name conflict with a file found in one of the TeX installation trees. Since the installation trees usually do not contain any aux files and the string -blx is fairly exotic in the name of a bib file, this is rather unlikely but theoretically possible. If you find out that this is indeed the case, you should redefine \blxauxsuffix permanently in the Biblatex configuration file, biblatex.cfg.

### 3.12.3 多个参考文献表 Multiple Bibliographies

In a collection of articles by different authors, such as a conference proceedings volume for example, it is very common to have one bibliography for each article rather than a global one for the entire book. In the example below, each article would be presented as a separate `\chapter` with its own bibliography.

Note that with the BibTeX backend, Biblatex creates one additional `aux` file for every `refsection` environment. These files have to be processed by BibTeX as well, see § 3.12.2 for details. BibTeX only

```
\documentclass{...}
\usepackage{biblatex}
\addbibresource{...}
\begin{document}
\chapter{...}
\begin{refsection}
...
\printbibliography[heading=subbibliography]
\end{refsection}
\chapter{...}
\begin{refsection}
...
\printbibliography[heading=subbibliography]
\end{refsection}
\end{document}
```

If `\printbibliography` is used inside a `refsection` environment, it automatically restricts the scope of the list of references to the enclosing `refsection` environment. For a cumulative bibliography which is subdivided by chapter but printed at the end of the book, use the `section` option of `\printbibliography` to select a reference section, as shown in the next example.

```
\documentclass{...}
\usepackage{biblatex}
\defbibheading{subbibliography}{%
  \section*{References for Chapter \ref{refsection:\therefsection}}}
\addbibresource{...}
\begin{document}
\chapter{...}
\begin{refsection}
...
\end{refsection}
\chapter{...}
```

```
\begin{refsection}
...
\end{refsection}
\printbibheading
\printbibliography[section=1,heading=subbibliography]
\printbibliography[section=2,heading=subbibliography]
\end{document}
```

Note the definition of the bibliography heading in the above example. This is the definition taking care of the subheadings in the bibliography. The main heading is generated with a plain `\chapter` command in this case. The Biblatex package automatically sets a label at the beginning of every `refsection` environment, using the standard `\label` command. The identifier used is the string `refsection:` followed by the number of the respective `refsection` environment. The number of the current section is accessible via the `refsection` counter. When using the `section` option of `\printbibliography`, this counter is also set locally. This means that you may use the counter in heading definitions to print subheadings like "References for Chapter 3", as shown above. You could also use the title of the respective chapter as a subheading by loading the `nameref` package and using `\nameref` instead of `\ref`:

```
\usepackage{nameref}
\defbibheading{subbibliography}{%
  \section*{\nameref{refsection:\therefsection}}}
```

Since giving one `\printbibliography` command for each part of a subdivided bibliography is tedious, Biblatex provides a shorthand. The `\bibbysection` command automatically loops over all reference sections. This is equivalent to giving one `\printbibliography` command for every section but has the additional benefit of automatically skipping sections without references. In the example above, the bibliography would then be generated as follows:

```
\printbibheading
\bibbysection[heading=subbibliography]
```

When using a format with one cumulative bibliography subdivided by chapter (or any other document division) it may be more appropriate to use `refsegment` rather than `refsection` environments. The difference is that the `refsection` environment generates labels local to the environment while `refsegment` does not affect the generation of labels, hence they will be unique across the entire document. Note that when using BibTeX as the backend, `refsegment` environments do not require additional `aux` files. The next example could also be given in § 3.12.4 because, visually, it creates one global bibliography subdivided into multiple segments.

```
\documentclass{...}
\usepackage{biblatex}
\defbibheading{subbibliography}{%
  \section*{References for Chapter \ref{refsegment:
    ↪ \therefsection\therefsegment}}}
\addbibresource{...}
\begin{document}
\chapter{...}
\begin{refsegment}
...
\end{refsegment}
\chapter{...}
\begin{refsegment}
...
\end{refsegment}
\printbibheading
\printbibliography[segment=1,heading=subbibliography]
\printbibliography[segment=2,heading=subbibliography]
\end{document}
```

The use of `refsegment` is similar to `refsection` and there is also a corresponding `segment` option for `\printbibliography`. The Biblatex package automatically sets a label at the beginning of every `refsegment` environment using the string `refsegment:` followed by the number of the respective `refsegment` environment as an identifier. There is a matching `refsegment` counter which may be used in heading definitions, as shown above. As with reference sections, there is also a shorthand command which automatically loops over all reference segments:

```
\printbibheading
\bibbysegment[heading=subbibliography]
```

This is equivalent to giving one `\printbibliography` command for every segment in the current `refsection`.

### 3.12.4 子参考文献表 Subdivided Bibliographies

It is very common to subdivide a bibliography by certain criteria. For example, you may want to list printed and online resources separately or divide a bibliography into primary and secondary sources. The former case is straightforward because you can use the entry type as a criterion for the `type` and `nottype` filters of `\printbibliography`. The next example also demonstrates how to generate matching subheadings for the two parts of the bibliography.

```
\documentclass{...}
```

```
\usepackage{biblatex}
\addbibresource{...}
\begin{document}
...
\printbibheading
\printbibliography[nottype=online,heading=subbibliography,
                   title={Printed Sources}]
\printbibliography[type=online,heading=subbibliography,
                   title={Online Sources}]

\end{document}
```

You may also use more than two subdivisions:

```
\printbibliography[type=article,...]
\printbibliography[type=book,...]
\printbibliography[nottype=article,nottype=book,...]
```

It is even possible to give a chain of different types of filters:

```
\printbibliography[section=2,type=book,keyword=abc,notkeyword=xyz]
```

This would print all works cited in reference section 2 whose entry type is `@book` and whose `keywords` field includes the keyword 'abc' but not 'xyz'. When using bibliography filters in conjunction with a numeric style, see § 3.13.5. If you need complex filters with conditional expressions, use the `filter` option in conjunction with a custom filter defined with `\defbibfilter`. See § 3.7.9 for details on custom filters.

```
\documentclass{...}
\usepackage{biblatex}
\addbibresource{...}
\begin{document}
...
\printbibheading
\printbibliography[keyword=primary,heading=subbibliography,%
                   title={Primary Sources}]
\printbibliography[keyword=secondary,heading=subbibliography,%
                   title={Secondary Sources}]
\end{document}
```

Dividing a bibliography into primary and secondary sources is possible with a `keyword` filter, as shown in the above example. In this case, with only two subdivisions, it would be sufficient to use one keyword as filter criterion:

```
\printbibliography[keyword=primary,...]
\printbibliography[notkeyword=primary,...]
```

Since Biblatex has no way of knowing if an item in the bibliography is considered to be primary or secondary literature, we need to supply the bibliography filter with the required data by adding a `keywords` field to each entry in the `bib` file. These keywords may then be used as targets for the `keyword` and `notkeyword` filters, as shown above. It may be a good idea to add such keywords right away while building a `bib` file.

```
@Book{key,
  keywords      = {primary,some,other,keywords},
  ...
```

An alternative way of subdividing the list of references are bibliography categories. They differ from the keywords-based approach shown in the example above in that they work on the document level and do not require any changes to the `bib` file.

```
\documentclass{...}
\usepackage{biblatex}
\DeclareBibliographyCategory{primary}
\DeclareBibliographyCategory{secondary}
\addtocategory{primary}{key1,key3,key6}
\addtocategory{secondary}{key2,key4,key5}
\addbibresource{...}
\begin{document}
...
\printbibheading
\printbibliography[category=primary,heading=subbibliography,%
                   title={Primary Sources}]
\printbibliography[category=secondary,heading=subbibliography,%
                   title={Secondary Sources}]
\end{document}
```

In this case it would also be sufficient to use one category only:

```
\printbibliography[category=primary,...]
\printbibliography[notcategory=primary,...]
```

It is still a good idea to declare all categories used in the bibliography explicitly because there is a `\bibbycategory` command which automatically loops over all categories. This is equivalent to giving one `\printbibliography` command for every category, in the order in which they were declared.

```
\documentclass{...}
\usepackage{biblatex}
\DeclareBibliographyCategory{primary}
\DeclareBibliographyCategory{secondary}
\addtocategory{primary}{key1,key3,key6}
\addtocategory{secondary}{key2,key4,key5}
\defbibheading{primary}{\section*{Primary Sources}}
\defbibheading{secondary}{\section*{Secondary Sources}}
\addbibresource{...}
\begin{document}
...
\printbibheading
\bibbycategory
\end{document}
```

The handling of the headings is different from `\bibbysection` and `\bibbysegment` in this case. `\bibbycategory` uses the name of the current category as a heading name. This is equivalent to passing `heading=`⟨*category*⟩ to `\printbibliography` and implies that you need to provide a matching heading for every category.

### 3.12.5 条目集 Entry Sets

An entry set is a group of entries which are cited as a single reference and listed as a single item in the bibliography. The individual entries in the set are separated by `\entrysetpunct` (§ 4.10.1). The Biblatex package supports two types of entry sets. Static entry sets are defined in the `bib` file like any other entry. Dynamic entry sets are defined with `\defbibentryset` (§ 3.7.11) on a per-document/per-refsection basis in the document preamble or the document body. This section deals with the definition of entry sets; style authors should also see § 4.11.1 for further information.

**3.12.5.1 Static entry sets** Static entry sets are defined in the `bib` file like any other entry. When using Biber as the backend, defining an entry set is as simple as adding an entry of type `@set`. The entry has an `entryset` field defining the members of the set as a separated list of entry keys:

Biber only

```
@Set{set1,
  entryset = {key1,key2,key3},
}
```

Entries may be part of a set in one document/refsection and stand-alone references in another one, depending on the presence of the `@set` entry. If the `@set` entry is cited, the set members are grouped automatically. If not, they will work like any regular entry.

When using BibTeX as the backend, which has no native support for entry sets, setting up entry sets involves more work. BibTeX requires `entryset` and `crossref`

fields to be used in a special way. The members of the set are given in the `entryset` field of the `@set` entry. The `@set` entry also requires a `crossref` field which points to the first key in the `entryset` field. In addition to that, all members of the set require `entryset` fields which are reverse pointers to the entry key of the `@set` head entry:

```
@Set{set1,
  entryset = {key1,key2,key3},
  crossref = {key1},
}
@Article{key1,
  entryset = {set1},
  author   = {...},
  title    = {...},
  ...
}
@InCollection{key2,
  entryset = {set1},
  author   = {...},
  title    = {...},
  ...
}
@Article{key3,
  entryset = {set1},
  author   = {...},
  title    = {...},
  ...
}
```

Note that citing any set member will automatically load the entire set with BibTeX. If you want to refer to an item as part of a set in one document/refsection and as a stand-alone reference in another one, you need two distinct entries with BibTeX.

#### 3.12.5.2 Dynamic entry sets

Biber only

Dynamic entry sets are set up and work much like static ones. The main difference is that they are defined in the document preamble or on the fly in the document body using the `\defbibentryset` command from § 3.7.11:

```
\defbibentryset{set1}{key1,key2,key3}
```

Dynamic entry sets in the document body are local to the enclosing `refsection` environment, if any. Otherwise, they are assigned to reference section 0. Those defined in the preamble are assigned to reference section 0. Note that dynamic entry sets require Biber. They will not work with any other backend.

### 3.12.6 数据容器 Data Containers

Biber only

The @xdata entry type serves as a data container holding one or more fields. These fields may be inherited by other entries using the xdata field. @xdata entries may not be cited or added to the bibliography, they only serve as a data source for other entries. This data inheritance mechanism is useful for fixed field combinations such as publisher/location and for other frequently used data:

```
@XData{hup,
  publisher  = {Harvard University Press},
  location   = {Cambridge, Mass.},
}
@Book{...,
  author     = {...},
  title      = {...},
  date       = {...},
  xdata      = {hup},
}
```

Using a separated list of keys in its xdata field, an entry may inherit data from several @xdata entries. Cascading @xdata entries are supported as well, i. e., an @xdata entry may reference one or more other @xdata entries:

```
@XData{macmillan:name,
  publisher  = {Macmillan},
}
@XData{macmillan:place,
  location   = {New York and London},
}
@XData{macmillan,
  xdata      = {macmillan:name,macmillan:place},
}
@Book{...,
  author     = {...},
  title      = {...},
  date       = {...},
  xdata      = {macmillan},
}
```

See also §§ 2.1.1 和 2.2.3.

### 3.12.7 电子出版信息 Electronic Publishing Information

The Biblatex package provides three fields for electronic publishing information: eprint, eprinttype, and eprintclass. The eprint field is a verbatim field similar to

doi which holds the identifier of the item. The eprinttype field holds the resource name, i. e., the name of the site or electronic archive. The optional eprintclass field is intended for additional information specific to the resource indicated by the eprinttype field. This could be a section, a path, classification information, etc. If the eprinttype field is available, the standard styles will use it as a literal label. In the following example, they would print "Resource: identifier" rather than the generic "eprint: identifier":

```
eprint     = {identifier},
eprinttype = {Resource},
```

The standard styles feature dedicated support for a few online archives. For arXiv references, put the identifier in the eprint field and the string arxiv in the eprinttype field:

```
eprint     = {math/0307200v3},
eprinttype = {arxiv},
```

For papers which use the new identifier scheme (April 2007 and later) add the primary classification in the eprintclass field:

```
eprint      = {1008.2849v1},
eprinttype  = {arxiv},
eprintclass = {cs.DS},
```

There are two aliases which ease the integration of arXiv entries. archiveprefix is treated as an alias for eprinttype; primaryclass is an alias for eprintclass. If hyperlinks are enabled, the eprint identifier will be transformed into a link to arxiv.org. See the package option arxiv in § 3.1.2.1 for further details.

For JSTOR references, put the stable JSTOR number in the eprint field and the string jstor in the eprinttype field:

```
eprint     = {number},
eprinttype = {jstor},
```

When using JSTOR's export feature to export citations in BibTeX format, JSTOR uses the url field by default (where the ⟨*number*⟩ is a unique and stable identifier):

```
url = {http://www.jstor.org/stable/number},
```

While this will work as expected, full URLs tend to clutter the bibliography. With the eprint fields, the standard styles will use the more readable "JSTOR: ⟨*number*⟩" format which also supports hyperlinks. The ⟨*number*⟩ becomes a clickable link if hyperref support is enabled.

For PubMed references, put the stable PubMed identifier in the eprint field and the string pubmed in the eprinttype field. This means that:

```
url = {http://www.ncbi.nlm.nih.gov/pubmed/pmid},
```

becomes:

```
eprint     = {pmid},
eprinttype = {pubmed},
```

and the standard styles will print "PMID: ⟨*pmid*⟩" instead of the lengthy URL. If hyperref support is enabled, the ⟨*pmid*⟩ will be a clickable link to PubMed.

For handles (HDLs), put the handle in the `eprint` field and the string `hdl` in the `eprinttype` field:

```
eprint     = {handle},
eprinttype = {hdl},
```

For Google Books references, put Google's identifier in the `eprint` field and the string `googlebooks` in the `eprinttype` field. This means that, for example:

```
url = {http://books.google.com/books?id=XXu4AkRVBBoC},
```

would become:

```
eprint     = {XXu4AkRVBBoC},
eprinttype = {googlebooks},
```

and the standard styles would print "Google Books: `XXu4AkRVBBoC`" instead of the full URL. If hyperref support is enabled, the identifier will be a clickable link to Google Books.[33]

Note that `eprint` is a verbatim field. Always give the identifier in its unmodified form. For example, there is no need to replace _ with _. Also see § 4.11.2 on how to add dedicated support for other eprint resources.

### 3.12.8 外部摘要和注释 External Abstracts and Annotations

Styles which print the fields `abstract` and/or `annotation` may support an alternative way of adding abstracts or annotations to the bibliography. Instead of including the text in the `bib` file, it may also be stored in an external LaTeX file. For example, instead of saying

```
@Article{key1,
  ...
  abstract        = {This is an abstract of entry ‘key1’.}
}
```

[33]Note that the Google Books ID seems to be a bit of an 'internal' value. As of this writing, there does not seem to be any way to search for an ID on Google Books. You may prefer to use the `url` in this case.

in the bib file, you may create a file named bibabstract-key1.tex and put the abstract in this file:

```
This is an abstract of entry ‘key1’.
\endinput
```

The name of the external file must be the entry key prefixed with bibabstract- or bibannotation-, respectively. You can change these prefixes by redefining \bibabstractprefix and \bibannotationprefix. Note that this feature needs to be enabled explicitly by setting the package option loadfiles from § 3.1.2.1. The option is disabled by default for performance reasons. Also note that any abstract and annotation fields in the bib file take precedence over the external files. Using external files is strongly recommended if you have long abstracts or a lot of annotations since this may increase memory requirements significantly. It is also more convenient to edit the text in a dedicated LaTeX file. Style authors should see § 4.11.3 for further information.

## 3.13 提示与警告 Hints and Caveats

This section provides additional usage hints and addresses some common problems and potential misconceptions.

### 3.13.1 与KOMA-Script 类共用的方法 Usage with KOMA-Script Classes

When using Biblatex in conjunction with one of the scrbook, scrreprt, or scrartcl classes, the headings bibliography and biblist from § 3.7.7 are responsive to the bibliography-related options of these classes.[34] You can override the default headings by using the heading option of \printbibliography, \printbibheading and \printbiblist. See §§ 3.7.2、3.7.3、3.7.7 for details. All default headings are adapted at load-time such that they blend with the behavior of these classes. If one of the above classes is detected, Biblatex will also provide the following additional tests which may be useful in custom heading definitions:

\ifkomabibtotoc{⟨*true*⟩}{⟨*false*⟩}

Expands to ⟨*true*⟩ if the class would add the bibliography to the table of contents, and to ⟨*false*⟩ otherwise.

\ifkomabibtotocnumbered{⟨*true*⟩}{⟨*false*⟩}

Expands to ⟨*true*⟩ if the class would add the bibliography to the table of contents as a numbered section, and to ⟨*false*⟩ otherwise. If this test yields ⟨*true*⟩, \ifkomabibtotoc will always yield ⟨*true*⟩ as well, but not vice versa.

[34] This applies to the traditional syntax of these options (bibtotoc and bibtotocnumbered) as well as to the ⟨*key*⟩=⟨*value*⟩ syntax introduced in KOMA-Script 3.x, i. e., to bibliography=nottotoc, bibliography=totoc, and bibliography=totocnumbered. The global toc=bibliography and toc=bibliographynumbered options as well as their aliases are detected as well. In any case, the options must be set globally in the optional argument to \documentclass.

### 3.13.2 与 Memoir 类共用的方法 Usage with the Memoir Class

When using Biblatex with the `memoir` class, most class facilities for adapting the bibliography have no effect. Use the corresponding facilities of this package instead (§§ 3.7.2、3.7.7、3.7.8). Instead of redefining `memoir`'s `\bibsection`, use the `heading` option of `\printbibliography` and `\defbibheading` (§§ 3.7.2 和 3.7.7). Instead of `\prebibhook` and `\postbibhook`, use the `prenote` and `postnote` options of `\printbibliography` and `\defbibnote` (§§ 3.7.2 和 3.7.8). All default headings are adapted at load-time such that they blend well with the default layout of this class. The default headings `bibliography` and `biblist` (§ 3.7.7) are also responsive to `memoir`'s `\bibintoc` and `\nobibintoc` switches. The length register `\bibitemsep` is used by Biblatex in a way similar to `memoir` (§ 3.10.3). This section also introduces some additional length registers which correspond to `memoir`'s `\biblistextra`. Lastly, `\setbiblabel` does not map to a single facility of the Biblatex package since the style of all labels in the bibliography is controlled by the bibliography style. See § 4.2.2 in the author section of this manual for details. If the `memoir` class is detected, Biblatex will also provide the following additional test which may be useful in custom heading definitions:

`\ifmemoirbibintoc{`⟨*true*⟩`}{`⟨*false*⟩`}`

Expands to ⟨*true*⟩ or ⟨*false*⟩, depending on `memoir`'s `\bibintoc` and `\nobibintoc` switches. This is a LaTeX frontend to `memoir`'s `\ifnobibintoc` test. Note that the logic of the test is reversed.

### 3.13.3 标注中的页码 Page Numbers in Citations

If the ⟨*postnote*⟩ argument to a citation command is a page number or page range, Biblatex will automatically prefix it with 'p.' or 'pp.' by default. This works reliably in typical cases, but sometimes manual intervention may be required. In this case, it is important to understand how this argument is handled in detail. First, Biblatex checks if the postnote is an Arabic or Roman numeral (case insensitive). If this test succeeds, the postnote is considered as a single page or other number which will be prefixed with 'p.' or some other string which depends on the `pagination` field (see § 2.3.10). If it fails, a second test is performed to find out if the postnote is a range or a list of Arabic or Roman numerals. If this test succeeds, the postnote will be prefixed with 'pp.' or some other string in the plural form. If it fails as well, the postnote is printed as is. Note that both tests expand the ⟨*postnote*⟩. All commands used in this argument must therefore be robust or prefixed with `\protect`. Here are a few examples of ⟨*postnote*⟩ arguments which will be correctly recognized as a single number, a range of numbers, or a list of numbers, respectively:

```
\cite[25]{key}
\cite[vii]{key}
\cite[XIV]{key}
\cite[34--38]{key}
```

```
\cite[iv--x]{key}
\cite[185/86]{key}
\cite[XI \& XV]{key}
\cite[3, 5, 7]{key}
\cite[vii--x; 5, 7]{key}
```

In some other cases, however, the tests may get it wrong and you need to resort to the auxiliary commands `\pno`, `\ppno`, and `\nopp` from § 3.8.8. For example, suppose a work is cited by a special pagination scheme consisting of numbers and letters. In this scheme, the string '27a' would mean 'page 27, part a'. Since this string does not look like a number or a range to Biblatex, you need to force the prefix for a single number manually:

```
\cite[\pno~27a]{key}
```

There is also a `\ppno` command which forces a range prefix as well as a `\nopp` command which suppresses all prefixes:

```
\cite[\ppno~27a--28c]{key}
\cite[\nopp 25]{key}
```

These commands may be used anywhere in the ⟨*postnote*⟩ argument. They may also be used multiple times. For example, when citing by volume and page number, you may want to suppress the prefix at the beginning of the postnote and add it in the middle of the string:

```
\cite[VII, \pno~5]{key}
\cite[VII, \pno~3, \ppno~40--45]{key}
\cite[see][\ppno~37--46, in particular \pno~40]{key}
```

There are also two auxiliary command for suffixes like 'the following page(s)'. Instead of inserting such suffixes literally (which would require `\ppno` to force a prefix):

```
\cite[\ppno~27~sq.]{key}
\cite[\ppno~55~sqq.]{key}
```

use the auxiliary commands `\psq` and `\psqq`. Note that there is no space between the number and the command. This space will be inserted automatically and may be modified by redefining the macro `\sqspace`.

```
\cite[27\psq]{key}
\cite[55\psqq]{key}
```

Since the postnote is printed without any prefix if it includes any character which is not an Arabic or Roman numeral, you may also type the prefix manually:

```
\cite[p.~5]{key}
```

It is possible to suppress the prefix on a per-entry basis by setting the `pagination` field of an entry to 'none', see § 2.3.10 for details. If you do not want any prefixes at all or prefer to type them manually, you can also disable the entire mechanism in the document preamble or the configuration file as follows:

```
\DeclareFieldFormat{postnote}{#1}
```

The ⟨*postnote*⟩ argument is handled as a field and the formatting of this field is controlled by a field formatting directive which may be freely redefined. The above definition will simply print the postnote as is. See §§ 4.3.2 和 4.4.2 in the author guide for further details.

### 3.13.4 姓名组成部分及其间距 Name Parts and Name Spacing

The Biblatex package gives users and style authors very fine-grained control of name spacing and the line-breaking behavior of names, especially when they are using Biber as the backend. The commands discussed in the following are documented in §§ 3.10.1 和 4.10.1. This section is meant to give an overview of how they are put together. A note on terminology: a name *part* is a basic part of the name, for example the first or the last name. Each part of a name may be a single name or it may be composed of multiple names. For example, the name part 'first name' may be composed of a first and a middle name. The latter are referred to as name *elements* in this section. Let's consider a simple name first: "John Edward Doe". This name is composed of the following parts:

| | |
|---|---|
| First | John Edward |
| Prefix | — |
| Last | Doe |
| Suffix | — |

The spacing, punctuation and line-breaking behavior of names is controlled by six macros:

| | |
|---|---|
| a=\bibnamedelima | Inserted by the backend after the first element of every name part if that element is less than three characters long and before the last element of every name part. |
| b=\bibnamedelimb | Inserted by the backend between all elements of a name part where \bibnamedelima does not apply. |
| c=\bibnamedelimc | Inserted by a formatting directive between the name prefix and the last name if useprefix=true. If useprefix=false, \bibnamedelimd is used instead. |
| d=\bibnamedelimd | Inserted by a formatting directive between name parts where \bibnamedelimc does not apply. |
| i=\bibnamedelimi | Replaces \bibnamedelima/b after initials |
| p=\revsdnamepunct | Inserted by a formatting directive after the last name when the name parts are reversed. |

This is how the delimiters are employed:

John$_{a}$ Edward$_{d}$ Doe

Doe$_{p}$,$_{d}$ John$_{a}$ Edward

Initials in the bib file get a special delimiter:

J.$_{i}$ Edward$_{d}$ Doe

Let's consider a more complex name: "Charles-Jean Étienne Gustave Nicolas de La Vallée Poussin". This name is composed of the following parts:

| | |
|---|---|
| First | Charles-Jean Étienne Gustave Nicolas |
| Prefix | de |
| Last | La Vallée Poussin |
| Suffix | — |

The delimiters:

Charles-Jean$_{b}$ Étienne$_{b}$ Gustave$_{a}$ Nicolas$_{d}$ de$_{c}$ La$_{a}$ Vallée$_{a}$ Poussin

Note that \bibnamedelima/b/i are inserted by the backend. The backend processes the name parts and takes care of the delimiters between the elements that make up a name part, processing each part individually. In contrast to that, the delimiters between the parts of the complete name (\bibnamedelimc/d) are added by name formatting directives at a later point in the processing chain. The spacing and punctuation of initials is also handled by the backend and may be customized by redefining the following three macros:

| | |
|---|---|
| a=\bibinitperiod | Inserted by the backend after initials. |
| b=\bibinitdelim | Inserted by the backend between multiple initials. |
| c=\bibinithyphendelim | Inserted by the backend between the initials of hyphenated name parts, replacing \bibinitperiod and \bibinitdelim. |

This is how they are employed:

J.[a][b] E.[a] Doe

K.-H.[c][a] Mustermann

### 3.13.5 参考文献筛选器和标注标签 Bibliography Filters and Citation Labels

The citation labels generated by this package are assigned to the full list of references before it is split up by any bibliography filters. They are guaranteed to be unique across the entire document (or a `refsection` environment), no matter how many bibliography filters you are using. When using a numeric citation scheme, however, this will most likely lead to discontinuous numbering in split bibliographies. Use the `defernumbers` package option to avoid this problem. If this option is enabled, numeric labels are assigned the first time an entry is printed in any bibliography.

### 3.13.6 参考文献标题中的活动字符 Active Characters in Bibliography Headings

Packages using active characters, such as `babel`, `polyglossia`, `csquotes`, or `underscore`, usually do not make them active until the beginning of the document body to avoid interference with other packages. A typical example of such an active character is the Ascii quote `"`, which is used by various language modules of the `babel`/`polyglossia` packages. If shorthands such as `"<` and `"a` are used in the argument to `\defbibheading` and the headings are defined in the document preamble, the non-active form of the characters is saved in the heading definition. When the heading is typeset, they do not function as a command but are simply printed literally. The most straightforward solution consists in moving `\defbibheading` after `\begin{document}`. Alternatively, you may use `babel`'s `\shorthandon` and `\shorthandoff` commands to temporarily make the shorthands active in the preamble. The above also applies to bibliography notes and the `\defbibnote` command.

### 3.13.7 在参考文献分节和分部中的编组 Grouping in Reference Sections and Segments

All LaTeX environments enclosed in `\begin` and `\end` form a group. This may have undesirable side effects if the environment contains anything that does not expect to be used within a group. This issue is not specific to `refsection` and `refsegment` environments, but it obviously applies to them as well. Since these environments will usually enclose much larger portions of the document than a typical `itemize` or similar environment, they are simply more likely to trigger problems related to grouping. If you observe any malfunctions after adding `refsection` environments to a document (for example, if anything seems to be 'trapped' inside the environment), try the following syntax instead:

```
\chapter{...}
```

```
\refsection
...
\endrefsection
```

This will not from a group, but otherwise works as usual. As far as Biblatex is concerned, it does not matter which syntax you use. The alternative syntax is also supported by the `refsegment` environment. Note that the commands `\newrefsection` and `\newrefsegment` do not form a group. See §§ 3.7.4 和 3.7.5 for details.

# 4 样式作者指南

本节内容是样式作者指南，主要介绍 Biblatex 包的接口。该指南囊括了设计参考文献著录和标注样式或者本地化模型所需知晓的所有内容。在阅读本部分内容前最好先阅读上一节的用户手册。

## 4.1 概述

在讨论 Biblatex 提供的命令和工具之前，我们首先介绍一些基本概念。Biblatex 包以一种特殊的方式使用辅助文件。最值得注意的是当使用 BibTeX 后端程序时，`bbl`文件的使用方式存在差别，即只有一个`bst`文件可用来实现结构化的数据接口，该文件并非用来输出可打印数据。

使用 LaTeX 的标准参考文献工具，一个文档通常包含任意数量的文献引用命令，以及常放在文档最后的`\bibliographystyle`和`\bibliography`命令。文献引用命令在文档中的位置是任意的，而`\bibliographystyle`和`\bibliography`命令则标记了打印参考文献表的位置，比如:

```
\documentclass{...}
\begin{document}
\cite{...}
...
\bibliographystyle{...}
\bibliography{...}
\end{document}
```

处理这些文件遵循一定的流程，其过程如下:

1. 运行 `latex`: 第一次运行`latex`，在 fileaux 文件中写入`\bibstyle`和 `\bibdata`命令，以及所有标注的`\citation`命令。这时，各引文标注[35]是未定义的，因为 LaTeX 等待 BibTeX 提供需要的数据，当然参考文献表也没生成。

2. 运行 `bibtex`:BibTeX 在`bbl`文件中写入一个`thebibliography`环境，用以提供`aux`文件中`\citation`命令所需求的所有条目，这些条目的数据来自`bib`文件。

[35]这里的 references 译为引文标注，指在引用命令导致在正文中出现的标注，这个标注由标签 label 构成。

3. 运行latex，第二次运行latex，thebibliography环境中的\bibitem命令在aux文件中为各参考文献条目写入\bibcite命令。这些\bibcite命令定义的标签将用于\cite命令。然而，各引文标注仍然未定义，因为这些标签在最后一次运行latex前仍未知。

4. 运行latex: 第三次运行，随着导言区最后读入了aux文件，引文标注的标签定义完成。这样所有的标注可以正确打印。

注意到所有的参考文献数据都以最终格式 (指最后打印出的格式) 写入bbl文件。该文件的读取和处理如同任何文档中的可打印章节。例如，考虑在一个bib文件中有如下条目:

```
@Book{companion,
  author    = {Michel Goossens and Frank Mittelbach and Alexander Samarin
    ↪ },
  title     = {The LaTeX Companion},
  publisher = {Addison-Wesley},
  address   = {Reading, Mass.},
  year      = {1994},
}
```

根据plain.bst 样式，BibTeX 在bbl文件中输出该条目如下:

```
\bibitem{companion}
Michel Goossens, Frank Mittelbach, and Alexander Samarin.
\newblock {\em The LaTeX Companion}.
\newblock Addison-Wesley, Reading, Mass., 1994.
```

默认情况下，LaTeX 生成顺序编码制标注标签，因此\bibitem命令在aux文件中写入的行如下所示:

```
\bibcite{companion}{1}
```

要实现一个不同的标注标签样式，意味着需要通过aux文件传递更多的数据。比如，当使用natbib包时，aux文件包含的标注 (或引用) 信息行，如下:

```
\bibcite{companion}{{1}{1994}{{Goossens et~al.}}{{Goossens, Mittelbach,
and Samarin}}}
```

Biblatex 包支持任何格式的标注标签，因此标注命令需要访问所有的参考文献数据。看一看同样需要在标注中提供所有参考文献数据的jurabib包的输出，我们将更清楚地理解这对上述处理过程的意义。

```
\bibcite{companion}{{Goossens\jbbfsasep Mittelbach\jbbstasep Samarin}%
```

```
{}{{0}{}{book}{1994}{}{}{}{}{Reading, Mass.\bpubaddr{}Addison-Wesley%
\bibbdsep{} 1994}}{{The LaTeX Companion}{}{}{2}{}{}{}{}{}}{\bibnf
{Goossens}{Michel}{M.}{}{}\Bibbfsasep\bibnf{Mittelbach}{Frank}{F.}%
{}{}\Bibbstasep\bibnf{Samarin}{Alexander}{A.}{}{}}{\bibtfont{The
LaTeX Companion}.\ \apyformat{Reading, Mass.\bpubaddr{}
Addison-Wesley\bibbdsep{} 1994}}}
```

在这种情况下，整个thebibliography环境的内容能通过aux文件有效传递。数据首先从bbl文件中读取出来，写入到aux中，然后再从aux读出保存到内存中。只有读入bbl文件，参考文献表才能生成。而 Biblatex 包将被迫通过aux文件回收所有的数据。这意味着处理过度且多余，因为不管怎么样数据都必须保存在内存中。

这种传统的处理过程都基于一个假设，即条目的完整数据只是参考文献表需要而所有的标注都使用短标签。这对于有内存限制的情况是非常有效的，但也意味着很难扩展。这就是 Biblatex 采取另一种方式的原因。首先，文档结构略有变化。取消在文档内使用\bibliography命令，数据库文件由导言区的\addbibresource命令指定，完全忽略\bibliographystyle命令 (所有的功能都将由包选项控制)，参考文献表使用\printbibliography命令打印:

```
\documentclass{...}
\usepackage[...]{biblatex}
\addbibresource{...}
\begin{document}
\cite{...}
...
\printbibliography
\end{document}
```

为了简化整个流程，Biblatex 基本上以应用aux文件的方式应用bbl文件，并舍弃了\bibcite命令。于是，我们得到如下流程:

1. 运行latex: 第一步类似于上述的传统方式:\bibstyle 和 \bibdata以及所有引用的\citation命令写入到aux文件中 (以 BibTeX 为后端程序) 或者写到bcf文件中 (以 Biber 为后端程序)。然后等待后端程序提供需要的数据。当以 BibTeX 为后端程序时，Biblatex 使用一个特殊 bst的文件，该文件用于实现 BibTeX 后端程序的数据接口，因此\bibstyle 命令则必须是\bibstyle{biblatex}。

2. 运行biber 或 bibtex: 后端程序提供了辅助文件中所有\citation命令所需的条目，这些条目的数据来自bib文件。然而，它并不在bbl文件中写出一个可打印的参考文献表，而是一个参考文献的结构化表达数据。类似于aux文件，读入该bbl文件时不打印任何东西，仅是将数据存入内存中。

3. 运行latex: 第二次运行，bbl文件在文档正文开始的时候处理，类似于aux文件。从这开始，所有参考文献数据都已在内存中，所以所有的引用都可以正

确打印。[36] 引用命令不仅可以访问预定义的标签，还可以访问完整的参考文献数据。参考文献表由内存中的相同数据生成，可以根据需要进行筛选和划分。

我们再次考虑上面给出的条目样例:

```
@Book{companion,
  author    = {Michel Goossens and Frank Mittelbach and Alexander Samarin
    ↪ },
  title     = {The LaTeX Companion},
  publisher = {Addison-Wesley},
  address   = {Reading, Mass.},
  year      = {1994},
}
```

使用 Biblatex 及 Biber 后端程序，这一条目实际上以如下格式输出:

```
\entry{companion}{book}{}
  \labelname{author}{3}{}{%
    {{uniquename=0,hash=...}{Goossens}{G.}{Michel}{M.}{}{}{}{}}%
    {{uniquename=0,hash=...}{Mittelbach}{M.}{Frank}{F.}{}{}{}{}}%
    {{uniquename=0,hash=...}{Samarin}{S.}{Alexander}{A.}{}{}{}{}}%
  }
  \name{author}{3}{}{%
    {{uniquename=0,hash=...}{Goossens}{G.}{Michel}{M.}{}{}{}{}}%
    {{uniquename=0,hash=...}{Mittelbach}{M.}{Frank}{F.}{}{}{}{}}%
    {{uniquename=0,hash=...}{Samarin}{S.}{Alexander}{A.}{}{}{}{}}%
  }
  \list{publisher}{1}{%
    {Addison-Wesley}%
  }
  \list{location}{1}{%
    {Reading, Mass.}%
  }
  \field{title}{The LaTeX Companion}
  \field{year}{1994}
\endentry
```

由这一例子可见，某种程度上说结构化的数据构成了bbl文件内容[37]。从这点上说，没有任何关于参考文献条目最终格式的决定。而参考文献表和引用标注的格式化由 LaTeX 宏控制，这些宏定义在参考文献和引用样式文件中。

[36] 如果defernumbers 包选项打开，Biblatex 以类似于传统过程的一种算法来生成顺序制标签。这种情况下，这些数字在参考文献表打印的时候指定且需从后端程序辅助文件中回收。因此需要额外运行一次 LaTeX 以在标注中获得它们。

[37] 这里应该是 bbl 文件而不是原文的 bib 文件

## 4.2 参考文献著录样式

一个参考文献著录样式是用于控制打印参考文献表中条目的宏的集合，定义在扩展名为bbx的文件中。Biblatex 包在其结尾加载所选择的参考文献样式文件。需要注意: 一些由多个标准样式文件共享的常用宏定义在biblatex.def 文件中。该文件同样在包结尾加载，但先于参考文献样式文件。

### 4.2.1 参考文献著录样式文件

在我们讨论参考文献著录样式的各部分之前，考虑一个典型的bbx文件总体结构，如下:

```
\ProvidesFile{example.bbx}[2006/03/15 v1.0 biblatex bibliography style]

\defbibenvironment{bibliography}
  {...}
  {...}
  {...}
\defbibenvironment{shorthand}
  {...}
  {...}
  {...}
\InitializeBibliographyStyle{...}
\DeclareBibliographyDriver{article}{...}
\DeclareBibliographyDriver{book}{...}
\DeclareBibliographyDriver{inbook}{...}
...
\DeclareBibliographyDriver{shorthand}{...}
\endinput
```

参考文献著录样式文件的主要结构包含如下命令:

\RequireBibliographyStyle{⟨*style*⟩}

该命令是可选的，用于引入一些建立在更一般的参考文献样式上的特殊样式。该命令加载样式文件style.bbx。

\InitializeBibliographyStyle{⟨*code*⟩}

该命令在参考文献表开始之前插入任意给定的⟨*code*⟩，但在参考文献表所形成的组内。该命令是可选的。它对于不同的参考文献驱动共享一些定义是有用的，但不能用于参考文献组外。记住，文档中可以有多个参考文献表，如果参考文献驱动进行了任何全局设置，应在下一个参考文献开始前重设[38]。

[38]这里不是很理解

`\DeclareBibliographyDriver{`⟨*entrytype*⟩`}{`⟨*code*⟩`}`

定义一个参考文献驱动。一个驱动 'driver' 是一个宏用于控制某一具体的参考文献条目 (当打印参考文献表的时候) 或者某一具体命名了的参考文献表 (当打印多个参考文献表的时候)。⟨*entrytype*⟩与bib文件中使用的条目类型对应，以小写字母给出 (见 § 2.1)。⟨*entrytype*⟩变量可以是一个星号。这种情况下，该驱动退化为没有具体驱动的条目类型。⟨*code*⟩是任意代码用于打印各自⟨*entrytype*⟩的参考文献条目。该命令是必须的。每个参考文献样式都应提供所用到的每类条目的驱动。

`\DeclareBibliographyAlias{`⟨*alias*⟩`}{`⟨*entrytype*⟩`}`

如果一个参考文献驱动用于处理多个参考文献条目类型，该命令可以用来定义某类已经定义驱动的⟨*entrytype*⟩别名。⟨*alias*⟩选项可以是一个星号，这种情况下，该驱动用于那些没有指定驱动的参考文献条目。

`\DeclareBibliographyOption[`⟨*datatype*⟩`]{`⟨*key*⟩`}[`⟨*value*⟩`]{`⟨*code*⟩`}`

该命令以⟨*key*⟩=⟨*value*⟩ 格式定义额外的导言区选项。⟨*key*⟩是选项键。⟨*code*⟩是当使用该选项时执行的任意 TeX 代码。键值作为`#1` 传递给⟨*code*⟩。可选的⟨*value*⟩是当该选项仅有键名而无键值给出时的默认键值。这对于布尔选项非常有用。⟨*datatype*⟩是选项的数据类型 (datatype)，如果缺省，那么默认为 'boolean'(布尔类型)，比如一个定义如下:

```
\DeclareBibliographyOption[boolean]{somekey}[true]{...}
```

给出'somekey' 而没有键值等价于'somekey=true'。有效的⟨*datatype*⟩值定义默认的 Biber 数据模型中，比如:

```
\DeclareDatamodelConstant[type=list]{optiondatatypes}{boolean,integer,
    ↪ string,xml}
```

`\DeclareEntryOption[`⟨*datatype*⟩`]{`⟨*key*⟩`}[`⟨*value*⟩`]{`⟨*code*⟩`}`

类似于`\DeclareBibliographyOption`，但用于定义 § 2.2.3节的`options`域中的选项，且仅基于 per-entry(条目) 进行设置。当 Biblatex 为标注命令和参考文献驱动准备数据时，执行⟨*code*⟩。

### 4.2.2 参考文献表环境

除了定义参考文献驱动，参考文献著录样式也要定义参考文献表环境用于控制参考文献表的输出。这些环境由命令`\defbibenvironment`名义。默认情况下，`\printbibliography`使用`bibliography`环境。下面是一个适用于不打印标签的参考文献表的环境定义:

```
\defbibenvironment{bibliography}
```

```
  {\list
     {}
     {\setlength{\leftmargin}{\bibhang}%
      \setlength{\itemindent}{-\leftmargin}%
      \setlength{\itemsep}{\bibitemsep}%
      \setlength{\parsep}{\bibparsep}}}
  {\endlist}
  {\item}
```

该定义使用 Biblatex 提供的\bibhang尺寸，应用了一个带悬挂缩进的list环境。它允许使用\bibitemsep 和 \bibparsep来实现一定程度的布局调整，Biblatex 提供的这两个尺寸就是为了该目的 (见 § 4.10.3)。作者年制 (authoryear) 和作者题名制 (authortitle) 的参考文献样式使用类似于该例的定义。

```
\defbibenvironment{bibliography}
  {\list
     {\printfield[labelnumberwidth]{labelnumber}}
     {\setlength{\labelwidth}{\labelnumberwidth}%
      \setlength{\leftmargin}{\labelwidth}%
      \setlength{\labelsep}{\biblabelsep}%
      \addtolength{\leftmargin}{\labelsep}%
      \setlength{\itemsep}{\bibitemsep}%
      \setlength{\parsep}{\bibparsep}}%
      \renewcommand*{\makelabel}[1]{\hss##1}}
  {\endlist}
  {\item}
```

一些参考文献样式在参考文献列表中打印标签。比如，设计一个顺序引用格式的参考文献样式需要在参考文献表的每个条目前面打印顺序数字，这样参考文献看起来就像一个顺序列表。在第一个例子中，\list命令的第一个参数是空的。在这个例子中，我们需要在其中插入数字，这些数字由 Biblatex 的labelnumber域中的数字提供。我们也应用 Biblatex 提供的几个尺寸和工具，详见 §§ 4.10.4 和 4.10.5。顺序制 (numeric) 参考文献样式使用如上的定义。除 labelnumber 由 labelalpha 代替和 labelnumberwidth 由 labelalphawidth 代替外，顺序字母制 (alphabetic) 的样式也是类似的。

各参考文献表以类似方式处理。\printbiblist命令默认使用以 bibliography list 命名的环境 (当使用 BibTeX 时，\printshorthands总是使用 shorthand 环境)。一个典型的例子如下，其中的尺寸和工具定义详见第 §§ 4.10.4 和 4.10.5节。

```
\defbibenvironment{shorthand}
  {\list
     {\printfield[shorthandwidth]{shorthand}}
     {\setlength{\labelwidth}{\shorthandwidth}%
```

```
    \setlength{\leftmargin}{\labelwidth}%
    \setlength{\labelsep}{\biblabelsep}%
    \addtolength{\leftmargin}{\labelsep}%
    \setlength{\itemsep}{\bibitemsep}%
    \setlength{\parsep}{\bibparsep}%
    \renewcommand*{\makelabel}[1]{##1\hss}}}
  {\endlist}
  {\item}
```

### 4.2.3 参考文献驱动

在我们讨论 Biblatex 包的数据接口命令前，了解一下参考文献驱动的结构是有益的。注意，虽然下面给出的例子是大为简化的，但仍具有说明价值。为可读性考虑，我们忽略了一些可能是`@book`条目的域，并且简化处理没有忽略的域。主要是为了说明驱动的结构。关于 BibTeX 文件的格式域与 Biblatex 包的数据类型的映射信息，见 § 2.2。

```
\DeclareBibliographyDriver{book}{%
  \printnames{author}%
  \newunit\newblock
  \printfield{title}%
  \newunit\newblock
  \printlist{publisher}%
  \newunit
  \printlist{location}%
  \newunit
  \printfield{year}%
  \finentry}
```

标准的参考文献样式应用两个参考文献宏 `begentry` 和 `finentry`。

```
\DeclareBibliographyDriver{entrytype}{%
  \usebibmacro{begentry}
  ...
  \usebibmacro{finentry}}
```

作为默认的定义。

```
\newbibmacro*{begentry}{}
\newbibmacro*{finentry}{\finentry}
```

推荐使用这两个宏，因为方便在驱动开始或结束时使用钩子。

回到上述给出 book 条目类型的驱动，我们发现有一些缺省: 即\printnames, \printlist, 和 \printfield命令所使用的格式命令。为了说明一个格式话指令是什么，这里给出上述驱动举例中所使用虚构指令。域的格式是直接的，域的值直接作为参数传递给格式命令，并根据需要格式化。下面的指令简单地将输入参数用一个\emph命令包裹:

```
\DeclareFieldFormat{title}{\emph{#1}}
```

列表格式则要复杂一些。在将列表划分为独立的项后，Biblatex 将对列表中的每一项执行格式化命令。各项作为参数传递给格式化命令。列表中各项间的分隔符由相应的命令控制，因此我们必须在插入分隔符前要检查是否在列表中或者是列表末尾。

```
\DeclareListFormat{location}{%
  #1%
  \ifthenelse{\value{listcount}<\value{liststop}}
    {\addcomma\space}
    {}}
```

姓名 (name) 的格式化指令类似于抄录列表。

依赖于数据模型常量‘nameparts’ 的姓名有如下默认定义:

```
\DeclareDatamodelConstant[type=list]{nameparts}
                                    {prefix,family,suffix,given}
```

这可以通过定制或者添加更多的姓名成分来处理比如来自父系姓的问题 (见文件93-nameparts.tex)。自然的，数据源需要一个扩展的姓名格式。biblatexml (§ D) 用来处理该问题，其中有一个扩展的姓名格式，可以处理自定义的姓名成分，当使用 Biber 后端的时候 (见 Biber 文档)。

在姓名格式中，姓名成分常量声明将为每个姓名成分提供数据模型定义的宏:

```
\namepart<namepart>
\namepart<namepart>i
```

姓名的格式化执行对姓名列表中的每一个姓名进行处理，看下面的例子:

```
\DeclareNameFormat{author}{%
  \ifthenelse{\value{listcount}=1}
    {\namepartfamily%
     \ifblank{\namepartgiven}{}{\addcomma\space\namepartgiven}}
    {\ifblank{\namepartgiven}{}{\namepartgiven\space}%
     \namepartfamily}%
  \ifthenelse{\value{listcount}<\value{liststop}}
```

```
    {\addcomma\space}
    {}}
```

上述各格式化命令调换了第一个作者的姓名前后顺序“Last, First”），而其余姓名则是常规顺序 (“First Last”)。注意: 必须要保证提供的姓名部分是姓 (last name)，因此我们必须要检查实际数据中姓名的哪些成分是存在的。如果姓名的一些成分不存在，则相关的变量就为空。如同抄录列表的命令，在各独立项之间插入的分隔符也由格式化命令控制，因为我们也要检查是否处于列表中还是在其末尾，这也是第二个`\ifthenelse`命令做的事情。

### 4.2.4 特殊域

下面的列表和域用于 Biblatex 给参考文献驱动和引用命令传递数据。它们由宏包自动定义，并不在`bib`文件中使用。从参考文献著录和标注样式的角度看，它们与`bib`文件中的域并没有什么不同。

#### 4.2.4.1 一般域

<datetype>dateunspecified 域 (string)

如果 <datetype>date 具有一个 EDTF 5.2.2 ‘unspecified’，该域将被设置为`yearindecade`, `yearincentury`, `monthinyear`, `dayinmonth`或`dayinyear`之一，这些字符串指定了 unspecified 信息的粒度。这些字符串可用于日期范围的判断，该日期范围自动为这些‘unspecified’ 日期创建，一个样式可能选择一种特殊方式来格式化日期。参见 § 2.3.8。例如: 一个条目的日期为:

```
@book{key,
  date     = {19uu},
  origdate = {199u}
}
```

将在`.bbl`产生如下信息:

```
@book{key,
  date     = {1900/1999},
  origdate = {1990/1999}
}
```

但也会额外的将域`dateunspecified`设置为‘yearincentury’，将`origdateunspecified`设置为‘yearindecade’。这一信息可以用来给`date`提供可能的信息‘20th century’，给`origdate`提供‘The 1990s’，这一信息无法单独从日期范围推算。因为这种自动生成的范围具有一个已知值，给出‘unspecified’ 元信息，因此使用该范围值来进行特殊的格式化相对容易。而标准样式不做此处理，`96-dates.tex`给出了一些例子。

entrykey 域 (string)

bib文件中某一项的条目关键词 (entry key)。这是一个字符串，用于 Biblatex 及其后端程序确定bib文件中的某一条目。

childentrykey 域 (string)

当引用一个条目集的子条目时，Biblatex 给引用数据提供了父@set条目的数据。这意味着entrykey表示的是父条目的关键词。而子条目的关键词在childentrykey域中提供。该域仅在引用一个条目集的某一子条目时使用。

labelnamesource 域 (literal)

保存给labelname提供信息的域的域名，由\DeclareLabelname确定。

labeltitlesource 域 (literal)

保存给labeltitle提供信息的域的域名，由\DeclareLabeltitle确定。

labeldatesource 域 (literal)

保存如下之一:

- 由\DeclareLabeldate选择的日期域域名的‘date’ 前的前缀。
- 一个域的域名。
- 一个抄录或本地化字符串。[39]

一般情况下保存由\DeclareLabeldate选择的日期域域名的‘date’ 前的前缀。例如，如果 labeldate 域是eventdate，那么labeldatesource就是‘event’。如果\DeclareLabeldate命令选择了date域，labeldatesource将会定义为一个空字符串作为‘date’ 的前缀，因为 date label 名中‘date’ 前为空。这就是说labeldatesource的内容可以用于构建对\DeclareLabeldate选择的域的指针。因为\DeclareLabeldate也可以选择抄录字符串作为备选，labeldatesource可以指向一个域或者不进行定义。记住:\DeclareLabeldate命令可以用于选择非日期域作为备选，所以labeldatesource可能包含一个域名。所以，总结起来，规则如下:

```
\iffieldundef{labeldatesource}
  {}% labeldate package option is not set
  {\iffieldundef{\thefield{labeldatesource}year}
    % \DeclareLabeldate resolved to either a literal/localisation
    % string or a non-date field since
    % if a date is defined by a date field, there is
    % at least a year
    {\iffieldundef{\thefield{labeldatesource}}
       {}% \DeclareLabeldate resolved to a literal/localisation string
```

[39] literal 译为抄录

```
      {}% \DeclareLabeldate resolved to a non-date field
    }
    {} % \DeclareLabeldate resolved a date field name prefix like "" or
    ↪ "orig"
  }
```

entrytype 域 (string)

条目类型 (@book, @inbook, 等)，以小写字母给出。

childentrytype 域 (string)

当引用一个条目集的子条目时，Biblatex 给引用命令提供父集条目的数据。这意味着entrytype保存父条目的类型。子条目的类型则由childentrytype域提供。该域仅在引用一个条目集的子条目时使用。

entrysetcount 域 (integer)

该域保存的整数用于指明一个集中某个集成员的位置 (起始值是 1)。该域仅对一个条目集的子条目有用。

hash 域 (string)

该域非常特殊，仅在姓名格式化命令中使用。它保存一个 hash 字符串，用于唯一的确定姓名列表中的单个姓名。该信息对于姓名列表中的所有姓名都有提供。另外参见namehash和fullhash。

namehash 域 (string)

一个 hash 字符串用于唯一确定labelname列表。这对再现检查很有用。比如，一个将再次出现的作者和编者用一个类似'idem' 的字符串代替的引用样式，可以用\savefield命令保存namehash域，并将其用于后面\iffieldequals(见 §§ 4.6.1 和 4.6.2) 命令的比较中。namehash域通过labelname列表的截短得到，即它的结果与maxnames和minnames选项相关。另外参见hash和fullhash。

<namelist>namehash 域 (string)

类似于namehash，但用于 'namelist' 姓名列表。

fullhash 域 (string)

一个 hash 字符串用于唯一确定labelname列表。该域域namehash有两点不同:1. 产生 hash 时忽略shortauthor和shorteditor列表。2. 该 hash 指的是完整的列表，忽略maxnames和minnames选项。另外见hash和namehash。

<namelist>fullhash 域 (string)

类似于fullhash,，但用于 'namelist' 姓名列表。

pageref 列表 (literal)

如果backref包选项打开，该域保存各被引用条目所在页的页码。如果文档中有refsection环境，反向引用是针对当前参考文献节的。

sortinit 域 (literal)

该域保存用于排序的信息首字符。使用 BibTeX 时，该域也用来代替sortinithash域。

sortinithash 域 (string)

使用 Biber 时，该域保存排序字符串的第一个扩展字素集群 (基本上是第一个字符) 的 Unicode 排序规则算法主要权重的 hash 值。当按照字母表顺序划分参考文献列表时很有用，该域有内部\bibinitsep所使用。(见 § 3.10.3)

clonesourcekey 域 (string)

该域保存复制条目源条目的关键词。复制条目常用于处理相关条目和related域。

#### 4.2.4.2 标注 (引用) 标签中使用的域

labelalpha 域 (literal)

当使用 BibTeX 为后端程序时，生成一个类似于传统 BibTeX 的alpha.bst样式的标签。这一默认标签由抽取labelname列表的首字母加上出版年的最后两个数字构成。label域可用来重写它的非数值部分 (non-numeric portion)。如果定义了label域，Biblatex 将使用它的值加上出版年的后两个数字生成labelalpha。shorthand域也可用来重写整个标签。如果定义了该域，labelalpha就是shorthand域，而不是一个自动生成的标签。使用 Biber 时，用户 Biber only
可以定义用来构建字母顺序标签的模板 (见 § 4.5.5)，而默认的模板域上面 bibtex 后端程序使用的格式相同。一个完整的字母顺序 ('alphabetic') 标签由以下域构成:

extraalpha 域 (integer)

当参考文献中包含同一作者同一年出版的多个引文时，'alphabetic' 引用格式常需要一个额外的字母加入标签来区分。这种情况下extraalpha域保存一个整数可用命令 \mknumalph 转换成字母或以其他方式格式化。该域类似于在作者年 (author-year) 格式中extrayear的作用。完整的 'alphabetic' 的标签由labelalpha 加extraalpha构成。注意包选项 labelalpha要求使用labelalpha和 extraalpha域 (详见 § 3.1.2.3)。另外参见 labelalpha和 § 3.10.1的\labelalphaothers。表8总结了不同的extra*非歧义计数器和他们追踪的信息。

labelname 列表 (name)

引用中打印的姓名。该列表可以是shortauthor, author, shorteditor, editor, 或translator域的复制值，正常情况以该顺序检测。如果没有作者 (authors) 和编者 (authors)，该列表时未定义的。注意该列表也与use<name>相关，见 § 3.1.3。引用样式打印引用中的姓名时使用这一列表。提供该列表仅为方便起见，没有附加的意义。使用 Biber 时，该域可以定制，详见 § 4.5.10。 Biber only

labelnumber 域 (literal)

参考文献条目的序号，用于顺序编码类的样式。如果定义了shorthand域，Biblatex 不再给各条目赋予一个数值。这种情况下，labelnumber就是 shorthand 而不是一个数字。顺序编码类的样式必须使用该域的值而不是一个计数器值。注意: 包选项labelnumber要求使用该域，详见 § 3.1.2.3。另可参见 § 3.1.2.1节的defernumbers选项。

labelprefix 域 (literal)

如果为了在一个 subbibliography 文献表的所有条目前都添加一个固定的字符串，设置了\newrefcontext命令的labelprefix选项，那么所有受影响的labelprefix域将提供该字符串。如果未设置前缀，相应条目的labelprefix域是未定义的。详见 § 3.7.10节\newrefcontext命令的labelprefix选项。如果定义了shorthand域，Biblatex 不会给相应条目的labelprefix域设置前缀。这种情况下labelprefix是未定义的。

labeltitle 域 (literal)

一篇文献可打印题名 (或标题)。在一些环境中，一个样式可能需要在一些可能的标题域中选择一个标题。例如，标注样式打印短标题可能需要打印shorttitle域，如果它存在的话，否则将打印title域。构建labeltitle时考虑的域的列表可以自定义。详见 § 4.5.10。注意:labeltitle包选项要求使用extratitle域，详见 § 3.1.2.3。另可参见extratitle。也要注意，labeltitleyear需要包选项需要extratitleyear域，另可参见extratitleyear。

extratitle 域 (integer)

该命令有时很有用，比如在 author-title 标注样式中，用于区别标题相同的文献。当有文献具有相同的labelname和labeltitle，extratitle域保存一个整数，可以利用\mknumalph转换为一个字母或者以其它方式格式化 (或者可以仅仅作为一个标志，用于表示将一些其它域比如日期与labeltitle域合并)。当文献表中具有相同labeltitle和labelname的文献只有一篇时，该域不定义。[40]注意:extratitle域是labeltitle包选项所要求使用，详见 § 3.1.2.3。另可参见labeltitle。8 总结了各种extra*计数器及其作用。

extratitleyear 域 (integer)

该命令有时很有用，比如在 author-title 标注样式中，用于区别标题相同年份相同但没有责任者的文献。当有文献具有相同的labeltitle和labelyear，extratitleyear域保存一个整数，可以利用\mknumalph转换为一个字母或者以其它方式格式化 (或者可以仅仅作为一个标志，用于表示将一些其它域比如出版者与labelyear域合并)。当文献表中具有相同labeltitle和labelyear的文献只有一篇时，该域不定义。注意: bibfieldextratitle 域是labeltitleyear包选项所要求使用，详见 § 3.1.2.3。另可参见labeltitleyear。8 总结了各种extra*计数器及其作用。

[40] there is only one work with the same labeltitle by the same labelname in the bibliography?

labelyear 域 (literal)

由\DeclareLabeldate(§ 4.5.10) 命令选择的日期域的年或者year域用于作者年制标签。一个完整的作者年标签由labelyear加extrayear域构成。注意labelyear和extrayear域是 labeldateparts包选项要求使用的，详见 § 3.1.2.3。另可参见extrayear。

labelendyear 域 (literal)

\DeclareLabeldate (§ 4.5.10) 命令选择的日期域的终止年，如果选择的日期是一个范围。

labelmonth 域 (datepart)

由\DeclareLabeldate(§ 4.5.10) 命令选择的日期域的月或者month域用于作者年制标签。注意labelmonth域是 labeldateparts包选项要求使用的，详见 § 3.1.2.3。

labelendmonth 域 (datepart)

\DeclareLabeldate (§ 4.5.10) 命令选择的日期域的终止月，如果选择的日期是一个范围。

labelday 域 (datepart)

由\DeclareLabeldate(§ 4.5.10) 命令选择的日期域的日或者month域用于作者年制标签。注意 labelday域是 labeldateparts包选项要求使用的，详见 § 3.1.2.3。

labelendday 域 (datepart)

\DeclareLabeldate (§ 4.5.10) 命令选择的日期域的终止日，如果选择的日期是一个范围。

extrayear 域 (integer)

当参考文献表中包含两个或更多的具有相同作者的文献且出版年份也相同时，author-year 标注样式常需要在年后面附加一个字母以示区别。这种情况下，extrayear域保存一个整数可以利用\mknumalph转换为一个字母或者以其它方式格式化。当文献表中某作者的文献只有一篇或者所有该作者的文献的出版年不同时，该域不定义。完整的作者年标签由labelyear加extrayear域构成。注意labelyear和extrayear域是labeldateparts包选项所需要使用的，详见 § 3.1.2.3。另可参见labelyear。8 总结了各种extra*计数器及其作用。

#### 4.2.4.3 Date 的成分域

注意，可以在数据模型中定义新的日期域，这些新定义的日期域的使用方式与本节将介绍的默认的数据模型类似。

bib文件中的日期域与央视接口提供的日期域如何关联详见表 11。当对样式中像origdate这样的域进行判断时，使用如下代码:

```
\iffieldundef{origyear}{...}{...}
```

它将告诉你相应的日期是否已定义。下面的判断:

```
\iffieldundef{origendyear}{...}{...}
```

将告诉你相应的日期和一个 (完全确定的) 范围是否已定义。下面的判断

```
\iffieldequalstr{origendyear}{}{...}{...}
```

将告诉你相应的日期和一个无终点的 (open-ended) 范围已经定义。Open-ended 范围由一个空的 `endyear` 成分表示 (而不是一个未定义的 `endyear` 成分)。更多例子详见 § 2.3.8节和41页的表 4。

| **bib File** | | **Data Interface** | |
|---|---|---|---|
| **Field** | **Value (Example)** | **Field** | **Value (Example)** |
| date | 1988 | day | undefined |
| | | month | undefined |
| | | year | 1988 |
| | | season | undefined |
| | | endday | undefined |
| | | endmonth | undefined |
| | | endyear | undefined |
| | | endseason | undefined |
| | | hour | undefined |
| | | minute | undefined |
| | | second | undefined |
| | | timezone | undefined |
| | | endhour | undefined |
| | | endminute | undefined |
| | | endsecond | undefined |
| | | endtimezone | undefined |
| date | 1997/ | day | undefined |
| | | month | undefined |
| | | year | 1997 |
| | | season | undefined |
| | | endday | undefined |
| | | endmonth | undefined |
| | | endyear | empty |
| | | endseason | undefined |
| | | hour | undefined |
| | | minute | undefined |
| | | second | undefined |
| | | timezone | undefined |
| | | endhour | undefined |
| | | endminute | undefined |
| | | endsecond | undefined |
| | | endtimezone | undefined |
| urldate | 2009-01-31 | urlday | 31 |
| | | urlmonth | 01 |
| | | urlyear | 2009 |
| | | urlseason | undefined |
| | | urlendday | undefined |
| | | urlendmonth | undefined |
| | | urlendyear | undefined |
| | | urlendseason | undefined |
| | | urlhour | undefined |

| | | | |
|---|---|---|---|
| | | urlminute | undefined |
| | | urlsecond | undefined |
| | | urltimezone | undefined |
| | | urlendhour | undefined |
| | | urlendminute | undefined |
| | | urlendsecond | undefined |
| | | urlendtimezone | undefined |
| urldate | 2009-01-31T15:34:04Z | urlday | 31 |
| | | urlmonth | 01 |
| | | urlyear | 2009 |
| | | urlseason | undefined |
| | | urlendday | undefined |
| | | urlendmonth | undefined |
| | | urlendyear | undefined |
| | | urlendseason | undefined |
| | | urlhour | 15 |
| | | urlminute | 34 |
| | | urlsecond | 04 |
| | | urltimezone | Z |
| | | urlendhour | undefined |
| | | urlendminute | undefined |
| | | urlendsecond | undefined |
| | | urlendtimezone | undefined |
| urldate | 2009-01-31T15:34:04+05:00 | urlday | 31 |
| | | urlmonth | 01 |
| | | urlyear | 2009 |
| | | urlseason | undefined |
| | | urlendday | undefined |
| | | urlendmonth | undefined |
| | | urlendyear | undefined |
| | | urlendseason | undefined |
| | | urlhour | 15 |
| | | urlminute | 34 |
| | | urlsecond | 04 |
| | | urltimezone | +0500 |
| | | urlendhour | undefined |
| | | urlendminute | undefined |
| | | urlendsecond | undefined |
| | | urlendtimezone | undefined |
| urldate | 2009-01-31T15:34:04/ | urlday | 31 |
| | 2009-01-31T16:04:34 | | |
| | | urlmonth | 1 |
| | | urlyear | 2009 |
| | | urlseason | undefined |
| | | urlendday | 31 |
| | | urlendmonth | 1 |
| | | urlendyear | 2009 |
| | | urlendseason | undefined |
| | | urlhour | 15 |
| | | urlminute | 34 |
| | | urlsecond | 4 |
| | | urltimezone | floating |
| | | urlendhour | 16 |
| | | urlendminute | 4 |
| | | urlendsecond | 34 |
| | | urlendtimezone | floating |
| origdate | 2002-21/2002-23 | origday | undefined |
| | | origmonth | 01 |
| | | origyear | 2002 |

| | | | |
|---|---|---|---|
| | | origseason | spring |
| | | origendday | undefined |
| | | origendmonth | 02 |
| | | origendyear | 2002 |
| | | origendseason | autumn |
| | | orighour | undefined |
| | | origminute | undefined |
| | | origsecond | undefined |
| | | origtimezone | undefined |
| | | origendhour | undefined |
| | | origendminute | undefined |
| | | origendsecond | undefined |
| | | origendtimezone | undefined |
| eventdate | 1995-01-31/1995-02-05 | eventday | 31 |
| | | eventmonth | 01 |
| | | eventyear | 1995 |
| | | eventseason | undefined |
| | | eventendday | 05 |
| | | eventendmonth | 02 |
| | | eventendyear | 1995 |
| | | eventendseason | undefined |
| | | eventhour | undefined |
| | | eventminute | undefined |
| | | eventsecond | undefined |
| | | eventtimezone | undefined |
| | | eventendhour | undefined |
| | | eventendminute | undefined |
| | | eventendsecond | undefined |
| | | eventendtimezone | undefined |

**Table 11: 日期接口 (注意:biblatex3.7 版提供的四个可解析日期接口，分别是 date，origdate，eventdate，urldate，在多数场合已经够用)**

hour 域 (datepart)

该域保存date域的小时 (hour) 成分，当日期是一个范围时，它保存开始日期的小时成分。

minute 域 (datepart)

该域保存date域的分钟成分，当日期是一个范围时，它保存开始日期的分钟成分。

second 域 (datepart)

该域保存date域的秒钟成分，当日期是一个范围时，它保存开始日期的秒钟成分。

timezone 域 (datepart)

该域保存date域的时区成分，当日期是一个范围时，它保存开始日期的时区成分。

day 域 (datepart)

该域保存date域的日成分，当日期是一个范围时，它保存开始日期的日成分。

month 域 (datepart)

该域保存数据源文件中的month域或者date域的月成分，当日期是一个范围时，它保存开始日期的月成分。

year 域 (datepart)

该域保存数据源文件中的year域或者date域的年成分，当日期是一个范围时，它保存开始日期的年成分。

season 域 (datepart)

该域保存由EDTF 5.2.5(见 § 2.3.8) 规定的date域的季节成分，它包含一个季节本地化字符串。当日期是一个范围时，它保存开始日期的季节成分。

endhour 域 (datepart)

如果date域中给出的日期是一个范围，该域保存结束日期的小时成分。

endminute 域 (datepart)

如果date域中给出的日期是一个范围，该域保存结束日期的分钟成分。

endsecond 域 (datepart)

如果date域中给出的日期是一个范围，该域保存结束日期的秒钟成分。

endtimezone 域 (datepart)

如果date域中给出的日期是一个范围，该域保存结束日期的时区成分。

endday 域 (datepart)

如果date域中给出的日期是一个范围，该域保存结束日期的日成分。

endmonth 域 (datepart)

如果date域中给出的日期是一个范围，该域保存结束日期的月成分。

endyear 域 (datepart)

如果date域中给出的日期是一个范围，该域保存结束日期的年成分。空的 (但已定义) 的endyear成分表示无终点的日期范围。

endseason 域 (datepart)

如果date域中给出的日期是一个范围，该域保存EDTF 5.2.5 (§ 2.3.8) 规定的结束日期的季节成分。它包含一个季节本地化字符串 (见 § 4.9.2.21)，空的 (但已定义) 的endseason成分表示无终点的日期范围。

orighour 域 (datepart)

该域保存origdate域的小时 (hour) 成分，当日期是一个范围时，它保存开始日期的小时成分。

origminute 域 (datepart)

该域保存origdate域的分钟成分，当日期是一个范围时，它保存开始日期的分钟成分。

origsecond 域 (datepart)

该域保存`origdate`域的秒钟成分，当日期是一个范围时，它保存开始日期的秒钟成分。

origtimezone 域 (datepart)

该域保存`origdate`域的时区成分，当日期是一个范围时，它保存开始日期的时区成分。

origday 域 (datepart)

该域保存`origdate`域的日成分，当日期是一个范围时，它保存开始日期的日成分。

origmonth 域 (datepart)

该域保存`origdate`域的月成分，当日期是一个范围时，它保存开始日期的月成分。

origyear 域 (datepart)

该域保存`origdate`域的年成分，当日期是一个范围时，它保存开始日期的年成分。

origseason 域 (datepart)

该域保存由EDTF 5.2.5(见 § 2.3.8) 规定的`origdate`域的季节成分，它包含一个季节本地化字符串。当日期是一个范围时，它保存开始日期的季节成分。

origendhour 域 (datepart)

如果`origdate`域中给出的日期是一个范围，该域保存结束日期的小时成分。

origendminute 域 (datepart)

如果`origdate`域中给出的日期是一个范围，该域保存结束日期的分钟成分。

origendsecond 域 (datepart)

如果`origdate`域中给出的日期是一个范围，该域保存结束日期的秒钟成分。

origendtimezone 域 (datepart)

如果`origdate`域中给出的日期是一个范围，该域保存结束日期的时区成分。

origendday 域 (datepart)

如果`origdate`域中给出的日期是一个范围，该域保存结束日期的日成分。

origendmonth 域 (datepart)

如果`origdate`域中给出的日期是一个范围，该域保存结束日期的月成分。

origendyear 域 (datepart)

如果`origdate` 域中给出的日期是一个范围，该域保存结束日期的年成分。空的 (但已定义) 的`origendyear`成分表示无终点的日期范围。

origendseason 域 (datepart)

如果origdate域中给出的日期是一个范围，该域保存EDTF 5.2.5 (§ 2.3.8) 规定的结束日期的季节成分。它包含一个季节本地化字符串 (见 § 4.9.2.21)，空的 (但已定义) 的origendseason成分表示无终点的origdate范围。

eventhour 域 (datepart)

该域保存eventdate域的小时 (hour) 成分，当日期是一个范围时，它保存开始日期的小时成分。

eventminute 域 (datepart)

该域保存eventdate域的分钟成分，当日期是一个范围时，它保存开始日期的分钟成分。

eventsecond 域 (datepart)

该域保存eventdate域的秒钟成分，当日期是一个范围时，它保存开始日期的秒钟成分。

eventtimezone 域 (datepart)

该域保存eventdate域的时区成分，当日期是一个范围时，它保存开始日期的时区成分。

eventday 域 (datepart)

该域保存eventdate域的日成分，当日期是一个范围时，它保存开始日期的日成分。

eventmonth 域 (datepart)

该域保存eventdate域的月成分，当日期是一个范围时，它保存开始日期的月成分。

eventyear 域 (datepart)

该域保存eventdate域的年成分，当日期是一个范围时，它保存开始日期的年成分

eventseason 域 (datepart)

该域保存由EDTF 5.2.5(见 § 2.3.8) 规定的eventdate域的季节成分，它包含一个季节本地化字符串。当日期是一个范围时，它保存开始日期的季节成分。

eventendhour 域 (datepart)

如果eventdate域中给出的日期是一个范围，该域保存结束日期的小时成分。

eventendminute 域 (datepart)

如果eventdate域中给出的日期是一个范围，该域保存结束日期的分钟成分。

eventendsecond 域 (datepart)

如果eventdate域中给出的日期是一个范围，该域保存结束日期的秒钟成分。

eventendtimezone 域 (datepart)

如果eventdate域中给出的日期是一个范围，该域保存结束日期的时区成分。

eventendday 域 (datepart)

如果eventdate域中给出的日期是一个范围，该域保存结束日期的日成分。

eventendmonth 域 (datepart)

如果eventdate域中给出的日期是一个范围，该域保存结束日期的月成分。

eventendyear 域 (datepart)

如果eventdate 域中给出的日期是一个范围，该域保存结束日期的年成分。空的 (但已定义) 的eventendyear成分表示无终点的日期范围。

eventendseason 域 (datepart)

如果eventdate域中给出的日期是一个范围，该域保存EDTF 5.2.5 (§ 2.3.8) 规定的结束日期的季节成分。它包含一个季节本地化字符串 (见 § 4.9.2.21)，空的 (但已定义) 的eventendseason成分表示无终点的eventdate范围。

urlhour 域 (datepart)

该域保存urldate域的小时 (hour) 成分，当日期是一个范围时，它保存开始日期的小时成分。

urlminute 域 (datepart)

该域保存urldate域的分钟成分，当日期是一个范围时，它保存开始日期的分钟成分。

urlsecond 域 (datepart)

该域保存urldate域的秒钟成分，当日期是一个范围时，它保存开始日期的秒钟成分。

timezone 域 (urldatepart)

该域保存urldate域的时区成分，当日期是一个范围时，它保存开始日期的时区成分。

urlday 域 (datepart)

该域保存urldate域的日成分。

urlmonth 域 (datepart)

该域保存urldate域的月成分。

urlyear 域 (datepart)

该域保存urldate域的年成分。

urlseason 域 (datepart)

该域保存由EDTF 5.2.5(见 § 2.3.8) 规定的 urldate 域的季节成分，它包含一个季节本地化字符串。当日期是一个范围时，它保存开始日期的季节成分。

urlendhour 域 (datepart)

如果urldate域中给出的日期是一个范围，该域保存结束日期的小时成分

urlendminute 域 (datepart)

如果urldate域中给出的日期是一个范围，该域保存结束日期的分钟成分

urlendsecond 域 (datepart)

如果urldate域中给出的日期是一个范围，该域保存结束日期的秒钟成分

urlendtimezone 域 (datepart)

如果urldate域中给出的日期是一个范围，该域保存结束日期的时区成分

urlendday 域 (datepart)

如果urldate域中给出的日期是一个范围，该域保存结束日期的日成分

urlendmonth 域 (datepart)

如果urldate域中给出的日期是一个范围，该域保存结束日期的月成分

urlendyear 域 (datepart)

如果urldate 域中给出的日期是一个范围，该域保存结束日期的年成分。空的 (但已定义) 的urlendyear成分表示无终点的日期范围。

urlendseason 域 (datepart)

如果urldate域中给出的日期是一个范围，该域保存EDTF 5.2.5 (§ 2.3.8) 规定的结束日期的季节成分。它包含一个季节本地化字符串 (见 § 4.9.2.21)，空的 (但已定义) 的urlendseason成分表示无终点的eventdate范围。

## 4.3 标注样式

参考文献标注样式是诸如\cite等用于打印不同类型标注的命令集。这些样式定义在后缀为cbx的文件中。Biblatex 在包末尾加载它们。注意: 一些标准标注样式的常用共享宏集放在biblatex.def 文件中。这一文件也在包末尾加载，先于选择的标注样式。它也包含由来自 § 3.8.5节的命令的定义。

### 4.3.1 标注样式文件

在讨论标注样式文件提供的各个命令前，考虑如下一个典型cbx文件的整体结构:

```
\ProvidesFile{example.cbx}[2006/03/15 v1.0 biblatex citation style]

\DeclareCiteCommand{\cite}{...}{...}{...}{...}
\DeclareCiteCommand{\parencite}[\mkbibparens]{...}{...}{...}{...}
\DeclareCiteCommand{\footcite}[\mkbibfootnote]{...}{...}{...}{...}
\DeclareCiteCommand{\textcite}{...}{...}{...}{...}
\endinput
```

\RequireCitationStyle{⟨*style*⟩}

这个命令是可选的，用于在一些更一般的样式基础上构建特殊的标注样式。它加载标注样式style.cbx。

\InitializeCitationStyle{⟨*code*⟩}

指定初始化或重设标注样式需要的任意⟨*code*⟩。这个钩子将在包加载的时候执行一次，并且每次都使用 § 3.8.8节的\citereset命令。\citereset命令也重设本宏包的内部标注追踪器。它会影响 § 4.6.2节中列出的 cmdifciteseen, \ifentryseen, \ifciteibid, 和\ifciteidem等判断。当使用refsection环境时，标注追踪器重设当前的refsection局部环境。

\OnManualCitation{⟨*code*⟩}指定重设部分标注样式需要的任意⟨*code*⟩。这一钩子将在 § 3.8.8中的\mancite命令使用时调用。它有时特别有用，可以代替像 'ibidem' 或'op. cit.' 等缩写表示的重复标注，因为当自动生成和人工产生的标注混合使用的时候这些缩写可能会有歧义。\mancite命令也会重设宏包的内部'ibidem' 和'idem' 追踪器，进而影响 § 4.6.2节讨论的\ifciteibid和\ifciteidem判断。

\DeclareCiteCommand{⟨*command*⟩}[⟨*wrapper*⟩]{⟨*precode*⟩}{⟨*loopcode*⟩}{⟨*sepcode*⟩}{⟨*postcode*⟩}
\DeclareCiteCommand*{⟨*command*⟩}[⟨*wrapper*⟩]{⟨*precode*⟩}{⟨*loopcode*⟩}{⟨*sepcode*⟩}{⟨*postcode*⟩}

这是用于定义所有标注 (引用) 命令的核心命令。它有 1 个可选参数和 5 个必选参数。⟨*command*⟩是要定义的命令，比如\cite。如果给出可选的⟨*wrapper*⟩参数，整个标注将会作为一个参数传递给⟨*wrapper*⟩，即包围 (wrapper) 命令必须要取得一个必选参数。[41] ⟨*precode*⟩是在标注开始时执行的任意代码。典型地，它将处理由prenote域提供的⟨*prenote*⟩参数。它可以可用来对⟨*loopcode*⟩所需的宏进行初始化。⟨*loopcode*⟩是每个条目关键词传递给⟨*command*⟩命令时执行的任意代码。它是打印标注标签或其它任意数据的核心代码。⟨*sepcode*⟩是每次执行⟨*loopcode*⟩完成后执行的代码。它仅在条目关键词列表传递给⟨*command*⟩时起作用。⟨*sepcode*⟩常用于插入一些分隔符，比如逗号或分号等。⟨*postcode*⟩是在标注结束时执行的代码。典型地，它将处理

[41] 典型的包围命令是\mkbibparens和\mkbibfootnote。

由postnote域提供的⟨*postnote*⟩参数。[42] 带星号的\DeclareCiteCommand命令定义了一个带星号的⟨*command*⟩。例如\DeclareCiteCommand*{cite} 命令将定义\cite*。[43]

\DeclareMultiCiteCommand{⟨*command*⟩}[⟨*wrapper*⟩]{⟨*cite*⟩}{⟨*delimiter*⟩}

该命令定义'multicite' 类命令 (见 § 3.8.3)。⟨*command*⟩是要定义的 multicite 命令，比如\cites。它自动在由\DeclareCiteCommand定义的后端命令基础上构建鲁棒的命令，其中⟨*cite*⟩参数用于指定使用的后端命令名。注意后端命令的包围命令 (封套)(即传递给\DeclareCiteCommand命令的⟨*wrapper*⟩参数) 自动忽略。使用可选⟨*wrapper*⟩参数作为其替换。⟨*delimiter*⟩是列表中单个标注之间的分隔字符串。下面给出的例子是典型的\multicitedelim命令，取自biblatex.def 中的实际定义:

```
\DeclareMultiCiteCommand{\cites}%
        {\cite}{\multicitedelim}
\DeclareMultiCiteCommand{\parencites}[\mkbibparens]%
        {\parencite}{\multicitedelim}
\DeclareMultiCiteCommand{\footcites}[\mkbibfootnote]%
        {\footcite}{\multicitedelim}
```

\DeclareAutoCiteCommand{⟨*name*⟩}[⟨*position*⟩]{⟨*cite*⟩}{⟨*multicite*⟩}

该命令为\autocite和\autocites类命令提供定义 (见 § 3.8.4)。要使定义生效需要打开 § 3.1.2.1节的autocite包选项。⟨*name*⟩是一个标识向包选项传递一个值。autocite 类命令是在\parencite和\parencites等后端命令基础上构建的。⟨*cite*⟩和⟨*multicite*⟩参数指定了使用的后端命令。⟨*cite*⟩参数用于\autocite，而⟨*multicite*⟩用于\autocites。⟨*position*⟩参数控制标注后的任何标点符号的处理。可能的值是 l, r, f。r 表示标点置于标注的右侧，即它不会移动。l 表示将标点移动到标注的左侧。f 在脚注中的作用类似于 r，在其它情况下则类似于 l。该参数是可选的默认是 r。另可参见 § 4.7.5节的\DeclareAutoPunctuation命令和 § 3.1.2.1节的autopunct包选项。下面的例子取自biblatex.def 中的实际定义:

```
\DeclareAutoCiteCommand{plain}{\cite}{\cites}
\DeclareAutoCiteCommand{inline}{\parencite}{\parencites}
\DeclareAutoCiteCommand{footnote}[l]{\footcite}{\footcites}
\DeclareAutoCiteCommand{footnote}[f]{\smartcite}{\smartcites}
```

文档导言区提供的定义可以利用如下方式随后采用 (见 § 3.2.2):

```
\ExecuteBibliographyOptions{autocite=name}
```

[42] 能给⟨*loopcode*⟩提供的参考文献数据是正在处理的条目的数据。此外，第一个 ('First') 条目的数据可以用于⟨*precode*⟩，最后一个 ('last') 条目的数据可以用于⟨*postcode*⟩。'First' and 'last' 指的是标注的打印顺序。如果sortcites包选项打开，这是经过排序处理后的顺序。注意: 没有任何参考文献数据可用于⟨*sepcode*⟩。

[43] 注意: 无星号的\DeclareCiteCommand命令也将定义隐式的定义一个带星号的标注命令，除非该标注命令前面已经定义。这只是用于提供备选。这种隐式方式定义的命令将等同于不带星号的命令。

### 4.3.2 特殊域

下面的域用于向标注命令传递数据。它们不用于bib文件中而由宏包自动定义。从标注样式的角度看，它们与bib中的域并无区别。另可参见 § 4.2.4。

prenote 域 (literal)

作为⟨*prenote*⟩参数向标注命令传递。该域仅用于标注而不能用在参考文献表中。如果⟨*prenote*⟩参数缺省或为空，该域不定义。

postnote 域 (literal)

作为⟨*postnote*⟩参数向标注命令传递。该域仅用于标注而不能用在参考文献表中。如果⟨*postnote*⟩参数缺省或为空，该域不定义。

multiprenote 域 (literal)

作为⟨*multiprenote*⟩参数向 multicite 类标注命令传递。该域仅用于标注而不能用在参考文献表中。如果⟨*multiprenote*⟩参数缺省或为空，该域不定义。

multipostnote 域 (literal)

作为⟨*multipostnote*⟩参数向 ulticite 类标注命令传递。该域仅用于标注而不能用在参考文献表中。如果⟨*multipostnote*⟩参数缺省或为空，该域不定义。

postpunct 域 (punctuation command)

作为拖尾的标点参数隐式地向标注命令传递。该域仅用于标注而不能用在参考文献表中。如果一个标注命令后面跟着的字符不在\DeclareAutoPunctuation (§ 4.7.5)命令的定义中，该域不定义。

## 4.4 数据接口

数据接口是用于格式化和打印全部参考文献数据的工具。这些工具均可在著录和标注样式中使用。

### 4.4.1 数据命令

本节介绍 Biblatex 包的主要数据接口。这些命令处理了绝大部分工作，即实际上由它们来对列表和域提供的数据进行打印。

\DeprecateField{⟨*field*⟩}{⟨*message*⟩}
\DeprecateList{⟨*list*⟩}{⟨*message*⟩}
\DeprecateName{⟨*name*⟩}{⟨*message*⟩}

用于在打印⟨*field*⟩, ⟨*list*⟩, ⟨*name*⟩时给出表示不允许的警告信息⟨*message*⟩。它帮助那些需要在样式中修改域名的样式作者。注意: 不允许的项只能是未在当前工作的数据模型中定义的项，⟨*field*⟩, ⟨*list*⟩或⟨*name*⟩不能出现在\DeclareDatamodelFields命令的参数中。

`\printfield[`⟨*format*⟩`]{`⟨*field*⟩`}`

该命令使用由`\DeclareFieldFormat`定义的格式化指令⟨*format*⟩打印⟨*field*⟩。如果声明了 type-specific 的格式指令，其将优先于设置的通用格式指令。如果⟨*field*⟩未定义则不作打印。如果⟨*format*⟩缺省，`\printfield`将尝试域名作为格式化指令名进行格式化。例如: 要打印`title`域，且⟨*format*⟩未指定，它将尝试用于域格式 `title`。[44] 这种情况下，任何 type-specific(具体类型的) 格式化指令将优先于通用指令采用。如果所有的这类格式都未定义，它将返回到 `default` 作为最后的方式。注意:`\printfield`为格式化指令提供当前正在`\currentfield`中处理的域名。

`\printlist[`⟨*format*⟩`][`⟨*start*⟩`–`⟨*stop*⟩`]{`⟨*literal list*⟩`}`该命令对所有在⟨*literal list*⟩中的项进行循环处理，从项数⟨*start*⟩开始，到项数⟨*stop*⟩结束，包括⟨*start*⟩和⟨*stop*⟩(所有的列表中项以 1 开始计数)。每一项都用由`\DeclareListFormat`定义的格式化指令⟨*format*⟩打印。如果声明了 type-specific 的格式指令，其将优先于设置的通用格式指令。如果⟨*literal list*⟩未定义则不作打印。如果⟨*format*⟩缺省，`\printlist`将尝试列表名作为格式化指令名进行格式化。这种情况下，任何 type-specific(具体类型的) 格式化指令将优先于通用指令采用。如果所有的这类格式都未定义，它将返回到 `default` 作为最后的方式。⟨*start*⟩参数默认是 1，⟨*stop*⟩默认是列表中项的总数。如果项的总数大于⟨*maxitems*⟩，⟨*stop*⟩默认为⟨*minitems*⟩(见 § 3.1.2.1)。更多细节参见`\printnames`。注意:`\printlist`为格式化指令提供当前正在`\currentlist`中处理的域名。

`\printnames[`⟨*format*⟩`][`⟨*start*⟩`–`⟨*stop*⟩`]{`⟨*name list*⟩`}`该命令对所有在⟨*name list*⟩中的项进行循环处理，从项数⟨*start*⟩开始，到项数⟨*stop*⟩结束，包括⟨*start*⟩和⟨*stop*⟩(所有的列表中项以 1 开始计数)。每一项都用由`\DeclareNameFormat`定义的格式化指令⟨*format*⟩打印。如果声明了 type-specific 的格式指令，其将优先于设置的通用格式指令。如果⟨*name list*⟩未定义则不作打印。如果⟨*format*⟩缺省，`\printnames`将尝试列表名作为格式化指令名进行格式化。这种情况下，任何 type-specific(具体类型的) 格式化指令将优先于通用指令采用。如果所有的这类格式都未定义，它将返回到 `default` 作为最后的方式。⟨*start*⟩参数默认是 1，⟨*stop*⟩默认是列表中项的总数。如果项的总数大于⟨*maxnames*⟩，⟨*stop*⟩默认为⟨*minnames*⟩(见 § 3.1.2.1)。如果你要自己制定一个范围而又要使用默认的列表格式，第一个可选参数必须给出但要留空:

```
\printnames[][1-3]{...}
```

⟨*start*⟩和⟨*stop*⟩之一可以缺省，因此下面的参数都是有效的:

```
\printnames[...][-1]{...}
\printnames[...][2-]{...}
\printnames[...][1-3]{...}
```

如果你要覆盖⟨*maxnames*⟩和⟨*minnames*⟩并打印整个列表，你可以在第二个可选参数中以如下方式设置`listtotal`计数器。

[44] 换句话说，`\printfield{title}` 等价于`\printfield[title]{title}`。

```
\printnames[...][-\value{listtotal}]{...}
```

当\printnames和\printlist处理一个列表时，当前状态的信息可以通过 4 个计数器获知: listtotal 计数器保存当前列表中项的总数，listcount保存当前正在处理的项的序号，liststart是传递给\printnames 或\printlist命令的 ⟨*start*⟩参数，liststop 则是 ⟨*stop*⟩参数。这些计数器用于列表的格式化指令。listtotal也可以在第二个可选参数中使用。注意，这些计数器仅在列表格式化指令中有意义在其它任何地方都无效。对于每类列表，都有一个具有相同名字的计数器保存该类列表的项的总数。例如，author计数器保存author列表中的项的总数。这些计数器类似于listtotal，但可用于列表格式化指令之外。还有maxnames ，minnames，maxitems和minitems计数器，用于保存相应的包选项的值。这些内部计数器的完整列表详见 § 4.10.5。注意: \printnames为格式化指令提供当前正在\currentname中处理的域名。

\printtext[⟨*format*⟩]{⟨*text*⟩}该命令用于打印⟨*text*⟩，可以是可打印的文本或者产生可打印文本的任意代码。它清除插入⟨*text*⟩之前的标点缓存并且通知 Biblatex 打印文本已经插入。这保证了所有之前和之后的\newblock和\newunit命令能产生预期的作用。\printfield、\printnames 、\bibstring及其相关命令都这般自动处理 (见 § 4.8)。如果一个参考文献样式需要插入抄录文本（包括来自 §§ 4.7.3 和 4.7.4的命令），需要使用该命令来确保 block 和 unit 标点在 § 4.7.1节中所述正常功能。可选参数⟨*format*⟩指定一个域格式指令用于格式化⟨*text*⟩。当需要把若干个域打印为某一格式的集合块，这就会很有用，比如把集合块用括号或引号包起来。如果声明了 type-specific 的格式指令，其将优先于设置的通用格式指令。如果⟨*format*⟩缺省，那么⟨*text*⟩如实输出 (原样打印)。更多实用细节见第 § 4.11.7节。

\printfile[⟨*format*⟩]{⟨*file*⟩}该命令类似于\printtext，差别在于第二个参数是一个文件名而不是抄录文本。⟨*file*⟩参数必须是一个能在 TeX 搜索路径找到的有效的 LaTeX 文件。\printfile将使用\input来加载该⟨*file*⟩。如果指定文件不存在，\printfile不做任何操作。可选的⟨*format*⟩参数指定了一个域格式化指令应用于该⟨*file*⟩。如果声明了 type-specific 的格式指令，其将优先于设置的通用格式指令。如果⟨*format*⟩缺省，那么⟨*file*⟩如实输出 (原样打印)。注意该功能需要显式的打开 § 3.1.2.1节的包选项loadfiles。默认情况下，\printfile不加载任何文件。

\printdate 该命令打印条目定义在date或month/year域中的日期。日期格式由 § 3.1.2.1节中的date包选项控制。另外也可以通过调整域格式 date (见 § 4.10.4) 来进一步格式化 (比如设置字体等)。注意: 该命令与标点追踪器自动交互，不必使用\printtext命令将其包围起来。

\printdateextra 类似于\printdate，但指定的日期域是extrayear域。用于设计作者年制的参考文献样式。

\printlabeldate 类似于\printdate，但打印的是日期域由\DeclareLabeldate决定。日期格式由 § 3.1.2.1节中的labeldate包选项控制。另外也可以通过调整域格式 labeldate (见 § 4.10.4) 来进一步格式化。

\printlabeldateextra 类似于\printlabeldate，但指定的日期域是extrayear域，用于设计作者年制的参考文献样式。

\print<datetype>date 类似于\printdate，但打印的是日期域是条目的由<datetype>date域。日期格式由 § 3.1.2.1节中的<datetype>date包选项控制。另外也可以通过调整域格式<datetype>date (见 § 4.10.4) 来进一步格式化。<datetype> 在默认的数据模型中有:'' (用于date域), 'orig', 'event' 和'url'。

\printtime 该命令打印条目定义在date域 (见 § 2.3.8) 中的时间范围，时间格式由 § 3.1.2.1节中的time包选项控制。另外也可以通过调整域格式 time (见 § 4.10.4) 来进一步格式化 (比如设置字体等)。时间格式化相关内容还包括timezeros选项，\bibtimesep和\bibtimezonesep宏 (§ 3.10.2)。注意: 该命令与标点追踪器自动交互，不必使用\printtext命令将其包围起来。注意该命令打印的是独立于日期元素的时间范围。当<datepart>dateusetime选项打开时，也可以与日期范围一起打印，而不是各自分别打印。

\print<datetype>time 类似于\printtime,但打印的条目的<datetype>time域。时间格式由§3.1.2.1节中的<datetype>time包选项控制。另外也可以通过调整域格式 <datetype>time (见 § 4.10.4) 来进一步格式化。<datetype> 在默认的数据模型中有:'' (用于date域), 'orig', 'event' 和'url'。

\indexfield[⟨*format*⟩]{⟨*field*⟩}

该命令类似于\printfield，差别在于不是打印⟨*field*⟩而是将其添加到索引中，其格式化指令⟨*format*⟩由\DeclareIndexFieldFormat命令定义。如果声明了 type-specific 的格式指令，其将优先于设置的通用格式指令。如果⟨*field*⟩域未定义，该命令不做任何操作。如果⟨*format*⟩缺省，那么\indexfield将采用与域名相同的格式名。这种情况下任何 type-specific 的格式指令都将优先于通用的格式指令。若所有的这些格式都未定义，那么将采用 default 格式作为最后的选择。

\indexlist[⟨*format*⟩][⟨*start*⟩–⟨*stop*⟩]{⟨*literal list*⟩}

该命令类似于\printlist，差别在于不是打印列表的项而是将其添加到索引中，其格式化指令⟨*format*⟩由\DeclareIndexListFormat命令定义。如果声明了 type-specific 的格式指令，其将优先于设置的通用格式指令。如果⟨*literal list*⟩未定义，该命令不做任何操作。如果⟨*format*⟩缺省，那么\indexlist将采用与列表名相同的格式名。这种情况下任何 type-specific 的格式指令都将优先于通用的格式指令。若所有的这些格式都未定义，那么将采用 default 格式作为最后的选择。

\indexnames[⟨*format*⟩][⟨*start*⟩–⟨*stop*⟩]{⟨*name list*⟩}

该命令类似于\printnames，差别在于不是打印姓名列表的项而是将其添加到索引中，其格式化指令⟨*format*⟩由\DeclareIndexNameFormat命令定义。如果声明了 type-specific 的格式指令，其将优先于设置的通用格式指令。如果⟨*name list*⟩未定义，该命令不做任何操作。如果⟨*format*⟩缺省，那么\indexnames将采用与列表名相同的格式名。这种情况下任何 type-specific 的格式指令都将优先于通用的格式指令。若所有的这些格式都未定义，那么将采用 default 格式作为最后的选择。

\entrydata{⟨*key*⟩}{⟨*code*⟩}
\entrydata*{⟨*key*⟩}{⟨*code*⟩}

像\printfield之类的数据命令正常情况下应用当前正在处理的条目的数据。可以使用\entrydata在局部环境中转换应用数据。⟨*key*⟩是要局部使用条目的关键词。⟨*code*⟩是在当前局部环境执行的任意代码。这一地面将在一个编组中执行。举例见 § 4.11.6节。注意该命令自动转换语言，如果autolang包选项打开的话。带星号的命令\entrydata*将复制封装条目 (the enclosing entry) 的所有域，使用域，计数器，和其它以字符串'saved' 为前缀命名的资源。这在比较两个数据集是很有用。例如，在⟨*code*⟩的参数中，author域保存了条目⟨*key*⟩的作者，而封装条目的作者保存在savedauthor域中。author计数器保存了⟨*key*⟩条目的author域的姓名数量，而封装条目的作者数量由savedauthor计数器计数。

\entryset{⟨*precode*⟩}{⟨*postcode*⟩}

该命令用在处理@set条目集的参考文献驱动中。它将对由entryset域指出的集的所有成员进行循环处理，对集的各个成员执行相应的驱动。这相当于对每个集成员执行 § 4.6.4节的\usedriver命令。⟨*precode*⟩是在集的每项处理之前执行的任意代码。⟨*postcode*⟩是在集的每项处理之后执行的任意代码。两个参数语法上必须的，但可以留空。用法举例见 § 4.11.1节。

\DeclareFieldInputHandler{⟨*field*⟩}{⟨*code*⟩}

给命令用于定义从.bbl读取数据所采用的域的数据输入处理器。在⟨*code*⟩内，宏\NewValue包含了域的值。比如，要忽略出现的volumes域，可以作:

```
\DeclareFieldInputHandler{volumes}{\def\NewValue{}}
```

一般情况下，要删除和修改域需要使用\DeclareSourcemap(见 § 4.5.3节)，而这一替代方法在一些情形下会很有用，例如当强调的是数据的外观而不是数据本身时，因为⟨*code*⟩可以是任意的 TeX 代码。

\DeclareListInputHandler{⟨*list*⟩}{⟨*code*⟩}

类似于\DeclareFieldInputHandler，但用于列表。在⟨*code*⟩内，宏\NewValue包含了列表的值，而\NewCount保存列表中项的序号。

\DeclareNameInputHandler{⟨*name*⟩}{⟨*code*⟩}

类似于\DeclareFieldInputHandler，但用于姓名列表。在⟨*code*⟩内，宏\NewValue包含了姓名列表的值，而\NewCount保存列表中姓名的序号，\NewOption保存了.bbl文件传递的任意姓名所属的选项。

### 4.4.2 格式化指令

本节介绍用于定义 § 4.4.1节的数据命令所需的格式化指令的命令。注意: 所有标注的格式定义在biblatex_.def 文件中。

\DeclareFieldFormat[⟨*entrytype, ...*⟩]{⟨*format*⟩}{⟨*code*⟩}
\DeclareFieldFormat*{⟨*format*⟩}{⟨*code*⟩}

定义域格式⟨*format*⟩。该格式化指令是由\printfield命令执行的任意⟨*code*⟩。域的值作为仅有的第一个参数传递给⟨*code*⟩。正在处理的域名在⟨*code*⟩中以\currentfield表示。如果指定一种条目类型 (⟨*entrytype*⟩)，那么格式是该类型专属的。⟨*entrytype*⟩可以是一个逗号分隔 (comma-separated) 的值列表。带星的命令类似于不带星的命令，区别在于它还将清除所有对条目类型做的格式定义。

\DeclareListFormat[⟨*entrytype, ...*⟩]{⟨*format*⟩}{⟨*code*⟩}
\DeclareListFormat*{⟨*format*⟩}{⟨*code*⟩}

定义文本列表[45]的格式⟨*format*⟩。格式化指令是\printlist命令处理列表中每一项时执行的任意⟨*code*⟩。当前正在处理的项作为第一个和唯一的参数传递给⟨*code*⟩。正在处理的文本列表名在⟨*code*⟩中以\currentlist表示。如果指定了⟨*entrytype*⟩，那么格式是该类型专属的。⟨*entrytype*⟩参数可以是一个逗号分隔 (comma-separated) 的值列表。注意格式化指令也会处理在列表各项间插入的标点。需要对当前项是在列表中间或者末尾进行检测，即检查listcount是否小于或等于liststop。带星的命令类似于不带星的命令，区别在于它还将清除所有对条目类型做的格式定义。

\DeclareNameFormat[⟨*entrytype, ...*⟩]{⟨*format*⟩}{⟨*code*⟩}
\DeclareNameFormat*{⟨*format*⟩}{⟨*code*⟩}

定义姓名列表的格式⟨*format*⟩。格式化指令是\printnames命令处理列表中每一项时执行的任意⟨*code*⟩。如果指定了⟨*entrytype*⟩，那么格式是该类型专属的。⟨*entrytype*⟩参数可以是一个逗号分隔 (comma-separated) 的值列表。单个姓名的各个组成部分由自动创建的宏表示 (见下)。默认的数据模型定义了四个组成部分对应于标准的 BibTeX 姓名成分参数。

`family` 姓，BibTeX 中为'last' name 部分。当一个姓名只有一个部分组成时 (比如 'Aristotle')，这一部分将被处理为姓。

`given` 名。注意名在 BibTeX 中为'first' name 部分。

`prefix` 前缀，比如 von, van, of, da, de, del, della 等。注意前缀在 BibTeX 格式文件中为'von' 部分。

`suffix` 后缀，比如 Jr, Sr 等。注意后缀在 BibTeX 格式文件中为'Jr' 部分。

数据模型'nameparts' 常量的值 (见 § 4.2.3) 在姓名的数据模型中为每个姓名组成部分创建了两个宏。比如，在默认的数据模型中，姓名格式由如下宏定义:

```
\namepartprefix %表示前缀部分
\namepartprefixi %表示前缀首字母
\namepartfamily %表示姓
\namepartfamilyi %表示姓首字母
```

[45]literal 译为文本

```
\namepartsuffix %表示后缀
\namepartsuffixi %表示后缀首字母
\namepartgiven %表示名
\namepartgiveni %表示名首字母
```

如果一个姓名的某些部分没有给出，相应的宏将为空，因此可以使用，比如etoolbox中\ifdefvoid这类的判断来检查姓名的各个组成部分。正在处理的姓名列表名在⟨*code*⟩中以\currentname表示。注意格式化指令也会处理在列表各项间插入的标点。需要对当前项是在列表中间或者末尾进行检测，即检查listcount是否小于或等于liststop(见 § 3.13.4节)。带星的命令类似于不带星的命令，区别在于它还将清除所有对条目类型做的格式定义。

\DeclareIndexFieldFormat[⟨*entrytype, ...*⟩]{⟨*format*⟩}{⟨*code*⟩}
\DeclareIndexFieldFormat*{⟨*format*⟩}{⟨*code*⟩}

定义域格式⟨*format*⟩。该格式化指令是由\indexfield命令执行的任意⟨*code*⟩。域的值作为仅有的第一个参数传递给⟨*code*⟩。正在处理的域名在⟨*code*⟩中以\currentfield表示。如果指定一种条目类型 (⟨*entrytype*⟩)，那么格式是该类型专属的。⟨*entrytype*⟩可以是一个逗号分隔 (comma-separated) 的值列表。该命令类似于\DeclareFieldFormat，差别在于⟨*code*⟩处理的数据不是用于打印而是用于索引。注意\indexfield将执行⟨*code*⟩本身，即⟨*code*⟩必须包含\index或类似命令。带星的命令类似于不带星的命令，区别在于它还将清除所有对条目类型做的格式定义。

\DeclareIndexListFormat[⟨*entrytype, ...*⟩]{⟨*format*⟩}{⟨*code*⟩}
\DeclareIndexListFormat*{⟨*format*⟩}{⟨*code*⟩}

定义文本列表格式⟨*format*⟩。该格式化指令是由\indexlist命令执行的任意⟨*code*⟩。列表中当前值作为唯一参数传递给⟨*code*⟩。正在处理的列表名在⟨*code*⟩中以\currentlist表示。如果指定一种条目类型 (⟨*entrytype*⟩)，那么格式是该类型专属的。⟨*entrytype*⟩可以是一个逗号分隔 (comma-separated) 的值列表。该命令类似于\DeclareListFormat，差别在于⟨*code*⟩处理的数据不是用于打印而是用于索引。注意\indexlist将执行⟨*code*⟩本身，即⟨*code*⟩必须包含\index或类似命令。带星的命令类似于不带星的命令，区别在于它还将清除所有对条目类型做的格式定义。

\DeclareIndexNameFormat[⟨*entrytype, ...*⟩]{⟨*format*⟩}{⟨*code*⟩}
\DeclareIndexNameFormat*{⟨*format*⟩}{⟨*code*⟩}

定义姓名列表格式⟨*format*⟩。该格式化指令是由\indexnames命令执行的任意⟨*code*⟩。列表中当前值作为唯一参数传递给⟨*code*⟩。正在处理的列表名在⟨*code*⟩中以\currentname表示。如果指定一种条目类型 (⟨*entrytype*⟩)，那么格式是该类型专属的。⟨*entrytype*⟩可以是一个逗号分隔 (comma-separated) 的值列表。该命令类似于\DeclareNameFormat，差别在于⟨*code*⟩处理的数据不是用于打印而是用于索引。

注意\indexnames将执行⟨*code*⟩本身，即⟨*code*⟩必须包含\index或类似命令。带星的命令类似于不带星的命令，区别在于它还将清除所有对条目类型做的格式定义。

\DeclareFieldAlias[⟨*entry type*⟩]{⟨*alias*⟩}[⟨*format entry type*⟩]{⟨*format*⟩}

声明⟨*alias*⟩作为域格式⟨*format*⟩的别名。如果指定一种条目类型(⟨*entrytype*⟩)，别名是该类型专属的。⟨*format entry type*⟩是后端格式的条目类型。这仅在声明某一具体条目类型的格式化指令的别名时需要。

\DeclareListAlias[⟨*entry type*⟩]{⟨*alias*⟩}[⟨*format entry type*⟩]{⟨*format*⟩}

声明⟨*alias*⟩作为文本列表格式⟨*format*⟩的别名。如果指定一种条目类型(⟨*entrytype*⟩)，别名是该类型专属的。⟨*format entry type*⟩是后端格式的条目类型。这仅在声明某一具体条目类型的格式化指令的别名时需要。

\DeclareNameAlias[⟨*entry type*⟩]{⟨*alias*⟩}[⟨*format entry type*⟩]{⟨*format*⟩}

声明⟨*alias*⟩作为姓名列表格式⟨*format*⟩的别名。如果指定一种条目类型(⟨*entrytype*⟩)，别名是该类型专属的。⟨*format entry type*⟩是后端格式的条目类型。这仅在声明某一具体条目类型的格式化指令的别名时需要。

\DeclareIndexFieldAlias[⟨*entry type*⟩]{⟨*alias*⟩}[⟨*format entry type*⟩]{⟨*format*⟩}

声明⟨*alias*⟩作为域格式⟨*format*⟩的别名。如果指定一种条目类型(⟨*entrytype*⟩)，别名是该类型专属的。⟨*format entry type*⟩是后端格式的条目类型。这仅在声明某一具体条目类型的格式化指令的别名时需要。

\DeclareIndexListAlias[⟨*entry type*⟩]{⟨*alias*⟩}[⟨*format entry type*⟩]{⟨*format*⟩}

声明⟨*alias*⟩作为文本列表格式⟨*format*⟩的别名。如果指定一种条目类型(⟨*entrytype*⟩)，别名是该类型专属的。⟨*format entry type*⟩是后端格式的条目类型。这仅在声明某一具体条目类型的格式化指令的别名时需要。

\DeclareIndexNameAlias[⟨*entry type*⟩]{⟨*alias*⟩}[⟨*format entry type*⟩]{⟨*format*⟩}

声明⟨*alias*⟩作为姓名列表格式⟨*format*⟩的别名。如果指定一种条目类型(⟨*entrytype*⟩)，别名是该类型专属的。⟨*format entry type*⟩是后端格式的条目类型。这仅在声明某一具体条目类型的格式化指令的别名时需要。

## 4.5 定制

### 4.5.1 关联条目

关联条目功能由如下部分构成:[46]

- 条目中的特殊域用于建立和描述关系
- 本地化字符串作为关联数据的前缀 (可选)
- 抽取和打印关联数据的宏

[46]这里 related data 用关联而不是相关，其它地方用到相关的应一并改过来

- 用于本地化字符串和关联数据格式化的格式

特殊域是`related`, `relatedtype`, `relatedstring`和 `relatedoptions`:

related 与当前条目某种程度关联的条目的条目关键词列表[47]。注意: 条目关键词[48]的顺序很重要。来自多个关联条目的数据是按该域中关键词的顺序打印。

relatedtype The type of relationship. This serves three purposes. If the value of this field resolves to a localisation string identifier, then the resulting localised string is printed before the data from the related entries. Secondly, if there is a macro called `related:`⟨*relatedtype*⟩, this is used to format the data from the related entries. If no such macro exists, then the macro `related:default` is used. Lastly, if there is a format named `related:`⟨*relatedtype*⟩, then it is used to format both the localised string and related entry data. If there is no related type specific format, the `related` format is used.

relatedstring If an entry contains this field, then if value of the field resolves to a localisation string identifier, the localisation key value specified is printed before data from the related entries. If the field does not specify a localisation key, its value is printed literally. If both `relatedtype` and `relatedstring` are present in an entry, `relatedstring` is used for the pre-data string (but `relatedtype` is still used to determine the macro and format to use when printing the data).

relatedoptions A list of per-entry options to set on the related entry (actually on the clone of the related entry which is used as a data source—the actual related entry is not modified because it might be cited directly itself).

The related entry feature is enabled by default by the package option `related` from § 3.1.2.1. The related information entry data from the related entries is included via a `\usebibmacro{related}` call. Standard styles call this macro towards the end of each driver. Style authors should ensure the existence of (or take note of existing) localisation strings which are useful as values for the `relatedtype` field, such as `translationof` or perhaps `translatedas`. A plural variant can be identified with the localisation key ⟨*relatedtype*⟩`s`. This key's corresponding string is printed whenever more than one entry is specified in `related`. Bibliography macros and formatting directives for printing entries related by ⟨*relatedtype*⟩ should be defined using the name `related:`⟨*relatedtype*⟩. The file `biblatex.def` contains macros and formats for some common relation types which can be used as templates. In particular, the `\entrydata*` command is essential in such macros in order to make the data of the related entries available. Examples of entries using this feature can be found in the Biblatex distribution examples file `biblatex-examples.bib`. There are some specific formatting macros for this feature which control delimiters and separators in related entry information, see § 4.10.1.

[47]这里 separated list 用分离列表有没有更好的说法

[48]这里的 key 关键词应该是引用关键词，即 bibtex 键

### 4.5.2 数据源集

能够定义在循环等中的使用的数据源 (datasource) 域名的集是有用的。而且，Biber 可以利用这种集来应用选项或者对某些数据源域的特定集进行操作。下面的宏允许用来定义数据源域的任意集合，这些数据源域以etoolbox列表的形式提供给 Biblatex 并通过.bcf文件中提供给 Biber 。

\DeclareDatafieldSet{⟨*name*⟩}{⟨*specification*⟩}

声明一个数据源域的集，其名为⟨*name*⟩。

name=⟨*set name*⟩

集的名。

⟨*specification*⟩是 1 个或更多的\member(成员) 项:

\member

fieldtype=⟨*fieldtype*⟩

datatype=⟨*datatype*⟩

field=⟨*fieldname*⟩

一个\member说明将域添加到集中。域可以由数据模型⟨*fieldtype*⟩ 和/或 ⟨*datatype*⟩指定 (见 § 4.5.4)。或者，域也可以通过使用⟨*field*⟩选项显式的以名字添加。一旦完成定义，集就以etoolbox列表的形式存在，命名为\datafieldset‘setname’并通过.bcf文件传递给 Biber 。

下面的例子就是 Biblatex 为姓名域和标题域定义的默认集:

```
\DeclareDatafieldSet{setnames}{
  \member[datatype=name, fieldtype=list]
}

\DeclareDatafieldSet{settitles}{
  \member[field=title]
  \member[field=booktitle]
  \member[field=eventtitle]
  \member[field=issuetitle]
  \member[field=journaltitle]
  \member[field=maintitle]
  \member[field=origtitle]
}
```

这将定义\datafieldsetsetnames和\datafieldsetsettitles宏以etoolbox列表形式包含指定的成员数据源域的名。

### 4.5.3 数据动态修改

对自动生成或者你无法控制的参考文献数据源进行修改在某种程度上会是一个问题。因此，Biber 提供了对它所读取的数据进行修改的能力，这样你可以对源数据流进行修改二不必实际改变它。这种改变可以在 Biber 的配置文件 (见 Biber 文档) 定义，或者通过 Biblatex 宏进行定义，通过宏定义的方法你可以在样式中或者以全局定义的方式，将修改应用在具体的文档中。

源映射发生在数据解析过程中，因此也在诸如继承和排序等任何其它操作之前。

源映射可以在不同的层 ( “levels” ) 进行定义，各层以某一定义的顺序进行处理。见 Biblatex 手册，考虑这些宏:

`\DeclareSourcemap`命令定义的 `user`-层映射 →
  在 Biber 配置文件定义的 `user`-层映射 (见 Biber 文档)→
    `\DeclareStyleSourcemap`定义的 `style`-层映射 →
      `\DeclareDriverSourcemap`定义的 `driver`-层映射

`\DeclareSourcemap{`⟨*specification*⟩`}`

定义源数据修改 (映射) 规则，可以用于执行如下任务或其任意组合:

- 将数据源条目类型映射为其它类型
- 将数据源域映射为其它其它域
- 给条目添加新域
- 从条目移除域
- 用标准的 Perl 正则表达式匹配和替换修改域的内容。
- 将上述操作限制在来自特定数据源的条目，这些特定数据源可以在`\addresource`宏中定义。
- 将上述操作限制在某些条目类型。
- 将上述操作限制在某一特定的参考文献节。

⟨*specification*⟩是一个不限范围的`\maps`指令列表，这些指令说明了应用于某一特定数据源类型的映射规则的容器 (§ 3.7.1)。可以自由地使用空格、制表符和行末符号来整理⟨*specification*⟩内容达到视觉效果。[49] 但空行是不允许的。这一命令仅能用于导言区并且只能使用一次–后面的命令将覆盖前面的定义。

`\maps[`⟨*options*⟩`]{`⟨*elements*⟩`}`

包含`\map`元素的有序集，每个`\map`都是应用于数据源的映射步的逻辑相关集。⟨*options*⟩包括:

[49] visually arrange the ⟨*specification*⟩

`datatype`=bibtex, biblatexml default: bibtex

包含的\map应用的数据源的类型 (见 § 3.7.1)

`overwrite`=true, false default: false

具体说明一个映射规则是否允许覆盖条目中已经存在数据。如果该选项未指定，默认是 false。简易形式overwrite等价于 overwrite=true。

`\map`[⟨*options*⟩]{⟨*restrictions,steps*⟩}

A container for an ordered set of map \steps, optionally restricted to particular entrytypes or data sources. This is a grouping element to allow a set of mapping steps to apply only to specific entrytypes or data sources. Mapping steps must always be contained within a \map element. The ⟨*options*⟩ are:

`overwrite`=true, false

As the same option on the parent \maps element. This option allows an override on a per-map group basis. If this option is not specified, the default is the parent \maps element option value. The short form overwrite is equivalent to overwrite=true.

`foreach`=⟨*loopval*⟩

Loop over all \steps in this \map, setting the special variable $MAPLOOP to each of the comma-separated values contained in ⟨*loopval*⟩. ⟨*loopval*⟩ can either be the name of a datafield set defined with \DeclareDatafieldSet (see § 4.5.2), a datasource field which is fetched and parsed as a comma-separated values list or an explicit comma-separated values list. ⟨*loopval*⟩ is determined in this order. This allows the user to repeat a group of \steps for each value ⟨*loopval*⟩. Using regexp maps, it is possible to create a CSV field for use with this functionality. The special variable $MAPUNIQ may also be used the \steps to generate a random unique string. This can be useful when creating keys for new entries. An example:

```
\DeclareSourcemap{
  \maps[datatype=bibtex]{
    \map[overwrite, foreach={author,editor, translator}]{
      \step[fieldsource=\regexp{$MAPLOOP}, match={Smith}, replace={Jones
    ↪ }]
    }
  }
}
```

`refsection`=⟨*integer*⟩

Only apply the contained \step commands to entries in the reference section with number ⟨*refsection*⟩.

`\perdatasource{`⟨*datasource*⟩`}`

Restricts all `\steps` in this `\map` element to entries from the named ⟨*datasource*⟩. The ⟨*datasource*⟩ name should be exactly as given in a `\addresource` macro defining a data source for the document. Multiple `\perdatasource` restrictions are allowed within a `\map` element.

`\pertype{`⟨*entrytype*⟩`}`

Restricts all `\steps` in this `\map` element to entries of the named ⟨*entrytype*⟩. Multiple `\pertype` restrictions are allowed within a `\map` element.

`\pernottype{`⟨*entrytype*⟩`}`

Restricts all `\steps` in this `\map` element to entries not of the named ⟨*entrytype*⟩. Multiple `\pernottype` restrictions are allowed within a `\map` element.

`\step[`⟨*options*⟩`]`

A mapping step. Each step is applied sequentially to every relevant entry where 'relevant' means those entries which correspond to the data source type, entrytype and data source name restrictions mentioned above. Each step is applied to the entry as it appears after the application of all previous steps. The mapping performed by the step is determined by the following ⟨*option*⟩s:

`typesource=`⟨*entrytype*⟩

`typetarget=`⟨*entrytype*⟩

`fieldsource=`⟨*entryfield*⟩

`notfield=`⟨*entryfield*⟩

`fieldtarget=`⟨*entryfield*⟩

`match=`⟨*regexp*⟩

`notmatch=`⟨*regexp*⟩

`replace=`⟨*regexp*⟩

`fieldset=`⟨*entryfield*⟩

`fieldvalue=`⟨*string*⟩

`entryclone=`⟨*clonekey*⟩

`entrynew=`⟨*entrynewkey*⟩

`entrynewtype=`⟨*string*⟩

`entrytarget=`⟨*string*⟩

`entrynull=true, false` default: `false`

`append=true, false` default: `false`

`final=true, false` default: `false`

`null`=true, `false` default: `false`

`origfield`=true, `false` default: `false`

`origfieldval`=true, `false` default: `false`

`origentrytype`=true, `false` default: `false`

For all boolean `\step` options, the short form `option` is equivalent to `option=true`. The following rules for a mapping step apply:

- If `entrynew` is set, a new entry is created with the entry key `entrynewkey` and the entry type given in the option `entrynewtype`. This entry is only in-scope during the processing of the current entry and can be referenced by `entrytarget`. In `entrynewkey`, you may use standard Perl regular expression backreferences to captures from a previous `match` step.
- When a `fieldset` step has `entrytarget` set to the entrykey of an entry created by `entrynew`, the target for the field set will be the `entrytarget` entry rather than the entry being currently processed. This allows users to create new entries and set fields in them.
- If `entrynull` is set, processing of the `\map` immediately terminates and the current entry is not created. It is as if it did not exist in the datasource. Obviously, you should select the entries which you want to apply this to using prior mapping steps.
- If `entryclone` is set, a clone of the entry is created with an entry key `clonekey`. Obviously this may cause labelling problems in author/year styles etc. and should be used with care. The cloned entry is in-scope during the processing of the current entry and can be modified by passing its key as the value to `entrytarget`. In `clonekey`, you may use standard Perl regular expression backreferences to captures from a previous `match` step.
- Change the `typesource` ⟨*entrytype*⟩ to the `typetarget` ⟨*entrytype*⟩, if defined. If `final` is `true` then if the ⟨*entrytype*⟩ of the entry is not `typesource`, processing of the parent `\map` immediately terminates.
- Change the `fieldsource` ⟨*entryfield*⟩ to `fieldtarget`, if defined. If `final` is `true` then if there is no `fieldsource` ⟨*entryfield*⟩ in the entry, processing of the parent `\map` immediately terminates.
- If `notfield` is used then only apply the step if the ⟨*entryfield*⟩ does not exist.
- If `match` is defined but `replace` is not, only apply the step if the `fieldsource` ⟨*entryfield*⟩ matches the `match` regular expression (logic is reversed if you use `notmatch` instead)[50]. You may use capture parenthesis as usual and refer to these ($1...$9) in later `fieldvalue` specifications. This allows you to pull out parts of some fields and put these parts in other fields.

[50] Regular expressions are full Perl 5.16 regular expressions. This means you may need to deal with special characters, see examples.

- Perform a regular expression match and replace on the value of the `fieldsource` ⟨*entryfield*⟩ if `match` and `replace` are defined.
- If `fieldset` is defined, then its value is ⟨*entryfield*⟩ which will be set to a value specified by further options. If `overwrite` is false for this step and the field to set already exists then the map step is ignored. If `final` is also true for this step, then processing of the parent map stops at this point. If `append` is true, then the value to set is appended to the current value of ⟨*entryfield*⟩. The value to set is specified by a mandatory one and only one of the following options:
  - `fieldvalue` — The `fieldset` ⟨*entryfield*⟩ is set to the `fieldvalue` ⟨*string*⟩
  - `null` — The `fieldset` ⟨*entryfield*⟩ is ignored, as if it did not exist in the datasource
  - `origentrytype` — The `fieldset` ⟨*entryfield*⟩ is set to the most recently mentioned `typesource` ⟨*entrytype*⟩ name
  - `origfield` — The `fieldset` ⟨*entryfield*⟩ is set to the most recently mentioned `fieldsource` ⟨*entryfield*⟩ name
  - `origfieldval` — The `fieldset` ⟨*entryfield*⟩ is set to the most recently mentioned `fieldsource` value

With BibTeX datasources, you may specify the pseudo-field `entrykey` for `fieldsource` which is the citation key of the entry. With `biblatexml` the `entrykey` is a normal attribute and can be reference like any other attribute. Naturally, this 'field' cannot be changed (used as `fieldset`, `fieldtarget` or changed using `replace`).

`\DeclareStyleSourcemap{`⟨*specification*⟩`}`

This command sets the source mappings used by a style. Such mappings are conceptually separate from user mappings defined with `\DeclareSourcemap` and are applied directly after user maps. The syntax is identical to `\DeclareSourcemap`. This command is provided for style authors so that any maps defined for the style do not interfere with user maps or the default driver maps defined with `\DeclareDriverSourcemap`. This command is for use in style files and can be used multiple times, the maps being run in order of definition.

`\DeclareDriverSourcemap[`⟨*datatype=driver*⟩`]{`⟨*specification*⟩`}`

This command sets the driver default source mappings for the specified ⟨*driver*⟩. Such mappings are conceptually separate from user mappings defined with `\DeclareSourcemap` and style mapping defined with `\DeclareStyleSourcemap`. They consist of mappings which are part of the driver setup. Users should not normally need to change these. Driver default mapping are applied after user mappings (`\DeclareSourcemap`) and style mappings (`\DeclareStyleSourcemap`). These defaults are described in Appendix § A. The ⟨*specification*⟩ is identical to that for `\DeclareSourcemap` but without the `\maps` elements: the ⟨*specification*⟩ is just a list

of \map elements since each \DeclareDriverSourcemap only applies to one datatype driver. See the default definitions in Appendix § A for examples.

Here are some data source mapping examples:

```
\DeclareSourcemap{
  \maps[datatype=bibtex]{
    \map{
      \perdatasource{example1.bib}
      \perdatasource{example2.bib}
      \step[fieldset=keywords, fieldvalue={keyw1, keyw2}]
      \step[fieldsource=entrykey]
      \step[fieldset=note, origfieldval]
    }
  }
}
```

This would add a keywords field with value 'keyw1, keyw2' and set the note field to the entry key to all entries which are found in either the examples1.bib or examples2.bib files.

```
\DeclareSourcemap{
  \maps[datatype=bibtex]{
    \map{
      \step[fieldsource=title]
      \step[fieldset=note, origfieldval]
    }
  }
}
```

Copy the title field to the note field unless the note field already exists.

```
\DeclareSourcemap{
  \maps[datatype=bibtex]{
    \map{
      \step[typesource=chat, typetarget=customa, final]
      \step[fieldset=type, origentrytype]
    }
  }
}
```

Any chat entrytypes would become customa entrytypes and would automatically have a type field set to 'chat' unless the type field already exists in the entry (because

`overwrite` is false by default). This mapping applies only to entries of type `@chat` since the first step has `final` set and so if the `typesource` does not match the entry entrytype, processing of this `\map` immediately terminates.

```
\DeclareSourcemap{
  \maps[datatype=bibtex]{
    \map{
      \perdatasource{examples.bib}
      \pertype{article}
      \pertype{book}
       \step[fieldset=abstract, null]
       \step[fieldset=note, fieldvalue={Auto-created this field}]
    }
  }
}
```

Any entries of entrytype `@article` or `@book` from the `examples.bib` datasource would have their `abstract` fields removed and a `note` field added with value 'Auto-created this field'.

```
\DeclareSourcemap{
  \maps[datatype=bibtex]{
    \map{
       \step[fieldset=abstract, null]
       \step[fieldsource=conductor, fieldtarget=namea]
       \step[fieldsource=gps, fieldtarget=usera]
    }
  }
}
```

This removes `abstract` fields from any entry, changes `conductor` fields to `namea` fields and changes `gps` fields to `usera` fields.

```
\DeclareSourcemap{
  \maps[datatype=bibtex]{
    \map{
       \step[fieldsource=pubmedid, fieldtarget=eprint, final]
       \step[fieldset=eprinttype, origfield]
       \step[fieldset=userd, fieldvalue={Some string of things}]
    }
  }
}
```

Applies only to entries with `pubmed` fields and maps `pubmedid` fields to `eprint` fields, sets the `eprinttype` field to 'pubmedid' and also sets the `userd` field to the string 'Some string of things'.

```
\DeclareSourcemap{
  \maps[datatype=bibtex]{
    \map{
       \step[fieldsource=series,
             match=\regexp{\A\d*(.+)},
             replace=\regexp{\L$1}]
    }
  }
}
```

Here, the contents of the `series` field have leading numbers stripped and the remainder of the contents lowercased. Since regular expressions usually contain all sort of special characters, it is best to enclose them in the provided `\regexp` macro as shown—this will pass the expression through to Biber correctly.

```
\DeclareSourcemap{
  \maps[datatype=bibtex]{
    \map{
       \step[fieldsource=maintitle,
             match=\regexp{Collected\s+Works.+Freud},
             final]
       \step[fieldset=keywords, fieldvalue=freud]
    }
  }
}
```

Here, if for an entry, the `maintitle` field matches a particular regular expression, we set a special keyword so we can, for example, make a references section just for certain items.

```
\DeclareSourcemap{
  \maps[datatype=bibtex]{
    \map{
       \step[fieldsource=lista, match=\regexp{regexp}, final]
       \step[fieldset=lista, null]
    }
  }
}
```

If an entry has a `lista` field which matches regular expression 'regexp', then it is removed.

```
\DeclareSourcemap{
  \maps[datatype=bibtex]{
    \map[overwrite=false]{
       \step[fieldsource=author]
       \step[fieldset=editor, origfieldval, final]
       \step[fieldsource=editor, match=\regexp{\A(.+?)\s+and.*}, replace
    ↪ ={$1}]
    }
  }
}
```

For any entry with an `author` field, try to set `editor` to the same as `author`. If this fails because `editor` already exists, stop, otherwise truncate `editor` to just the first name in the name list.

```
\DeclareSourcemap{
  \maps[datatype=bibtex]{
    \map{
       \step[fieldsource=author,
             match={Smith, Bill},
             replace={Smith, William}]
       \step[fieldsource=author,
             match={Jones, Baz},
             replace={Jones, Barry}]
    }
  }
}
```

Here, we use multiple match/replace for the same field to regularise some inconstant name variants. Bear in mind that `\step` processing within a `map` element is sequential and so the changes from a previous `\steps` are already committed. Note that we don't need the `\regexp` macro to protect the regular expressions in this example as they contain no characters which need special escaping. Please note that due to the difficulty of protecting regular expressions in LaTeX, there should be no literal spaces in the argument to `\regexp`. Please use escape code equivalents if spaces are needed. For example, this example, if using `\regexp`, should be:

```
\DeclareSourcemap{
  \maps[datatype=bibtex]{
    \map{
```

```
      \step[fieldsource=author,
            match=\regexp{Smith,\s+Bill},
            replace=\regexp{Smith,\x20William}]
      \step[fieldsource=author,
            match=\regexp{Jones,\s+Baz},
            replace=\regexp{Jones,\x20Barry}]
    }
  }
}
```

Here, we have used the hexadecimal escape sequence '`\x20`' in place of literal spaces in the replacement strings.

```
\DeclareSourcemap{
  \maps[datatype=bibtex]{
    \map[overwrite]{
      \step[fieldsource=author, match={Doe,}, final]
      \step[fieldset=shortauthor, origfieldval]
      \step[fieldset=sortname, origfieldval]
      \step[fieldsource=shortauthor,
            match=\regexp{Doe,\s*(?:\.|ohn)(?:[-]*)(?:P\.|Paul)*},
            replace={Doe, John Paul}]
      \step[fieldsource=sortname,
            match=\regexp{Doe,\s*(?:\.|ohn)(?:[-]*)(?:P\.|Paul)*},
            replace={Doe, John Paul}]
    }
  }
}
```

Only applies to entries with an `author` field matching 'Doe,'. First the `author` field is copied to both the `shortauthor` and `sortname` fields, overwriting them if they already exist. Then, these two new fields are modified to canonicalise a particular name, which presumably has some variants in the data source.

```
\DeclareSourcemap{
  \maps[datatype=bibtex]{
    \map[overwrite]{
      \step[fieldsource=verba, final]
      \step[fieldset=verbb, fieldvalue=/, append]
      \step[fieldset=verbb, origfieldval, append]
      \step[fieldsource=verbb, final]
      \step[fieldset=verbc, fieldvalue=/, append]
      \step[fieldset=verbc, origfieldval, append]
```

```
    }
  }
}
```

This example demonstrates the sequential nature of the step processing and the `append` option. If an entry has a `verba` field then first, a forward slash is appended to the `verbb` field. Then, the contents of `verba` are appended to the `verbb` field. A slash is then appended to the `verbc` field and the contents of `verbb` are appended to the `verbc` field.

```
\DeclareSourcemap{
  \maps[datatype=bibtex]{
    \map[overwrite]{
      \step[fieldset=autourl, fieldvalue={http://scholar.google.com/
    ↪ scholar?q="}]
      \step[fieldsource=title]
      \step[fieldset=autourl, origfieldval, append]
      \step[fieldset=autourl, fieldvalue={"+author:}, append]
      \step[fieldsource=author, match=\regexp{\A([^,]+)\s*,}]
      \step[fieldset=autourl, fieldvalue={$1}, append]
      \step[fieldset=autourl, fieldvalue={&as_ylo=}, append]
      \step[fieldsource=year]
      \step[fieldset=autourl, origfieldval, append]
      \step[fieldset=autourl, fieldvalue={&as_yhi=}, append]
      \step[fieldset=autourl, origfieldval, append]
    }
  }
}
```

This example assumes you have created a field called `autourl` using the datamodel macros from § 4.5.4 in order to hold, for example, a Google Scholar query URL auto-created from elements of the entry. The example progressively extracts information from the entry, constructing the URL as it goes. It demonstrates that it is possible to refer to parenthetical matches from the most recent `match` in any following `fieldvalue` which allows extracting the family name from the `author`, assuming a 'family, given' format. The resulting field could then be used as a hyperlink from, for example, the title of the work in the bibliography.

```
\DeclareSourcemap{
  \maps[datatype=bibtex]{
    \map{
      \step[fieldsource=title, match={A Title}, final]
      \step[entrynull]
    }
```

```
  }
}
```

Any entry with a `title` field matching 'A Title' will be completely ignored.

```
\DeclareSourcemap{
  \maps[datatype=bibtex]{
    \map{
      \pernottype{book}
      \pernottype{article}
      \step[entrynull]
    }
  }
}
```

Any entry which is not a `@book` or `@article` will be ignored.

```
\DeclareSourcemap{
  \maps[datatype=bibtex]{
    \map{
      \perdatasource{biblatex-examples.bib}
      \step[entryclone={rel-}]
    }
  }
}
```

Here, a clone of an entry from the specified data source will be created. The entry key of the clone will be the same as the original but prefixed by the value of the `entryclone` parameter. The cloned entry would still need to be cited in the document using its new entry key. This type of mapping step should be used with care as it may produce labelling problems in authoryear styles which use, for example, `extrayear`. One use case is for numeric styles which contain multiple bibliographies containing the same entry. In this case, you may need different bibliography number labeld for the same entry and this is very tricky when there is only one entry which needs different labels. Creating clones with different entry keys solves this problem.

`biblatexml` datasources are more structured than BibTeX since they are XML. Sourcemapping is possible with `biblatexml` too but the specifications of source and target fields etc. also support XPath 1.0 paths in order to be able to work with the structured data. Fields can be specified as per the BibTeX examples above and these are converted into XPath 1.0 queries internally as necessary. For example:

```
\DeclareSourcemap{
  \maps[datatype=biblatexml]{
```

```
    \map{
  \step[fieldsource=\regexp{./bltx:names[@type='author']/bltx:name[2]/
    ↪ bltx:namepart[@type='family']},
      match=\regexp{\ASmith},
      replace={Jones}]
    }
    \map{
      \step[fieldsource=editor, fieldtarget=translator]
    }
    \map{
      \step[fieldsource=\regexp{./bltx:names[@type='editor']},
            fieldtarget=\regexp{./bltx:names[@type='translator']}]
    }
    \map{
      \step[fieldset=\regexp{./bltx:names[@type='author']/bltx:name[2]/
    ↪ @useprefix},
            fieldvalue={false}]
    }
  }
}
```

These maps, respectively,

- Replace the family name 'Smith' of the second `author` name with 'Jones'
- Move the `editor` to `translator`
- Move the `editor` to `translator` but with explicit XPaths
- Set the per-namelist `useprefix` option on the `author` name list to 'false'

### 4.5.4 数据模型规范

Biblatex 使用的数据模型包括 4 个主要元素:

- Specification of constant strings and lists of strings
- Specification of valid Entrytypes
- Specification of valid Fields along with their type, datatype and any special flags
- Specification of which Fields are valid in which Entrytypes
- Specification of constraints which can be used to validate data against the data model

The default data model is defined in the core Biblatex file `blx-dm.def` using the macros described in this section. The default data model is described in detail in § 2. The data model is used internally by Biblatex and also by the backend. In practice, changing the data model means that you can define the entrytypes and fields for your datasources and validate your data against the data model. Naturally, this is not much use unless your style supports any new entrytypes or fields and it raises issues of portability between styles (although this can be mitigated by using the dynamic data modification functionality described in § 4.5.3).

Validation against the data model means that after mapping your data sources into the data model, Biber (using its `--validate_datamodel` option) can check:

- Whether all entrytypes are valid entrytypes
- Whether all fields are valid fields for their entrytype
- Whether the fields obey various constraints on their format which you specify

Redefining the data model can be done in several places. Style authors can create a `.dbx` file which contains the data model macros required and this will be loaded automatically when using the Biblatex package `style` option by looking for a file named after the style with a `.dbx` extension (just like the `.cbx` and `.bbx` files for a style). If the `style` option is not used but rather the `citestyle` and `bibstyle` options, then the package will try to load `.dbx` files called `<citestyle>.dbx` and `<bibstyle>.dbx`. Alternatively, the name of the data model file can be different from any of the style option names by specifying the name (without `.dbx` extension) to the package `datamodel` option. After loading the style data model file, Biblatex then loads, if present, a users `biblatex-dm.cfg` which should be put somewhere Biblatex can find it, just like the main configuration file `biblatex.cfg`. To summarise, the data model is determined by adding to the data model from each of these locations, in order:

```
blx-dm.def→
  <datamodel option>.dbx →
    <style option>.dbx →
      <citestyle option>.dbx and <bibstyle option>.dbx →
        biblatex-dm.cfg
```

It is not possible to add to a loaded data model by using the macros below in your preamble as the preamble is read after Biblatex has defined critical internal macros based on the data model. If any data model macro is used in a document, it will be ignored and a warning will be generated. The data model is defined using the following macros:

`\DeclareDatamodelConstant[`⟨*options*⟩`]{`⟨*name*⟩`}{`⟨*constantdef*⟩`}`

Declares the ⟨*name*⟩ as a datamodel constant with definition ⟨*constantdef*⟩. Such constants are typically used internally by Biber.

`type=string, list` default: `string`

A constant can be a simple string (default if the ⟨*type*⟩ option is omitted) or a comma-separated list of strings.

`\DeclareDatamodelEntrytypes[`⟨*options*⟩`]{`⟨*entrytypes*⟩`}`

Declares the comma-separated list of ⟨*entrytypes*⟩ to be valid entrytypes in the data model. As usual in TeX csv lists, make sure each element is immediately followed by a comma or the closing brace—no extraneous whitespace.

`skipout=true, false` default: `false`

This entrytype is not output to the `.bbl`. Typically used for special entrytypes which are processed and consumed by the backend such as `@xdata`.

`\DeclareDatamodelFields[`⟨*options*⟩`]{`⟨*fields*⟩`}`

Declares the comma-separated list of ⟨*fields*⟩ to be valid fields in the data model with associated comma-separated ⟨*options*⟩. The ⟨*type*⟩ and ⟨*datatype*⟩ options are mandatory. All valid ⟨*options*⟩ are:

`type=`⟨*field type*⟩

Set the type of the field as described in § 2.2.1, typically 'field' or 'list'.

`format=`⟨*field format*⟩

Any special format of the field. Normally unspecified but can take the value 'xsv' which tells Biber that this field has a separated values format. The exact separator can be controlled with the Biber option `xsvsep` and defaults to the expected comma surrounded by optional whitespace.

`datatype=`⟨*field datatype*⟩

Set the datatype of the field as described in § 2.2.1. For example, 'name' or 'literal'.

`nullok=true, false` default: `false`

The field is allowed to be defined but empty.

`skipout=true, false` default: `false`

The field is not output to the `.bbl` and is therefore not present during Biblatex style processing. As usual in TeX csv lists, make sure each element is immediately followed by a comma or the closing brace—no extraneous whitespace.

`label=true, false` default: `false`

The field can be used as a label in a bibliography or bibliography list. Specifying this causes Biblatex to create several helper macros for the field so that there are some internal lengths and headings etc. defined.

`\DeclareDatamodelEntryfields[`⟨*entrytypes*⟩`]{`⟨*fields*⟩`}`

Declares that the comma-separated list of ⟨*fields*⟩ is valid for the comma-separated list of ⟨*entrytypes*⟩. If ⟨*entrytypes*⟩ is not given, the fields are valid for all entrytypes. As usual in TeX csv lists, make sure each element is immediately followed by a comma or the closing brace—no extraneous whitespace.

`\DeclareDatamodelConstraints[`⟨*entrytypes*⟩`]{`⟨*specification*⟩`}`

If a comma-separated list of ⟨*entrytypes*⟩ is given, the constraints apply only to those entrytypes. The ⟨*specification*⟩ is an undelimited list of `\constraint` directives which specify a constraint. Spaces, tabs, and line endings may be used freely to visually arrange the ⟨*specification*⟩. Blank lines are not permissible.

`\constraint[`⟨*type=constrainttype*⟩`]{`⟨*elements*⟩`}`

Specifies a constraint of type ⟨*constrainttype*⟩. Valid constraint types are:

`type=data`, `mandatory`, `conditional`

Constraints of type 'data' put restrictions on the value of a field. Constraints of type 'mandatory' specify which fields or combinations of fields an entrytype should have. Constraints of type 'conditional' allow more sophisticated conditional and quantified field constraints.

`datatype=integer`, `isbn`, `issn`, `ismn`, `date`, `pattern`

For constraints of type ⟨*data*⟩, constrain field values to be the given datatype.

`rangemin=`⟨*num*⟩

For constraints of ⟨*type*⟩ 'data' and ⟨*datatype*⟩ 'integer', constrain field values to be at least ⟨*num*⟩.

`rangemax=`⟨*num*⟩

For constraints of ⟨*type*⟩ 'data' and ⟨*datatype*⟩ 'integer', constrain field values to be at most ⟨*num*⟩.

`pattern=`⟨*patt*⟩

For constraints of ⟨*type*⟩ 'data' and ⟨*datatype*⟩ 'pattern', constrain field values to match regular expression pattern ⟨*patt*⟩. It is best to wrap any regular expression in the macro `\regexp`, see § 4.5.3.

A `\constraint` macro may contain any of the following:

`\constraintfieldsor{`⟨*fields*⟩`}`

For constraints of ⟨*type*⟩ 'mandatory', specifies that an entry must contain a boolean OR of the `\constraintfields`.

`\constraintfieldsxor{`⟨*fields*⟩`}`

For constraints of ⟨*type*⟩ 'mandatory', specifies that an entry must contain a boolean XOR of the `\constraintfields`.

`\antecedent[`⟨*quantifier=quantspec*⟩`]{`⟨*fields*⟩`}`

For constraints of ⟨*type*⟩ 'conditional', specifies a quantified set of `\constraintfields` which must be satisfied before the `\consequent` of the constraint is checked. ⟨*quantspec*⟩ should have one of the following values:

`quantifier=all`, `one`, `none`

Specifies how many of the `\constrainfield`'s inside the `\antecedent` have to be present to satisfy the antecedent of the conditional constraint.

`\consequent[`⟨*quantifier=quantspec*⟩`]{`⟨*fields*⟩`}`

For constraints of ⟨*type*⟩ 'conditional', specifies a quantified set of `\constraintfields` which must be satisfied if the preceding `\antecedent` of the constraint was satisfied. ⟨*quantspec*⟩ should have one of the following values:

`quantifier=all`, `one`, `none`

Specifies how many of the `\constraintfield`'s inside the `\consequent` have to be present to satisfy the consequent of the conditional constraint.

`\constraintfield{`⟨*field*⟩`}`

For constraints of ⟨*type*⟩ 'data', the constraint applies to this ⟨*field*⟩. For constraints of ⟨*type*⟩ 'mandatory', the entry must contain this ⟨*field*⟩.

The data model declaration macros may be used multiple times as they append to the previous definitions. In order to replace, change or remove existing definitions (such as the default model which is loaded with Biblatex), you should reset (clear) the current definition and then set what you want using the following macros. Typically, these macros will be the first things in any `biblatex-dm.cfg` file:

`\ResetDatamodelEntrytypes`

Clear all data model entrytype information.

`\ResetDatamodelFields`

Clear all data model field information.

`\ResetDatamodelEntryfields`

Clear all data model fields for entrytypes information.

`\ResetDatamodelConstraints`

Clear all data model fields Constraints information.

Here is an example of a simple data model. Refer to the core Biblatex file `blx-dm.def` for the default data model specification.

```
\ResetDatamodelEntrytypes
\ResetDatamodelFields
\ResetDatamodelEntryfields
\ResetDatamodelConstraints

\DeclareDatamodelEntrytypes{entrytype1, entrytype2}

\DeclareDatamodelFields[type=field, datatype=literal]{field1,field2,
    ↪ field3,field4}

\DeclareDatamodelEntryfields{field1}
\DeclareDatamodelEntryfields[entrytype1]{field2,field3}
\DeclareDatamodelEntryfields[entrytype2]{field2,field3,field4}

\DeclareDatamodelConstraints[entrytype1]{
  \constraint[type=data, datatype=integer, rangemin=3, rangemax=10]{
    \constraintfield{field1}
  }
  \constraint[type=mandatory]{
    \constraintfield{field1}
    \constraintfieldsxor{
      \constraintfield{field2}
      \constraintfield{field3}
    }
  }
}
\DeclareDatamodelConstraints{
  \constraint[type=conditional]{
    \antecedent[quantifier=none]{
      \constraintfield{field2}
    }
    \consequent[quantifier=all]{
      \constraintfield{field3}
      \constraintfield{field4}
    }
  }
}
```

This model specifies:

- Clear the default data model completely
- Two valid entry types `@entrytype1` and `@entrytype2`

- Four valid literal field fields
- `field1` is valid for all entrytypes
- `field2` and `field3` are valid for `entrytype1`
- `field2`, `field3` and `field4` are valid for `@entrytype2`
- For `@entrytype1`:
  - `field1` must be an integer between 3 and 10
  - `field1` must be present
  - One and only one of `field2` or `field3` must be present
- For any entrytype, if `field2` is not present, `field3` and `field4` must be present

### 4.5.5 标签

字母顺序制样式使用一个标签来区分参考文献条目。这个标签由条目的内容使用一个描述怎么构建标签的模板构建。该模板可以全局自定义或者分条目类型定义。如何抽取姓名域的部分作为标签使用一个独立的模板，因为姓名域是相当复杂的域。标签的自定义需要用 Biber 后端程序而不能用其它后端程序。

`\DeclareLabelalphaTemplate[`⟨*entrytype, ...*⟩`]{`⟨*specification*⟩`}`

Defines the alphabetic label template for the given entrytypes. If no entrytypes are specified in the first argument, then the global label template is defined. The ⟨*specification*⟩ is an undelimited list of `\labelelement` directives which specify the elements used to build the label. Spaces, tabs, and line endings may be used freely to visually arrange the ⟨*specification*⟩. Blank lines are not permissible. This command may only be used in the preamble.

`\labelelement{`⟨*elements*⟩`}`

Specifies the elements used to build the label. The ⟨*elements*⟩ are an undelimited list of `\field` or `\literal` commands which are evaluated in the order in which they are given. The first `\field` or `\literal` which expands to a non-empty string is used as the `\labelelement` expansion and the next `\labelelement`, if any, is then processed.

`\field[`⟨*options*⟩`]{`⟨*field*⟩`}`

If ⟨*field*⟩ is non-empty, use it as the current label `\labelelement`, subject to the options below. Useful values for ⟨*field*⟩ are typically the name list type fields, date fields, and title fields. You may also use the 'citekey' pseudo-field to specify the citation key as part of the label. Name list fields are treated specially and when a name list field is specified, the template defined with `\DeclareLabelalphaNameTemplate` is used to extract parts from the name which then returns the string that the `\field` option uses.

`final`=`true`, `false` default: `false`

This option marks a `\field` directive as the final one in the ⟨*specification*⟩. If the ⟨*field*⟩ is non-empty, then this field is used for the label and the remainder of the ⟨*specification*⟩ will be ignored. The short form `final` is equivalent to `final=true`.

`lowercase`=`true`, `false` default: `false`

Forces the label part derived from the field to lowercase. By default, the case is taken from the field source and not modified.

`strwidth`=⟨*integer*⟩ default: `1`

The number of characters of the ⟨*field*⟩ to use. This setting may be overridden by an individual name part when extracting characters from a name. See `\DeclareLabelalphaNameTemplate` below.

`strside`=`left`, `right` default: `left`

The side of the string from which to take the `strwidth` number of characters. This setting may be overridden by an individual name part when extracting characters from a name. See `\DeclareLabelalphaNameTemplate` below.

`padside`=`left`, `right` default: `right`

Side to pad the label part when using the `padchar` option. Only for use with fixed-width label strings (`strwidth`).

`padchar`=⟨*character*⟩

If present, pads the label part on the `padside` side with the specified character to the length of `strwidth`. Only for use with fixed-width label strings (`strwidth`).

`uppercase`=`true`, `false` default: `false`

Forces the label part derived from the field to uppercase. By default, the case is taken from the field source and not modified.

`varwidth`=`true`, `false` default: `false`

Use a variable width, left-side substring of characters from the string returned for ⟨*field*⟩. The length of the string is determined by the minimum length needed to disambiguate the substring from all other ⟨*field*⟩ elements in the same position in the label. For name list fields, this means that each name substring is disambiguated from all other name substrings which occur in the same position in the name list (see examples below). This option overrides `strwidth` if both are used. The short form `varwidth` is equivalent to `varwidth=true`. For name list fields, the `\nameparts` with the `pre` option set are prepended to the string returned from this disambiguation.

`varwidthnorm`=`true`, `false` default: `false`

As `varwidth` but will force the disambiguated substrings for the ⟨*field*⟩ to be the same length as the longest disambiguated substring. This can be used to regularise the format of the labels if desired. This option overrides `strwidth` if both are used. The short form `varwidthnorm` is equivalent to `varwidthnorm=true`.

`varwidthlist`=`true`, `false` default: `false`

Alternative method of automatic label disambiguation where the field as a whole is disambiguated from all other fields in the same label position. For non-name list fields, this is equivalent to `varwidth`. For name list fields, names in a name list are not disambiguated from other names in the same position in their name lists but instead the entire name list is disambiguated as a whole from other name lists (see examples below). This option overrides `strwidth` if both are used. The short form `varwidthlist` is equivalent to `varwidthlist=true`. For name list fields, the `\nameparts` with the `pre` option set are prepended to the string returned from this disambiguation.

`strwidthmax`=⟨*integer*⟩

When using `varwidth`, this option sets a limit (in number of characters) on the length of variable width substrings. This option can be used to regularise the label.

`strfixedcount`=⟨*integer*⟩ default: 1

When using `varwidthnorm`, there must be at least `strfixedcount` disambiguated substrings with the same, maximal length to trigger the forcing of all disambiguated substrings to this same maximal length.

`ifnames`=⟨*range*⟩

Only use this `\field` specification if it is a name list field with a number of names matching the `ifnames` range value. This allows a `\labelelement` to be conditionalised on name length (see below). The range can specified as in the following examples:

```
ifnames=3     -> Only apply to name lists containing exactly 3 names
ifnames={2-4} -> Only apply to name lists containing minimum 2 and
    ↪ maximum 4 names
ifnames={-3}  -> Only apply to name lists containing at most 3 names
ifnames={2-}  -> Only apply to name lists containing at least 2 names
```

`names`=⟨*range*⟩

By default, for name list fields, the names used range from the first name to the `maxalphanames`/`minalphanames` truncation. This option can be used to override this with an explicit range of names to consider. The plus '+' sign is a special end of range marker denoting the truncation point of max/minalphanames. The range separator can be any number of characters with the Unicode Dash property. For example:

```
name=3     -> Use first 3 names in the name list
name={2-3} -> Use second and thirds names only
name={-3}  -> Same as 1-3
name={2-}  -> Use all names starting with the second name (ignoring max/
    ↪ minalphanames truncation)
name={2-+} -> Use all names starting with the second name (respecting max
    ↪ /minalphanames truncation)
```

namessep=⟨*string*⟩ default: empty

An arbitrary string separator to put between names in a namelist.

noalphaothers=true, false default: false

By default, \labelalphaothers is appended to label parts derived from name lists if there are more names in the list than are shown in the label part. This option can be used to disable the default behaviour.

\literal{⟨*characters*⟩}

Insert the literal ⟨*characters*⟩ into the label at this point.

When a name list \field is specified, the method of extracting the string is specified by a separate template specified by the following command:

\DeclareLabelalphaNameTemplate[⟨*entrytype, ...*⟩]{⟨*specification*⟩}

Specifies the template to use to extract a label string from a name list when a \field specification in \DeclareLabelalphaTemplate contains a name list. The template can be specified per-entrytype.

\namepart[⟨*options*⟩]{⟨*namepart*⟩}

⟨*namepart*⟩ is one of the datamodel nameparts defined with the \DeclareDatamodelConstant command (see § 4.2.3). The options are:

use=true, false default: false

Only use the ⟨*namepart*⟩ in constructing the label information if there is a corresponding option use'namepart' and that option is true.

pre=true, false default: false

When constructing label strings from names, the \namepart *without* a pre option will be used to construct label string, passing through disambiguation, substring etc. operations as specified by the \field options in \DeclareLabelalpaTemplate. Then the \namepart options *with* the pre option set will be prepended to the result, (in the order given, if there are more than one such \nameparts). This allows to unconditionally prepend certain namepart information to name label strings, like name prefices. Note that the uppercase and lowercase options of \field in \DeclareLabelalphaTemplate are applied to the entire label returned from \DeclareLabelalphaTemplate, both pre parts and non pre.

compound=true, false default: false

For static (non-varwidth) disambiguation in \DeclareLabelalphaTemplate, nameparts separated by whitespace or hyphens (compound names) as separate names for label generation. This means that when forming a label out of, for example the surname 'Ballam Forsyth' with a 1 character, left-side substring, this name would give 'BF' with compound=true and 'B' with compound=false. The short form compound is equivalent to compound=true.

`strwidth`=⟨*integer*⟩ default: 1

The number of characters of the ⟨*namepart*⟩ to use.

`strside`=`left`, `right` default: `left`

The side of the string from which to take the `strwidth` number of characters.

Note that the templates for labels can be defined per-type and you should be aware of this when using the automatically disambiguated label functionality. Disambiguation is not per-type as this might lead to ambiguity due to different label formats for different types being isolated from each others disambiguation process. Normally, you will want to use very different label formats for different types to make the type obvious by the label.

Here are some examples. The default global Biblatex alphabetic label template is defined below. Firstly, `shorthand` has `final=true` and so if there is a `shorthand` field, it is used as the label and nothing more of the template is considered. Next, the `label` field is used as the first label element if it exists. Otherwise, if there is only one name (`ifnames=1`) in the `labelname` list, then three characters from the left side of the family name in the `labelname` are used as the first label element. If the `labelname` has more than one name in it, one character from the left side of each family name is used as the first label element. The second label element consists of 2 characters from the right side of the `year` field.

The default template for constructing labels from names is also shown. This prepends the first character from the left side of any prefix (if the `useprefix` option is true) to a label extracted from the family name (according to the options on the calling `\field` option from `\DeclareLabelalphaTemplate`), allowing for compound family names.

```
\DeclareLabelalphaTemplate{
  \labelelement{
    \field[final]{shorthand}
    \field{label}
    \field[strwidth=3,strside=left,ifnames=1]{labelname}
    \field[strwidth=1,strside=left]{labelname}
  }
  \labelelement{
    \field[strwidth=2,strside=right]{year}
  }
}

\DeclareLabelalphaNameTemplate{
  \namepart[use=true, pre=true, strwidth=1, compound=true]{prefix}
  \namepart{family}
}
```

To get an idea of how the label automatic disambiguation works, consider the following author lists:

```
Agassi, Chang, Laver   (2000)
Agassi, Connors, Lendl (2001)
Agassi, Courier, Laver (2002)
Borg, Connors, Edberg  (2003)
Borg, Connors, Emerson (2004)
```

Assuming a template declaration such as:

```
\DeclareLabelalphaTemplate{
  \labelelement{
    \field[varwidth]{labelname}
  }
}
```

Then the labels would be:

```
Agassi, Chang, Laver    [AChLa]
Agassi, Connors, Lendl  [AConLe]
Agassi, Courier, Laver  [ACouLa]
Borg, Connors, Edberg   [BConEd]
Borg, Connors, Emerson  [BConEm]
```

With normalised variable width labels defined:

```
\DeclareLabelalphaTemplate{
  \labelelement{
    \field[varwidthnorm]{labelname}
  }
}
```

You would get the following as the substrings of names in each position are extended to the length of the longest substring in that same position:

```
Agassi, Chang, Laver    [AChaLa]
Agassi, Connors, Lendl  [AConLe]
Agassi, Courier, Laver  [ACouLa]
Borg, Connors, Edberg   [BConEd]
Borg, Connors, Emerson  [BConEm]
```

With a restriction to two characters for the name components of the label element defined like this:

```
\DeclareLabelalphaTemplate{
  \labelelement{
    \field[varwidthnorm,strwidthmax=2]{labelname}
  }
}
```

This would be the result (note that the individual family name label parts are no longer unambiguous):

```
Agassi, Chang, Laver    [AChLa]
Agassi, Connors, Lendl  [ACoLe]
Agassi, Courier, Laver  [ACoLa]
Borg, Connors, Edberg   [BCoEd]
Borg, Connors, Emerson  [BCoEm]
```

Alternatively, you could choose to disambiguate the name lists as a whole with:

```
\DeclareLabelalphaTemplate{
  \labelelement{
    \field[varwidthlist]{labelname}
  }
}
```

Which would result in:

```
Agassi, Chang, Laver    [AChL]
Agassi, Connors, Lendl  [ACoL]
Agassi, Courier, Laver  [ACL]
Borg, Connors, Edberg   [BCEd]
Borg, Connors, Emerson  [BCE]
```

Perhaps you only want to consider at most two names for label generation but disambiguate at the whole name list level:

```
\DeclareLabelalphaTemplate{
  \labelelement{
    \field[varwidthlist,names=2]{labelname}
  }
}
```

Which would result in:

```
Agassi, Chang, Laver    [ACh+]
Agassi, Connors, Lendl  [ACo+]
Agassi, Courier, Laver  [AC+]
Borg, Connors, Edberg   [BC+a]
Borg, Connors, Emerson  [BC+b]
```

In this last example, you can see `\labelalphaothers` has been appended to show that there are more names. The last two labels now require disambiguating with `\extraalpha` as there is no way of disambiguating this label name list with only two names.

Finally, here is an example using multiple label elements:

```
\DeclareLabelalphaTemplate{
  \labelelement{
    \field[varwidthlist]{labelname}
  }
  \labelelement{
    \literal{-}
  }
  \labelelement{
    \field[strwidth=3,strside=right]{labelyear}
  }
}
```

Which would result in:

```
Agassi, Chang, Laver    [AChL-000]
Agassi, Connors, Lendl  [AConL-001]
Agassi, Courier, Laver  [ACouL-002]
Borg, Connors, Edberg   [BCEd-003]
Borg, Connors, Emerson  [BCEm-004]
```

Here is another rather contrived example showing that you don't need to specially quote LaTeX special characters (apart from '%', obviously) when specifying padding characters and literals:

```
\DeclareLabelalphaTemplate{
  \labelelement{
    \literal{>}
  }
  \labelelement{
    \literal{\%}
```

```
  }
  \labelelement{
    \field[namessep={/}, strwidth=4, padchar=_]{labelname}
  }
  \labelelement{
    \field[strwidth=3, padchar=&, padside=left]{title}
  }
  \labelelement{
    \field[strwidth=2,strside=right]{year}
  }
}
```

which given:

```
@Book{test,
  author    = {XXX YY and WWW ZZ},
  title     = {T},
  year      = {2007},
}
```

would resulting a label looking like this:

```
[>%YY/ZZ__&&T07]
```

Generating labels from fields may involve some difficulties when you have fields containing diacritics, hyphens, spaces etc. Often, you want to ignore things like separator characters or spaces when generating labels. An option is provided to customise the regular expression(s) to strip from a field before it is passed to the label generation system.

`\DeclareNolabel{`⟨*specification*⟩`}`

Defines regular expressions to strip from any field before generating a label part for the field. The ⟨*specification*⟩ is an undelimited list of `\nolabel` directives which specify the regular expressions to remove from fields. Spaces, tabs and line endings may be used freely to visually arrange the ⟨*specification*⟩. Blank lines are not permissible. This command may only be used in the preamble.

`\nolabel{`⟨*regexp*⟩`}`

Any number of `\nolabel` commands can be given each of which specifies to remove the ⟨*regexp*⟩ from the copy of the field which the label generation system sees. Since regular expressions usually contain special characters, it is best to enclose them in the provided `\regexp` macro as shown—this will pass the expression through to Biber correctly.

If there is no `\DeclareNolabel` specification, Biber will default to:

```
\DeclareNolabel{
  % strip punctuation, symbols, separator and control characters
  \nolabel{\regexp{[\p{P}\p{S}\p{C}]+}}
}
```

This Biber default strips punctuation, symbol, separator and control characters from fields before passing the field string to the label generation system.

`\DeclareNolabelwidthcount{`⟨*specification*⟩`}`

Defines regular expressions to ignore from any field when counting characters in fixed-width substrings. The ⟨*specification*⟩ is an undelimited list of `\nolabelwidthcount` directives which specify the regular expressions to ignore when counting characters for fixed-width substrings. Spaces, tabs and line endings may be used freely to visually arrange the ⟨*specification*⟩. Blank lines are not permissible. This command may only be used in the preamble.

`\nolabelwidthcount{`⟨*regexp*⟩`}`

Any number of `\nolabelwidthcount` commands can be given each of which specifies to ignore the ⟨*regexp*⟩ when generating fixed-width substrings during label generation. Since regular expressions usually contain special characters, it is best to enclose them in the provided `\regexp` macro as shown—this will pass the expression through to Biber correctly.

There is no default `\DeclareNolabelwidthcount` specification. Note that this setting is only taken into account when using fixed-width substrings (non-varwidth) during label part generation. See § 4.5.5.

### 4.5.6 排序

In addition to the predefined sorting schemes discussed in § 3.5, it is possible to define new ones or modify the default definitions. The sorting process may be customized further by excluding certain fields from sorting on a per-type basis and by automatically populating the `presort` field on a per-type basis.

`\DeclareSortingScheme[`⟨*options*⟩`]{`⟨*name*⟩`}{`⟨*specification*⟩`}`

Defines the sorting scheme ⟨*name*⟩. The ⟨*name*⟩ is the identifier passed to the `sorting` option (§ 3.1.2.1) when selecting the sorting scheme. The `\DeclareSortingScheme` command supports the following optional arguments:

`locale=`⟨*locale*⟩

The locale for the sorting scheme which then overrides the global sorting locale in the `sortlocale` option discussed in § 3.1.2.1.

The ⟨*specification*⟩ is an undelimited list of `\sort` directives which specify the elements to be considered in the sorting process. Spaces, tabs, and line endings may be used freely to visually arrange the ⟨*specification*⟩. Blank lines are not permissible. This command may only be used in the preamble.

`\sort{`⟨*elements*⟩`}`

Specifies the elements considered in the sorting process. The ⟨*elements*⟩ are an undelimited list of `\field`, `\literal`, and `\citeorder` commands which are evaluated in the order in which they are given. If an element is defined, it is added to the sort key and the sorting routine skips to the next `\sort` directive. If it is undefined, the next element is evaluated. Since literal strings are always defined, any `\literal` commands should be the sole or the last element in a `\sort` directive. All ⟨*elements*⟩ should be the same datatype as described in § 2.2.2 since they will be potentially compared to any of the other ⟨*elements*⟩ in other entries.. The `\sort` command supports the following optional arguments:

`locale=`⟨*locale*⟩

Override the locale used for sorting at the level of a particular set of sorting elements. If specified, the locale overrides the locale set at the level of `\DeclareSortingScheme` and also the global setting. See also the discussion of the global sorting locale option `sortlocale` in § 3.1.2.1.

`direction=ascending, descending` default: `ascending`

The sort direction, which may be either `ascending` or `descending`. The default is ascending order.

`final=true, false` default: `false`

This option marks a `\sort` directive as the final one in the ⟨*specification*⟩. If one of the ⟨*elements*⟩ is available, the remainder of the ⟨*specification*⟩ will be ignored. The short form `final` is equivalent to `final=true`.

`sortcase=true, false`

Whether or not to sort case-sensitively. The default setting depends on the global `sortcase` option.

`sortupper=true, false`

Whether or not to sort in 'uppercase before lowercase' (`true`) or 'lowercase before uppercase' order (`false`). The default setting depends on the global `sortupper` option.

`\field[`⟨*key=value, ...*⟩`]{`⟨*field*⟩`}`

The `\field` element adds a ⟨*field*⟩ to the sorting specification. If the ⟨*field*⟩ is undefined, the element is skipped. The `\field` command supports the following optional arguments:

`padside=left, right` default: `left`

Pads a field on the `left` or `right` side using `padchar` so that its width is `padwidth`. If no padding option is set, no padding is done at all. If any padding option is specified, then padding is performed and the missing options are assigned built-in default values. If padding and substring matching are both specified, the substring match is performed first.

`padwidth=`⟨*integer*⟩ default: `4`

The target width in characters.

`padchar=`⟨*character*⟩ default: `0`

The character to be used when padding the field.

`strside=left, right` default: `left`

Performs a substring match on the `left` or `right` side of the field. The number of characters to match is specified by the corresponding `strwidth` option. If no substring option is set, no substring matching is performed at all. If any substring option is specified, then substring matching is performed and the missing options are assigned built-in default values. If padding and substring matching are both specified, the substring match is performed first.

`strwidth=`⟨*integer*⟩ default: `4`

The number of characters to match.

`\literal{`⟨*string*⟩`}`

The `\literal` element adds a literal ⟨*string*⟩ to the sorting specification. This is useful as a fallback if some fields are not available.

`\citeorder` The `\citeorder` element has a special meaning. It requests a sort based on the lexical order of the actual citations. For entries cited within the same citation command like:

```
\cite{one,two}
```

there is a distinction between the lexical order and the semantic order. Here “one” and “two” have the same semantic order but a unique lexical order. The semantic order only matters if you specify further sorting to disambiguate entries with the same semantic order. For example, this is the definition of the `none` sorting scheme:

```
\DeclareSortingScheme{none}{
  \sort{\citeorder}
}
```

This sorts the bibliography purely lexically by the order of the keys in the citation commands. In the example above, it sorts “one” before “two”. However, suppose that

you consider “one” and “two” to have the same order (semantic order) since they are cited at the same time and want to further sort these by year. Suppose “two” has an earlier `year` than “one”:

```
\DeclareSortingScheme{noneyear}{
  \sort{\citeorder}
  \sort{year}
}
```

This sorts “two” before “one”, even though lexically, “one” would sort before “two”. This is possible because the semantic order can be disambiguated by the further sorting on year. With the standard `none` sorting scheme, the lexical order and semantic order are identical because there is nothing further to disambiguate them. This means that you can use `\citeorder` just like any other sorting specification element, choosing how to further sort entries cited at the same time (in the same citation command).

`\DeclareSortingNamekeyScheme[`⟨*schemename*⟩`]{`⟨*specification*⟩`}`

Defines how the sorting keys for names are constructed. This can change the sorting order of names arbitrarily because you can choose how to put together the name parts when constructing the string to compare when sorting. The sorting key construction scheme so defined is called ⟨*schemename*⟩ which defaults to “global” if this optional parameter is absent. When constructing the sorting key for a name, a sorting key for each name part is constructed and the key for each name is formed into an ordered key list with a special internal separator. The point of this option is to accommodate languages or situations where sorting of names needs to be customised (for example, Icelandic names are sometimes sorted by given names rather than by family names). This macro may be used multiple times to define schemes with different names which can then be referred to later. Sorting name key schemes can have the following scopes, in order of increasing precedence:

- The default scheme defined without the optional name argument
- Given as the `sortingnamekeyscheme` option to a reference context (see § 3.7.10)
- Given as a per-entry option `sortnamekeyscheme` in a bibliography data source entry
- Given as a per-namelist option `sortnamekeyscheme`
- Given as a per-name option `sortnamekeyscheme`

By default there is only a global scheme which has the following ⟨*specification*⟩:

```
\DeclareSortingNamekeyScheme{
  \keypart{
    \namepart[use=true]{prefix}
```

```
  }
  \keypart{
    \namepart{family}
  }
  \keypart{
    \namepart{given}
  }
  \keypart{
    \namepart{suffix}
  }
  \keypart{
    \namepart[use=false]{prefix}
  }
}
```

This means that the key is constructed by concatenating, in order, the name prefix (only if the `useprefix` option is true), the family name(s), the given names(s), the name suffix and then the name prefix (only if the `useprefix` option is false).

`\keypart{`⟨*part*⟩`}`

⟨*part*⟩ is an ordered list of of `\namepart` and `\literal` specifications which are concatenated together when constructing a part of the name sorting key.

`\literal{`⟨*string*⟩`}`

A literal string to insert into the name sorting key.

`\namepart{`⟨*name*⟩`}`

Specifies the ⟨*name*⟩ of a namepart to use in constructing the name sorting key.

`use`=`true`, `false` default: `true`

Indicates that the namepart ⟨*name*⟩ is only to be used in this concatenation position if the corresponding `use'name'` option is set to the specified boolean value.

`inits`=`true`, `false` default: `true`

Indicates that only the initials of namepart ⟨*name*⟩ are to be used in constructing the sorting specification.

As an example, suppose you wanted to be able to sort names by given name rather than family name, you could define a sorting name key scheme like this:

```
\DeclareSortingNamekeyScheme[givenfirst]{
  \keypart{
    \namepart{given}
```

```
  }
  \keypart{
    \namepart[use=true]{prefix}
  }
  \keypart{
    \namepart{family}
  }
  \keypart{
    \namepart[use=false]{prefix}
  }
}
```

You can then use the name `givenfirst` at the appropriate scope in order to make Biber use this scheme when constructing sorting name keys. For example, you could enable this for one bibliography list like this:

```
\begin{refcontext}[sortnamekeyscheme=givenfirst]
\printbibliography
\end{refcontext}
```

or perhaps you only want to do this for a particular entry:

```
@BOOK{key,
  OPTIONS = {sortnamekeyscheme=givenfirst},
  AUTHOR = {Arnar Vigfusson}
}
```

or just a name list by using the option as a pseudo-name which will be ignored:

```
@BOOK{key,
  AUTHOR = {sortnamekeyscheme=givenfirst and Arnar Vigfusson}
}
```

or just a single name by passing the option as part of the extended name information format which Biber supports (see Biber doc):

```
@BOOK{key,
  AUTHOR = {given=Arnar, family=Vigfusson, sortnamekeyscheme=givenfirst}
}
```

Now we give some examples of sorting schemes. In the first example, we define a simple name/title/year scheme. The name element may be either the `author`, the `editor`, or the `translator`. Given this specification, the sorting routine will use the first element which is available and continue with the `title`. Note that the options `use<name>` options are considered automatically in the sorting process:

```
\DeclareSortingScheme{sample}{
  \sort{
    \field{author}
    \field{editor}
    \field{translator}
  }
  \sort{
    \field{title}
  }
  \sort{
    \field{year}
  }
}
```

In the next example, we define the same scheme in a more elaborate way, considering special fields such as `presort`, `sortkey`, `sortname`, etc. Since the `sortkey` field specifies the master sort key, it needs to override all other elements except for `presort`. This is indicated by the `final` option. If the `sortkey` field is available, processing will stop at this point. If not, the sorting routine continues with the next `\sort` directive. This setup corresponds to the default definition of the `nty` scheme:

```
\DeclareSortingScheme{nty}{
  \sort{
    \field{presort}
  }
  \sort[final]{
    \field{sortkey}
  }
  \sort{
    \field{sortname}
    \field{author}
    \field{editor}
    \field{translator}
    \field{sorttitle}
    \field{title}
  }
  \sort{
    \field{sorttitle}
    \field{title}
  }
  \sort{
    \field{sortyear}
```

```
    \field{year}
  }
}
```

Finally, here is an example of a sorting scheme which overrides the global sorting locale and additionally overrides again when sorting by the `origtitle` field. Note the use in the scheme-level override of a babel/polyglossia language name instead of a real locale identifier. Biber will map this to a suitable, real locale identifier (in this case, `sv_SE`):

```
\DeclareSortingScheme[locale=swedish]{custom}{
  \sort{
    \field{sortname}
    \field{author}
    \field{editor}
    \field{translator}
    \field{sorttitle}
    \field{title}
  }
  \sort[locale=de_DE_phonebook]{
    \field{origtitle}
  }
}
```

`\DeclareSortExclusion{`⟨*entrytype, ...*⟩`}{`⟨*field, ...*⟩`}`

Specifies fields to be excluded from sorting on a per-type basis. The ⟨*entrytype*⟩ argument and the ⟨*field*⟩ argument may be a comma-separated list of values. A blank ⟨*field*⟩ argument will clear all exclusions for this ⟨*entrytype*⟩. A value of '*' for ⟨*entrytype*⟩ will exclude ⟨*field,...*⟩ for every entrytype. This is equivalent to simply deleting the field from the sorting specification and is only normally used in combination with `\DeclareSortInclusion` when one wishes to exclude a field for all but explicitly included entrytypes. See example in `\DeclareSortInclusion` below. This command may only be used in the preamble.

`\DeclareSortInclusion{`⟨*entrytype, ...*⟩`}{`⟨*field, ...*⟩`}`

Only used along with `\DeclareSortExclusion`. Specifies fields to be included in sorting on a per-type basis. This allows the user to exclude a field from sorting for all entrytypes and then to override this for certain entrytypes. This is easier sometimes than using `\DeclareSortExclusion` to list exclusions for many entrytypes. The ⟨*entrytype*⟩ argument and the ⟨*field*⟩ argument may be a comma-separated list of values. This command may only be used in the preamble. For example, this would use `title` during sorting only for `@articles`:

```
\DeclareSortExclusion{*}{title}
\DeclareSortInclusion{article}{title}
```

\DeclarePresort[⟨*entrytype, ...*⟩]{⟨*string*⟩}

Specifies a string to be used to automatically populate the `presort` field of entries without a `presort` field. The `presort` may be defined globally or on a per-type basis. If the optional ⟨*entrytype*⟩ argument is given, the ⟨*string*⟩ applies to the respective entry type. If not, it serves as the global default value. Specifying an ⟨*entrytype*⟩ in conjunction with a blank ⟨*string*⟩ will clear the type-specific setting. The ⟨*entrytype*⟩ argument may be a comma-separated list of values. This command may only be used in the preamble.

\DeclareSortTranslit[⟨*entrytype*⟩]{⟨*specification*⟩}

Languages which can be written in different scripts or alphabets often only have CLDR sorting tailoring for one script and it is expected that you transliterate into the supported script for sorting purposes. A common example is Sanskrit which is often written in academic contexts in IAST romanised script but which needs to be sorted in the 'sa' locale which expects the Devanāgarī script. This means that it is necessary to transliterate into the sorting script internally. `\DeclareSortTranslit` declares which parts of an entry you would like to transliterate for sorting purposes. Without the ⟨*entrytype*⟩ parameter, the ⟨*specification*⟩ applies to all entrytypes. The ⟨*specification*⟩ is one or more `\translit` commands:

\translit{⟨*field or fieldset*⟩}{⟨*from*⟩}{⟨*to*⟩}

Specifies that the data field `field` or all fields in a fieldset ⟨*fieldset*⟩ declared with `\DeclareDatafieldSet` (see § 4.5.2) should be transliterated from script ⟨*from*⟩ to script ⟨*to*⟩ for sorting purposes. The field/set argument can also be '*' to apply transliteration to all fields. The valid ⟨*from*⟩ and ⟨*to*⟩ values are given in table 12. Note that Biblatex does not aim to support general transliteration, only those which are useful for sorting purposes. Please open a GitHub ticket for Biblatex if you think you need additional transliterations.

An example of transliterating titles so that they sort correctly in Sanskrit:

```
\DeclareDatafieldSet{settitles}{
  \member[field=title]
  \member[field=booktitle]
  \member[field=eventtitle]
  \member[field=issuetitle]
  \member[field=journaltitle]
  \member[field=maintitle]
```

| From | To | Description |
|---|---|---|
| iast | devanagari | Sanskrit IAST[51] to Devanāgarī |

**Table 12: Valid transliteration pairs**

```
  \member[field=origtitle]
}

\DeclareSortTranslit{
  \translit[settitles]{iast}{devanagari}
}
```

### 4.5.7 Bibliography List Filters

When using customisable bibliography lists (See § 3.7.3), usually one wants to return in the `.bbl` only those entries which have the particular fields which the bibliography list is summarising. For example, when printing a normal list of shorthands, you want the list returned by Biber in the `.bbl` to contain only those entries which have a `shorthand` field. This is accomplished by defining a bibliography list filter using the `\DeclareBiblistFilter` command. This differs from the filters defined using `\defbibfilter` (see § 3.7.9) since the filters defined by `\defbibfilter` run inside Biblatex after the `.bbl` has been generated. In addition, bibliography lists in the `.bbl` do not contain entry data, only the citation keys for the entries and so no filtering by Biblatex using `\defbibfilter` is possible for bibliography lists.

`\DeclareBiblistFilter{`⟨*name*⟩`}{`⟨*specification*⟩`}`

Defines a bibliography list filter with ⟨*name*⟩. The ⟨*specification*⟩ consists of one or more `\filter` or `\filteror` macros, all of which must be satisfied for the entry to pass the filter:

`\filter[`⟨*filterspec*⟩`]{`⟨*filter*⟩`}`

Filter entries according to the ⟨*filterspec*⟩ and ⟨*filter*⟩. ⟨*filterspec*⟩ can be one of:

**type/nottype** Entry is/is not of `entrytype` ⟨*filter*⟩

**subtype/notsubtype** Entry is/is not of `subtype` ⟨*filter*⟩

**keyword/notkeyword** Entry has/does not have `keyword` ⟨*filter*⟩

**field/notfield** Entry has/does not have a field called ⟨*filter*⟩

`\filteror{`⟨*type*⟩`}{`⟨*filters*⟩`}`

A wrapper around one or more `\filter` commands specifying that they form a disjunctive set, i.e. any one of the ⟨*filters*⟩ must be satisfied.

Fields in the datamodel which are marked as 'Label fields' (see § 4.5.4) automatically have a filter defined for them with the same name and which filters out any entries which do no contain the field. For example, Biblatex automatically generates a filter for the `shorthand` field:

```
\DeclareBiblistFilter{shorthand}{
  \filter[type=field,filter=shorthand]
}
```

### 4.5.8 Controlling Name Initials Generation

Generating initials for name parts from a given name involves some difficulties when you have names with prefixes, diacritics, hyphens etc. Often, you want to ignore things like prefixes when generating initials so that the initials for "al-Hasan" is just "H" instead of "a-H". This is tricky when you also have names like "Ho-Pun" where you want the initials to be "H-P", for example.

`\DeclareNoinit{`⟨*specification*⟩`}`

Defines regular expressions to strip from names before generating initials. The ⟨*specification*⟩ is an undelimited list of `\noinit` directives which specify the regular expressions to remove from the name. Spaces, tabs and line endings may be used freely to visually arrange the ⟨*specification*⟩. Blank lines are not permissible. This command may only be used in the preamble.

`\noinit{`⟨*regexp*⟩`}`

Any number of `\noinit` commands can be given each of which specifies to remove the ⟨*regexp*⟩ from the copy of the name which the initials generation system sees. Since regular expressions usually contain special characters, it is best to enclose them in the provided `\regexp` macro as shown—this will pass the expression through to Biber correctly.

If there is no `\DeclareNoinit` specification, Biber will default to:

```
\DeclareNoinit{
  % strip lowercase prefixes like 'al-' when generating initials from
    ↪ names
  \noinit{\regexp{\b\p{Ll}{2}\p{Pd}}}
  % strip some common diacritics when generating initials from names
  \noinit{\regexp{[\x{2bf}\x{2018}]}}
}
```

This Biber default strips a couple of diacritics and also strips lowercase prefixes from names before generating initials.

### 4.5.9 排序微调 Fine Tuning Sorting

对排序微调是很有用的，它可以忽略一些特殊域的某些部分。It can be useful to fine tune sorting so that it ignores certain parts of particular fields.

`\DeclareNosort{`⟨*specification*⟩`}`

Defines regular expressions to strip from particular fields or types of fields when sorting. The ⟨*specification*⟩ is an undelimited list of `\nosort` directives which specify the regular expressions to remove from particular fields or type of field. Spaces, tabs and line endings may be used freely to visually arrange the ⟨*specification*⟩. Blank lines are not permissible. This command may only be used in the preamble.

`\nosort{`⟨*field or field type*⟩`}{`⟨*regexp*⟩`}`

Any number of `\nosort` commands can be given each of which specifies to remove the ⟨*regexp*⟩ from the ⟨*field*⟩ or ⟨*field type*⟩. A ⟨*field type*⟩ is simple a convenience grouping of semantically similar fields from which you might want to remove a regexp. Table 13 shows the available field types. Since regular expressions usually contain special characters, it is best to enclose them in the provided `\regexp` macro as shown—this will pass the expression through to Biber correctly.

The default is:

```
\DeclareNosort{
  % strip prefixes like 'al-' when sorting names
  \nosort{type_names}{\regexp{\A\p{L}{2}\p{Pd}}}
  % strip some diacritics when sorting names
  \nosort{type_names}{\regexp{[\x{2bf}\x{2018}]}}
}
```

This Biber default strips a couple of diacritics and also strips prefixes from names when sorting. Suppose you wanted to ignore "The" at the beginning of a `title` field when sorting:

```
\DeclareNosort{
  \nosort{title}{\regexp{\AThe\s+}}
}
```

Or if you wanted to ignore "The" at the beginning of any title field:

```
\DeclareNosort{
  \nosort{type_title}{\regexp{\AThe\s+}}
}
```

| Field Type | Fields |
|---|---|
| type_name | author |
| | afterword |
| | annotator |
| | bookauthor |
| | commentator |
| | editor |
| | editora |
| | editorb |
| | editorc |
| | foreword |
| | holder |
| | introduction |
| | namea |
| | nameb |
| | namec |
| | shortauthor |
| | shorteditor |
| | translator |
| type_title | booktitle |
| | eventtitle |
| | issuetitle |
| | journaltitle |
| | maintitle |
| | origtitle |
| | title |

**Table 13: Field types for `\nosort`**

### 4.5.10 特殊域 Special Fields

Some of the automatically generated fields from § 4.2.4.2 may be customized.

`\DeclareLabelname[`⟨*entrytype, ...*⟩`]{`⟨*specification*⟩`}`

Defines the fields to consider when generating the `labelname` field (see § 4.2.4.2). The ⟨*specification*⟩ is an ordered list of `\field` commands. The fields are checked in the order listed and the first field which is available will be used as `labelname`. This is the default definition:

```
\DeclareLabelname{%
  \field{shortauthor}
  \field{author}
  \field{shorteditor}
  \field{editor}
  \field{translator}
}
```

The `labelname` field may be customized globally or on a per-type basis. If the optional ⟨*entrytype*⟩ argument is given, the specification applies to the respective entry type. If

not, it is applied globally. The ⟨*entrytype*⟩ argument may be a comma-separated list of values. This command may only be used in the preamble.

`\DeclareLabeldate[`⟨*entrytype, ...*⟩`]{`⟨*specification*⟩`}`

Defines the date components to consider when generating `labelyear`, `labelmonth`, `labelday`, `labelendyear`, `labelendmonth` and `labelendday` fields (see § 4.2.4.2). The ⟨*specification*⟩ is an ordered list of `\field` or `\literal` commands. The items are checked in the order listed and the first item which is available will be used to poplu-ate the mentioned fields. Note that the `\field` items do not have to be datetype 'date' in the data model so that you can create pseudo-year labels by, for example, using a `pubstate` field contents, if available, as the year label by defining `\DeclareLabeldate` suitably. Note also that a `\literal` command will always be used when found and so this should always be the last thing in the list. If the value of a `\literal` command is a valid localisation string, then this will be resolved in the current language, otherwise the value is used as a literal string as-is. This is the default definition:

```
\DeclareLabeldate{%
  \field{date}
  \field{year}
  \field{eventdate}
  \field{origdate}
  \field{urldate}
  \literal{nodate}
}
```

Note that the `date` field is split by the backend into `year`, `month` which are also valid fields in the default data model. In order to support legacy data which directly sets `year` and/or `month`, the specification '`date`' in `\DeclareLabeldate` will also match `year` and `month` fields, if present. The `label*` fields may be customized globally or on a per-type basis. If the optional ⟨*entrytype*⟩ argument is given, the specification applies to the respective entry type. If not, it is applied globally. The ⟨*entrytype*⟩ argument may be a comma-separated list of values. This command may only be used in the preamble. See also § 4.2.4.3.

`\DeclareLabeltitle[`⟨*entrytype, ...*⟩`]{`⟨*specification*⟩`}`

Defines the fields to consider when generating the `labeltitle` field (see § 4.2.4.2). The ⟨*specification*⟩ is an ordered list of `\field` commands. The fields are checked in the order listed and the first field which is available will be used as `labeltitle`. This is the default definition:

```
\DeclareLabeltitle{%
  \field{shorttitle}
```

```
  \field{title}
}
```

The `labeltitle` field may be customized globally or on a per-type basis. If the optional ⟨*entrytype*⟩ argument is given, the specification applies to the respective entry type. If not, it is applied globally. The ⟨*entrytype*⟩ argument may be a comma-separated list of values. This command may only be used in the preamble.

### 4.5.11 数据继承 Data Inheritance (crossref)

Biber features a highly customizable cross-referencing mechanism with flexible data inheritance rules. This sections deals with the configuration interface. See 附录 B for the default configuration. A note on terminology: the *child* or *target* is the entry with the `crossref` field, the *parent* or *source* is the entry the `crossref` field points to. The child inherits data from the parent.

`\DefaultInheritance[`⟨*exceptions*⟩`]{`⟨*options*⟩`}`

Configures the default inheritance behavior. This command may only be used in the preamble. The default behavior may be customized be setting the following ⟨*options*⟩:

`all=true, false` default: `true`

Whether or not to inherit all fields from the parent by default.

`all=true` means that the child entry inherits all fields from the parent, unless a more specific inheritance rule has been set up with `\DeclareDataInheritance`. If an inheritance rule is defined for a field, data inheritance is controlled by that rule. `all=false` means that no data is inherited from the parent by default and each field to be inherited requires an explicit inheritance rule set up with `\DeclareDataInheritance`. The package default is `all=true`.

`override=true, false` default: `false`

Whether or not to overwrite target fields with source fields if both are defined. This applies both to automatic inheritance and to explicit inheritance rules. The package default is `override=false`, i. e., existing fields of the child entry are not overwritten.

`ignore=`⟨*csv list of uniqueness options*⟩

This option takes a comma-separated list of one of more of 'singletitle', 'uniquetitle', 'uniquebaretitle' and/or 'uniquework'. The purpose of this option is to ignore tracking information for these three options when the field which would trigger the tracking (表 7) is inherited. An example—Suppose that you have several `@book` entries which all crossref a `@mvbook` from which they get their `author` field. You might reasonably want the `\ifsingletitle` test to return 'true' for this author as their only 'work' is the `@mvbook`. Similar comments would apply to situations involving the `\ifuniquetitle`, `\ifuniquebaretitle` and `\ifuniquework` tests. The `ignore` option lists which of these should have their tracking information ignored when the fields which would trigger

them are inherited. The idea is that the presence of an inherited field does not contribute towards the determination of whether some combination of name/title is unique in the bibliographic data. For example, this modified default setting would ignore `singletitle` and `uniquetitle` tracking:

```
\DefaultInheritance{ignore={singletitle,uniquetitle}, all=true, override
    ↪ =false}
```

Of course, the ignoring of tracking does nothing if the fields inherited do not play a role in tracking. Only the fields listed in 表 7 are relevant to this option.

The optional ⟨*exceptions*⟩ are an undelimited list of `\except` directives. Spaces, tabs, and line endings may be used freely to visually arrange the ⟨*exceptions*⟩. Blank lines are not permissible.

`\except{`⟨*source*⟩`}{`⟨*target*⟩`}{`⟨*options*⟩`}`

Defines an exception to the default inheritance rules.

`\DeclareDataInheritance` sets the inheritance ⟨*options*⟩ for a specific ⟨*source*⟩ and ⟨*target*⟩ combination. The ⟨*source*⟩ and ⟨*target*⟩ arguments specify the parent and the child entry type. The asterisk matches all types and is permissible in either argument.

`\DeclareDataInheritance[`⟨*options*⟩`]{`⟨*source, ...*⟩`}{`⟨*target, ...*⟩`}{`⟨*rules*⟩`}`

Declares inheritance rules. The ⟨*source*⟩ and ⟨*target*⟩ arguments specify the parent and the child entry type. Either argument may be a single entry type, a comma-separated list of types, or an asterisk. The asterisk matches all entry types. The ⟨*rules*⟩ are an undelimited list of `\inherit` and/or `\noinherit` directives. Spaces, tabs, and line endings may be used freely to visually arrange the ⟨*rules*⟩. Blank lines are not permissible. This command may only be used in the preamble. The options are:

`ignore=`⟨*csv list of uniqueness options*⟩

As the `ignore` option on `\DefaultInheritance` explained above. When set here, it takes precedence over any global options set with `\DefaultInheritance`. For example, this would ignore `singletitle` and `uniquetitle` tracking for a `@book` inheriting from a `@mvbook`.

```
\DeclareDataInheritance[ignore={singletitle,uniquetitle}]{mvbook}{book}{
    ↪ ...}
```

`\inherit[`⟨*option*⟩`]{`⟨*source*⟩`}{`⟨*target*⟩`}`

Defines an inheritance rule by mapping a ⟨*source*⟩ field to a ⟨*target*⟩ field. ⟨*option*⟩ can be one of

`override`=`true`, `false` default: `false`

As the `override` option for `\DefaultInheritance` explained above. When set here, it takes precedence over any global options set with `\DefaultInheritance`.

`\noinherit{`⟨*source*⟩`}`

Unconditionally prevents inheritance of the ⟨*source*⟩ field.

`\ResetDataInheritance` Clears all inheritance rules defined with `\DeclareDataInheritance`. This command may only be used in the preamble.

Here are some practical examples:

```
\DefaultInheritance{all=true,override=false}
```

This example shows how to configure the default inheritance behavior. The above settings are the package defaults.

```
\DefaultInheritance[
  \except{*}{online}{all=false}
]{all=true,override=false}
```

This example is similar to the one above but adds one exception: entries of type `@online` will, by default, not inherit any data from any parent.

```
\DeclareDataInheritance{collection}{incollection}{
  \inherit{title}{booktitle}
  \inherit{subtitle}{booksubtitle}
  \inherit{titleaddon}{booktitleaddon}
}
```

So far we have looked at setting up standard inheritance. For example, `all=true` means that the `publisher` field of a source entry is copied to the `publisher` field of the target entry. In some cases, however, asymmetric mappings are required. They are defined with `\DeclareDataInheritance`. The above example sets up three typical rules for `@incollection` entries referencing a `@collection`. We map the `title` and related fields of the source to the corresponding `booktitle` fields of the target.

```
\DeclareDataInheritance{mvbook,book}{inbook,bookinbook}{
  \inherit{author}{author}
  \inherit{author}{bookauthor}
}
```

This rule is an example of one-to-many mapping: it maps the `author` field of the source to both the `author` and the `bookauthor` fields of the target in order to allow for compact `inbook`/`bookinbook` entries. The source may be either a `@mvbook` or a `@book` entry, the target either an `@inbook` or a `@bookinbook` entry.

```
\DeclareDataInheritance{*}{inbook,incollection}{
  \noinherit{introduction}
}
```

This rule prevents inheritance of the `introduction` field. It applies to all targets of type `@inbook` or `@incollection`, regardless of the source entry type.

```
\DeclareDataInheritance{*}{*}{
  \noinherit{abstract}
}
```

This rule, which applies to all entries, regardless of the source and target entry types, prevents inheritance of the `abstract` field.

```
\DefaultInheritance{all=true,override=false}
\ResetDataInheritance
```

This example demonstrates how to emulate traditional BibTeX's cross-referencing mechanism. It enables inheritance by default, disables overwriting, and clears all other inheritance rules and mappings.

In a bibliography entry, you can give an option 'noinherit' where the value is a datafield set defined with `\DeclareDatafieldSet` (§ 4.5.2). This will block inheritance of the fields in the set on a per-entry basis. For example:

```
\DeclareDatafieldSet{nobtitle}{
  \member[field=booktitle]
}
```

```
@INBOOK{s1,
  OPTIONS  = {noinherit=nobtitle},
  TITLE    = {Subtitle},
  CROSSREF = {s2}
}

@BOOK{s2,
  TITLE = {Title}
}
```

Here, s1 will not inherit the TITLE of s2 as BOOKTITLE as this is blocked by the datafield set given as the value to the noinherit option. One important thing to note is that children will never inherit any dateparts of a given type if they already contain a datepart of that type. So, for example:

```
@INBOOK{b1,
  DATE     = {2004-03-03},
  ORIGDATE = {2004-03},
  CROSSREF = {b2}
}

@BOOK{b2,
  DATE      = {2004-03-03/2005-08-09},
  ORIGDATE  = {2004-03/2005-08},
  EVENTDATE = {2004-03/2005-08},
}
```

Here, b1 will not inherit any of endyear, endmonth, endday, origendyear or origendmonth as this would make a mess of its own dates. It will, given the inheritance defaults, inherit all of the event* date parts.

## 4.6 辅助命令

本节的工具用来分析和保存参考文献数据而不是对其进行格式化或者打印。

### 4.6.1 数据命令

本节的命令允许以 low-level 方式访问未格式化的参考文献数据。这些命令不是用来输出，而是用来将数据保存到临时宏中，可以用于下一步的比较。

`\thefield{`⟨*field*⟩`}`

展开为未格式化的⟨*field*⟩。如果⟨*field*⟩未定义那么展开为一个空字符串。

`\strfield{`⟨*field*⟩`}`

类似于`\thefield`命令，但其值经自动净化，以便安全的用于构成控制序列名。

`\csfield{`⟨*field*⟩`}`

类似于`\thefield`命令，但禁止展开

`\usefield{`⟨*command*⟩`}{`⟨*field*⟩`}`

执行⟨*command*⟩命令使用未格式化的⟨*field*⟩作为其参数

`\thelist{`⟨*literal list*⟩`}`

展开为未格式化的⟨*literal list*⟩。如果 list 未定义那么展开为一个空字符串。注意该命令中将⟨*literal list*⟩转存为本宏包使用的内部格式。这一格式不适合打印。

`\strlist{`⟨*literal list*⟩`}`

类似于`\thelist`，差别在于该命令能自动处理列表的内部表示，因此列表的值可以安全地用于控制序列名的构建。

`\thename{`⟨*name list*⟩`}`

展开为未格式化的⟨*name list*⟩。如果 list 未定义那么展开为一个空字符串。注意该命令中将⟨*name list*⟩转存为本宏包使用的内部格式。这一格式不适合打印。

`\strname{`⟨*name list*⟩`}`

类似于`\thename`，差别在于该命令能自动处理列表的内部表示，因此列表的值可以安全地用于控制序列名的构建。

`\savefield{`⟨*field*⟩`}{`⟨*macro*⟩`}`
`\savefield*{`⟨*field*⟩`}{`⟨*macro*⟩`}`

将未格式化的⟨*field*⟩拷贝到一个⟨*macro*⟩中。不带星的命令全局的定义⟨*macro*⟩，而带星的命令是局部定义。

`\savelist{`⟨*literal list*⟩`}{`⟨*macro*⟩`}`
`\savelist*{`⟨*literal list*⟩`}{`⟨*macro*⟩`}`

将未格式化的⟨*literal list*⟩拷贝到一个⟨*macro*⟩中。不带星的命令全局的定义⟨*macro*⟩，而带星的命令是局部定义。

`\savename{`⟨*name list*⟩`}{`⟨*macro*⟩`}`
`\savename*{`⟨*name list*⟩`}{`⟨*macro*⟩`}`

将未格式化的⟨*name list*⟩拷贝到一个⟨*macro*⟩中。不带星的命令全局的定义⟨*macro*⟩，而带星的命令是局部定义。

`\savefieldcs{`⟨*field*⟩`}{`⟨*csname*⟩`}`
`\savefieldcs*{`⟨*field*⟩`}{`⟨*csname*⟩`}`

类似于`\savefield`命令，当将控制序列名⟨*csname*⟩(即没有斜杠) 作为参数，而不是宏。

`\savelistcs{`⟨*literal list*⟩`}{`⟨*csname*⟩`}`
`\savelistcs*{`⟨*literal list*⟩`}{`⟨*csname*⟩`}`

类似于`\savelist`命令，当将控制序列名⟨*csname*⟩(即没有斜杠) 作为参数，而不是宏。

`\savenamecs{`⟨*name list*⟩`}{`⟨*csname*⟩`}`
`\savenamecs*{`⟨*name list*⟩`}{`⟨*csname*⟩`}`

类似于`\savename`命令，当将控制序列名⟨*csname*⟩(即没有斜杠) 作为参数，而不是宏。

`\restorefield{`⟨*field*⟩`}{`⟨*macro*⟩`}`

从之前用`\savefield`命令定义的⟨*macro*⟩中将⟨*field*⟩恢复回来。该域是在局部范围内恢复。

`\restorelist{`⟨*literal list*⟩`}{`⟨*macro*⟩`}`

从之前用`\savelist`命令定义的⟨*macro*⟩中将⟨*literal list*⟩恢复回来。该 list 是在局部范围内恢复。

`\restorename{`⟨*name list*⟩`}{`⟨*macro*⟩`}`

从之前用`\savename`命令定义的⟨*macro*⟩中将⟨*name list*⟩恢复回来。该 list 是在局部范围内恢复。

`\clearfield{`⟨*field*⟩`}`

在局部范围内清除⟨*field*⟩。以这种方式清除的域对于后续的数据命令来说相当于没有定义。

`\clearlist{`⟨*literal list*⟩`}`

在局部范围内清除⟨*literal list*⟩。以这种方式清除的 list 对于后续的数据命令来说相当于没有定义。

`\clearname{`⟨*name list*⟩`}`

在局部范围内清除⟨*name list*⟩。以这种方式清除的 list 对于后续的数据命令来说相当于没有定义。

### 4.6.2 独立判断命令

本节的命令是不同类型的 stand-alone 判断命令，用于参考文献著录和标注样式中。

`\if<datetype>julian{`⟨*true*⟩`}{`⟨*false*⟩`}`

当日期'datetype'date 因为`julian`和`gregorianstart`选项的设置转换为儒略历 (Julian Calendar) 时，展开为⟨*true*⟩。

`\ifdatejulian{`⟨*true*⟩`}{`⟨*false*⟩`}`

类似于`\if<datetype>julian`但用于`\mkbibdate*`格式化命令中 (§ 4.10.2)，在这些格式化命令中恰当使用的`\if<datetype>julian`命令等价于该命令。

\if<datetype>dateera{⟨*era*⟩}{⟨*true*⟩}{⟨*false*⟩}

当日期'datetype'date(date, urldate, eventdate等) 指定了一个时区等于⟨*era*⟩，则展开为⟨*true*⟩，否则展开为⟨*false*⟩。Biber 确认并在.bbl文件中传递的可用⟨*era*⟩字符串是:

**bce**BCE/BC era

**ce**CE/AD era

该命令用于确定是否打印 § 4.9.2.21节的地址字符串。

\ifdateera{⟨*era*⟩}{⟨*true*⟩}{⟨*false*⟩}

类似于\if<datetype>dateera，但用于\mkbibdate*格式化命令 (§ 4.10.2)，在这些格式化命令中恰当使用的\if<datetype>dateera命令等价于该命令。

\if<datetype>datecirca{⟨*true*⟩}{⟨*false*⟩}

当日期'datetype'date(date, urldate, eventdate等) 在数据源中具有一个'circa' 标记时，则展开为⟨*true*⟩，否则展开为⟨*false*⟩。参见 § 2.3.8。该命令用于确定是否打印 § 4.9.2.21节中的字符串。

\ifdatecirca{⟨*true*⟩}{⟨*false*⟩}

类似于\if<datetype>datecirca，但用于\mkbibdate*格式化命令 (§ 4.10.2)，在这些格式化命令中恰当使用的\if<datetype>datecirca 命令等价于该命令。

\if<datetype>dateuncertain{⟨*true*⟩}{⟨*false*⟩}

当日期'datetype'date(date, urldate, eventdate等) 在数据源中具有一个不确定标记时，则展开为⟨*true*⟩，否则展开为⟨*false*⟩。参见 § 2.3.8。该命令用于确定是否打印例如年份后的一个问号。

\ifdateuncertain{⟨*true*⟩}{⟨*false*⟩}

类似于\if<datetype>dateuncertain，但用于\mkbibdate*格式化命令 (§ 4.10.2)，在这些格式化命令中恰当使用的\if<datetype>dateuncertain命令等价于该命令。

\ifenddateuncertain{⟨*true*⟩}{⟨*false*⟩}

类似于\ifend<datetype>dateuncertain，但用于\mkbibdate*格式化命令 (§ 4.10.2)，在这些格式化命令中恰当使用的\ifend<datetype>dateuncertain命令等价于该命令。

\ifcaselang[⟨*language*⟩]{⟨*true*⟩}{⟨*false*⟩}

如果可选的⟨*language*⟩是\DeclareCaseLangs(见 § 4.6.4) 声明的语言之一，展开为⟨*true*⟩，否则展开为⟨*false*⟩。但可选参数不给出时，对\currentlang值进行判断。

\ifsortnamekeyscheme{⟨*string*⟩}{⟨*true*⟩}{⟨*false*⟩}

如果⟨*string*⟩等于范围排序名关键词格式名[52](4.5.6)，否则展开为⟨*false*⟩。

\iffieldundef{⟨*field*⟩}{⟨*true*⟩}{⟨*false*⟩}

展开为⟨*true*⟩，如果⟨*field*⟩未定义，否则展开为⟨*false*⟩

\iflistundef{⟨*literal list*⟩}{⟨*true*⟩}{⟨*false*⟩}

展开为⟨*true*⟩，如果⟨*literal list*⟩未定义，否则展开为⟨*false*⟩

\ifnameundef{⟨*name list*⟩}{⟨*true*⟩}{⟨*false*⟩}

展开为⟨*true*⟩，如果⟨*name list*⟩未定义，否则展开为⟨*false*⟩

\iffieldsequal{⟨*field 1*⟩}{⟨*field 2*⟩}{⟨*true*⟩}{⟨*false*⟩}

展开为⟨*true*⟩，如果⟨*field 1*⟩和⟨*field 2*⟩相等，否则展开为⟨*false*⟩

\iflistsequal{⟨*literal list 1*⟩}{⟨*literal list 2*⟩}{⟨*true*⟩}{⟨*false*⟩}

展开为⟨*true*⟩，如果⟨*literal list 1*⟩和⟨*literal list 2*⟩相等，否则展开为⟨*false*⟩

\ifnamesequal{⟨*name list 1*⟩}{⟨*name list 2*⟩}{⟨*true*⟩}{⟨*false*⟩}

展开为⟨*true*⟩，如果⟨*name list 1*⟩和⟨*name list 2*⟩相等，否则展开为⟨*false*⟩

\iffieldequals{⟨*field*⟩}{⟨*macro*⟩}{⟨*true*⟩}{⟨*false*⟩}

展开为⟨*true*⟩，如果⟨*field*⟩的值和⟨*macro*⟩的定义相等，否则展开为⟨*false*⟩。[53]

\iflistequals{⟨*literal list*⟩}{⟨*macro*⟩}{⟨*true*⟩}{⟨*false*⟩}

展开为⟨*true*⟩，如果⟨*literal list*⟩的值和⟨*macro*⟩的定义相等，否则展开为⟨*false*⟩。

\ifnameequals{⟨*name list*⟩}{⟨*macro*⟩}{⟨*true*⟩}{⟨*false*⟩}

展开为⟨*true*⟩，如果⟨*name list*⟩的值和⟨*macro*⟩的定义相等，否则展开为⟨*false*⟩。

\iffieldequalcs{⟨*field*⟩}{⟨*csname*⟩}{⟨*true*⟩}{⟨*false*⟩}

类似于`\iffieldequals`，但将控制序列名⟨*csname*⟩(不带斜杠) 作为参数，而不是一个宏名。

\iflistequalcs{⟨*literal list*⟩}{⟨*csname*⟩}{⟨*true*⟩}{⟨*false*⟩}

类似于`\iflistequals`，但将控制序列名⟨*csname*⟩(不带斜杠) 作为参数，而不是一个宏名。

[52] the current in scope sorting name key scheme name 待议

[53] 可以用于改进 gb7714-2015 中的新闻和标准的判断

\ifnameequalcs{⟨*name list*⟩}{⟨*csname*⟩}{⟨*true*⟩}{⟨*false*⟩}

类似于\ifnameequals，但将控制序列名⟨*csname*⟩(不带斜杠) 作为参数，而不是一个宏名。

\iffieldequalstr{⟨*field*⟩}{⟨*string*⟩}{⟨*true*⟩}{⟨*false*⟩}

展开为⟨*true*⟩，如果⟨*field*⟩的值和字符串⟨*string*⟩的定义相等，否则展开为⟨*false*⟩。该命令是鲁棒的。

\iffieldxref{⟨*field*⟩}{⟨*true*⟩}{⟨*false*⟩}

如果一个条目定义了crossref/xref，该命令检测⟨*field*⟩是否与 cross-referenced 父条目相关联。如果子条目的⟨*field*⟩与父条目对应的⟨*field*⟩相等，那么执行⟨*true*⟩，否则执行⟨*false*⟩。如果crossref/xref未定义，总是执行⟨*false*⟩。该命令是鲁棒的。crossref和xref域的描述见 § 2.2.3，更多关于 cross-referencing 的信息见 § 2.4.1。

\iflistxref{⟨*literal list*⟩}{⟨*true*⟩}{⟨*false*⟩}

类似于\iffieldxref命令，但检测⟨*literal list*⟩是否与 cross-referenced 父条目相关联。crossref和 xref域的描述见 § 2.2.3，更多关于 cross-referencing 的信息见 § 2.4.1。

\ifnamexref{⟨*name list*⟩}{⟨*true*⟩}{⟨*false*⟩}

类似于\iffieldxref命令，但检测⟨*name list*⟩是否与 cross-referenced 父条目相关联。crossref和 xref域的描述见 § 2.2.3，更多关于 cross-referencing 的信息见 § 2.4.1。

\ifcurrentfield{⟨*field*⟩}{⟨*true*⟩}{⟨*false*⟩}

执行⟨*true*⟩，如果当前域为⟨*field*⟩，否则执行⟨*false*⟩。该命令是鲁棒的。它主要用于域格式指令中，如果在其它环境中总是执行⟨*false*⟩。

\ifcurrentlist{⟨*literal list*⟩}{⟨*true*⟩}{⟨*false*⟩}

执行⟨*true*⟩，如果当前 list 为⟨*literal list*⟩，否则执行⟨*false*⟩。该命令是鲁棒的。它主要用于域格式指令中，如果在其它环境中总是执行⟨*false*⟩。

\ifcurrentname{⟨*name list*⟩}{⟨*true*⟩}{⟨*false*⟩}

执行⟨*true*⟩，如果当前 list 为⟨*name list*⟩，否则执行⟨*false*⟩。该命令是鲁棒的。它主要用于域格式指令中，如果在其它环境中总是执行⟨*false*⟩。

\ifuseprefix{⟨*true*⟩}{⟨*false*⟩}

执行⟨*true*⟩，如果useprefix选项打开 (无论是全局的还是针对当前条目)，否则执行⟨*false*⟩。该选项的细节见 § 3.1.3。

\ifuseauthor{⟨*true*⟩}{⟨*false*⟩}

这只是下面的\ifuse<name>宏的一个特例，因为author是默认数据模型的一部分所以放到这里来说。执行⟨*true*⟩，如果useauthor选项打开 (无论是全局的还是针对当前条目)，否则执行⟨*false*⟩。该选项的细节见 § 3.1.3。

`\ifuseeditor{`⟨*true*⟩`}{`⟨*false*⟩`}`

这只是下面的`\ifuse<name>`宏的一个特例，因为`editor`是默认数据模型的一部分所以放到这里来说。执行⟨*true*⟩，如果`useeditor`选项打开 (无论是全局的还是针对当前条目)，否则执行⟨*false*⟩。该选项的细节见 § 3.1.3。

`\ifusetranslator{`⟨*true*⟩`}{`⟨*false*⟩`}`

这只是下面的`\ifuse<name>`宏的一个特例，因为`translator`是默认数据模型的一部分所以放到这里来说。执行⟨*true*⟩，如果`usetranslator`选项打开 (无论是全局的还是针对当前条目)，否则执行⟨*false*⟩。该选项的细节见 § 3.1.3。

`\ifuse<name>{`⟨*true*⟩`}{`⟨*false*⟩`}`

展开为⟨*true*⟩，如果选项`use<name>`打开 (无论全局还是当前条目的选项)，否则展开为⟨*false*⟩。这一选项的细节详见第 § 3.1.3节。

`\ifcrossrefsource{`⟨*true*⟩`}{`⟨*false*⟩`}`

展开为⟨*true*⟩，如果包含在`.bbl`中的条目的间接引用 (referenced)[54]次数大于`mincrossrefs`，否则展开为⟨*false*⟩。见 § 3.1.2.1。如果条目被直接引用则展开为⟨*false*⟩。

`\ifxrefsource{`⟨*true*⟩`}{`⟨*false*⟩`}`

展开为⟨*true*⟩，如果包含在`.bbl`中的条目的间接引用 (referenced)[55]次数大于optminxrefs，否则展开为⟨*false*⟩。见 § 3.1.2.1。如果条目被直接引用则展开为⟨*false*⟩。

`\ifsingletitle{`⟨*true*⟩`}{`⟨*false*⟩`}`

展开为⟨*true*⟩，如果文献表中只有以`labelname`为名的一片文献，否则展开为⟨*false*⟩。如果没有`labelname`为名的条目，当文献表中有以`labeltitle`为题的文献则展开为⟨*true*⟩，否则展开为⟨*false*⟩。如果条目既没设置`labelname`也没设置`labeltitle`，总是展开为⟨*false*⟩。注意该功能需要显式的打开宏包选项`singletitle`才行。

`\ifuniquetitle{`⟨*true*⟩`}{`⟨*false*⟩`}`

展开为⟨*true*⟩，如果只有一篇文献的题名是`labeltitle`，否则展开为⟨*false*⟩。如果条目的`labeltitle`未设置也展开为⟨*false*⟩。注意: 要使用这一功能需要显式地打开包选项`uniquetitle`。

`\ifuniquebaretitle{`⟨*true*⟩`}{`⟨*false*⟩`}`

展开为⟨*true*⟩，如果`labelname`域为空且只有一篇文献的题名是`labeltitle`，否则展开为⟨*false*⟩。如果条目的`labeltitle`未设置也展开为⟨*false*⟩。注意: 要使用这一功能需要显式地打开包选项`uniquebaretitle`。

[54]应该是交叉引用次数
[55]应该是交叉引用次数

`\ifuniquework{`⟨*true*⟩`}{`⟨*false*⟩`}`

展开为⟨*true*⟩，如果文献表中只有一篇文献的标签名是`labelname`且题名是`labeltitle`，否则展开为⟨*false*⟩。如果条目的`labelname`和`labeltitle`均未设置也展开为⟨*false*⟩。注意: 要使用这一功能需要显式地打开包选项`uniquework`。如果同一条目的`singletitle`和`uniquetitle`都是 false，可能是因为其他条目也有相同的`labelname`或者`labeltitle`。`uniquework`可以让我们知道有另一条目具有相同的`labelname`和`labeltitle`。这对于一种多人合作的情况很有用，当多个同时维护参考文献数据源时，有可能会添加内容相同但引用关键词不同的文献。这一判断能帮助找到这中存在副本情况。

`\ifuniqueprimaryauthor{`⟨*true*⟩`}{`⟨*false*⟩`}`

展开为⟨*true*⟩，如果一篇文献的对于其`labelname`的第一作者的姓是唯一的，否则展开为⟨*false*⟩。如果条目的`labelname`未设置，将展开为⟨*false*⟩。注意使用该功能需要显式的打开包选项`uniqueprimaryauthor`。

`\ifandothers{`⟨*list*⟩`}{`⟨*true*⟩`}{`⟨*false*⟩`}`

展开为⟨*true*⟩，如果⟨*list*⟩已定义并且在`bib`文件中以关键词'`and others`'截短了，否则展开为⟨*false*⟩。⟨*list*⟩可以是 literal 或 name 列表。

`\ifmorenames{`⟨*true*⟩`}{`⟨*false*⟩`}`

展开为⟨*true*⟩，如果当前姓名列表已经截短或将截短，否则展开为⟨*false*⟩。该命令用于姓名列表的格式化指令中，在其它地方使用将展开为⟨*false*⟩。该命令对当前列表执行与`\ifandothers`判断一样的操作。如果判断结果为否，它将检测`listtotal`是否大于`liststop`。该命令用于格式化命令中用以决定是否需要在列表默认打印"and others" or "et al."这样的标注。注意: 当需要检测实在列表中间或者末尾时，即`listcount`是否小于或等于`liststop`，详见第 § 4.4.1节。

`\ifmoreitems{`⟨*true*⟩`}{`⟨*false*⟩`}`

类似于`\ifmorenames`，但检测 literal 列表。用于 literal 列表的格式化指令，其它地方用总是展开为⟨*false*⟩。

`\if<namepart>inits{`⟨*true*⟩`}{`⟨*false*⟩`}`

根据`firstinits`包选项的状态，展开为⟨*true*⟩或⟨*false*⟩(见第 § 3.1.2.3节)。该命令用于姓名列表的格式化指令。

`\ifterseinits{`⟨*true*⟩`}{`⟨*false*⟩`}`

根据`terseinits`包选项的状态，展开为⟨*true*⟩或⟨*false*⟩(见第 § 3.1.2.3节)。该命令用于姓名列表的格式化指令。

`\ifentrytype{`⟨*type*⟩`}{`⟨*true*⟩`}{`⟨*false*⟩`}`

如果当前处理条目类型是⟨*type*⟩，则展开为⟨*true*⟩，否则展开为⟨*false*⟩。

\ifkeyword{⟨*keyword*⟩}{⟨*true*⟩}{⟨*false*⟩}

如果⟨*keyword*⟩能在当前处理的条目的keywords域中找到，展开为⟨*true*⟩，否则展开为⟨*false*⟩。

\ifentrykeyword{⟨*entrykey*⟩}{⟨*keyword*⟩}{⟨*true*⟩}{⟨*false*⟩}

当条目关键词作为\ifkeyword命令参数的变化形式，在判断当前处理条目是否是某一条目时很有用。

\ifcategory{⟨*category*⟩}{⟨*true*⟩}{⟨*false*⟩}

执行⟨*true*⟩，如果当前正在处理条目被指派为由\addtocategory命令定义的⟨*category*⟩中，否则执行⟨*false*⟩。

\ifentrycategory{⟨*entrykey*⟩}{⟨*category*⟩}{⟨*true*⟩}{⟨*false*⟩}

当条目关键词作为\ifcategory命令参数时的变化形式，在判断当前处理条目是否是某一条目时很有用。

\ifciteseen{⟨*true*⟩}{⟨*false*⟩}

展开为⟨*true*⟩，如果当前条目之前已经被引用过，否则展开为⟨*false*⟩。该命令是鲁棒的，用于标注样式中。如果文档中有refsection环境，引用追踪是基于这些环境的。注意：引用追踪器需要显式的以包选项citetracker打开，如果追踪器未打开，该命令总是展开为⟨*false*⟩。另可参见第 § 4.6.4节的\citetrackertrue和\citetrackerfalse开关。

\ifentryseen{⟨*entrykey*⟩}{⟨*true*⟩}{⟨*false*⟩}

当条目关键词作为\ifciteseen命令参数时的变化形式。因为⟨*entrykey*⟩先于判断展开，它也可以用来测试在xref等域中的条目关键词。

```
\ifentryseen{\thefield{xref}}{true}{false}
```

除了一个额外参数，\ifentryseen的操作类似于\ifciteseen。

\ifentryinbib{⟨*entrykey*⟩}{⟨*true*⟩}{⟨*false*⟩}

如果⟨*entrykey*⟩出现当前文献表中，执行⟨*true*⟩，否则执行⟨*false*⟩。该命令用于参考文献著录样式。

\iffirstcitekey{⟨*true*⟩}{⟨*false*⟩}

如果当前处理条目是引用列表中的第一个，执行⟨*true*⟩，否则执行⟨*false*⟩。该命令依赖于citecount, citetotal, multicitecount 和 multicitetotal计数器 (见 § 4.10.5)，因此只能用于\DeclareCiteCommand命令定义的标注命令的循环执行代码⟨*loopcode*⟩中。

\iflastcitekey{⟨*true*⟩}{⟨*false*⟩}

类似于\iffirstcitekey，但判断的是是否为引用列表中的最后一个。

\ifciteibid{⟨*true*⟩}{⟨*false*⟩}

如果当前处理条目于前一条相同，展开为⟨*true*⟩，否则展开为⟨*false*⟩。该命令用于标注样式。如果有refsection环境，追踪器是基于这些环境的。注意:'ibidem' 追踪器需要由ibidtracker包选项显式的打开。该判断命令的运行方式与追踪器运行的模式相关，详见 § 3.1.2.3。如果追踪器未打开，总是展开为⟨*false*⟩。另可参见 § 4.6.4节的\citetrackertrue和\citetrackerfalse开关。

\ifciteidem{⟨*true*⟩}{⟨*false*⟩}

如果当前处理条目的责任者 (即作者或编者) 于前一条目的相同，展开为⟨*true*⟩，否则展开为⟨*false*⟩。该命令用于标注样式。如果有refsection环境，追踪器是基于这些环境的。注意: 'idem' 追踪器需要由idemtracker包选项显式的打开。该判断命令的运行方式与追踪器运行的模式相关，详见 § 3.1.2.3。如果追踪器未打开，总是展开为⟨*false*⟩。另可参见 § 4.6.4节的\citetrackertrue和\citetrackerfalse开关。

\ifopcit{⟨*true*⟩}{⟨*false*⟩}

该命令类似于\ifciteibid,但只要当前处理等条目的作者或编者与前一条目相同，则展开为⟨*true*⟩。注意: 'opcit' 追踪器需要由opcittracker包选项显式的打开。该判断命令的运行方式与追踪器运行的模式相关，详见 § 3.1.2.3。如果追踪器未打开，总是展开为⟨*false*⟩。另可参见 § 4.6.4节的\citetrackertrue和\citetrackerfalse开关。

\ifloccit{⟨*true*⟩}{⟨*false*⟩}

该命令类似于\ifopcit，但还要比较⟨*postnote*⟩的参数，如果他们相同且是数值 (§ 4.6.2节的\ifnumerals命令判断)，则展开为⟨*true*⟩。即: 如果引文的页码与前一文献相同则展开为 true。注意: 'loccit' 追踪器需要由loccittracker包选项显式的打开。该判断命令的运行方式与追踪器运行的模式相关，详见 § 3.1.2.3。如果追踪器未打开，总是展开为⟨*false*⟩。另可参见 § 4.6.4节的\citetrackertrue和\citetrackerfalse开关。

\iffirstonpage{⟨*true*⟩}{⟨*false*⟩}

该命令的运行与pagetracker包选项相关，如果选项设置成 page，当当前项是页中的第一项，展开为⟨*true*⟩，否则展开为⟨*false*⟩。如果选项设置成 spread，当当前项是合页中的第一项，展开为⟨*true*⟩，否则展开为⟨*false*⟩。如果选项未打开，总是展开为⟨*false*⟩。根据所处环境不同，'item' 可以是一个标注，或者参考文献表中的条目。注意该命令区分正文文本和脚注，例如，当在某页的第一个脚注中使用，即便是文中有一个标注且先于该脚注。另可参见 § 4.6.4节的\pagetrackertrue和\pagetrackerfalse开关。

\ifsamepage{⟨*instance 1*⟩}{⟨*instance 2*⟩}{⟨*true*⟩}{⟨*false*⟩}

如果两个引用实例位于同于页或者同一合页中，展开为⟨*true*⟩，否则为⟨*false*⟩。一个引用实例可以是一个标注也可以是文献表中的条目。这些实例用instcount计数区分，见 § 4.10.5。该命令的运行与pagetracker包选项相关，如果选项设置成spread，其本质是'if same spread'(是否同意合页) 的判断。如果选项未打开，总是展开为⟨*false*⟩。参数⟨*instance 1*⟩和⟨*instance 2*⟩以e-TeX's \numexpr方式当成整数表达式处理。这意味着可以在参数中计算。比如:

```
\ifsamepage{\value{instcount}}{\value{instcount}-1}{true}{false}
```

注意:\value命令不是以\the为前缀，在第二个参数中做了减法运算。如果⟨*instance 1*⟩ 或 ⟨*instance 2*⟩是无效数字 (比如一个负值)，总是展开为⟨*false*⟩。也要注意该命令不区分正文和脚注。另可参见 § 4.6.4节的\pagetrackertrue和\pagetrackerfalse开关。

\ifinteger{⟨*string*⟩}{⟨*true*⟩}{⟨*false*⟩}

如果⟨*string*⟩是一个正整数，展开为⟨*true*⟩，否则为⟨*false*⟩，该命令鲁棒。

\ifnumeral{⟨*string*⟩}{⟨*true*⟩}{⟨*false*⟩}

如果⟨*string*⟩是一个阿拉伯或者罗马数字，展开为⟨*true*⟩，否则为⟨*false*⟩，该命令鲁棒。另可参见 § 4.6.4节的\DeclareNumChars和\NumCheckSetup命令。

\ifnumerals{⟨*string*⟩}{⟨*true*⟩}{⟨*false*⟩}

如果⟨*string*⟩是一个阿拉伯或者罗马数字的范围或列表，展开为⟨*true*⟩，否则为⟨*false*⟩，该命令鲁棒。相比于\ifnumeral命令，当参数像 "52–58", "14/15", "1, 3, 5"等时，该命令会执行⟨*true*⟩。另可参见 § 4.6.4节的\DeclareNumChars，\NumCheckSetup，\DeclareRangeCommands和 \NumCheckSetup命令。

\ifpages{⟨*string*⟩}{⟨*true*⟩}{⟨*false*⟩}

类似于\ifnumerals，但也考虑 § 4.6.4节的\DeclarePageCommands命令。

\iffieldint{⟨*field*⟩}{⟨*true*⟩}{⟨*false*⟩}

类似于\ifinteger命令，但使用⟨*field*⟩的值而不是一个字符串，如果域未定义，执行⟨*false*⟩。

\iffieldnum{⟨*field*⟩}{⟨*true*⟩}{⟨*false*⟩}

类似于\ifnumeral命令，但使用⟨*field*⟩的值而不是一个字符串，如果域未定义，执行⟨*false*⟩。

\iffieldnums{⟨*field*⟩}{⟨*true*⟩}{⟨*false*⟩}

类似于\ifnumerals命令，但使用⟨*field*⟩的值而不是一个字符串，如果域未定义，执行⟨*false*⟩。[56]

[56]是否可以用来解析卷期的范围?

\iffieldpages{⟨*field*⟩}{⟨*true*⟩}{⟨*false*⟩}

类似于\ifpages命令，但使用⟨*field*⟩的值而不是一个字符串，如果域未定义，执行⟨*false*⟩。

\ifbibstring{⟨*string*⟩}{⟨*true*⟩}{⟨*false*⟩}

如果⟨*string*⟩是已知的本地化关键词，展开为⟨*true*⟩，否则⟨*false*⟩。默认定义的本地化字符串见 § 4.9.2。新的字符串可以用命令\NewBibliographyString定义。

\ifbibxstring{⟨*string*⟩}{⟨*true*⟩}{⟨*false*⟩}

类似于\ifbibstring，但⟨*string*⟩是展开的。

\iffieldbibstring{⟨*field*⟩}{⟨*true*⟩}{⟨*false*⟩}

类似于\ifbibstring，但使用⟨*field*⟩域的值而不是一个字符串，如果域未定义，执行⟨*false*⟩。

\ifdriver{⟨*entrytype*⟩}{⟨*true*⟩}{⟨*false*⟩}

展开为⟨*true*⟩如果⟨*entrytype*⟩的驱动存在，否则为⟨*false*⟩。

\ifcapital{⟨*true*⟩}{⟨*false*⟩}

如果 Biblatex 的标点追踪器将当前位置的本地化字符串大写，则执行⟨*true*⟩，否则执行⟨*false*⟩。给命令在格式化指令中对于姓名的某一部分做有条件的大写处理时有用。

\ifcitation{⟨*true*⟩}{⟨*false*⟩}

当处于标注中则展开为⟨*true*⟩，否则为⟨*false*⟩。注意这一命令与其所在的最外层环境有关。比如，当由\DeclareCiteCommand命令定义的标注命令执行一个由\DeclareBibliographyDriver定义的驱动，则任何在该驱动中的\ifcitation都会展开为⟨*true*⟩。在 § 4.11.6可以看到一个实例。

\ifbibliography{⟨*true*⟩}{⟨*false*⟩}

当处于文献表中则展开为⟨*true*⟩，否则为⟨*false*⟩。注意这一命令与其所在的最外层环境有关。比如，当由\DeclareBibliographyDriver命令定义的驱动执行一个由\DeclareCiteCommand定义的标注，则任何在该标注中的\ifbibliography都会展开为⟨*true*⟩。在 § 4.11.6可以看到一个实例。

\ifnatbibmode{⟨*true*⟩}{⟨*false*⟩}

根据 § 3.1.1的natbib选项展开为⟨*true*⟩或⟨*false*⟩。

\ifciteindex{⟨*true*⟩}{⟨*false*⟩}

根据 § 3.1.2.1的indexing选项展开为⟨*true*⟩或⟨*false*⟩。

`\ifbibindex{⟨true⟩}{⟨false⟩}`

根据 § 3.1.2.1的`indexing`选项展开为⟨*true*⟩或⟨*false*⟩。

`\iffootnote{⟨true⟩}{⟨false⟩}`

当处于脚注中时，展开为⟨*true*⟩，否则为⟨*false*⟩。注意: 在`minipage`中的脚注被认为正文的一部分。当处于页面底部的脚注中或者由`endnotes`提供的 endnotes 中时，只会展开为⟨*true*⟩。

`citecounter` 这一计数器表示当前处理条目在当前 reference section 中的引用次数。注意该功能需要以包选项`citecounter`显式的打开。如果选项设置为 `context`，正文和脚注中的引用分别计数。这种情况下，`citecounter`记录其所在环境中的值。

`uniquename` 这一计数器用于`labelname`列表。它以每个名字为基础进行设置。如果姓不同，它的值设置为 0，当增加姓名的其它部分的首字母使得姓名能区分，则设置为 1，如果需要完整的姓名才能区分，则设置为 2。作者年值和作者标题值得标注格式需要这一信息来增加姓名的其它部分以对姓相同的作者进行引用。比如: 当引用列表中有一个'John Doe' 和一个'Edward Doe'，该计数器将设置为 1。如果有有一个'John Doe' 和一个'John Doe'，该计数器将设置为 2。如果选项设置成 `init`/`allinit`/`mininit`，那么计数器将限制值最大为 1。这对于标注样式不打印全名而使用首字母来区分姓名很有用。如果添加首字母还无法区分姓名，`uniquename`将设置为 0。该功能需要以包选项`uniquename`显式的打开。注意`uniquename` 是对`\printnames`局部的，仅根据`labelname`列表或其来源姓名列表 (典型如`author` 或`editor`) 设置。它的值在任何正文中都是 0, 即它仅在处理姓名的格式化指令中计算，更多细节和实例见 § 4.11.4。

`uniquelist` 该计数器用于`labelname`列表。它以每个域为基础进行设置。它的值表示当使用`maxnames`/`minnames`自动将姓名列表截短后导致标注歧义时，消除歧义需要的最小姓名数。比如，有一篇作者是'Doe/Smith/Johnson' 的文献和另一篇作者是'Doe/Edwards/Williams' 的文献，设置 `maxnames=1` 将导致两篇的作者都是'Doe et al.'。这种情况下，两个条目的`labelname`列表的`uniquelist`将设置成 2，因为至少需要两个名字来区分。注意`uniquelist`是对`\printnames`命令局部的，仅根据`labelname`列表或其来源姓名列表 (典型如`author` 或`editor`) 设置。它的值在任何正文中都是 0, 即它仅在处理姓名的格式化指令中计算，如果该值存在，则`\printnames`命令在处理姓名列表时将自动应用，即自动覆盖`maxnames`/`minnames`。该功能需要以包选项`uniquelist`显式的打开。更多细节和实例见 § 4.11.4。

`parenlevel` 圆括号和/或方括号的嵌套层级。该信息仅在 § 3.1.2.3的`parentracker`选项打开的情况下提供。

### 4.6.3 使用`\ifboolexpr`和`\ifthenelse`的判断

第 § 4.6.2节介绍的判断可以与`etoolbox`宏包提供的`\ifboolexpr`命令和`ifthen`宏包提供的`\ifthenelse`命令一同使用。这种情况下，其语法略有差异，判断命令的⟨*true*⟩和⟨*false*⟩参数自动省略，而直接传递给`\ifboolexpr` 或`\ifthenelse`。注意，使用这些命令需要一些计算代价。如果不需要一些布尔操作，使用 § 4.6.2节的 stand-alone 判断命令更高效。

\ifboolexpr{⟨*expression*⟩}{⟨*true*⟩}{⟨*false*⟩}

该etoolbox包命令允许进行包括布尔运算和编组的复杂判断。

```
\ifboolexpr{ (
               test {\ifnameundef{editor}}
               and
               not test {\iflistundef{location}}
             )
             or test {\iffieldundef{year}}
  }
  {...}
  {...}
```

\ifthenelse{⟨*tests*⟩}{⟨*true*⟩}{⟨*false*⟩}

该ifthen包命令允许进行包括布尔运算和编组的复杂判断。

```
\ifthenelse{ \(
               \ifnameundef{editor}
               \and
               \not \iflistundef{location}
             \)
             \or \iffieldundef{year}
  }
  {...}
  {...}
```

Biblatex 提供的附加判断命令仅在标注命令和文献表中使用\ifboolexpr或\ifthenelse命令时可用。

### 4.6.4 综合命令

本节介绍参考文献著录和标注样式中使用的一些综合命令和小巧工具。

\newbibmacro{⟨*name*⟩}[⟨*arguments*⟩][⟨*optional*⟩]{⟨*definition*⟩}
\newbibmacro*{⟨*name*⟩}[⟨*arguments*⟩][⟨*optional*⟩]{⟨*definition*⟩}

定义一个用于后面\usebibmacro调用的宏。该命令的语法类似于\newcommand，除了⟨*name*⟩可以包含一些数字或标点，但不以斜杠开头。可选参数⟨*arguments*⟩是一个整数用于指定宏需要处理的参数数量。如果⟨*optional*⟩给出，它指定了该宏的第一个参数的默认值，这第一个参数自动变成为可选参数。相比于\newcommand，当宏已经定义时，\newbibmacro命令会给出一个警告信息，并自动转换为\renewbibmacro命令。类似于\newcommand，该命令的常规形式在定义中使用\long前缀，而带星的命令则没有。如果一个宏声明为 long，它的参数可以包含\par记号。

提供\newbibmacro和\renewbibmacro命令是为了方便使用，样式作者也可以使用\newcommand 或\def。然而，需要注意，共享文件 biblatex.def 中的绝大多数定义都是用\newbibmacro定义的，因此，要使用和修改它们要用相应的方式处理。

\renewbibmacro{⟨*name*⟩}[⟨*arguments*⟩][⟨*optional*⟩]{⟨*definition*⟩}
\renewbibmacro*{⟨*name*⟩}[⟨*arguments*⟩][⟨*optional*⟩]{⟨*definition*⟩}

类似于\newbibmacro，但用于重定义⟨*name*⟩。相比于\newcommand，当宏未定义时，\renewbibmacro命令给出一个警告信息，并自动转换为\newbibmacro命令。

\providebibmacro{⟨*name*⟩}[⟨*arguments*⟩][⟨*optional*⟩]{⟨*definition*⟩}
\providebibmacro*{⟨*name*⟩}[⟨*arguments*⟩][⟨*optional*⟩]{⟨*definition*⟩}

类似于\newbibmacro，但仅在⟨*name*⟩未定义时定义宏。该命令概念上类似于\providecommand。

\usebibmacro{⟨*name*⟩}
\usebibmacro*{⟨*name*⟩}

该命令执行由\newbibmacro定义的宏⟨*name*⟩。如果宏带参数，只要简单的跟在⟨*name*⟩后面即可。该命令的常规形式会处理⟨*name*⟩，而带星的命令不会。⟨*name*⟩ while the starred variant does not.

\savecommand{⟨*command*⟩}
\restorecommand{⟨*command*⟩}

这两个命令用来保存和恢复⟨*command*⟩，其中⟨*command*⟩必须是以斜杠开头的命令。两个命令都在局部范围内起作用。它们主要用于本地化文件中。

\savebibmacro{⟨*name*⟩}
\restorebibmacro{⟨*name*⟩}

这两个命令用来保存和恢复宏⟨*name*⟩，其中⟨*name*⟩由\newbibmacro定义的宏的标识。两个命令都在局部范围内起作用。它们主要用于本地化文件中。

\savefieldformat[⟨*entry type*⟩]{⟨*format*⟩}
\restorefieldformat[⟨*entry type*⟩]{⟨*format*⟩}

这两个命令用来保存和恢复格式化指令⟨*format*⟩，其中⟨*format*⟩由\DeclareFieldFormat定义。两个命令都在局部范围内起作用。它们主要用于本地化文件中。

\savelistformat[⟨*entry type*⟩]{⟨*format*⟩}
\restorelistformat[⟨*entry type*⟩]{⟨*format*⟩}

这两个命令用来保存和恢复格式化指令⟨*format*⟩，其中⟨*format*⟩由\DeclareListFormat定义。两个命令都在局部范围内起作用。它们主要用于本地化文件中。

\savenameformat[⟨*entry type*⟩]{⟨*format*⟩}
\restorenameformat[⟨*entry type*⟩]{⟨*format*⟩}

这两个命令用来保存和恢复格式化指令⟨*format*⟩，其中⟨*format*⟩由\DeclareNameFormat定义。两个命令都在局部范围内起作用。它们主要用于本地化文件中。

\ifbibmacroundef{⟨*name*⟩}{⟨*true*⟩}{⟨*false*⟩}

如果参考文献宏⟨*name*⟩未定义，展开为⟨*true*⟩否则为⟨*false*⟩。

\iffieldformatundef[⟨*entry type*⟩]{⟨*name*⟩}{⟨*true*⟩}{⟨*false*⟩}
\iflistformatundef[⟨*entry type*⟩]{⟨*name*⟩}{⟨*true*⟩}{⟨*false*⟩}
\ifnameformatundef[⟨*entry type*⟩]{⟨*name*⟩}{⟨*true*⟩}{⟨*false*⟩}

如果参考文献格式化指令⟨*format*⟩未定义，展开为⟨*true*⟩否则为⟨*false*⟩。otherwise.

\usedriver{⟨*code*⟩}{⟨*entrytype*⟩}

执行⟨*entrytype*⟩类条目的参考文献驱动。在由\DeclareCiteCommand定义的标注命令的⟨*loopcode*⟩中调用该命令是打印类似于一个参考文献条目的完整标注的简单方法。诸如\newblock等命令无法用于标注，自动省略。附加的初始化命令可以通过⟨*code*⟩参数传递。该参数在一个编组内执行，这一编组用于运行相应驱动。注意:该参数语法上是必须的，但可以留空。也要注意如果autolang包选项打开的话，该命令会自动切换语言。

\bibhypertarget{⟨*name*⟩}{⟨*text*⟩}

hyperref的\hypertarget命令的封套[57]。⟨*name*⟩是超链接锚的名字，⟨*text*⟩的内容作为超链接锚，可以是任意可打印文字或代码。如果文档中存在refsection环境，⟨*name*⟩是基于当前 refsection 环境。如果hyperref包选项未打开或者hyperref包未加载，该命令简单的传递⟨*text*⟩变量。另可参见 § 4.10.4节的格式化指令bibhypertarget。

\bibhyperlink{⟨*name*⟩}{⟨*text*⟩}

hyperref的\hyperlink命令的包套。⟨*name*⟩是由\bibhypertarget定义的超链接锚的名字，⟨*text*⟩的内容将转变成超链接，可以是任意可打印文字或代码。如果文档中存在refsection环境，⟨*name*⟩是基于当前 refsection 环境。如果hyperref包选项未打开或者hyperref包未加载，该命令简单的传递⟨*text*⟩变量。另可参见 § 4.10.4节的格式化指令 bibhyperlink。

\bibhyperref[⟨*entrykey*⟩]{⟨*text*⟩}

将⟨*text*⟩转变为指向参考文献表中的⟨*entrykey*⟩(即某一条目) 的内部链接。如果⟨*entrykey*⟩省略，该命令使用当前正在处理的条目的引用关键词。该命令用于将标注转换为可点击的超链接，可以链接到参考文献表中的相应条目。链接目标

[57] wrapper 译为包围器，封套，包套？

由 Biblatex 自动标记。如果文档中有多个文献表，链接目标将是所有文献表中第一个出现的⟨*entrykey*⟩条目。如果文档中存在`refsection`环境，则超链接基于当前 refsection 环境。另可参见 § 4.10.4节的格式化指令 `bibhyperref`。

`\ifhyperref{`⟨*true*⟩`}{`⟨*false*⟩`}`

展开为⟨*true*⟩，如果`hyperref`包选项已打开 (意味着`hyperref`包已加载)，否则展开为⟨*false*⟩。

`\docsvfield{`⟨*field*⟩`}`

类似于`etoolbox`包的`\docsvlist`命令，差别在于它的参数是一个域名。域的值将以一个 comma-separated(英文逗号分隔) 的列表进行解析。如果⟨*field*⟩为定义，该命令展开为空字符串。

`\forcsvfield{`⟨*handler*⟩`}{`⟨*field*⟩`}`

类似于`etoolbox`包的`\forcsvlist`命令，差别在于它的参数是一个域名。域的值将以一个 comma-separated(英文逗号分隔) 的列表进行解析。如果⟨*field*⟩为定义，该命令展开为空字符串

`\MakeCapital{`⟨*text*⟩`}`

类似于`\MakeUppercase`，但仅将⟨*text*⟩的第一个可打印字符转换为大写。注意:`\MakeUppercase`命令的限制也适用于这一命令。即: ⟨*text*⟩中的所有命令必须是鲁棒的或者以`\protect`为前缀，因为在大写操作中⟨*text*⟩需要展开。除了 Ascii 字符和标准重音命令外，该命令也处理`inputenc`包的活动字符和`babel`包的缩略词。如果⟨*text*⟩以一个控制序列开头，不做任何大写操作。该命令是鲁棒的。

`\MakeSentenceCase{`⟨*text*⟩`}`
`\MakeSentenceCase*{`⟨*text*⟩`}`

将⟨*text*⟩参数转换为 sentence case(句子模式)，即字符串中的第一个单词首字母大写而剩下其他部分转换为小写。该命令是鲁棒的。带星号的命令与常规命令 (不带星号) 的差别在于它能考虑条目的语言，根据`langid`域指定。只有当`langid`未定义或者定为由`\DeclareCaseLangs`命令 (见后面) 声明的某种语言时，它才将⟨*text*⟩转换为句子模式。[58] 否则⟨*text*⟩不做任何改变。推荐使用`\MakeSentenceCase*`而不是常规命令。两个命令都支持`bib`文件的传统 BibTeX 规范，即: 遇到任何以花括号包围的内容大小写都不作变化，例如:

```
\MakeSentenceCase{an Introduction to LaTeX}
\MakeSentenceCase{an Introduction to {LaTeX}}
```

将得到:

[58] 默认情况下，如下语言支持转换: `american`, `british`, `canadian`, `english`, `australian`, `newzealand` as well as the aliases `USenglish` and `UKenglish`. 要扩展或修改该列表请使用`\DeclareCaseLangs`命令。

```
An introduction to latex
An introduction to LaTeX
```

在以传统 BibTeX 方式设计的bib文件中，为阻止字母的 case-changing(大小写变化)，将单个字母用花括号包围是一种相当常见的方法。

```
title = {An Introduction to {L}a{T}e{X}}
```

这种方式存在一个问题是括号会压缩被包围字母两侧的字距。最好的方式是如第一个例子所示的那样，将整个单词都包围起来。

\mkpageprefix[⟨*pagination*⟩][⟨*postpro*⟩]{⟨*text*⟩}

该命令用于域格式化指令中，包括标注命令的⟨*postnote*⟩参数和文献条目的pages域的格式化。默认情况下，它将会解析⟨*text*⟩参数，并且以'p.' or 'pp.' 做为前缀。可选参数⟨*pagination*⟩保存指示 pagination 类型的域名，可以是pagination或bookpagination，默认是pagination。前缀与⟨*text*⟩之间的间距可以通过重定义\ppspace命令来调整。默认是一个不可断行的词内空格。详见 §§ 2.3.10 和 3.13.3。另可参见\DeclareNumChars, \DeclareRangeChars, \DeclareRangeCommands, 和\NumCheckSetup。可选参数⟨*postpro*⟩指定了用于对⟨*text*⟩后处理的宏。如果只给出一个可选参数，将作为⟨*pagination*⟩，下面是两个典型例子:

```
\DeclareFieldFormat{postnote}{\mkpageprefix[pagination]{#1}}
\DeclareFieldFormat{pages}{\mkpageprefix[bookpagination]{#1}}
```

第一个例子中的可选参数pagination可以省略。

\mkpagetotal[⟨*pagination*⟩][⟨*postpro*⟩]{⟨*text*⟩}

该命令类似于\mkpageprefix，差别在于它用于条目的pagetotal域，即它将打印"123 pages"而不是"page 123"。可选参数⟨*pagination*⟩默认是bookpagination。在⟨*text*⟩和后缀之间的间距可由对\ppspace重定义进行调整。可选参数⟨*postpro*⟩指定了用于对⟨*text*⟩后处理的宏。如果只给出一个可选参数，将作为⟨*pagination*⟩，下面是一个典型例子:

```
\DeclareFieldFormat{pagetotal}{\mkpagetotal[bookpagination]{#1}}
```

在本例中可选参数bookpagination可省略。

\mkcomprange[⟨*postpro*⟩]{⟨*text*⟩}
\mkcomprange*[⟨*postpro*⟩]{⟨*text*⟩}

该命令，用于域格式化指令，将⟨*text*⟩参数解析为页码范围并且压缩这些范围。扫描程序将\bibrangedash和 hyphens 作为范围间隔符。支持范围列表

| Input | Output | | |
|---|---|---|---|
| | mincomprange=10 | mincomprange=100 | mincomprange=1000 |
| 11–15 | 11–5 | 11–15 | 11–15 |
| 111–115 | 111–5 | 111–5 | 111–115 |
| 1111–1115 | 1111–5 | 1111–5 | 1111–5 |
| | maxcomprange=1000 | maxcomprange=100 | maxcomprange=10 |
| 1111–1115 | 1111–5 | 1111–5 | 1111–5 |
| 1111–1155 | 1111–55 | 1111–55 | 1111–1155 |
| 1111–1555 | 1111–555 | 1111–1555 | 1111–1555 |
| | mincompwidth=1 | mincompwidth=10 | mincompwidth=100 |
| 1111–1115 | 1111–5 | 1111–15 | 1111–115 |
| 1111–1155 | 1111–55 | 1111–55 | 1111–155 |
| 1111–1555 | 1111–555 | 1111–555 | 1111–555 |

**Table 14: \mkcomprange setup**

以\bibrangessep(Biber[59]) 或 commas/semicolon(BibTeX) 分隔。如果因为某些原因需要隐藏来自 list/range 扫描程序的一个字符，那么可以将该字符或者整个字符串用花括号包围起来。可选参数⟨*postpro*⟩指定了一个用于对⟨*text*⟩进行后处理的宏。怎么使用该参数见\mkcomprange命令。带星的命令的差别在于⟨*postpro*⟩参数应用于列表的各项。例如:

```
\DeclareFieldFormat{postnote}{\mkcomprange[{\mkpageprefix[pagination]}
    ↪ ]{#1}}
```

注意:\mkcomprange命令首先处理，\mkpageprefix则作为后处理器。也要注意⟨*postpro*⟩被额外的一对花括号包围。这仅在特殊情况下需要，为阻止 LaTeX 的可选参数扫描器与嵌套的方括号混淆。带星的命令与不带星命令的差别是它应用于值得列表，例如:

```
\mkcomprange[\mkpageprefix]{5, 123-129, 423-439}
\mkcomprange*[\mkpageprefix]{5, 123-129, 423-439}
```

将输出:

```
pp. 5, 123-9, 423-39
p. 5, pp. 123-9, pp. 423-39
```

\mkfirstpage[⟨*postpro*⟩]{⟨*text*⟩}
\mkfirstpage*[⟨*postpro*⟩]{⟨*text*⟩}

该命令，用于域格式化指令，将⟨*text*⟩参数解析为页码范围并且仅打印这些范围的起始页码。扫描程序将\bibrangedash和 hyphens 作为范围间隔符。支持范围列表

[59]Biber 总会将 commas/semicolon(逗号或冒号) 的多范围分隔符转换为 \bibrangessep，因此可以在样式中控制。

以\bibrangessep(Biber[60]) 或 commas/semicolon(BibTeX) 分隔。如果因为某些原因需要隐藏来自 list/range 扫描程序的一个字符，那么可以将该字符或者整个字符串用花括号包围起来。可选参数⟨*postpro*⟩指定了一个用于对⟨*text*⟩进行后处理的宏。怎么使用该参数见\mkcomprange命令。带星的命令的差别在于⟨*postpro*⟩参数应用于列表的各项。例如:

```
\mkfirstpage[\mkpageprefix]{5, 123-129, 423-439}
\mkfirstpage*[\mkpageprefix]{5, 123-129, 423-439}
```

将输出:

```
pp. 5, 123, 423
p. 5, p. 123, p. 423
```

\rangelen{⟨*rangefield*⟩}该命令将其参数解析为一个范围，并返回范围的长度。对于开口的范围将返回-1。这可以作为样式中一些判断的一部分，例如将‘f’ 作为只有两页的范围的后缀，比如范围‘36-37’ 将打印‘36f’。这可以通过命令\ifnumcomp实现:

- Calculate the total of multiple ranges in the same field such as ‘1-10, 20-30’
- Handle implicit ranges such as ‘22-4’ and ‘130-33’
- Handle roman numeral ranges in upper and lower case and consisting of both ASCII and Unicode roman numeral representations.

下面是一些例子:

| | |
|---|---|
| pages = ‘10’ | \rangelen{pages} returns ’1’ |
| pages = ‘10-15’ | \rangelen{pages} returns ’6’ |
| pages = ‘10-15,47-53’ | \rangelen{pages} returns ’13’ |
| pages = ‘10-’ | \rangelen{pages} returns ’-1’ |
| pages = ‘-10’ | \rangelen{pages} returns ’-1’ |
| pages = ‘48-9’ | \rangelen{pages} returns ’2’ |
| pages = ‘172-77’ | \rangelen{pages} returns ’6’ |
| pages = ‘i-vi’ | \rangelen{pages} returns ’6’ |
| pages = ‘X-XX’ | \rangelen{pages} returns ’11’ |
| pages = ‘VII-xii’ | \rangelen{pages} returns ’6’ |
| pages = ‘VII-xii, 145-7, 135-39’ | \rangelen{pages} returns ’14’ |

\rangelen命令可以用于判断中:

```
\ifnumcomp{\rangelen{pages}}{=}{1}{add ’f’}{do nothing}
```

[60]Biber 总会将 commas/semicolon(逗号或冒号) 的多范围分隔符转换为 \bibrangessep ，因此可以在样式中控制。

\DeclareNumChars{⟨*characters*⟩}
\DeclareNumChars*{⟨*characters*⟩}

该命令设置 § 4.6.2节的\ifnumeral, \ifnumerals, 和\ifpages命令。该设置也将影响\iffieldnum, \iffieldnums, \iffieldpages, \mkpageprefix 和 \mkpagetotal命令。⟨*characters*⟩参数是一个无分隔符的符号列表，将作为数字的一部分进行处理。不带星命令将替换当前设置，带星命令则将其参数附加到当前列表中。默认设置为:

```
\DeclareNumChars{.}
```

这意味着 (节或者其他) 数值比如 '3.4.5' 将被认为是一个数字。注意，默认检测的是阿拉伯和罗马数字，没有必要对此做显式声明。

\DeclareRangeChars{⟨*characters*⟩}
\DeclareRangeChars*{⟨*characters*⟩}

该命令设置 § 4.6.2的\ifnumerals和\ifpages命令。其设置还将影响\iffieldnums，\iffieldpages，\mkpageprefix和\mkpagetotal。⟨*characters*⟩参数是一个无分隔符的符号列表，将作为范围指示符进行处理。不带星命令将替换当前设置，带星命令则将其参数附加到当前列表中。默认设置为:

```
\DeclareRangeChars{~,;-+/}
```

这意味着比如'3–5', '35+', '8/9' 等字符串会被\ifnumerals和\ifpages认为是一个范围。这些字符串中的非范围字符将被认为是数字。因此，类似于'3a–5a' 和'35b+' 之类的字符串默认情况下不被认为是范围。更多细节详见 §§ 2.3.10 和 3.13.3。

\DeclareRangeCommands{⟨*commands*⟩}
\DeclareRangeCommands*{⟨*commands*⟩}

该命令类似于\DeclareRangeChars，差别在于⟨*commands*⟩参数是一个无分隔符的命令列表，将被视为范围指示符。不带星命令将替换当前设置，带星命令则将其参数附加到当前列表中。默认列表相当长，应该覆盖所有一般情况。下面是一个简单例子:

```
\DeclareRangeCommands{\&\bibrangedash\textendash\textemdash\psq\psqq}
```

更多细节参见 §§ 2.3.10 和 3.13.3。

\DeclarePageCommands{⟨*commands*⟩}
\DeclarePageCommands*{⟨*commands*⟩}

该命令类似于\DeclareRangeCommands，差别在于它仅影响\ifpages和\iffieldpages判断，而不影响\ifnumerals 和\iffieldnums。默认设置为:

```
\DeclarePageCommands{\pno\ppno}
```

\NumCheckSetup{⟨*code*⟩}

该命令用于临时重定义一些命令，若不重定义，这些命令将与 § 4.6.2节的\ifnumeral, \ifnumerals, \ifpages命令执行的判断冲突。该设置也将影响\iffieldnum, \iffieldnums, \iffieldpages, \mkpageprefix和\mkpagetotal。这些命令将在组内执行⟨*code*⟩。因为上述命令将展开为字符串用于分析，可以利用将冲突命令展开为空字符串 (将被判断命令忽略) 的方式来移除这些命令。更多细节参见 §§ 2.3.10 和 3.13.3。

\DeclareCaseLangs{⟨*languages*⟩}
\DeclareCaseLangs*{⟨*languages*⟩}

定义语言列表，该列表在\MakeSentenceCase*命令将一个字符串转换成句子时考虑。⟨*languages*⟩参数是一个由babel/polyglossia语言标识构成的 comma-separated (逗号分隔) 列表。不带星命令用于替换当前设置，而带星的命令用于附加当前列表。默认的设置为:

```
\DeclareCaseLangs{%
  american,british,canadian,english,australian,newzealand,USenglish,
    ↪ UKenglish}
```

语言标识的列表见babel/polyglossia手册和表 3。

\BibliographyWarning{⟨*message*⟩}

该命令类似于\PackageWarning，但打印内容除了输入行号外还有当前处理条目的引用关键词。如果⟨*message*⟩相当长，可以使用\MessageBreak命令来断行。注意: 标准的\PackageWarning命令在参考文献中使用时无法提供一个有意义的提示，因为其打印的输入行号只是\printbibliography命令所在的行号。

\pagetrackertrue
\pagetrackerfalse

这些命令将打开或关闭局部引用追踪器 (这将影响来自 § 4.6.2节的\iffirstonpage和\ifsamepage判断)。可在标注命令定义或者正文中的任意位置使用。要使标注命令完全排除页码追踪，可以在\DeclareCiteCommand命令的⟨*precode*⟩参数中使用\pagetrackerfalse。详见 § 4.3.1。注意: 当全局页码追踪关闭时，这些命令无效。

\citetrackertrue
\citetrackerfalse

这些命令将打开或关闭所有的局部引用追踪器 (这将影响来自 § 4.6.2节的 \ifciteseen, \ifentryseen, \ifciteibid, 和\ifciteidem判断)。可在标注命令定义或者正文中的任意位置使用。要使标注命令完全排除页码追踪，可以在\DeclareCiteCommand命令的⟨*precode*⟩参数中使用\citetrackerfalse。详见 § 4.3.1。注意: 当全局追踪关闭时，这些命令无效。

\backtrackertrue
\backtrackerfalse

这些命令将打开或关闭所有的局部 backref 追踪器。可在标注命令定义或者正文中的任意位置使用。要使标注命令完全排除反向链接追踪，可以在\DeclareCiteCommand命令的⟨*precode*⟩参数中使用\backtrackerfalse。注意：当 backref 选项未进行全局设置，这些命令无效。

## 4.7 标点和间距

The Biblatex package provides elaborate facilities designed to manage and track punctuation and spacing in the bibliography and in citations. These facilities work on two levels. The high-level commands discussed in § 4.7.1 deal with punctuation and whitespace inserted by the bibliography style between the individual segments of a bibliography entry. The commands in §§ 4.7.2、4.7.3、4.7.4 work at a lower level. They use TeX's space factor and modified space factor codes to track punctuation in a robust and efficient way. This way it is possible to detect trailing punctuation marks within fields, not only those explicitly inserted between fields. The same technique is also used for automatic capitalization of localisation strings, see \DeclareCapitalPunctuation in § 4.7.5 as well as § 4.8 for details. Note that these facilities are only made available locally in citations and bibliographies. They will not affect any other part of a document.

### 4.7.1 块和单元标点 Block and Unit Punctuation

The major segments of a bibliography entry are 'blocks' and 'units'. A block is the larger segment of the two, a unit is shorter or at most equal in length. For example, the values of fields such as title or note usually form a unit which is separated from subsequent data by a period or a comma. A block may comprise several fields which are treated as separate units, for example publisher, location, and year. The segmentation of an entry into blocks and units is at the discretion of the bibliography style. An entry is segmented by inserting \newblock and \newunit commands at suitable places and \finentry at the very end (see § 4.2.3 for an example). See also § 4.11.7 for some practical hints.

\newblock Records the end of a block. This command does not print anything, it merely marks the end of the block. The block delimiter \newblockpunct will be inserted by a subsequent \printtext, \printfield, \printlist, \printnames, or \bibstring command. You may use \newblock at suitable places without having to worry about spurious blocks. A new block will only be started by the next \printfield (or similar) command if this command prints anything. See § 4.11.7 for further details.

\newunit Records the end of a unit and puts the default delimiter \newunitpunct in the punctuation buffer. This command does not print anything, it merely marks the end of the unit. The punctuation buffer will be inserted by the next \printtext, \printfield, \printlist, \printnames, or \bibstring command. You may use \newunit after commands like \printfield without having to worry about spurious punctuation and whitespace. The buffer will only be inserted by the next \printfield or similar

command if *both* fields are non-empty. This also applies to `\printtext`, `\printlist`, `\printnames`, and `\bibstring`. See § 4.11.7 for further details.

`\finentry` Inserts `\finentrypunct`. This command should be used at the very end of every bibliography entry.

`\setunit{`⟨*punctuation*⟩`}`

`\setunit*{`⟨*punctuation*⟩`}`

The `\setunit` command is similar to `\newunit` except that it uses ⟨*punctuation*⟩ instead of `\newunitpunct`. The starred variant differs from the regular version in that it checks if the last `\printtext`, `\printfield`, `\printlist`, `\printnames`, or `\bibstring` command did actually print anything. If not, it does nothing.

`\printunit{`⟨*punctuation*⟩`}`

`\printunit*{`⟨*punctuation*⟩`}`

The `\printunit` command is similar to `\setunit` except that ⟨*punctuation*⟩ persists in the buffer. This ensures that ⟨*punctuation*⟩ is inserted before the next non-empty field printed by the `\printtext`, `\printfield`, `\printlist`, `\printnames`, or `\bibstring` commands—regardless of any intermediate calls to `\newunit` or `\setunit`.

`\setpunctfont{`⟨*command*⟩`}`

This command, which is intended for use in field formatting directives, provides an alternative way of dealing with unit punctuation after a field printed in a different font (for example, a title printed in italics). The standard LaTeX way of dealing with this is adding a small amount of space, the so-called italic correction. This command allows adapting the punctuation to the font of the preceding field. The ⟨*command*⟩ should be a text font command which takes one argument, such as `\emph` or `\textbf`. This command will only affect punctuation marks inserted by one of the commands from § 4.7.3. The font adaption is applied to the next punctuation mark only and will be reset automatically thereafter. If you want to reset it manually before it takes effect, issue `\resetpunctfont`. If the `punctfont` package option is disabled, this command does nothing. Note that the `\mkbibemph`, `\mkbibitalic` and `\mkbibbold` wrappers from § 4.10.4 incorporate this feature by default.

`\resetpunctfont` This command resets the unit punctuation font defined with `\setpunctfont` before it takes effect. If the `punctfont` package option is disabled, this command does nothing.

### 4.7.2 标点判断 Punctuation Tests

The following commands may be used to test for preceding punctuation marks at any point in citations and the bibliography.

`\ifpunct{`⟨*true*⟩`}{`⟨*false*⟩`}`

Executes ⟨*true*⟩ if preceded by any punctuation mark except for an abbreviation dot, and ⟨*false*⟩ otherwise.

\ifterm{⟨*true*⟩}{⟨*false*⟩}

Executes ⟨*true*⟩ if preceded by a terminal punctuation mark, and ⟨*false*⟩ otherwise. A terminal punctuation mark is any punctuation mark which has been registered for automatic capitalization, either with \DeclareCapitalPunctuation or by default, see § 4.7.5 for details. By default, this applies to periods, exclamation marks, and question marks.

\ifpunctmark{⟨*character*⟩}{⟨*true*⟩}{⟨*false*⟩}

Executes ⟨*true*⟩ if preceded by the punctuation mark ⟨*character*⟩, and ⟨*false*⟩ otherwise. The ⟨*character*⟩ may be a comma, a semicolon, a colon, a period, an exclamation mark, a question mark, or an asterisk. Note that a period denotes an end-of-sentence period. Use the asterisk to test for the dot after an abbreviation. If this command is used in a formatting directive for name lists, i. e., in the argument to \DeclareNameFormat, the ⟨*character*⟩ may also be an apostrophe.

\ifprefchar{⟨*true*⟩}{⟨*false*⟩}

Executes ⟨*true*⟩ if preceded by any prefix character declared by \DeclarePrefChars.

### 4.7.3 添加标点 Adding Punctuation

下面的命令设计用来重复标点。参考文献和标注样式总需要使用这些命令来代替原样输出标点符号。本节中所有的\add...命令自动利用\unspace命令移除前面的空白 (见 § 4.7.4)。注意: 下面讨论的所有的\add...命令的作用是宏包默认的，无论 Biblatex 换哪种语言都会重新恢复。其作用可以通过\DeclarePunctuationPairs命令进行调整，见 § 4.7.5。节

The following commands are designed to prevent double punctuation marks. Bibliography and citation styles should always use these commands instead of literal punctuation marks. All \add... commands in this section automatically remove preceding whitespace with \unspace (see § 4.7.4). Note that the behavior of all \add... commands discussed below is the package default, which is restored whenever Biblatex switches languages. This behavior may be adjusted with \DeclarePunctuationPairs from § 4.7.5.

\adddot 如果前面输出的不是任何一种标点符号，那么添加一个句点 (period)。该命令的目的是在一个缩写后面插入点 (dot)。以这种方式插入的点被认为与其它标点命令插入的标点性质相同。该命令也用来将前面如实输出的句点 (原样输出的句点，literal period) 转换成一个缩写的点。Adds a period unless it is preceded by any punctuation mark. The purpose of this command is inserting the dot after an abbreviation. Any dot inserted this way is recognized as such by the other punctuation commands. This command may also be used to turn a previously inserted literal period into an abbreviation dot.

\addcomma 如果前面输出不是一个逗号 (comma)、分号 (semicolon)、冒号 (colon) 和句点 (period)，那么添加一个逗号。Adds a comma unless it is preceded by another comma, a semicolon, a colon, or a period.

`\addsemicolon` Adds a semicolon unless it is preceded by a comma, another semicolon, a colon, or a period.

`\addcolon` Adds a colon unless it is preceded by a comma, a semicolon, another colon, or a period.

`\addperiod` 如果前面输出不是一个缩写点或其他任何标点符号，那么添加一个句号。该命令也可以用来将前面插入的缩写点转换为句号，比如在句子的末尾[61]。Adds a period unless it is preceded by an abbreviation dot or any other punctuation mark. This command may also be used to turn a previously inserted abbreviation dot into a period, for example at the end of a sentence.

`\addexclam` Adds an exclamation mark unless it is preceded by any punctuation mark except for an abbreviation dot.

`\addquestion` Adds a question mark unless it is preceded by any punctuation mark except for an abbreviation dot.

`\isdot` 当前面输出的是句号的时候，将其转换为缩写的点，如果前面是其它符号那么不添加任何符号。Turns a previously inserted literal period into an abbreviation dot. In contrast to `\adddot`, nothing is inserted if this command is not preceded by a period.

`\nopunct` Adds an internal marker which will cause the next punctuation command to print nothing.

### 4.7.4 添加空格 Adding Whitespace

The following commands are designed to prevent spurious whitespace. Bibliography and citation styles should always use these commands instead of literal whitespace. In contrast to the commands in §§ 4.7.2 和 4.7.3, they are not restricted to citations and the bibliography but available globally.

`\unspace` Removes preceding whitespace, i. e., removes all skips and penalties from the end of the current horizontal list. This command is implicitly executed by all of the following commands.

`\addspace` Adds a breakable interword space.

`\addnbspace` Adds a non-breakable interword space.

`\addthinspace` Adds a *breakable* thin space.

`\addnbthinspace` Adds a non-breakable thin space. This is similar to `\,` and `\thinspace`.

`\addlowpenspace` Adds a space penalized by the value of the `lownamepenalty` counter, see §§ 3.10.3 和 4.10.3 for details.

`\addhighpenspace` Adds a space penalized by the value of the `highnamepenalty` counter, see §§ 3.10.3 和 4.10.3 for details.

[61] 不是很理解，前面如果是缩写的点，那么不加入句号，那么缩写的点就转变为句号了？

`\addlpthinspace` Similar to `\addlowpenspace` but adds a breakable thin space.

`\addhpthinspace` Similar to `\addhighpenspace` but adds a breakable thin space.

`\addabbrvspace` Adds a space penalized by the value of the `abbrvpenalty` counter, see §§ 3.10.3 和 4.10.3 for details.

`\addabthinspace` Similar to `\addabbrvspace` but using a thin space.

`\adddotspace` Executes `\adddot` and adds a space penalized by the value of the `abbrvpenalty` counter, see §§ 3.10.3 和 4.10.3 for details.

`\addslash` Adds a breakable slash. This command differs from the `\slash` command in the LaTeX kernel in that a linebreak after the slash is not penalized at all.

Note that the commands in this section implicitly execute `\unspace` to remove spurious whitespace, hence they may be used to override each other. For example, you may use `\addnbspace` to transform a previously inserted interword space into a non-breakable one and `\addspace` to turn a non-breakable space into a breakable one. [62]

### 4.7.5 标点设置和大写 Configuring Punctuation and Capitalization

The following commands configure various features related to punctuation and automatic capitalization. [63]

`\DeclarePrefChars{`⟨*characters*⟩`}`

This command declares characters that are to be treated specially when testing to see if `\bibnamedelimc` is to be inserted between a name prefix and a family name. If a character is in the list of ⟨*characters*⟩, `\bibnamedelimc` is not inserted. It is used to allow abbreviated name prefices like 'd'Argent' where no space should be inserted after the apostrophe. The default setting is:

```
\DeclarePrefChars{'}
```

`\DeclareAutoPunctuation{`⟨*characters*⟩`}`

This command defines the punctuation marks to be considered by the citation commands as they scan ahead for punctuation. Note that ⟨*characters*⟩ is an undelimited list of characters. Valid ⟨*characters*⟩ are period, comma, semicolon, colon, exclamation and question mark. The default setting is:

```
\DeclareAutoPunctuation{.,;:!?}
```

[62] 注意有的时候`\unspace`看似能够起到作用，但其实并不能随意使用的。在 beamer 中 printtext 老是有些问题，可能是实现 printtext 命令的依赖命令，在 beamer 中重定义了，跟 aritcle 文档类中的情况差别很大。

[63] 这里的 capitalization 是大写的意思么？

This definition is restored automatically whenever the `autopunct` package option is set to `true`. Executing `\DeclareAutoPunctuation{}` is equivalent to setting `autopunct=false`, i. e., it disables this feature.

`\DeclareCapitalPunctuation{`⟨*characters*⟩`}`

When Biblatex inserts localisation strings, i. e., key terms such as 'edition' or 'volume', it automatically capitalizes them after terminal punctuation marks. This command defines the punctuation marks which will cause localisation strings to be capitalized if one of them precedes a string. Note that ⟨*characters*⟩ is an undelimited list of characters. Valid ⟨*characters*⟩ are period, comma, semicolon, colon, exclamation and question mark. The package default is:

```
\DeclareCapitalPunctuation{.!?}
```

Using `\DeclareCapitalPunctuation` with an empty argument is equivalent to disabling automatic capitalization. Since this feature is language specific, this command must be used in the argument to `\DefineBibliographyExtras` (when used in the preamble) or `\DeclareBibliographyExtras` (when used in a localisation module). See §§ 3.9 和 4.9 for details. By default, strings are capitalized after periods, exclamation marks, and question marks. All strings are generally capitalized at the beginning of a paragraph (in fact whenever TeX is in vertical mode).

`\DeclarePunctuationPairs{`⟨*identifier*⟩`}{`⟨*characters*⟩`}`

Use this command to declare valid pairs of punctuation marks. This will affect the punctuation commands discussed in § 4.7.3. For example, the description of `\addcomma` states that this command adds a comma unless it is preceded by another comma, a semicolon, a colon, or a period. In other words, commas after abbreviation dots, exclamation marks, and question marks are permitted. These valid pairs are declared as follows:

```
\DeclarePunctuationPairs{comma}{*!?}
```

The ⟨*identifier*⟩ selects the command to be configured. The identifiers correspond to the names of the punctuation commands from § 4.7.3 without the `\add` prefix, i. e., valid ⟨*identifier*⟩ strings are `dot`, `comma`, `semicolon`, `colon`, `period`, `exclam`, `question`. The ⟨*characters*⟩ argument is an undelimited list of punctuation marks. Valid ⟨*characters*⟩ are comma, semicolon, colon, period, exclamation mark, question mark, and asterisk. A period in the ⟨*characters*⟩ argument denotes an end-of-sentence period, an asterisk the dot after an abbreviation. This is the default setup, which is automatically restored whenever Biblatex switches languages and corresponds to the behavior described in § 4.7.3:

```
\DeclarePunctuationPairs{dot}{}
```

```
\DeclarePunctuationPairs{comma}{*!?}
\DeclarePunctuationPairs{semicolon}{*!?}
\DeclarePunctuationPairs{colon}{*!?}
\DeclarePunctuationPairs{period}{}
\DeclarePunctuationPairs{exclam}{*}
\DeclarePunctuationPairs{question}{*}
```

Since this feature is language specific, `\DeclarePunctuationPairs` must be used in the argument to `\DefineBibliographyExtras` (when used in the preamble) or `\DeclareBibliographyExtras` (when used in a localisation module). See §§ 3.9 和 4.9 for details. Note that some localisation modules may use a setup which is different from the package default.[64]

`\DeclareQuotePunctuation{`⟨*characters*⟩`}`

This command controls 'American-style' punctuation. The `\mkbibquote` wrapper from § 4.10.4 can interact with the punctuation facilities discussed in §§ 4.7.1、4.7.3、4.7.4. Punctuation marks after `\mkbibquote` will be moved inside the quotes if they have been registered with `\DeclareQuotePunctuation`. Note that ⟨*characters*⟩ is an undelimited list of characters. Valid ⟨*characters*⟩ are period, comma, semicolon, colon, exclamation and question mark. Here is an example:

```
\DeclareQuotePunctuation{.,}
```

Executing `\DeclareQuotePunctuation{}` is equivalent to disabling this feature. This is the package default. Since this feature is language specific, this command must be used in the argument to `\DefineBibliographyExtras` (when used in the preamble) or `\DeclareBibliographyExtras` (when used in a localisation module). See §§ 3.9 和 4.9 for details. See also § 3.11.1.

`\uspunctuation` A shorthand using the lower-level commands `\DeclareQuotePunctuation` and `\DeclarePunctuationPairs` to activate 'American-style' punctuation. See § 3.11.1 for details. This shorthand is provided for convenience only. The effective settings are applied by the lower-level commands.

`\stdpunctuation` Undoes the settings applied by `\uspunctuation`, restoring standard punctuation. As standard punctuation is the default setting, you only need this command to override a previously executed `\uspunctuation` command. See § 3.11.1 for details.

### 4.7.6 Correcting Punctuation Tracking

The facilities for punctuation tracking and automatic capitalization are very reliable under normal circumstances, but there are always marginal cases which may

[64] As of this writing, the `american` module uses different settings for 'American-style' punctuation.

require manual intervention. Typical cases are localisation strings printed as the first word in a footnote (which is usually treated as the beginning of a paragraph as far as capitalization is concerned, but TeX is not in vertical mode at this point) or punctuation after periods which are not really end-of-sentence periods (for example, after an ellipsis like "[…]" a command such as `\addperiod` would do nothing since parentheses and brackets are transparent to the punctuation tracker). In such cases, use the following commands in bibliography and citation styles to mark the beginning or middle of a sentence if and where required:

`\bibsentence` This command marks the beginning of a sentence. A localisation string immediately after this command will be capitalized and the punctuation tracker is reset, i. e., this command hides all preceding punctuation marks from the punctuation tracker and enforces capitalization.

`\midsentence` This command marks the middle of a sentence. A localisation string immediately after this command will not be capitalized and the punctuation tracker is reset, i. e., this command hides all preceding punctuation marks from the punctuation tracker and suppresses capitalization.

`\midsentence*` The starred variant of `\midsentence` differs from the regular one in that a preceding abbreviation dot is not hidden from the the punctuation tracker, i. e., any code after `\midsentence*` will see a preceding abbreviation dot. All other punctuation marks are hidden from the punctuation tracker and capitalization is suppressed.

## 4.8 本地化字符串 Localization Strings

Localization strings are key terms such as 'edition' or 'volume' which are automatically translated by Biblatex's localisation modules. See § 4.9 for an overview and § 4.9.2 for a list of all strings supported by default. The commands in this section are used to print the localised term.

`\bibstring[`⟨*wrapper*⟩`]{`⟨*key*⟩`}`

Prints the localisation string ⟨*key*⟩, where ⟨*key*⟩ is an identifier in lowercase letters (see § 4.9.2). The string will be capitalized as required, see § 4.7.5 for details. Depending on the `abbreviate` package option from § 3.1.2.1, `\bibstring` prints the short or the long version of the string. If localisation strings are nested, i. e., if `\bibstring` is used in another string, it will behave like `\bibxstring`. If the ⟨*wrapper*⟩ argument is given, the string is passed to the ⟨*wrapper*⟩ for formatting. This is intended for font commands such as `\emph`.

`\biblstring[`⟨*wrapper*⟩`]{`⟨*key*⟩`}`

Similar to `\bibstring` but always prints the long string, ignoring the `abbreviate` option.

\bibsstring[⟨*wrapper*⟩]{⟨*key*⟩}

Similar to \bibstring but always prints the short string, ignoring the abbreviate option.

\bibcpstring[⟨*wrapper*⟩]{⟨*key*⟩}

Similar to \bibstring but the term is always capitalized.

\bibcplstring[⟨*wrapper*⟩]{⟨*key*⟩}

Similar to \biblstring but the term is always capitalized.

\bibcpsstring[⟨*wrapper*⟩]{⟨*key*⟩}

Similar to \bibsstring but the term is always capitalized.

\bibucstring[⟨*wrapper*⟩]{⟨*key*⟩}

Similar to \bibstring but the whole term is uppercased.

\bibuclstring[⟨*wrapper*⟩]{⟨*key*⟩}

Similar to \biblstring but the whole term is uppercased.

\bibucsstring[⟨*wrapper*⟩]{⟨*key*⟩}

Similar to \bibsstring but the whole term is uppercased.

\biblcstring[⟨*wrapper*⟩]{⟨*key*⟩}

Similar to \bibstring but the whole term is lowercased.

\biblclstring[⟨*wrapper*⟩]{⟨*key*⟩}

Similar to \biblstring but the whole term is lowercased.

\biblcsstring[⟨*wrapper*⟩]{⟨*key*⟩}

Similar to \bibsstring but the whole term is lowercased.

\bibxstring{⟨*key*⟩}

A simplified but expandable version of \bibstring. Note that this variant does not capitalize automatically, nor does it hook into the punctuation tracker. It is intended for special cases in which strings are nested or an expanded localisation string is required in a test.

\bibxlstring[⟨*wrapper*⟩]{⟨*key*⟩}

Similar to \bibxstring but always uses the long string, ignoring the abbreviate option.

`\bibxsstring[`⟨*wrapper*⟩`]{`⟨*key*⟩`}`

Similar to `\bibxstring` but always uses the short string, ignoring the `abbreviate` option.

`\mainlang`

Switches from the current language to the main document language. This can be used the ⟨*wrapper*⟩ argument in the localisation string commands above.

## 4.9 本地化模块 Localization Modules

A localisation module provides translations for key terms such as 'edition' or 'volume' as well as definitions for language specific features such as the date format and ordinals. These definitions are provided in files with the suffix `lbx`. The base name of the file must be a language name known to the `babel`/`polyglossia` packages. The `lbx` files may also be used to map `babel`/`polyglossia` language names to the backend modules of the Biblatex package. All localisation modules are loaded on demand in the document body. Note that the contents of the file are processed in a group and that the category code of the character `@` is temporarily set to 'letter'.

### 4.9.1 本地化命令 Localization Commands

The user-level versions of the localisation commands were already introduced in § 3.9. When used in `lbx` files, however, the syntax of localisation commands is different from the user syntax in the preamble and the configuration file. When used in localisation files, there is no need to specify the ⟨*language*⟩ because the mapping of strings to a language is already provided by the name of the `lbx` file.

`\DeclareBibliographyStrings{`⟨*definitions*⟩`}`

This command is only available in `lbx` files. It is used to define localisation strings. The ⟨*definitions*⟩ consist of ⟨*key*⟩=⟨*value*⟩ pairs which assign an expression to an identifier. A complete list of all keys supported by default is given is § 4.9.2. Note that the syntax of the value is different in `lbx` files. The value assigned to a key consists of two expressions, each of which is wrapped in an additional pair of brackets. This is best shown by example:

```
\DeclareBibliographyStrings{%
  bibliography  = {{Bibliography}{Bibliography}},
  shorthands    = {{List of Abbreviations}{Abbreviations}},
  editor        = {{editor}{ed.}},
  editors       = {{editors}{eds.}},
}
```

The first value is the long, written out expression, the second one is an abbreviated or short form. Both strings must always be given even though they may be identical if an expression is always (or never) abbreviated. Depending on the setting of the `abbreviate` package option (see § 3.1.2.1), Biblatex selects one expression when loading the `lbx` file. There is also a special key named `inherit` which copies the strings from a different language. This is intended for languages which only differ in a few expressions, such as German and Austrian or American and British English. For example, here are the complete definitions for Austrian:

```
\DeclareBibliographyStrings{%
  inherit       = {german},
  january       = {{J\"anner}{J\"an.}},
}
```

The above examples are slightly simplified. Real localisation files should use the punctuation and formatting commands discussed in §§ 4.7.3 和 3.10 instead of literal punctuation. Here is an excerpt from a real localisation file:

```
  bibliography     = {{Bibliography}{Bibliography}},
  shorthands       = {{List of Abbreviations}{Abbreviations}},
  editor           = {{editor}{ed\adddot}},
  editors          = {{editors}{eds\adddot}},
  byeditor         = {{edited by}{ed\adddotspace by}},
  mathesis         = {{Master's thesis}{MA\addabbrvspace thesis}},
```

Note the handling of abbreviation dots, the spacing in abbreviated expressions, and the capitalization in the example above. All expressions should be capitalized as they usually are when used in the middle of a sentence. The Biblatex package will automatically capitalize the first word when required at the beginning of a sentence, see `\DeclareCapitalPunctuation` in § 4.7.5 for details. Expressions intended for use in headings are special. They should be capitalized in a way that is suitable for titling and should not be abbreviated (but they may have a short form).

`\InheritBibliographyStrings{`⟨*language*⟩`}`

This command is only available in `lbx` files. It copies the localisation strings for ⟨*language*⟩ to the current language, as specified by the name of the `lbx` file.

`\DeclareBibliographyExtras{`⟨*code*⟩`}`

This command is only available in `lbx` files. It is used to adapt language specific features such as the date format and ordinals. The ⟨*code*⟩, which may be arbitrary LaTeX code, will usually consist of redefinitions of the formatting commands from § 4.10.2.

`\UndeclareBibliographyExtras{`⟨*code*⟩`}`

This command is only available in `lbx` files. It is used to restore any formatting commands modified with `\DeclareBibliographyExtras`. If a redefined command is included in § 4.10.2, there is no need to restore its previous definition since these commands are localised by all language modules anyway.

`\InheritBibliographyExtras{`⟨*language*⟩`}`

This command is only available in `lbx` files. It copies the bibliography extras for ⟨*language*⟩ to the current language, as specified by the name of the `lbx` file.

`\DeclareHyphenationExceptions{`⟨*text*⟩`}`

This command corresponds to `\DefineHyphenationExceptions` from § 3.9. The difference is that it is only available in `lbx` files and that the ⟨*language*⟩ argument is omitted. The hyphenation exceptions will affect the language of the `lbx` file currently being processed.

`\DeclareRedundantLanguages{`⟨*language, language, ...*⟩`}{`⟨*langid, langid, ...*⟩`}`

This command provides the language mappings required by the `clearlang` option from § 3.1.2.1. The ⟨*language*⟩ is the string given in the `language` field (without the optional `lang` prefix); ⟨*langid*⟩ is `babel`/`polyglossia`'s language identifier, as given in the optional argument of `\usepackage` when loading `babel` or the argument of `\setdefaultlanguage` or `\setotherlanguages` when using `polyglossia`. This command may be used in `lbx` files or in the document preamble. Here are some examples:

```
\DeclareRedundantLanguages{french}{french}
\DeclareRedundantLanguages{german}{german,ngerman,austrian,naustrian,
        nswissgerman,swissgerman}
\DeclareRedundantLanguages{english,american}{english,american,british,
        canadian,australian,newzealand,USenglish,UKenglish}
```

Note that this feature needs to be enabled globally with the `clearlang` option from § 3.1.2.1. If it is disabled, all mappings will be ignored. If the ⟨*langid*⟩ parameter is blank, Biblatex will clear the mappings for the corresponding ⟨*language*⟩, i. e., the feature will be disabled for this ⟨*language*⟩ only.

`\DeclareLanguageMapping{`⟨*language*⟩`}{`⟨*file*⟩`}`

This command maps a `babel`/`polyglossia` language identifier to an `lbx` file. The ⟨*language*⟩ must be a language name known to the `babel`/`polyglossia` package, i. e., one of the identifiers listed in 表 3. The ⟨*file*⟩ argument is the name of an alternative `lbx` file without the `.lbx` suffix. Declaring the same mapping more than once is possible. Subsequent declarations will simply overwrite any previous ones. This command may only be used in the preamble. See § 4.11.8 for further details.

`\NewBibliographyString{`⟨*key*⟩`}`

This command, which may be used in the preamble (including `cbx` and `bbx` files) as well as in `lbx` files, declares new localisation strings, i. e., it initializes a new ⟨*key*⟩ to be used in the ⟨*definitions*⟩ of `\DefineBibliographyStrings` or `\DeclareBibliographyStrings`. The ⟨*key*⟩ argument may also be a comma-separated list of key names. When used in an `lbx`, the ⟨*key*⟩ is initialized only for the language specified by the name of the `lbx` file. The keys listed in § 4.9.2 are defined by default.

### 4.9.2 Localization Keys

The localisation keys in this section are defined by default and covered by the localisation files which come with Biblatex. Note that these strings are only available in citations, the bibliography and bibliography lists. All expressions should be capitalized as they usually are when used in the middle of a sentence. Biblatex will capitalize them automatically at the beginning of a sentence. The only exceptions to these rules are the three strings intended for use in headings.

**4.9.2.1 Headings** The following strings are special because they are intended for use in headings and made available globally via macros. For this reason, they should be capitalized for use in headings and they must not include any local commands which are part of Biblatex's author interface.

bibliography The term 'bibliography', also available as `\bibname`.

references The term 'references', also available as `\refname`.

shorthands The term 'list of shorthands' or 'list of abbreviations', also available as `\biblistname`.

**4.9.2.2 Roles, Expressed as Functions** The following keys refer to roles which are expressed as a function ('editor', 'translator') rather than as an action ('edited by', 'translated by').

editor The term 'editor', referring to the main editor. This is the most generic editorial role.

editors The plural form of `editor`.

compiler The term 'compiler', referring to an editor whose task is to compile a work.

compilers The plural form of `compiler`.

founder The term 'founder', referring to a founding editor.

founders The plural form of `founder`.

continuator An expression like 'continuator', 'continuation', or 'continued', referring to a past editor who continued the work of the founding editor but was subsequently replaced by the current editor.

continuators The plural form of `continuator`.

redactor The term 'redactor', referring to a secondary editor.

redactors The plural form of `redactor`.

reviser The term ‘reviser’, referring to a secondary editor.

revisers The plural form of `reviser`.

collaborator A term like ‘collaborator’, ‘collaboration’, ‘cooperator’, or ‘cooperation’, referring to a secondary editor.

collaborators The plural form of `collaborator`.

translator The term ‘translator’.

translators The plural form of `translator`.

commentator The term ‘commentator’, referring to the author of a commentary to a work.

commentators The plural form of `commentators`.

annotator The term ‘annotator’, referring to the author of annotations to a work.

annotators The plural form of `annotators`.

#### 4.9.2.3 Concatenated Editor Roles, Expressed as Functions

The following keys are similar in function to `editor`, `translator`, etc. They are used to indicate additional roles of the editor, e. g., ‘editor and translator’, ‘editor and foreword’.

editortr Used if `editor`/`translator` are identical.

editorstr The plural form of `editortr`.

editorco Used if `editor`/`commentator` are identical.

editorsco The plural form of `editorco`.

editoran Used if `editor`/`annotator` are identical.

editorsan The plural form of `editoran`.

editorin Used if `editor`/`introduction` are identical.

editorsin The plural form of `editorin`.

editorfo Used if `editor`/`foreword` are identical.

editorsfo The plural form of `editorfo`.

editoraf Used if `editor`/`aftword` are identical.

editorsaf The plural form of `editoraf`.

Keys for `editor`/`translator`/⟨*role*⟩ combinations:

editortrco Used if `editor`/`translator`/`commentator` are identical.

editorstrco The plural form of `editortrco`.

editortran Used if `editor`/`translator`/`annotator` are identical.

editorstran The plural form of `editortran`.

editortrin Used if `editor`/`translator`/`introduction` are identical.

editorstrin The plural form of `editortrin`.

editortrfo Used if `editor/translator/foreword` are identical.

editorstrfo The plural form of `editortrfo`.

editortraf Used if `editor/translator/aftword` are identical.

editorstraf The plural form of `editortraf`.

Keys for `editor/commentator/`⟨*role*⟩ combinations:

editorcoin Used if `editor/commentator/introduction` are identical.

editorscoin The plural form of `editorcoin`.

editorcofo Used if `editor/commentator/foreword` are identical.

editorscofo The plural form of `editorcofo`.

editorcoaf Used if `editor/commentator/aftword` are identical.

editorscoaf The plural form of `editorcoaf`.

Keys for `editor/annotator/`⟨*role*⟩ combinations:

editoranin Used if `editor/annotator/introduction` are identical.

editorsanin The plural form of `editoranin`.

editoranfo Used if `editor/annotator/foreword` are identical.

editorsanfo The plural form of `editoranfo`.

editoranaf Used if `editor/annotator/aftword` are identical.

editorsanaf The plural form of `editoranaf`.

Keys for `editor/translator/commentator/`⟨*role*⟩ combinations:

editortrcoin Used if `editor/translator/commentator/introduction` are identical.

editorstrcoin The plural form of `editortrcoin`.

editortrcofo Used if `editor/translator/commentator/foreword` are identical.

editorstrcofo The plural form of `editortrcofo`.

editortrcoaf Used if `editor/translator/commentator/aftword` are identical.

editorstrcoaf The plural form of `editortrcoaf`.

Keys for `editor/annotator/commentator/`⟨*role*⟩ combinations:

editortranin Used if `editor/annotator/commentator/introduction` are identical.

editorstranin The plural form of `editortranin`.

editortranfo Used if `editor/annotator/commentator/foreword` are identical.

editorstranfo The plural form of `editortranfo`.

editortranaf Used if `editor/annotator/commentator/aftword` are identical.

editorstranaf The plural form of `editortranaf`.

#### 4.9.2.4 Concatenated Translator Roles, Expressed as Functions

The following keys are similar in function to `translator`. They are used to indicate additional roles of the translator, e. g., 'translator and commentator', 'translator and introduction'.

translatorco — Used if `translator/commentator` are identical.

translatorsco — The plural form of `translatorco`.

translatoran — Used if `translator/annotator` are identical.

translatorsan — The plural form of `translatoran`.

translatorin — Used if `translator/introduction` are identical.

translatorsin — The plural form of `translatorin`.

translatorfo — Used if `translator/foreword` are identical.

translatorsfo — The plural form of `translatorfo`.

translatoraf — Used if `translator/aftword` are identical.

translatorsaf — The plural form of `translatoraf`.

Keys for `translator/commentator/`⟨*role*⟩ combinations:

translatorcoin — Used if `translator/commentator/introduction` are identical.

translatorscoin — The plural form of `translatorcoin`.

translatorcofo — Used if `translator/commentator/foreword` are identical.

translatorscofo — The plural form of `translatorcofo`.

translatorcoaf — Used if `translator/commentator/aftword` are identical.

translatorscoaf — The plural form of `translatorcoaf`.

Keys for `translator/annotator/`⟨*role*⟩ combinations:

translatoranin — Used if `translator/annotator/introduction` are identical.

translatorsanin — The plural form of `translatoranin`.

translatoranfo — Used if `translator/annotator/foreword` are identical.

translatorsanfo — The plural form of `translatoranfo`.

translatoranaf — Used if `translator/annotator/aftword` are identical.

translatorsanaf — The plural form of `translatoranaf`.

#### 4.9.2.5 Roles, Expressed as Actions

The following keys refer to roles which are expressed as an action ('edited by', 'translated by') rather than as a function ('editor', 'translator').

byauthor — The expression '[created] by ⟨*name*⟩'.

byeditor — The expression 'edited by ⟨*name*⟩'.

bycompiler The expression ‘compiled by ⟨*name*⟩’.

byfounder The expression ‘founded by ⟨*name*⟩’.

bycontinuator The expression ‘continued by ⟨*name*⟩’.

byredactor The expression ‘redacted by ⟨*name*⟩’.

byreviser The expression ‘revised by ⟨*name*⟩’.

byreviewer The expression ‘reviewed by ⟨*name*⟩’.

bycollaborator An expression like ‘in collaboration with ⟨*name*⟩’ or ‘in cooperation with ⟨*name*⟩’.

bytranslator The expression ‘translated by ⟨*name*⟩’ or ‘translated from ⟨*language*⟩ by ⟨*name*⟩’.

bycommentator The expression ‘commented by ⟨*name*⟩’.

byannotator The expression ‘annotated by ⟨*name*⟩’.

#### 4.9.2.6 Concatenated Editor Roles, Expressed as Actions

The following keys are similar in function to `byeditor`, `bytranslator`, etc. They are used to indicate additional roles of the editor, e. g., ‘edited and translated by’, ‘edited and furnished with an introduction by’, ‘edited, with a foreword, by’.

byeditortr Used if `editor/translator` are identical.

byeditorco Used if `editor/commentator` are identical.

byeditoran Used if `editor/annotator` are identical.

byeditorin Used if `editor/introduction` are identical.

byeditorfo Used if `editor/foreword` are identical.

byeditoraf Used if `editor/aftword` are identical.

Keys for `editor/translator/`⟨*role*⟩ combinations:

byeditortrco Used if `editor/translator/commentator` are identical.

byeditortran Used if `editor/translator/annotator` are identical.

byeditortrin Used if `editor/translator/introduction` are identical.

byeditortrfo Used if `editor/translator/foreword` are identical.

byeditortraf Used if `editor/translator/aftword` are identical.

Keys for `editor/commentator/`⟨*role*⟩ combinations:

byeditorcoin Used if `editor/commentator/introduction` are identical.

byeditorcofo Used if `editor/commentator/foreword` are identical.

byeditorcoaf Used if `editor/commentator/aftword` are identical.

Keys for `editor/annotator/`⟨*role*⟩ combinations:

byeditoranin Used if `editor/annotator/introduction` are identical.

byeditoranfo Used if `editor/annotator/foreword` are identical.

byeditoranaf Used if `editor/annotator/aftword` are identical.

Keys for `editor/translator/commentator/`⟨*role*⟩ combinations:

byeditortrcoin Used if `editor/translator/commentator/introduction` are identical.

byeditortrcofo Used if `editor/translator/commentator/foreword` are identical.

byeditortrcoaf Used if `editor/translator/commentator/aftword` are identical.

Keys for `editor/translator/annotator/`⟨*role*⟩ combinations:

byeditortranin Used if `editor/annotator/commentator/introduction` are identical.

byeditortranfo Used if `editor/annotator/commentator/foreword` are identical.

byeditortranaf Used if `editor/annotator/commentator/aftword` are identical.

#### 4.9.2.7 Concatenated Translator Roles, Expressed as Actions

The following keys are similar in function to `bytranslator`. They are used to indicate additional roles of the translator, e. g., 'translated and commented by', 'translated and furnished with an introduction by', 'translated, with a foreword, by'.

bytranslatorco Used if `translator/commentator` are identical.

bytranslatoran Used if `translator/annotator` are identical.

bytranslatorin Used if `translator/introduction` are identical.

bytranslatorfo Used if `translator/foreword` are identical.

bytranslatoraf Used if `translator/aftword` are identical.

Keys for `translator/commentator/`⟨*role*⟩ combinations:

bytranslatorcoin Used if `translator/commentator/introduction` are identical.

bytranslatorcofo Used if `translator/commentator/foreword` are identical.

bytranslatorcoaf Used if `translator/commentator/aftword` are identical.

Keys for `translator/annotator/`⟨*role*⟩ combinations:

bytranslatoranin Used if `translator/annotator/introduction` are identical.

bytranslatoranfo Used if `translator/annotator/foreword` are identical.

bytranslatoranaf Used if `translator/annotator/aftword` are identical.

#### 4.9.2.8 Roles, Expressed as Objects

Roles which are related to supplementary material may also be expressed as objects ('with a commentary by') rather than as functions ('commentator') or as actions ('commented by').

withcommentator The expression 'with a commentary by ⟨*name*⟩'.

withannotator The expression 'with annotations by ⟨*name*⟩'.

withintroduction The expression 'with an introduction by ⟨*name*⟩'.

withforeword The expression 'with a foreword by ⟨*name*⟩'.

withafterword The expression 'with an afterword by ⟨*name*⟩'.

#### 4.9.2.9 Supplementary Material

commentary The term 'commentary'.

annotations The term 'annotations'.

introduction The term 'introduction'.

foreword The term 'foreword'.

afterword The term 'afterword'.

#### 4.9.2.10 Publication Details

volume The term 'volume', referring to a book.

volumes The plural form of `volume`.

involumes The term 'in', as used in expressions like 'in ⟨*number of volumes*⟩ volumes'.

jourvol The term 'volume', referring to a journal.

jourser The term 'series', referring to a journal.

book The term 'book', referring to a document division.

part The term 'part', referring to a part of a book or a periodical.

issue The term 'issue', referring to a periodical.

newseries The expression 'new series', referring to a journal.

oldseries The expression 'old series', referring to a journal.

edition The term 'edition'.

in The term 'in', referring to the title of a work published as part of another one, e. g., '⟨*title of article*⟩ in ⟨*title of journal*⟩'.

inseries The term 'in', as used in expressions like 'volume ⟨*number*⟩ in ⟨*name of series*⟩'.

ofseries The term 'of', as used in expressions like 'volume ⟨*number*⟩ of ⟨*name of series*⟩'.

number The term 'number', referring to an issue of a journal.

chapter The term 'chapter', referring to a chapter in a book.

version The term 'version', referring to a revision number.

reprint The term ‘reprint’.

reprintof The expression ‘reprint of ⟨*title*⟩’.

reprintas The expression ‘reprinted as ⟨*title*⟩’.

reprintfrom The expression ‘reprinted from ⟨*title*⟩’.

translationof The expression ‘translation of ⟨*title*⟩’.

translationas The expression ‘translated as ⟨*title*⟩’.

translationfrom The expression ‘translated from [the] ⟨*language*⟩’.

reviewof The expression ‘review of ⟨*title*⟩’.

origpubas The expression ‘originally published as ⟨*title*⟩’.

origpubin The expression ‘originally published in ⟨*year*⟩’.

astitle The term ‘as’, as used in expressions like ‘published by ⟨*publisher*⟩ as ⟨*title*⟩’.

bypublisher The term ‘by’, as used in expressions like ‘published by ⟨*publisher*⟩’.

#### 4.9.2.11 Publication State

inpreparation The expression ‘in preparation’ (the manuscript is being prepared for publication).

submitted The expression ‘submitted’ (the manuscript has been submitted to a journal or conference).

forthcoming The expression ‘forthcoming’ (the manuscript has been accepted by a press or journal).

inpress The expression ‘in press’ (the manuscript is fully copyedited and out of the author’s hands; it is in the final stages of the production process).

prepublished The expression ‘pre-published’ (the manuscript is published in a preliminary form or location, such as online version in advance of print publication).

#### 4.9.2.12 Pagination

page The term ‘page’.

pages The plural form of `page`.

column The term ‘column’, referring to a column on a page.

columns The plural form of `column`.

section The term ‘section’, referring to a document division (usually abbreviated as §).

sections The plural form of `section` (usually abbreviated as §§).

paragraph The term ‘paragraph’ (i. e., a block of text, not to be confused with `section`).

paragraphs The plural form of `paragraph`.

verse The term ‘verse’ as used when referring to a work which is cited by verse numbers.

verses The plural form of `verse`.

line The term ‘line’ as used when referring to a work which is cited by line numbers.

lines The plural form of `line`.

#### 4.9.2.13 Types

The following keys are typically used in the `type` field of `@thesis`, `@report`, `@misc`, and other entries:

mathesis — An expression equivalent to the term 'Master's thesis'.

phdthesis — The term 'PhD thesis', 'PhD dissertation', 'doctoral thesis', etc.

candthesis — An expression equivalent to the term 'Candidate thesis'. Used for 'Candidate' degrees that have no clear equivalent to the Master's or doctoral level.

techreport — The term 'technical report'.

resreport — The term 'research report'.

software — The term 'computer software'.

datacd — The term 'data CD' or 'CD-ROM'.

audiocd — The term 'audio CD'.

#### 4.9.2.14 Miscellaneous

nodate — The term to use in place of a date when there is no date for an entry e. g., 'n.d.'

and — The term 'and', as used in a list of authors or editors, for example.

andothers — The expression 'and others' or 'et alii', used to mark the truncation of a name list.

andmore — Like `andothers` but used to mark the truncation of a literal list.

#### 4.9.2.15 Labels

The following strings are intended for use as labels, e. g., 'Address: ⟨*url*⟩' or 'Abstract: ⟨*abstract*⟩'.

url — The term 'address' in the sense of an internet address.

urlfrom — An expression like 'available from ⟨*url*⟩' or 'available at ⟨*url*⟩'.

urlseen — An expression like 'accessed on ⟨*date*⟩', 'retrieved on ⟨*date*⟩', 'visited on ⟨*date*⟩', referring to the access date of an online resource.

file — The term 'file'.

library — The term 'library'.

abstract — The term 'abstract'.

annotation — The term 'annotations'.

#### 4.9.2.16 Citations

Traditional scholarly expressions used in citations:

idem — The term equivalent to the Latin 'idem' ('the same [person]').

idemsf — The feminine singular form of `idem`.

idemsm — The masculine singular form of `idem`.

idemsn — The neuter singular form of `idem`.

idempf — The feminine plural form of `idem`.

idempm The masculine plural form of `idem`.

idempn The neuter plural form of `idem`.

idempp The plural form of `idem` suitable for a mixed gender list of names.

ibidem The term equivalent to the Latin 'ibidem' ('in the same place').

opcit The term equivalent to the Latin term 'opere citato' ('[in] the work [already] cited').

loccit The term equivalent to the Latin term 'loco citato' ('[at] the place [already] cited').

confer The term equivalent to the Latin 'confer' ('compare').

sequens The term equivalent to the Latin 'sequens' ('[and] the following [page]'), as used to indicate a range of two pages when only the starting page is provided (e. g., '25 sq.' or '25 f.' instead of '25–26').

sequentes The term equivalent to the Latin 'sequentes' ('[and] the following [pages]'), as used to indicate an open-ended range of pages when only the starting page is provided (e. g., '25 sqq.' or '25 ff.').

passim The term equivalent to the Latin 'passim' ('throughout', 'here and there', 'scatteredly').

Other expressions frequently used in citations:

see The term 'see'.

seealso The expression 'see also'.

seenote An expression like 'see note ⟨*footnote*⟩' or 'as in ⟨*footnote*⟩', used to refer to a previous footnote in a citation.

backrefpage An expression like 'see page ⟨*page*⟩' or 'cited on page ⟨*page*⟩', used to introduce back references in the bibliography.

backrefpages The plural form of `backrefpage`, e. g., 'see pages ⟨*pages*⟩' or 'cited on pages ⟨*pages*⟩'.

quotedin An expression like 'quoted in ⟨*citation*⟩', used when quoting a passage which was already a quotation in the cited work.

citedas An expression like 'henceforth cited as ⟨*shorthand*⟩', used to introduce a shorthand in a citation.

thiscite The expression used in some verbose citation styles to differentiate between the page range of the cited item (typically an article in a journal, collection, or conference proceedings) and the page number the citation refers to. For example: "Author, Title, in: Book, pp. 45–61, `thiscite` p. 52."

#### 4.9.2.17 Month Names

january The name 'January'.

february The name 'February'.

march The name 'March'.

april The name 'April'.

may The name ‘May’.

june The name ‘June’.

july The name ‘July’.

august The name ‘August’.

september The name ‘September’.

october The name ‘October’.

november The name ‘November’.

december The name ‘December’.

#### 4.9.2.18 Language Names

langamerican The language ‘American’ or ‘American English’.

langbrazilian The language ‘Brazilian’ or ‘Brazilian Portuguese’.

langcatalan The language ‘Catalan’.

langcroatian The language ‘Croatian’.

langczech The language ‘Czech’.

langdanish The language ‘Danish’.

langdutch The language ‘Dutch’.

langenglish The language ‘English’.

langestonian The language ‘Estonian’.

langfinnish The language ‘Finnish’.

langfrench The language ‘French’.

langgerman The language ‘German’.

langgreek The language ‘Greek’.

langitalian The language ‘Italian’.

langjapanese The language ‘Japanese’.

langlatin The language ‘Latin’.

langnorwegian The language ‘Norwegian’.

langpolish The language ‘Polish’.

langportuguese The language ‘Portuguese’.

langrussian The language ‘Russian’.

langslovak The language ‘Slovak’.

langslovene The language ‘Slovene’.

langspanish The language ‘Spanish’.

langswedish The language ‘Swedish’.

The following strings are intended for use in phrases like ‘translated from [the] English by ⟨*translator*⟩’:

fromamerican The expression ‘from [the] American’ or ‘from [the] American English’.

frombrazilian The expression ‘from [the] Brazilian’ or ‘from [the] Brazilian Portuguese’.

fromcatalan The expression ‘from [the] Catalan’.

fromcroatian The expression ‘from [the] Croatian’.

fromczech The expression ‘from [the] Czech’.

fromdanish The expression ‘from [the] Danish’.

fromdutch The expression ‘from [the] Dutch’.

fromenglish The expression ‘from [the] English’.

fromestonian The expression ‘from [the] Estonian’.

fromfinnish The expression ‘from [the] Finnish’.

fromfrench The expression ‘from [the] French’.

fromgerman The expression ‘from [the] German’.

fromgreek The expression ‘from [the] Greek’.

fromitalian The expression ‘from [the] Italian’.

fromjapanese The expression ‘from [the] Japanese’.

fromlatin The expression ‘from [the] Latin’.

fromnorwegian The expression ‘from [the] Norwegian’.

frompolish The expression ‘from [the] Polish’.

fromportuguese The expression ‘from [the] Portuguese’.

fromrussian The expression ‘from [the] Russian’.

fromslovak The expression ‘from [the] Slovak’.

fromslovene The expression ‘from [the] Slovene’.

fromspanish The expression ‘from [the] Spanish’.

fromswedish The expression ‘from [the] Swedish’.

#### 4.9.2.19 Country Names

Country names are localised by using the string `country` plus the ISO-3166 country code as the key. The short version of the translation should be the ISO-3166 country code. Note that only a small number of country names is defined by default, mainly to illustrate this scheme. These keys are used in the `location` list of `@patent` entries but they may be useful for other purposes as well.

countryde The name ‘Germany’, abbreviated as `DE`.

countryeu The name ‘European Union’, abbreviated as `EU`.

countryep Similar to `countryeu` but abbreviated as `EP`. This is intended for `patent` entries.

countryfr The name ‘France’, abbreviated as FR.

countryuk The name ‘United Kingdom’, abbreviated (according to ISO-3166) as GB.

countryus The name ‘United States of America’, abbreviated as US.

#### 4.9.2.20 Patents and Patent Requests

Strings related to patents are localised by using the term patent plus the ISO-3166 country code as the key. Note that only a small number of patent keys is defined by default, mainly to illustrate this scheme. These keys are used in the type field of @patent entries.

patent The generic term ‘patent’.

patentde The expression ‘German patent’.

patenteu The expression ‘European patent’.

patentfr The expression ‘French patent’.

patentuk The expression ‘British patent’.

patentus The expression ‘U.S. patent’.

Patent requests are handled in a similar way, using the string patreq as the base name of the key:

patreq The generic term ‘patent request’.

patreqde The expression ‘German patent request’.

patreqeu The expression ‘European patent request’.

patreqfr The expression ‘French patent request’.

patrequk The expression ‘British patent request’.

patrequs The expression ‘U.S. patent request’.

#### 4.9.2.21 Dates and Times

Abbreviation strings for standard eras. Both secular and Christian variants are supported.

commonera The era ‘CE’

beforecommonera The era ‘BCE’

annodomini The era ‘AD’

beforechrist The era ‘BC’

Abbreviation strings for ‘circa’ dates:

circa The string ‘circa’

Abbreviation strings for seasons parsed from EDTF dates:

spring The string ‘spring’

summer The string ‘summer’

autumn The string ‘autumn’

winter The string ‘winter’

Abbreviation strings for AM/PM:

am The string ‘AM’

pm The string ‘PM’

## 4.10 Formatting Commands

This section corresponds to § 3.10 in the user part of this manual. Bibliography and citation styles should incorporate the commands and facilities discussed in this section in order to provide a certain degree of high-level configurability. Users should not be forced to write new styles if all they want to do is modify the spacing in the bibliography or the punctuation used in citations.

### 4.10.1 User-definable Commands and Hooks

This section corresponds to § 3.10.1 in the user part of the manual. The commands and hooks discussed here are meant to be redefined by users, but bibliography and citation styles may provide a default definition which is different from the package default. These commands are defined in `biblatex.def`. Note that all commands starting with `\mk`… take one mandatory argument.

`\bibnamedelima` This delimiter controls the spacing between the elements which make up a name part. It is inserted automatically by the backend after the first name element if the element is less than three characters long and before the last element. The default definition is `\addhighpenspace`, i. e., a space penalized by the value of the `highnamepenalty` counter (§ 3.10.3). Please refer to § 3.13.4 for further details.

`\bibnamedelimb` This delimiter controls the spacing between the elements which make up a name part. It is inserted automatically by the backend between all name elements where `\bibnamedelima` does not apply. The default definition is `\addlowpenspace`, i. e., a space penalized by the value of the `lownamepenalty` counter (§ 3.10.3). Please refer to § 3.13.4 for further details.

`\bibnamedelimc` This delimiter controls the spacing between name parts. The default name formats use it between the name prefix and the last name if `useprefix=true`. The default definition is `\addhighpenspace`, i. e., a space penalized by the value of the `highnamepenalty` counter (§ 3.10.3). Please refer to § 3.13.4 for further details.

`\bibnamedelimd` This delimiter controls the spacing between name parts. The default name formats use it between all name parts where `\bibnamedelimc` does not apply. The default definition is `\addlowpenspace`, i. e., a space penalized by the value of the `lownamepenalty` counter (§ 3.10.3). Please refer to § 3.13.4 for further details.

\bibnamedelimi This delimiter replaces \bibnamedelima/b after initials. Note that this only applies to initials given as such in the bib file, not to the initials automatically generated by Biblatex which use their own set of delimiters.

\bibinitperiod The punctuation inserted automatically by the backend after all initials unless \bibinithyphendelim applies. The default definition is a period (\adddot). Please refer to § 3.13.4 for further details.

\bibinitdelim The spacing inserted automatically by the backend between multiple initials unless \bibinithyphendelim applies. The default definition is an unbreakable interword space. Please refer to § 3.13.4 for further details.

\bibinithyphendelim The punctuation inserted automatically by the backend between the initials of hyphenated name parts, replacing \bibinitperiod and \bibinitdelim. The default definition is a period followed by an unbreakable hyphen. Please refer to § 3.13.4 for further details.

\bibindexnamedelima Replaces \bibnamedelima in the index.

\bibindexnamedelimb Replaces \bibnamedelimb in the index.

\bibindexnamedelimc Replaces \bibnamedelimc in the index.

\bibindexnamedelimd Replaces \bibnamedelimd in the index.

\bibindexnamedelimi Replaces \bibnamedelimi in the index.

\bibindexinitperiod Replaces \bibinitperiod in the index.

\bibindexinitdelim Replaces \bibinitdelim in the index.

\bibindexinithyphendelim Replaces \bibinithyphendelim in the index.

\revsdnamepunct The punctuation to be printed between the first and last name parts when a name is reversed. The default is a comma. This command should be incorporated in formatting directives for name lists. Please refer to § 3.13.4 for further details.

\bibnamedash The dash to be used as a replacement for recurrent authors or editors in the bibliography. The default is an 'em' or an 'en' dash, depending on the indentation of the list of references.

\labelnamepunct The separator to be printed after the name used for alphabetizing in the bibliography (author or editor, if the author field is undefined). Use this separator instead of \newunitpunct at this location. The default is \newunitpunct, i. e., it is not handled differently from regular unit punctuation but permits convenient reconfiguration.

\subtitlepunct The separator to be printed between the fields title and subtitle, booktitle and booksubtitle, as well as maintitle and mainsubtitle. Use this separator instead of \newunitpunct at this location. The default is \newunitpunct, i. e., it is not handled differently from regular unit punctuation but permits convenient reconfiguration.

`\intitlepunct` The separator to be printed between the word "in" and the following title in entry types such as `@article`, `@inbook`, `@incollection`, etc. Use this separator instead of `\newunitpunct` at this location. The default definition is a colon plus an interword space.

`\bibpagespunct` The separator to be printed before the `pages` field. Use this separator instead of `\newunitpunct` at this location. The default is a comma plus an interword space.

`\bibpagerefpunct` The separator to be printed before the `pageref` field. Use this separator instead of `\newunitpunct` at this location. The default is an interword space.

`\multinamedelim` The delimiter to be printed between multiple items in a name list like `author` or `editor` if there are more than two names in the list. If there are only two names in the list, use the `\finalnamedelim` instead. This command should be incorporated in all formatting directives for name lists.

`\finalnamedelim` Use this command instead of `\multinamedelim` before the final name in a name list.

`\revsdnamedelim` The extra delimiter to be printed after the first name in a name list consisting of two names (in addition to `\finalnamedelim`) if the first name is reversed. This command should be incorporated in all formatting directives for name lists.

`\andothersdelim` The delimiter to be printed before the localisation string '`andothers`' if a name list like `author` or `editor` is truncated. This command should be incorporated in all formatting directives for name lists.

`\multilistdelim` The delimiter to be printed between multiple items in a literal list like `publisher` or `location` if there are more than two names in the list. If there are only two items in the list, use the `\finallistdelim` instead. This command should be incorporated in all formatting directives for literal lists.

`\finallistdelim` Use this command instead of `\multilistdelim` before the final item in a literal list.

`\andmoredelim` The delimiter to be printed before the localisation string '`andmore`' if a literal list like `publisher` or `location` is truncated. This command should be incorporated in all formatting directives for literal lists.

`\multicitedelim` The delimiter printed between citations if multiple entry keys are passed to a single citation command. This command should be incorporated in the definition of all citation commands, for example in the ⟨*sepcode*⟩ argument passed to `\DeclareCiteCommand`. See § 4.3.1 for details.

`\supercitedelim` Similar to `\multinamedelim`, but intended for the `\supercite` command only.

`\compcitedelim` Similar to `\multicitedelim`, but intended for citation styles that 'compress' multiple citations, i. e., print the author only once if subsequent citations share the same author etc.

`\textcitedelim` Similar to `\multicitedelim`, but intended for `\textcite` and related commands (§ 3.8.2).

`\nametitledelim` The delimiter to be printed between the author/editor and the title. This command should be incorporated in the definition of all citation commands of author-title and some verbose citation styles.

`\nameyeardelim` The delimiter to be printed between the author/editor and the year. This command should be incorporated in the definition of all citation commands of author-year citation styles.

`\namelabeldelim` The delimiter printed between the name/title and the label. This command should be incorporated in the definition of all citation commands of alphabetic and numeric citation styles.

`\nonameyeardelim` The delimiter printed between the substitute for the labelname when it does not exist (usually the label or title in standard styles) and the year in author-year citation styles. This is only used when there is no labelname since when the labelname exists, `\nameyeardelim` is used.

`\volcitedelim` The delimiter to be printed between the volume portion and the page/text portion of `\volcite` and related commands (§ 3.8.6).

`\prenotedelim` The delimiter to be printed after the ⟨*prenote*⟩ argument of a citation command.

`\postnotedelim` The delimiter to be printed after the ⟨*postnote*⟩ argument of a citation command.

`\extpostnotedelim` The delimiter printed between the citation and the parenthetical ⟨*postnote*⟩ argument of a citation command when the postnote occurs outside of the citation parentheses. In the standard styles, this occurs when the citation uses the shorthand field of the entry.

`\mkbibnamefamily{`⟨*text*⟩`}` Formatting hook for the family name, to be used in all formatting directives for name lists.

`\mkbibnamegiven{`⟨*text*⟩`}` Similar to `\mkbibnamefamily`, but intended for the given name.

`\mkbibnameprefix{`⟨*text*⟩`}` Similar to `\mkbibnamefamily`, but intended for the name prefix.

`\mkbibnamesuffix{`⟨*text*⟩`}` Similar to `\mkbibnamefamily`, but intended for the name suffix.

`\relatedpunct` The separator between the relatedtype bibliography localisation string and the data from the first related entry.

`\relateddelim` The separator between the data of multiple related entries. The default definition is a linebreak.

\relateddelim<relatedtype> The separator between the data of multiple related entries inside related entries of type 'relatedtype'. There is no default, if such a type-specific delimiter does not exist, \relateddelim is used.

### 4.10.2 Language-specific Commands

This section corresponds to § 3.10.2 in the user part of the manual. The commands discussed here are usually handled by the localisation modules, but may also be redefined by users on a per-language basis. Note that all commands starting with \mk… take one or more mandatory arguments.

\bibrangedash The language specific range dash. Defaults to \textendash.

\bibrangessep The language specific separator to be used between multiple ranges. Defaults to a comma followed by a space.

\bibdatesep The language specific separator used between date components in terse date formats. Defaults to \hyphen.

\bibdaterangesep The language specific separator to be used for date ranges. Defaults to \textendash for all date formats apart from ymd which defaults to a \slash. The date format option edtf is hard-coded to \slash since this is a standards compliant format.

\mkbibdatelong Takes the names of three field as arguments which correspond to three date components (in the order year/month/day) and uses their values to print the date in the language specific long date format.

\mkbibdateshort Similar to \mkbibdatelong but using the language specific short date format.

\mkbibtimezone Modifies a timezone string passed in as the only argument. By default this changes 'Z' to the value of \bibtimezone.

\bibdateuncertain The language specific marker to be used after uncertain dates when the global option dateuncertain is enabled. Defaults to a space followed by a question mark.

\bibdateeraprefix The language specific marker which is printed as a prefix to beginning BCE/BC dates in a date range when the option dateera is set to 'astronomical'. Defaults to \textminus, if defined and \textendash otherwise.

\bibdateeraendprefix The language specific marker which is printed as a prefix to end BCE/BC dates in a date range when the option dateera is set to 'astronomical'. Defaults to a thin space followed by \bibdateeraprefix when \bibdaterangesep is set to a dash and to \bibdateeraprefix otherwise. This is a separate macro so that you may add extra space before a negative date marker which, for example follows a dash date range marker as this can look a little odd.

\bibtimesep The language specific marker which separates time components. Default to a colon.

\bibutctimezone The language specific string printed for the UTC timezone. Defaults to 'Z'.

\bibtimezonesep The language specific marker which separates an optional time zone component from a time. Empty by default.

\bibdatetimesep The language specific separator printed between date and time components when printing time components along with date components (see the <datetype>dateusetime option in § 3.1.2.1). Defaults to a space for non-EDTF output formats, and 'T' for EDTF output format.

\finalandcomma Prints the comma to be inserted before the final 'and' in an enumeration, if applicable in the respective language.

\finalandsemicolon Prints the semicolon to be inserted before the final 'and' in an enumeration, if applicable in the respective language.

\mkbibordinal{⟨*integer*⟩}

Takes an integer argument and prints it as an ordinal number.

\mkbibmascord{⟨*integer*⟩}

Similar to \mkbibordinal, but prints a masculine ordinal, if applicable in the respective language.

\mkbibfemord{⟨*integer*⟩}

Similar to \mkbibordinal, but prints a feminine ordinal, if applicable in the respective language.

\mkbibneutord{⟨*integer*⟩}

Similar to \mkbibordinal, but prints a neuter ordinal, if applicable in the respective language.

\mkbibordedition{⟨*integer*⟩}

Similar to \mkbibordinal, but intended for use with the term 'edition'.

\mkbibordseries{⟨*integer*⟩}

Similar to \mkbibordinal, but intended for use with the term 'series'.

### 4.10.3 用户可定义的尺寸和计数器 User-definable Lengths and Counters

This section corresponds to § 3.10.3 in the user part of the manual. The length registers and counters discussed here are meant to be altered by users. Bibliography and citation styles should incorporate them where applicable and may also provide a default setting which is different from the package default.

\bibhang The hanging indentation of the bibliography, if applicable. This length is initialized to \parindent at load-time. If \parindent is zero length for some reason, \bibhang will default to 1em.

`\biblabelsep` The horizontal space between entries and their corresponding labels. Bibliography styles which use `list` environments and print a label should set `\labelsep` to `\biblabelsep` in the definition of the respective environment.

`\bibitemsep` The vertical space between the individual entries in the bibliography. Bibliography styles using `list` environments should set `\itemsep` to `\bibitemsep` in the definition of the respective environment.

`\bibparsep` The vertical space between paragraphs within an entry in the bibliography. Bibliography styles using `list` environments should set `\parsep` to `\bibparsep` in the definition of the respective environment.

`abbrvpenalty` The penalty used by `\addabbrvspace`, `\addabthinspace`, and `\adddotspace`, see § 4.7.4 for details.

`lownamepenalty` The penalty used by `\addlowpenspace` and `\addlpthinspace`, see § 4.7.4 for details.

`highnamepenalty` The penalty used by `\addhighpenspace` and `\addhpthinspace`, see § 4.7.4 for details.

`biburlnumpenalty` If this counter is set to a value greater than zero, Biblatex will permit linebreaks after numbers in all strings formatted with the `\url` command from the `url` package. This will affect URLs and DOIs in the bibliography. The breakpoints will be penalized by the value of this counter. If URLs and/or DOIs in the bibliography run into the margin, try setting this counter to a value greater than zero but less than 10000 (you normally want to use a high value like 9000). Setting the counter to zero disables this feature. This is the default setting.

`biburlucpenalty` Similar to `biburlnumpenalty`, except that it will add a breakpoint after all uppercase letters.

`biburllcpenalty` Similar to `biburlnumpenalty`, except that it will add a breakpoint after all lowercase letters.

### 4.10.4 辅助命令和钩子 Auxiliary Commands and Hooks

The auxiliary commands and facilities in this section serve a special purpose. Some of them are used by Biblatex to communicate with bibliography and citation styles in some way or other.

`\mkbibemph{`⟨*text*⟩`}`

A generic command which prints its argument as emphasized text. This is a simple wrapper around the standard `\emph` command. Apart from that, it uses `\setpunctfont` from § 4.7.1 to adapt the font of the next punctuation mark following the text set in italics. If the `punctfont` package option is disabled, this command behaves like `\emph`.

\mkbibitalic{⟨*text*⟩}

Similar in concept to \mkbibemph but prints italicized text. This is a simple wrapper around the standard \textit command which incorporates \setpunctfont. If the punctfont package option is disabled, this command behaves like \textit.

\mkbibbold{⟨*text*⟩}

Similar in concept to \mkbibemph but prints bold text. This is a simple wrapper around the standard \textbf command which incorporates \setpunctfont. If the punctfont package option is disabled, this command behaves like \textbf.

\mkbibquote{⟨*text*⟩}

A generic command which wraps its argument in quotation marks. If the csquotes package is loaded, this command uses the language sensitive quotation marks provided by that package. \mkbibquote also supports 'American-style' punctuation, see \DeclareQuotePunctuation in § 4.7.5 for details.

\mkbibparens{⟨*text*⟩}

A generic command which wraps its argument in parentheses. This command is nestable. When nested, it will alternate between parentheses and brackets, depending on the nesting level.

\mkbibbrackets{⟨*text*⟩}

A generic command which wraps its argument in square brackets. This command is nestable. When nested, it will alternate between brackets and parentheses, depending on the nesting level.

\bibopenparen⟨*text*⟩\bibcloseparen

Alternative syntax for \mkbibparens. This will also work across groups. Note that \bibopenparen and \bibcloseparen must always be balanced.

\bibopenbracket⟨*text*⟩\bibclosebracket

Alternative syntax for \mkbibbrackets. This will also work across groups. Note that \bibopenbracket and \bibclosebracket must always be balanced.

\mkbibfootnote{⟨*text*⟩}

A generic command which prints its argument as a footnote. This is a wrapper around the standard LaTeX \footnote command which removes spurious whitespace preceding the footnote mark and prevents nested footnotes. By default, \mkbibfootnote requests capitalization at the beginning of the note and automatically adds a period at the end. You may change this behavior by redefining the \bibfootnotewrapper macro introduced below.

\mkbibfootnotetext{⟨*text*⟩}

Similar to \mkbibfootnote but uses the \footnotetext command.

\mkbibendnote{⟨*text*⟩}

Similar in concept to \mkbibfootnote except that it prints its argument as an endnote. \mkbibendnote removes spurious whitespace preceding the endnote mark and prevents nested notes. It supports the \endnote command provided by the endnotes package as well as the \pagenote command provided by the pagenote package and the memoir class. If both commands are available, \endnote takes precedence. If no endnote support is available, \mkbibendnote issues an error and falls back to \footnote. By default, \mkbibendnote requests capitalization at the beginning of the note and automatically adds a period at the end. You may change this behavior by redefining the \bibendnotewrapper macro introduced below.

\mkbibendnotetext{⟨*text*⟩}

Similar to \mkbibendnote but uses the \endnotetext command. Please note that as of this writing, neither the pagenote package nor the memoir class provide a corresponding \pagenotetext command. In this case, \mkbibendnote will issue an error and fall back to \footnotetext.

\bibfootnotewrapper{⟨*text*⟩}

An inner wrapper which encloses the ⟨*text*⟩ argument of \mkbibfootnote and \mkbibfootnotetext. For example, \mkbibfootnote eventually boils down to this:

```
\footnote{\bibfootnotewrapper{text}}
```

The wrapper ensures capitalization at the beginning of the note and adds a period at the end. The default definition is:

```
\newcommand{\bibfootnotewrapper}[1]{\bibsentence #1\addperiod}
```

If you don't want capitalization at the beginning or a period at the end of the note, do not modify \mkbibfootnote but redefine \bibfootnotewrapper instead.

\bibendnotewrapper{⟨*text*⟩}

Similar in concept to \bibfootnotewrapper but related to the \mkbibendnote and \mkbibendnotetext commands.

\mkbibsuperscript{⟨*text*⟩}

A generic command which prints its argument as superscripted text. This is a simple wrapper around the standard LaTeX \textsuperscript command which removes spurious whitespace and allows hyphenation of the preceding word.

`\mkbibmonth{`⟨*integer*⟩`}`

This command takes an integer argument and prints it as a month name. Even though the output of this command is language specific, its definition is not, hence it is normally not redefined in localisation modules.

`\mkbibseason{`⟨*string*⟩`}`

This command takes a season localisation string and prints the version of the string corresponding to the setting of the `dateabbrev` package option. Even though the output of this command is language specific, its definition is not, hence it is normally not redefined in localisation modules.

`\mkyearzeros{`⟨*integer*⟩`}`

This command strips leading zeros from a year or enforces them, depending on the `datezeros` package option (§ 3.1.2.1). It is intended for use in the definition of date formatting macros.

`\mkmonthzeros{`⟨*integer*⟩`}`

This command strips leading zeros from a month or enforces them, depending on the `datezeros` package option (§ 3.1.2.1). It is intended for use in the definition of date formatting macros.

`\mkdayzeros{`⟨*integer*⟩`}`

This command strips leading zeros from a day or enforces them, depending on the `datezeros` package option (§ 3.1.2.1). It is intended for use in the definition of date formatting macros.

`\mktimezeros{`⟨*integer*⟩`}`

This command strips leading zeros from a number or preserves them, depending on the `timezeros` package option (§ 3.1.2.1). It is intended for use in the definition of time formatting macros.

`\forcezerosy{`⟨*integer*⟩`}`

This command adds zeros to a year (or any number supposed to be 4-digits). It is intended for date formatting and ordinals.

`\forcezerosmdt{`⟨*integer*⟩`}`

This command adds zeros to a month, day or time part (or any number supposed to be 2-digits). It is intended for date/time formatting and ordinals.

`\stripzeros{`⟨*integer*⟩`}`

This command strips leading zeros from a number. It is intended for date formatting and ordinals.

<labelfield>width For every field marked as a 'Label field' in the data model, a formatting directive is created as per `shorthandwidth` above. Since `shorthand` is so marked in the default data model, this functionality is a superset of that described for `shorthandwidth`.

labelnumberwidth Similar to `shorthandwidth`, but referring to the `labelnumber` field and the length register `\labelnumberwidth`. Numeric styles should adjust this directive such that it corresponds to the format used in the bibliography.

labelalphawidth Similar to `shorthandwidth`, but referring to the `labelalpha` field and the length register `\labelalphawidth`. Alphabetic styles should adjust this directive such that it corresponds to the format used in the bibliography.

bibhyperref A special formatting directive for use with `\printfield` and `\printtext`. This directive wraps its argument in a `\bibhyperref` command, see § 4.6.4 for details.

bibhyperlink A special formatting directive for use with `\printfield` and `\printtext`. It wraps its argument in a `\bibhyperlink` command, see § 4.6.4 for details. The ⟨*name*⟩ argument passed to `\bibhyperlink` is the value of the `entrykey` field.

bibhypertarget A special formatting directive for use with `\printfield` and `\printtext`. It wraps its argument in a `\bibhypertarget` command, see § 4.6.4 for details. The ⟨*name*⟩ argument passed to `\bibhypertarget` is the value of the `entrykey` field.

volcitepages A special formatting directive which controls the format of the page/text portion in the argument of citation commands like `\volcite`.

volcitevolume A special formatting directive which controls the format of the volume portion in the argument of citation commands like `\volcite`.

date A special formatting directive which controls the format of `\printdate` (§ 4.4.1). Note that the date format (long/short etc.) is controlled by the package option `date` from § 3.1.2.1. This formatting directive only controls additional formatting such as fonts etc.

labeldate As `date` but controls the format of `\printlabeldate`.

<datetype>date As `date` but controls the format of `\print<datetype>date`.

time A special formatting directive which controls the format of `\printtime` (§ 4.4.1). Note that the time format (24h/12h etc.) is controlled by the package option `time` from § 3.1.2.1. This formatting directive only controls additional formatting such as fonts etc.

labeltime As `time` but controls the format of `\printlabeltime`.

<datetype>time As `time` but controls the format of `\print<datetype>time`.

### 4.10.5 辅助长度计数器和其它功能 Auxiliary Lengths, Counters, and Other Features

The length registers and counters discussed here are used by Biblatex to pass information to bibliography and citation styles. Think of them as read-only registers. Note that all counters are LaTeX counters. Use `\value{counter}` to read out the current value.

`\<labelfield>width` For every field marked as a 'label' field in the data model, a length register is created as per `shorthandwidth` above. Since `shorthand` is so marked in the default data model, this functionality is a superset of that described for `shorthandwidth`.

`\labelnumberwidth` This length register indicates the width of the widest `labelnumber`. Numeric bibliography styles should incorporate this length in the definition of the bibliography environment.

`\labelalphawidth` This length register indicates the width of the widest `labelalpha`. Alphabetic bibliography styles should incorporate this length in the definition of the bibliography environment.

`maxextraalpha` This counter holds the highest number found in any `extraalpha` field.

`maxextrayear` This counter holds the highest number found in any `extrayear` field.

`refsection` This counter indicates the current `refsection` environment. When queried in a bibliography heading, the counter returns the value of the `refsection` option passed to `\printbibliography`.

`refsegment` This counter indicates the current `refsegment` environment. When queried in a bibliography heading, this counter returns the value of the `refsegment` option passed to `\printbibliography`.

`maxnames` This counter holds the setting of the `maxnames` package option.

`minnames` This counter holds the setting of the `minnames` package option.

`maxitems` This counter holds the setting of the `maxitems` package option.

`minitems` This counter holds the setting of the `minitems` package option.

`instcount` This counter is incremented by Biblatex for every citation as well as for every entry in the bibliography and bibliography lists. The value of this counter uniquely identifies a single instance of a reference in the document.

`citetotal` This counter, which is only available in the ⟨*loopcode*⟩ of a citation command defined with `\DeclareCiteCommand`, holds the total number of valid entry keys passed to the citation command.

`citecount` This counter, which is only available in the ⟨*loopcode*⟩ of a citation command defined with `\DeclareCiteCommand`, holds the number of the entry key currently being processed by the ⟨*loopcode*⟩.

`multicitetotal` This counter is similar to `citetotal` but only available in multicite commands. It holds the total number of citations passed to the multicite command. Note that each of these citations may consist of more than one entry key. This information is provided by the `citetotal` counter.

`multicitecount` This counter is similar to `citecount` but only available in multicite commands. It holds the number of the citation currently being processed. Note that this citation may consist of more than one entry key. This information is provided by the `citetotal` and `citecount` counters.

`listtotal` This counter holds the total number of items in the current list. It is intended for use in list formatting directives and does not hold a meaningful value when used anywhere else. As an exception, it may also be used in the second optional argument to `\printnames` and `\printlist`, see § 4.4.1 for details. For every list, there is also a counter by the same name which holds the total number of items in the corresponding list. For example, the `author` counter holds the total number of items in the `author` list. This applies to both name lists and literal lists. These counters are similar to `listtotal` except that they may also be used independently of list formatting directives. For example, a bibliography style might check the `editor` counter to decide Whether or not to print the term "editor" or rather its plural form "editors" after the list of editors.

`listcount` This counter holds the number of the list item currently being processed. It is intended for use in list formatting directives and does not hold a meaningful value when used anywhere else.

`liststart` This counter holds the ⟨*start*⟩ argument passed to `\printnames` or `\printlist`. It is intended for use in list formatting directives and does not hold a meaningful value when used anywhere else.

`liststop` This counter holds the ⟨*stop*⟩ argument passed to `\printnames` or `\printlist`. It is intended for use in list formatting directives and does not hold a meaningful value when used anywhere else.

`\currentlang` The name of the currently active language for Biblatex. Can be used anywhere and defaults to the main document language. This is automatically switched inside entries which define `langid`, given suitable settings of the `autolang` and `language` options. Note that this does not track all document language changes, only the current Biblatex setting.

`\currentfield` The name of the field currently being processed by `\printfield`. This information is only available locally in field formatting directives.

`\currentlist` The name of the literal list currently being processed by `\printlist`. This information is only available locally in list formatting directives.

`\currentname` The name of the name list currently being processed by `\printnames`. This information is only available locally in name formatting directives.

### 4.10.6 多用途钩子 General Purpose Hooks

`\AtBeginBibliography{`⟨*code*⟩`}`

Appends the ⟨*code*⟩ to an internal hook executed at the beginning of the bibliography. The ⟨*code*⟩ is executed at the beginning of the list of references, immediately after the ⟨*begin code*⟩ of `\defbibenvironment`. This command may only be used in the preamble.

`\AtBeginShorthands{`⟨*code*⟩`}`

Appends the ⟨*code*⟩ to an internal hook executed at the beginning of the list of shorthands. The ⟨*code*⟩ is executed at the beginning of the list of shorthands, immediately after the ⟨*begin code*⟩ of `\defbibenvironment`. This command may only be used in the preamble.

This is just an alias for:

```
\AtBeginBiblist{shorthand}{code}
```

`\AtBeginBiblist{`⟨*biblistname*⟩`}{`⟨*code*⟩`}`

Appends the ⟨*code*⟩ to an internal hook executed at the beginning of the bibliography list ⟨*biblistname*⟩. The ⟨*code*⟩ is executed at the beginning of the bibliography list, immediately after the ⟨*begin code*⟩ of `\defbibenvironment`. This command may only be used in the preamble.

`\AtEveryBibitem{`⟨*code*⟩`}`

Appends the ⟨*code*⟩ to an internal hook executed at the beginning of every item in the bibliography. The ⟨*code*⟩ is executed immediately after the ⟨*item code*⟩ of `\defbibenvironment`. The bibliographic data of the respective entry is available at this point. This command may only be used in the preamble.

`\AtEveryLositem{`⟨*code*⟩`}`

Appends the ⟨*code*⟩ to an internal hook executed at the beginning of every item in the list of shorthands. The ⟨*code*⟩ is executed immediately after the ⟨*item code*⟩ of `\defbibenvironment`. The bibliographic data of the respective entry is available at this point. This command may only be used in the preamble.

This is just an alias for:

```
\AtEveryBiblistitem{shorthand}{code}
```

\AtEveryBiblistitem{⟨*biblistname*⟩}{⟨*code*⟩}

Appends the ⟨*code*⟩ to an internal hook executed at the beginning of every item in the bibliography list named ⟨*biblistname*⟩. The ⟨*code*⟩ is executed immediately after the ⟨*item code*⟩ of `\defbibenvironment`. The bibliographic data of the respective entry is available at this point. This command may only be used in the preamble.

\AtNextBibliography{⟨*code*⟩}

Similar to `\AtBeginBibliography` but only affecting the next `\printbibliography`. The internal hook is cleared after being executed once. This command may be used in the document body.

\AtEveryCite{⟨*code*⟩}

Appends the ⟨*code*⟩ to an internal hook executed at the beginning of every citation command. The ⟨*code*⟩ is executed immediately before the ⟨*precode*⟩ of the command (see § 4.3.1). No bibliographic data is available at this point. This command may only be used in the preamble.

\AtEveryCitekey{⟨*code*⟩}

Appends the ⟨*code*⟩ to an internal hook executed once for every entry key passed to a citation command. The ⟨*code*⟩ is executed immediately before the ⟨*loopcode*⟩ of the command (see § 4.3.1). The bibliographic data of the respective entry is available at this point. This command may only be used in the preamble.

\AtEveryMultiCite{⟨*code*⟩}

Appends the ⟨*code*⟩ to an internal hook executed at the beginning of every multicite command. The ⟨*code*⟩ is executed immediately before the `multiprenote` field (§ 4.3.2) is printed. No bibliographic data is available at this point. This command may only be used in the preamble.

\AtNextCite{⟨*code*⟩}

Similar to `\AtEveryCite` but only affecting the next citation command. The internal hook is cleared after being executed once. This command may be used in the document body.

\AtEachCitekey{⟨*code*⟩}

Similar to `\AtEveryCitekey` but only affecting the current citation command. This command may be used in the document body. The ⟨*code*⟩ is appended to the internal hook locally when located in a citation, as determined by `\ifcitation`.

\AtNextCitekey{⟨*code*⟩}

Similar to `\AtEveryCitekey` but only affecting the next entry key. The internal hook is cleared after being executed once. This command may be used in the document body.

`\AtNextMultiCite{`⟨*code*⟩`}`

Similar to `\AtEveryMultiCite` but only affecting the next multicite command. The internal hook is cleared after being executed once. This command may be used in the document body.

`\AtDataInput[`⟨*entrytype*⟩`]{`⟨*code*⟩`}`

Appends the ⟨*code*⟩ to an internal hook executed once for every entry as the bibliographic data is imported from the `bbl` file. The ⟨*entrytype*⟩ is the entry type the ⟨*code*⟩ applies to. If it applies to all entry types, omit the optional argument. The ⟨*code*⟩ is executed immediately after the entry has been imported. This command may only be used in the preamble. Note that ⟨*code*⟩ may be executed multiple times for an entry. This occurs when the same entry is cited in different `refsection` environments or the `sorting` option settings incorporate more than one sorting scheme. The `refsection` counter holds the number of the respective reference section while the data is imported.

`\UseBibitemHook`

Executes the internal hook corresponding to `\AtEveryBibitem`.

`\UseEveryCiteHook`

Executes the internal hook corresponding to `\AtEveryCite`.

`\UseEveryCitekeyHook`

Executes the internal hook corresponding to `\AtEveryCitekey`.

`\UseEveryMultiCiteHook`

Executes the internal hook corresponding to `\AtMultiEveryCite`.

`\UseNextCiteHook`

Executes and clears the internal hook corresponding to `\AtNextCite`.

`\UseNextCitekeyHook`

Executes and clears the internal hook corresponding to `\AtNextCitekey`.

`\UseNextMultiCiteHook`

Executes and clears the internal hook corresponding to `\AtNextMultiCite`.

`\DeferNextCitekeyHook`

Locally un-defines the internal hook specified by `\AtNextCitekey`. This essentially defers the hook to the next entry key in the citation list, when executed in the ⟨*precode*⟩ argument of `\DeclareCiteCommand` (§ 4.3.1).

## 4.11 提示与警告

本节提供了关于 biblatex 宏包接口的一些附加提示，也将论述一些普遍性的问题和容易误解的概念。

### 4.11.1 条目集

条目集已经在§3.12.5节介绍过，本节主要讨论怎么在著录样式中处理条目集。从驱动的角度看，静态和动态的条目集并无差别。两者都以相同方式处理。只需要使用§4.4.1的`\entryset`命令遍历集的所有成员 (以在`@set`条目的`entryset`域中的出现的顺序，或者它们传递给`\defbibentryset`命令的顺序进行遍历)，并在最后加上`\finentry`命令即可。格式化则有集的成员各自的条目类型的驱动控制。

```
\DeclareBibliographyDriver{set}{%
  \entryset{}{}%
  \finentry}
```

需要注意: 本宏包附带的 `numeric` 样式支持条目集细分，即集成员以一个字母或者其他记号来标记，标注命令可以引用整个集或者其中的某一具体成员。记号由样式文件以如下方式生成:

```
\DeclareBibliographyDriver{set}{%
  \entryset
    {\printfield{entrysetcount}%
     \setunit*{\addnbspace}}
    {}%
  \finentry}
```

`entrysetcount`域保存了一个整数用于指示集成员在整个集中的位置。数字是转换为一个字母还是其他记号由域格式`entrysetcount`控制。所有驱动需要做的是打印域和一些空格 (或者换行)。在标注中打印记号的方式类似。当顺序编码制样式给出`\printfield{labelnumber}` 时，可以简单地加上`entrysetcount`域。

```
\printfield{labelnumber}\printfield{entrysetcount}
```

因为该域仅在处理标注指向一个集成员时定义，所以没有必要添加任何更多的判断。

### 4.11.2 电子出版信息

标准样式主要支持 arXiv 网站的文献[65]。其它资源的支持很容易增加。标准样式以如下方式处理`eprint`域:

[65]译者:arXiv 原先是由物理学家保罗·金斯巴格在 1991 年建立的网站，本意在收集物理学的论文预印本，随后括及天文、数学等其它领域。金斯巴格因为这个网站获得了 2002 年的麦克阿瑟奖。arXiv 原先挂在洛斯阿拉莫斯国家实验室，是故早期被称为「LANL 预印本数据库」。目前的 arXiv 落脚于康乃尔大学 [1]，并在全球各地设有镜像站点。网站在 1999 年改名为 arXiv.org 。

```
\iffieldundef{eprinttype}
  {\printfield{eprint}}
  {\printfield[eprint:\strfield{eprinttype}]{eprint}}
```

如果eprinttype域存在，上述代码将使用域格式 eprint:⟨*eprinttype*⟩。如果该格式未定义，\printfield自动退回到使用域格式 eprint。有两种预定义的域格式，type-specific(具体类型的) 域格式 eprint:arxiv 和通用域格式 eprint。

```
\DeclareFieldFormat{eprint}{...}
\DeclareFieldFormat{eprint:arxiv}{...}
```

换句话说，增加其他数据源的支持只需要定义一个名为 eprint:⟨*resource*⟩的与格式，其中⟨*resource*⟩是在eprinttype域中使用的标识。

### 4.11.3 外部摘要和注释

外部摘要和注释已经在 § 3.12.8节讨论过，本节为样式作者提供更多的背景知识。标准样式使用如下的宏 (来自 biblatex.def) 来处理摘要和注释:

```
\newbibmacro*{annotation}{%
  \iffieldundef{annotation}
    {\printfile[annotation]{\bibannotationprefix\thefield{entrykey}.tex
    ↪ }}%
    {\printfield{annotation}}}
\newcommand*{\bibannotationprefix}{bibannotation-}

\newbibmacro*{abstract}{%
  \iffieldundef{abstract}
    {\printfile[abstract]{\bibabstractprefix\thefield{entrykey}.tex}}%
    {\printfield{abstract}}}
\newcommand*{\bibabstractprefix}{bibabstract-}
```

如果abstract/annotation域未定义，上述代码将从外部文件中加载摘要和注释。\printfile将根据用户定义的前缀来搜索文件名。注意: 必须显式地设置 § 3.1.2.1节的loadfiles包选项来打开\printfile功能。基于性能原因该功能默认是关闭的。

### 4.11.4 消除姓名歧义

在 § 3.1.2.3节引入的uniquename和uniquelist选项支持多种操作模式。本节用举例方式介绍不同模式的差别。uniquename选项消除labelname列表中各姓名间的歧义，uniquelist消除因maxnames/minnames截短导致的labelname列表歧义。两个选项可以单独使用也可以联合使用:

消除姓名歧义原理是根据由一个或多个姓名成分构成的‘base’ 来确定需要在其基础上添加什么 (如果存在的话) 使得姓名在当前参考文献节中是唯一的。消除姓名歧义由如下命令声明的 uniquename 模板控制:

`\DeclareUniquenameTemplate{`⟨*specification*⟩`}`

⟨*specification*⟩是`\namepart`命令列表，定义了确定 uniquename 信息使用的姓名成分。

`\namepart[`⟨*options*⟩`]{`⟨*namepart*⟩`}`

⟨*namepart*⟩是数据模型中的姓名成分，由`\DeclareDatamodelConstant`命令定义 (见 § 4.2.3)。选项包括:

`use`=`true`, `false` default: `false`

在构建 uniquename 信息中仅使用⟨*namepart*⟩，如果存在相应的选项`use‘namepart’`并且值为 true。

`base`=`true`, `false` default: `false`

⟨*namepart*⟩是‘base’ 的部分，‘base’ 是用作唯一性区分的 namepart(s) 信息的主段。

默认的 uniquename 模板是:

```
\DeclareUniquenameTemplate{
  \namepart[use=true, base=true]{prefix}
  \namepart[base=true]{family}
  \namepart{given}
}
```

这意味着要区分的‘base’ 由姓 (‘family’) 和前缀 (如果`useprefix`选项是 true) 构成。消除歧义主要通过增加模板中任何非‘base’ 姓名成分来实现，这里就是名 (‘given’) 成分。

#### 4.11.4.1 单个姓名 (姓名间的区分)(uniquename)

下面从一些`uniquename`例子开始，考虑如下数据:

```
John Doe   2008
Edward Doe 2008
John Smith 2008
Jane Smith 2008
```

假设我们使用作者年制且设置 `uniquename=false`，这种情况下，我们得到如下引用标注:

```
Doe 2008a
Doe 2008b
Smith 2008a
Smith 2008b
```

因为姓有歧义，且所有的年都相同，所以年后附加的字符用来区分并消除歧义。然而，很多样式指南强制要求附加字符只能用于相同作者的区分，而不能用于作者相同的姓的区分。为了消除作者姓的歧义，需要增加姓名的其它完整部分或者缩写来区分。这一需要由uniquename选项处理，下面是使用了 uniquename=init 的引用标注:

```
J. Doe 2008
E. Doe 2008
Smith 2008a
Smith 2008b
```

uniquename=init 限制了用缩写来区分姓名。但因为'J. Smith' 仍然有歧义，所以没有增加。而使用 uniquename=full，标注如下:

```
J. Doe 2008
E. Doe 2008
John Smith 2008
Jane Smith 2008
```

为了说明 uniquename=init/full 和 allinit/allfull 的差别，我们下面介绍 'visible' 姓名的概念。'visible' 姓名是位于maxnames/minnames/uniquelist截短点前的姓名，比如，给出数据:

```
William Jones/Edward Doe/Jane Smith
John Doe
John Smith
```

当 maxnames=1, minnames=1, uniquename=init/full 时，我们得到如下的引用标注:

```
Jones et al.
Doe
Smith
```

在消除歧义的时候，uniquename=init/full 仅考虑可见的姓名。因为本例中所有的可见姓名的姓都是不同的，所有没有姓名的其他部分附加进来。比较一下使用 uniquename=allinit 的输出:

```
Jones et al.
J. Doe
Smith
```

allinit 认为所有在labelname列表中的姓名，包括列表截短后已经隐藏并且由'et al.' 代替的姓名。在本例中，'John Doe' 与 'Edward Doe' 存在歧义。因为两个 'Smiths' 无法通过添加缩写的方式区分，所以没有添加。现在来比较一下 uniquename = allfull 的输出:

```
Jones et al.
J. Doe
John Smith
```

`uniquename=mininit/minfull` 选项类似于 `init/full` 仅考虑可见姓名，但仅执行最小的歧义消除。即，仅对姓列表的歧义进行处理，考虑如下数据:

```
John Doe/William Jones
Edward Doe/William Jones
John Smith/William Edwards
Edward Smith/Allan Johnson
```

使用 `uniquename=init/full`，得到:

```
J. Doe and Jones
E. Doe and Jones
J. Smith and Edwards
E. Smith and Johnson
```

使用 `uniquename=mininit/minfull`，得到:

```
J. Doe and Jones
E. Doe and Jones
Smith and Edwards
Smith and Johnson
```

'Smiths' 并无歧义，因为姓名列表时没有歧义。`mininit/minfull`选项仅对姓的列表相同情况进行处理。全局的看姓的列表，注意当未截短的列表的可见名相同的时候，截短的列表时也可能是不同的，比如下面的数据:

```
John Doe/William Jones
Edward Doe
```

使用 `maxnames=1`, `uniquename=init/full`:

```
J. Doe et al.
E. Doe
```

使用 `uniquename=mininit/minfull`:

```
Doe et al.
Doe
```

因为列表有 'et al.' 的不同，姓名列表就不歧义。

#### 4.11.4.2 姓名列表（列表间的区分）(uniquelist)

姓名列表也可能存在歧义问题。如果labelname列表由maxnames/minnames选项截短就可能产生歧义。这类问题由uniquelist选处理，考虑如下数据:

```
Doe/Jones/Smith   2005
Smith/Johnson/Doe 2005
Smith/Doe/Edwards 2005
Smith/Doe/Jones   2005
```

很多作者年制样式需要在标注中截短，比如使用 maxnames=1 选项，得到:

```
Doe et al. 2005
Smith et al. 2005a
Smith et al. 2005b
Smith et al. 2005c
```

因为截短后作者存在歧义，所以添加额外字符确保引用标注的唯一性。同样的，一些样式强制要求额外字符只能用于所有作者都相同的情况。为了区分作者列表，必须增加更多的姓名，这样就会超出maxnames/minnames选项设定的截短点。uniquelist选项即描述这一需求，当 uniquelist=true，有:

```
Doe et al. 2005
Smith, Johnson et al. 2005
Smith, Doe and Edwards 2005
Smith, Doe and Jones 2005
```

uniquelist选项以条目为限重设maxnames/minnames。大体上，标注的 'et al.' 部分扩展到无歧义的点–而且也基本到此为止。uniquelist也可以与uniquename联合使用，考虑如下数据:

```
John Doe/Allan Johnson/William Jones  2009
John Doe/Edward Johnson/William Jones 2009
John Doe/Jane Smith/William Jones     2009
John Doe/John Smith/William Jones     2009
John Doe/John Edwards/William Jones   2009
John Doe/John Edwards/Jack Johnson    2009
```

使用 maxnames=1，得到:

```
Doe et al. 2009a
Doe et al. 2009b
Doe et al. 2009c
Doe et al. 2009d
Doe et al. 2009e
Doe et al. 2009f
```

使用 maxnames=1, uniquename=full, uniquelist=true 则有:

```
Doe, A. Johnson et al. 2009
Doe, E. Johnson et al. 2009
Doe, Jane Smith et al. 2009
Doe, John Smith et al. 2009
Doe, Edwards and Jones 2009
Doe, Edwards and Johnson 2009
```

使用 uniquelist=minyear，消除列表歧义仅在可见列表和labelyear相同的时候。这对于仅仅需要整个标注整体具有唯一性的作者年制样式是很有用的，但是不保证作者姓名的非歧义性。这一模式概念上域 uniquename=mininit/minfull 选项相关。考虑如下例子:

```
Smith/Jones   2000
Smith/Johnson 2001
```

使用 maxnames=1 和 uniquelist=true，得到:

```
Smith and Jones 2000
Smith and Johnson 2001
```

使用 uniquelist=minyear，则得到:

```
Smith et al. 2000
Smith et al. 2001
```

使用 uniquelist=minyear，两个文献的作者是否相同并不清楚，但标注的整体是非歧义的，因为年份的不同。与此相反，uniquelist=true 需要消除作者列表的歧义即便这一信息对于参考文献表的唯一引用是不必要的，看看如下例子:

```
Vogel/Beast/Garble/Rook  2000
Vogel/Beast/Tremble/Bite 2000
Vogel/Beast/Acid/Squeeze 2001
```

使用 maxnames=3, minnames=1, uniquelist=true，得到

```
Vogel, Beast, Garble et al. 2000
Vogel, Beast, Tremble et al. 2000
Vogel, Beast, Acid et al. 2001
```

使用 uniquelist=minyear 选项，则有:

```
Vogel, Beast, Garble et al. 2000
Vogel, Beast, Tremble et al. 2000
Vogel et al. 2001
```

在最后一个引用中，uniquelist=minyear 不重写maxnames/minnames，因为年份的不同，所以不需要消除与其它两个间的歧义。

### 4.11.5 浮动体和TOC/LOT/LOF中的追踪器

当引用命令出现在浮动体 (比如图和表的题注) 中，因为浮动体无论是物理上还是逻辑上都在文本流之外，这会导致的学术反向引用 (比如'ibidem') 和基于页码追踪器的反向引用难以区分，因此这种引用的逻辑很难在其中应用。为避免这种问题，引用和页码追踪器在所有的浮动体中临时关闭。而且，这些追踪器加上反向引用追踪器 (backref) 在目录，图和表目录中也临时关闭。[666768]

### 4.11.6 混合编程接口

Biblatex 宏包给样式作者提供了 2 个主要的编程接口即: bbx文件中使用的\DeclareBibliographyDriver 命令用来定义各类参考文献条目的驱动 (即条目的格式处理器)，cbx文件中使用的\DeclareCiteCommand命令用来定义新的标注命令。然而有时候，混合使用这两个接口会很方便。比如\fullcite命令就可以打印类似于完整参考文献条目的长串引用，该命令定义大体如下:

```
\DeclareCiteCommand{\fullcite}
  {...}
  {\usedriver{...}{\thefield{entrytype}}}
  {...}
  {...}
```

如上所见，打印标注的核心代码简单地为当前的条目类型执行了\DeclareBibliographyDriver定义的驱动命令。当为长标注格式[69]编写标注样式文件的时候，使用下面的结构是非常方便的:

```
\ProvidesFile{example.cbx}[2007/06/09 v1.0 biblatex citation style]

\DeclareCiteCommand{\cite}
  {...}
  {\usedriver{...}{cite:\thefield{entrytype}}}
  {...}
  {...}

\DeclareBibliographyDriver{cite:article}{...}
\DeclareBibliographyDriver{cite:book}{...}
\DeclareBibliographyDriver{cite:inbook}{...}
...
```

[66]译者:reference 引用，链接，引文？从交叉引用角度看呢？
[67]译者:citation 引用，标注？
[68]译者:ibidem 也需要看一下前面，到底什么意义
[69]译者:verbose: 冗长的

混合接口的另一个有用情况是在参考文献表中使用交叉引用 (cross-references) 时。比如当打印`@incollection` 类型的条目，数据继承自`@collection`父条目，可由一个指向对应父条目的简短指针来代替。

[1] Audrey Author: *Title of article.* In: [2], pp. 134–165.

[2] Edward Editor, ed.: *Title of collection.* Publisher: Location, 1995.

实现参考文献表内的这种交叉引用的一种方法是将它们当成标注，并使用`xref` 或 `crossref`域的值作为条目关键词 (条目 bibtex 键)，示例如下:

```
\ProvidesFile{example.bbx}[2007/06/09 v1.0 biblatex bibliography style]

\DeclareCiteCommand{\bbx@xref}
  {}
  {...}% code for cross-references
  {}
  {}

\DeclareBibliographyDriver{incollection}{%
  ...
  \iffieldundef{xref}
    {...}% code if no cross-reference
    {\bbx@xref{\thefield{xref}}}%
  ...
}
```

当定义`\bbx@xref`命令时，`\DeclareCiteCommand`命令的⟨*precode*⟩, ⟨*postcode*⟩, 和⟨*sepcode*⟩参数留空，是因为上面例子中没有用到。交叉引用由`\bbx@xref`命令的⟨*loopcode*⟩参数打印。更多的关于`xref`域的细节见 § 2.2.3节以及 § 2.4.1节中的注意事项。在 § 4.6.2节我们也看到了`\iffieldxref`, `\iflistxref`, 和`\ifnamexref`测试命令。这些都也可以用 § 4.4.1节的`\entrydata`命令来实现。

```
\ProvidesFile{example.bbx}[2007/06/09 v1.0 biblatex bibliography style]

\DeclareBibliographyDriver{incollection}{%
  ...
  \iffieldundef{xref}
    {...}% code if no cross-reference
    {\entrydata{\thefield{xref}}{%
      % code for cross-references
      ...
    }}%
  ...
```

```
}
```

### 4.11.7 使用标点追踪

**4.11.7.1 标点基础** 样式作者设计参考文献驱动时需要记住一点原则: 块和单元的标点是异步处理的。用例子最容易解释这一点，看下面一段代码:

```
\printfield{title}%
\newunit
\printfield{edition}%
\newunit
\printfield{note}%
```

如果没有edition域，那么这段代码的打印结果不会是:

```
Title. . Note
```

而会是

```
Title. Note
```

因为单元的标点追踪器是异步方式工作的。\newunit命令将不会立即打印标点。它仅是记录一个单元的边界并且将\newunitpunct命令放入标点缓存中。该缓存会有接下来的\printfield、\printlist或类似命令进行处理，且仅当这些命令各自处理的域或列表已定义的时候才会处理。像\printfield这样的命令在插入任何块和单元的标点之前将首先考虑 3 个因素:

- 是否有新的单元/块的输出请求？

  = 前面是否有\newunit或者\newblock命令？

- 前面的命令是否有打印输出？

  = 前面是否有\printfield或者相似命令？

  = 该命令是否实际打印了任何东西？

- 现在是否要打印一些东西？

  要进行打印处理的域或列表是否已定义？

块和单元的标点只会在上述所有条件满足的时候才会输出。让我们再次考虑上面的例子:

```
\printfield{title}%
\newunit
\printfield{edition}%
\newunit
\printfield{note}%
```

如果edition域没有定义会发生什么呢？第一个\printfield命令打印了标题并设置一个内部的‘new text’ 标志。第一个\newunit命令设置一个内部的‘new unit’ 标志。这使没有任何标点输出。第二个\printfield命令不进行任何处理因为edition域未定义。接下来的\newunit命令再次设置‘new unit’ 标志，仍然没有标点输出。第三个\printfield命令检测note域是否已定义，如果是，它会寻找‘new text’ 和‘new unit’ 标志。如果两个标志都存在，那么它会在打印 note 前插入标点缓存。然后它会清除‘new unit’ 标志然后再次设置‘new text’ 标志。

所有这些听起来似乎很复杂，但实际上，这意味着可以用顺序的方式写一个具有很多部件的参考文献驱动。这种方法的优势在不使用标点追踪而实现上述代码功能时会体现的很明显。如果不用标点追踪，那么会因为大量对所有可能存在域的判断产生一个复杂的\iffieldundef判断命令集合。

```
\iffieldundef{title}%
  {\iffieldundef{edition}
     {\printfield{note}}
     {\printfield{edition}%
      \iffieldundef{note}%
        {}
        {. \printfield{note}}}}
  {\printfield{title}%
   \iffieldundef{edition}
     {}
     {. \printfield{edition}}%
   \iffieldundef{note}
     {}
     {. \printfield{note}}}%
```

#### 4.11.7.2 常见错误

把单元的标点处理认为是同步处理的是一个相当常见的误解。这会导致当驱动中包含抄录文本[70]时出现一些典型错误。考虑下面导致标点错位的错误代码段:

```
\printfield{title}%
\newunit
(\printfield{series} \printfield{number})%
```

这段代码将产生下面的结果:

```
Title (. Series Number)
```

这里发生了什么呢？第一个\printfield命令打印了标题，然后\newunit命令标记了一个新的单元边界但不打印任何内容。单元的标点由下一个\printfield命

[70]这里 literal text 理解为原样文本，如实文本，逐字文本，抄录文本，照抄文本

令打印。这是前面提过的异步机制。然而因为左括号在下一个 \printfield命令插入标点前立即打印，所以导致了错误的句点。当插入任何原样文本比如括号(还包括由 \bibopenparen和\mkbibparens命令打印的括号) 时，总需要将这些文本用\printtext命令包起来。要让标点追踪正常运转，需要让驱动知道所有插入的原样文本。这是\printtext命令的作用所在。\printtext命令联系标点追踪器确保标点缓存在原样文本打印前插入。它也设定内部 'new text' 标志。注意本例中还有第三处原样文本即\printfield{series} 后面的空格。在改正的例子中，我们将使用标点追踪器来处理该空格。

```
\printfield{title}%
\newunit
\printtext{(}%
\printfield{series}%
\setunit*{\addspace}%
\printfield{number}%
\printtext{)}%
```

尽管上面的代码能够如常工作，但处理括号、引号和其它包围某个域的标点是，推荐的方式是定义一个域格式:

```
\DeclareFieldFormat{parens}{\mkbibparens{#1}}
```

域格式可以同时用于\printfield和\printtext命令，因此我们可以利用它对若干个域用一堆括号进行包裹。

```
\printtext[parens]{%
  \printfield{series}%
  \setunit*{\addspace}%
  \printfield{number}%
}%
```

这里我们还需要处理没有 series 信息时的情况，因此进一步改进代码如下:

```
\iffieldundef{series}
  {}
  {\printtext[parens]{%
     \printfield{series}%
     \setunit*{\addspace}%
     \printfield{number}}}%
```

最后的一点提示: 本地化字符串对于标点追踪器来说不是原样文本。因为\bibstring和相似命令能联系标点追踪器，因此就不需要用\printtext包裹起来。

**4.11.7.3 高级用法** 标点追踪器也可用来处理更复杂的情况。比如，考虑需要对location、publisher和 year根据数据是否提供以如下的格式打印：

```
...text. Location: Publisher, Year. Text...
...text. Location: Publisher. Text...
...text. Location: Year. Text...
...text. Publisher, Year. Text...
...text. Location. Text...
...text. Publisher. Text...
...text. Year. Text...
```

这个问题可以用一个相当复杂的\iflistundef和\iffieldundef判断集进行处理，通过这些判断可以确定所有可能的域的组合：

```
\iflistundef{location}
  {\iflistundef{publisher}
     {\printfield{year}}
     {\printlist{publisher}%
      \iffieldundef{year}
        {}
        {, \printfield{year}}}}
  {\printlist{location}%
   \iflistundef{publisher}%
     {\iffieldundef{year}
        {}
        {: \printfield{year}}}
     {: \printlist{publisher}%
      \iffieldundef{year}
        {}
        {, \printfield{year}}}}%
```

可以应用\ifthenelse命令和 § 4.6.3讨论的布尔运算可使上面的代码更具可读性。但本质上是一样的。然而，也可以按顺序写成如下方式:

```
\newunit
\printlist{location}%
\setunit*{\addcolon\space}%
\printlist{publisher}%
\setunit*{\addcomma\space}%
\printfield{year}%
\newunit
```

在实际使用中，你会经常使用标点追踪器执行一些显式或隐式的组合判断，比如，考虑如下格式 (注意当没有 publisher 时 location 后面的标点)

```
...text. Location: Publisher, Year. Text...
...text. Location: Publisher. Text...
...text. Location, Year. Text...
...text. Publisher, Year. Text...
...text. Location. Text...
...text. Publisher. Text...
...text. Year. Text...
```

这可以用如下代码进行处理:

```
\newunit
\printlist{location}%
\iflistundef{publisher}
  {\setunit*{\addcomma\space}}
  {\setunit*{\addcolon\space}}%
\printlist{publisher}%
\setunit*{\addcomma\space}%
\printfield{year}%
\newunit
```

因为当没有 publisher 时 location 后面的标点的特殊性，我们需要用一个`\iflistundef`判断来确保正确性。剩下其它的则有标点追踪器处理。

### 4.11.8 本地化定制模型

样式规则可能包含某些规定，比如像'edition' 之类的字符串怎么缩写或者需要表示成一些固定的形式。例如MLA样式指南要求作者在参考文献表的标题中使用'Works Cited' 而不是'Bibliography' 或'References'。本地化命令比如 § 3.9节的`\DefineBibliographyStrings`可以在`cbx`和`bbx`文件中处理这些情况。然而，重载样式文件及其翻译内容是相当不便的。这是§4.9.1节的`\DeclareLanguageMapping`命令发挥作用的地方。这一命令将一个`lbx`文件映射为某一`babel`/`polyglossia`语言的替代翻译。例如，可以创建一个名为`french-humanities.lbx` 的文件提供适用于人文学科的法语翻译并在导言区或者配置文件中将其映射为`babel`/`polyglossia`语言`french`。

```
\DeclareLanguageMapping{french}{french-humanities}
```

如果正文的语言设置为 `french`，`french-humanities.lbx` 将替换`french.lbx`。回到前述的MLA示例，一个MLA样式可能带有一个`american-mla.lbx` 文件来提供符合MLA样式规则的字符串。它将在`cbx`和/或 `bbx` 文件中声明如下映射:

```
\DeclareLanguageMapping{american}{american-mla}
```

因为替换的lbx文件可以从标准的american.lbx 模型中继承字符串，所以american-mla.lbx 可以简化为:

```
\ProvidesFile{american-mla.lbx}[2008/10/01 v1.0 biblatex localization]
\InheritBibliographyExtras{american}
\DeclareBibliographyStrings{%
  inherit          = {american},
  bibliography     = {{Works Cited}{Works Cited}},
  references       = {{Works Cited}{Works Cited}},
}
\endinput
```

替换的lbx文件必须保证本地化模型是完整的。这可以通过从相应的标准模型中继承来实现。如果语言 american 映射为american-mla.lbx，Biblatex 将不会加载american.lbx 触发该模型被明确要求加载。在上述示例中，继承'strings' 和'extras' 将导致 Biblatex 在american-mla.lbx 中应用修改前加载american.lbx。

注意:\DeclareLanguageMapping不用于处理语言的变体 (比如 American English 和 British English)) 或者babel/polyglossia语言别名 (比如 USenglish vs. american)。例如，babel/polyglossia提供了 USenglish 选项类似于 american。因此 Biblatex 附带一个USenglish.lbx 文件，该文件简单的从american.lbx 文件中继承所有数据 (而该文件又从english.lbx 文件中获取字符串)。换句话说，语言变体和babel/polyglossia 语言别名的映射发射在文件层，因此 Biblatex 的语言支持可以简单地增加额外的lbx文件来实现拓展。没有必要进行集中的映射，如果需要支持比如 Portuguese (babel/polyglossia: portuges)，可以创建一个名为portuges.lbx 的文件。如果babel/polyglossia提供一个名为 brasil 的别名，可以创建一个brasil.lbx 文件并可从portuges.lbx 中继承数据。相比之下，\DeclareLanguageMapping主要用于处理文体上的变化，像'humanities 对 natural sciences' or 'MLA 对 APA' 等，这通常是建立在现有的lbx文件基础上的。

### 4.11.9 编组

在标注和著录样式中，可能需要设置标志或保存一些值以便后面使用。这种情况下，理解宏包执行的基本编组结构很关键。一条经验法则是，无论诸如 § 4.6节讨论的样式作者命令是否存在，所有的工作都是在一个大的编组内进行的，因为本宏包的作者接口都是局部的。当存在参考文献数据时，至少存在一个额外的编组，下面是一些基本规则:

- 由\printbibliography或类似命令打印的整个文献表是在一个编组中处理。表中的每个条目都是在一个额外的编组中，这一编组包含了\defbibenvironment的⟨*item code*⟩和所有的驱动代码。

- 由\printbiblist命令打印的整个文献表是在一个编组中处理。表中的每个条目都是在一个额外的编组中，这一编组包含了\defbibenvironment的⟨*item code*⟩和所有的驱动代码。
- 所有由\DeclareCiteCommand命令定义的标注命令都是在一个编组中处理，该编组包含由⟨*precode*⟩, ⟨*sepcode*⟩, ⟨*loopcode*⟩, 和⟨*postcode*⟩等参数构成的完整标注代码。每词执行的⟨*loopcode*⟩都包含在一个额外的编组中。如果指定了任何的⟨*wrapper*⟩，包含⟨*wrapper*⟩代码和标注代码的整个单元都在一个额外的编组中。
- 除了由\DeclareCiteCommand定义的所有后端命令会产生编组外，所有的'autocite' 和 'multicite' 定义也都会产生一个额外的编组。
- \printfile, \printtext, \printfield, \printlist, 和\printnames命令也形成编组。这意味着所有的格式化指令都是在它们自身的编组中处理。
- 所有的lbx文件都是在一个编组中加载和处理。如果一个lbx文件包含的代码不是\DeclareBibliographyExtras的一部分，那么该定义是全局的。

注意: 在标注和著录样式中使用\aftergroup是不可靠的，因为在一定环境中应用的编组的精确层数在宏包的未来版本中可能发生变化。上述说明中如果说某些东西在一个编组中处理，这意味着至少存在一个编组，也可能存在多层嵌套的编组。

### 4.11.10 命名空间

为减小命名冲突的风险，LaTeX 宏包通常在其内部宏名前加上一个代表该宏包的短字符串。例如: 如果foobar宏包需要一个内部使用的宏，它通常会命名为\FB@macro或\foo@macro而不是\macro or \@macro。下面是 Biblatex 使用或推荐使用的前缀字符串:

blx 所有的宏名像\blx@name的宏严格作为内部使用。这也应用于计数器名，长度名，布尔开关等。这些宏可能会以非后向兼容的方式改变，它们可能会重命名设置删除掉而不会有更多的说明。这种改变也不会在版本修改历史和版本发布信息中出现。简而言之: 不要在任何样式中使用以 blx 字符串开头的宏。

abx 以 abx 为前缀的宏也是内部使用，但会相当稳定。但仍应优先使用正式的作者接口提供的工具，当有些情况下使用某一 abx 宏可能会比较方便。

bbx 建议在著录样式中内部使用的宏名的前缀

cbx 建议在标注样式中内部使用的宏名的前缀

lbx 建议在本地化模型中内部使用的宏名的前缀。本地化模型需要添加第二个前缀来指定语言，比如一个为西班牙语本地化模型定义的内部宏可以命名为\lbx@es@macro。

# 附录 Appendix

## A 驱动层的默认数据源映射

These are the driver default source mappings.

### A.1 bibtex

The bibtex driver is of course the most comprehensive and mature of the Biblatex/Biber supported data formats. These source mapping defaults are how the aliases from sections § 2.1.2 and § 2.2.5 are implemented.

```
\DeclareDriverSourcemap[datatype=bibtex]{
  \map{
    \step[typesource=conference, typetarget=inproceedings]
    \step[typesource=electronic, typetarget=online]
    \step[typesource=www,        typetarget=online]
  }
  \map{
    \step[typesource=mastersthesis, typetarget=thesis, final]
    \step[fieldset=type,            fieldvalue=mathesis]
  }
  \map{
    \step[typesource=phdthesis, typetarget=thesis, final]
    \step[fieldset=type,        fieldvalue=phdthesis]
  }
  \map{
    \step[typesource=techreport, typetarget=report, final]
    \step[fieldset=type,         fieldvalue=techreport]
  }
  \map{
    \step[fieldsource=address,       fieldtarget=location]
    \step[fieldsource=school,        fieldtarget=institution]
    \step[fieldsource=annote,        fieldtarget=annotation]
    \step[fieldsource=archiveprefix, fieldtarget=eprinttype]
    \step[fieldsource=journal,       fieldtarget=journaltitle]
    \step[fieldsource=primaryclass,  fieldtarget=eprintclass]
    \step[fieldsource=key,           fieldtarget=sortkey]
    \step[fieldsource=pdf,           fieldtarget=file]
  }
}
```

# B 默认继承设置 Default Inheritance Setup

The following table shows the Biber cross-referencing rules defined by default. Please refer to §§ 2.4.1 和 4.5.11 for explanation.

| Types | | Fields | |
|---|---|---|---|
| **Source** | **Target** | **Source** | **Target** |
| * | * | ids | – |
| | | crossref | – |
| | | xref | – |
| | | entryset | – |
| | | entrysubtype | – |
| | | execute | – |
| | | label | – |
| | | options | – |
| | | presort | – |
| | | related | – |
| | | relatedoptions | – |
| | | relatedstring | – |
| | | relatedtype | – |
| | | shorthand | – |
| | | shorthandintro | |
| | | sortkey | |
| mvbook, book | inbook, bookinbook, suppbook | author | author |
| | | author | bookauthor |
| mvbook | book, inbook, bookinbook, suppbook | title | maintitle |
| | | subtitle | mainsubtitle |
| | | titleaddon | maintitleaddon |
| | | shorttitle | – |
| | | sorttitle | – |
| | | indextitle | – |
| | | indexsorttitle | – |
| mvcollection, mvreference | collection, reference, incollection, inreference, suppcollection | title | maintitle |
| | | subtitle | mainsubtitle |
| | | titleaddon | maintitleaddon |
| | | shorttitle | – |
| | | sorttitle | – |
| | | indextitle | – |
| | | indexsorttitle | – |
| mvproceedings | proceedings, inproceedings | title | maintitle |
| | | subtitle | mainsubtitle |
| | | titleaddon | maintitleaddon |
| | | shorttitle | – |
| | | sorttitle | – |
| | | indextitle | – |
| | | indexsorttitle | – |
| book | inbook, bookinbook, suppbook | title | booktitle |
| | | subtitle | booksubtitle |
| | | titleaddon | booktitleaddon |
| | | shorttitle | – |
| | | sorttitle | – |
| | | indextitle | – |
| | | indexsorttitle | – |

| Types | | Fields | |
|---|---|---|---|
| **Source** | **Target** | **Source** | **Target** |
| collection, reference | incollection, inreference, suppcollection | title | booktitle |
| | | subtitle | booksubtitle |
| | | titleaddon | booktitleaddon |
| | | shorttitle | – |
| | | sorttitle | – |
| | | indextitle | – |
| | | indexsorttitle | – |
| proceedings | inproceedings | title | booktitle |
| | | subtitle | booksubtitle |
| | | titleaddon | booktitleaddon |
| | | shorttitle | – |
| | | sorttitle | – |
| | | indextitle | – |
| | | indexsorttitle | – |
| periodical | article, suppperiodical | title | journaltitle |
| | | subtitle | journalsubtitle |
| | | shorttitle | – |
| | | sorttitle | – |
| | | indextitle | – |
| | | indexsorttitle | – |

# C 默认的排序方式

## C.1 Alphabetic Schemes 1

The following table shows the standard alphabetic sorting schemes defined by default. Please refer to § 3.5 for explanation.

| Option | Sorting scheme | | | | |
|---|---|---|---|---|---|
| nty | presort | →sortname | →sorttitle | →sortyear | →volume |
| | ↪mm | ↪author | ↪title | ↪year | |
| | | ↪editor | | | |
| | | ↪translator | | | |
| | | ↪sorttitle | | | |
| | | ↪title | | | |
| nyt | presort | →sortname | →sortyear | →sorttitle | →volume |
| | ↪mm | ↪author | ↪year | ↪title | |
| | | ↪editor | | | |
| | | ↪translator | | | |
| | | ↪sorttitle | | | |
| | | ↪title | | | |
| nyvt | presort | →sortname | →sortyear | →volume | →sorttitle |
| | ↪mm | ↪author | ↪year | | ↪title |
| | | ↪editor | | | |
| | | ↪translator | | | |
| | | ↪sorttitle | | | |
| | | ↪title | | | |
| all | presort | →sortkey | | | |
| | ↪mm | | | | |

## C.2 Alphabetic Schemes 2

The following table shows the alphabetic sorting schemes for `alphabetic` styles defined by default. Please refer to § 3.5 for explanation.

| Option | Sorting scheme | | | | | |
|---|---|---|---|---|---|---|
| anyt | presort<br>↪ mm | → labelalpha | → sortname<br>↪ author<br>↪ editor<br>↪ translator<br>↪ sorttitle<br>↪ title | → sortyear<br>↪ year | → sorttitle<br>↪ title | → volume |
| anyvt | presort<br>↪ mm | → labelalpha | → sortname<br>↪ author<br>↪ editor<br>↪ translator<br>↪ sorttitle<br>↪ title | → sortyear<br>↪ year | → volume | → sorttitle<br>↪ title |
| all | presort<br>↪ mm | → labelalpha | → sortkey | | | |

## C.3 Chronological Schemes

The following table shows the chronological sorting schemes defined by default. Please refer to § 3.5 for explanation.

| Option | Sorting scheme | | | |
|---|---|---|---|---|
| ynt | presort<br>↪ mm | → sortyear<br>↪ year<br>↪ 9999 | → sortname<br>↪ author<br>↪ editor<br>↪ translator<br>↪ sorttitle<br>↪ title | → sorttitle<br>↪ title |
| ydnt | presort<br>↪ mm | → sortyear (desc.)<br>↪ year (desc.)<br>↪ 9999 | → sortname<br>↪ author<br>↪ editor<br>↪ translator<br>↪ sorttitle<br>↪ title | → sorttitle<br>↪ title |
| all | presort<br>↪ mm | → sortkey | | |

# D `biblatexml`

The `biblatexml` XML datasource format is designed to be an extensible and modern data source format for Biblatex users. There are limitations with BibTeX format `.bib` files, in particular one might mention UTF-8 support and name formats. Biber goes some way to addressing the UTF-8 limitations by using a modified version of the `btparse` C library but the rather archaic name parsing rules for BibTeX are hard-coded and specific to simple Western names.

`biblatexml` is an XML format for bibliographic data. When Biber either reads or writes `biblatexml` format datasources, it automatically writes a RelaXNG XML schema for the datasources which is dynamically generated from the active Biblatex datamodel. There is no static schema for `biblatexml` datasources because the allowable fields etc. depend on the data model. The format of `biblatexml` datasources is relatively self-explanatory—it is usually only necessary to generate a `biblatexml` datasource from existing BibTeX format datasources (using Biber's 'tool' mode) in order to understand the format. Biber also allows users to validate `biblatexml` datasources against the data model generated schema.

Since the `biblatexml` format is XML and depends on the data model and the data model is extensible by the user (see § 4.5.4), the `biblatexml` format can deal with extensions that BibTeX format data sources cannot, e.g. new nameparts, options at sub-entry scope. Since it is an XML format, it is relatively easy to transform it into other XML formats or HTML using standard XML processing libraries and tools.

Here is an explanation of the format with examples. By convention, `biblatexml` files have a `.bltxml` extension and `kpsewhich` understands this file extension.

## D.1 Header

`biblatexml` files begin with the standard XML header:

```
<?xml version="1.0" encoding="UTF-8"?>
```

The schema model, type and schema type namespace are given in the following line:

```
<?xml-model href="biblatexml.rng"
            type="application/xml"
            schematypens="http://relaxng.org/ns/structure/1.0"?>
```

When Biber generates `biblatexml` data sources, it automatically adds this line and points the schema model (href) attribute at the automatically generated RelaXNG XML schema for ease of validation.

## D.2 Body

The body of a `biblatexml` data source looks like:

```
<bltx:entries
  xmlns:bltx="http://biblatex-biber.sourceforge.net/biblatexml">

  <bltx:entry id="" entrytype="">
  </bltx:entry>
       .
       .
       .
```

```
  <bltx:entry id="" entrytype="">
  </bltx:entry>

</bltx:entries>
```

The body is one or more `entry` elements inside the top-level `entries` element and everything is in the `bltx` namespace. An entry has an `id` attribute corresponding to a BibTeX entry key and a `entrytype` attribute corresponding to a BibTeX entrytype. For example, the `biblatexml`

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-model href="biblatexml.rng"
            type="application/xml"
            schematypens="http://relaxng.org/ns/structure/1.0"?>
<bltx:entries
  xmlns:bltx="http://biblatex-biber.sourceforge.net/biblatexml">
  <bltx:entry id="key1" entrytype="book">
  </bltx:entry>
</bltx:entries>
```

Corresponds to the BibTeX `.bib`

```
@book{key1,
}
```

In general, the XML elements in a `biblatexml` format datasource file have names corresponding to the fields in the datamodel, just like BibTeX format datasources. So for example, the BibTeX format source

```
@book{key1,
  TITLE = {...},
  ISSUE = {...},
  NOTE = {...}
}
```

would be, in `biblatexml`

```
  <bltx:entry id="key1" entrytype="book">
    <bltx:title>...</bltx:title>
    <bltx:issue>...</bltx:issue>
    <bltx:note>...</bltx:note>
  </bltx:entry>
```

The following exceptions to this simple mapping are to be noted

### D.2.1 Key aliases

Citation key aliases are specified like this:

```
<bltx:ids>
  <bltx:key>alias1</bltx:key>
  <bltx:key>alias2</bltx:key>
</bltx:ids>
```

this corresponds to the BibTeX format

```
@book{key1,
  IDS = {alias1,alias2}
}
```

### D.2.2 Names

Name specifications in `biblatexml` are somewhat more complex in order to generalise the name handling abilities of Biblatex. The user has to be more explicit about the name parts and this allows a much great scope for the handling of different types of names and name parts. A name in `biblatexml` format looks like this

```
<bltx:names type="author" morenames="1" useprefix="true">
  <bltx:name gender="sm">
    <bltx:namepart type="given">
      <bltx:namepart initial="J">John</bltx:namepart>
      <bltx:namepart initial="A">Arthur</bltx:namepart>
    </bltx:namepart>
    <bltx:namepart type="family">Smith</bltx:namepart>
    <bltx:namepart type="prefix" initial="v">von</bltx:namepart>
  </bltx:name>
  <bltx:name useprefix="false">
    <bltx:namepart type="given">
      <bltx:namepart>Raymond</bltx:namepart>
    </bltx:namepart>
    <bltx:namepart type="family">Brown</bltx:namepart>
  </bltx:name>
</bltx:names>
```

A name list field is contained in the `names` element with the mandatory `type` attribute giving the name of the name list. Things to note:

- The optional `morenames` attribute performs the same task as the BibTeX datasource format 'and others' string at the end of a name.

- Note that optional `useprefix` option can be specified can be specified at the level of a name list or an individual name in the name list. This is impossible with BibTeX datasources.
- Individual names may have an optional `gender` attribute which must be one of those defined in the datamodel 'gender' constant list. This is currently not used by standard styles but is available in Biblatex name formats if necessary.
- A name list is composed of one or more `name` elements.
- Each name is composed of name parts of a `type` defined by the data model 'nameparts' constant.
- Each name part may have an option `initial` attribute which makes explicit the initial of the name part. If this is not present, Biber attempts to automatically determine the initial from the name part.
- Name parts may have name parts so that compound names can be handled.

Ignoring the `biblatexml`-only features, a corresponding BibTeX format datasource would look like this:

```
AUTHOR = {von Smith, John Arthur and Brown, Raymond and others}
```

### D.2.3 Lists

Datasource list fields (see § 2.2.1) can be represented in two ways, depending on whether there is more than one element in the list:

```
<bltx:publisher>London</bltx:publisher>
<bltx:location>
  <bltx:item>London</bltx:item>
  <bltx:item>Moscow</bltx:item>
</bltx:location>
```

### D.2.4 Ranges

Datasource range fields (see § 2.2.1) are represented like this:

```
<bltx:pages>
  <bltx:item>
    <bltx:start>1</bltx:start>
    <bltx:end>10</bltx:end>
  </bltx:item>
  <bltx:item>
    <bltx:start>30</bltx:start>
```

```
      <bltx:end>34</bltx:end>
    </bltx:item>
  </bltx:pages>
```

A range field is a list of ranges, each with its own `item`. A range item has a `start` element and an optional `end` element, since ranges can be open-ended.

### D.2.5 Dates

Datasource date fields (see § 2.2.1) can be represented in two ways, depending on whether they constitute a date range:

```
  <bltx:date>1985-04-02</bltx:date>
  <bltx:date type="event">
    <bltx:start>1990-05-16</bltx:start>
    <bltx:end>1990-05-17</bltx:end>
  </bltx:date>
```

The `type` attribute on a date element corresponds to a particular type of date defined in the data model.

### D.2.6 Related Entries

Related entries are specified as follows:

```
  <bltx:related>
    <bltx:item type="reprint"
               ids="rel1,rel2"
               string="Somestring"
               options="skipbiblist"/>
  </bltx:related>
```

This corresponds to the BibTeX format:

```
@book{key1,
  RELATED         = {rel2,rel2},
  RELATEDTYPE     = {reprint},
  RELATEDSTRING   = {Somestring},
  RELATEDOPTIONS  = {skipbiblist}
}
```

As per § 4.5.1, the `string` and `options` attributes are optional.

# E 选项范围 Option Scope

The following table provides an overview of the scope (global/per-type/per-entry) of various package options.

| Option | Scope | | | |
|---|---|---|---|---|
| | Load-time | Global | Per-type | Per-entry |
| abbreviate | • | • | – | – |
| alldates | • | • | – | – |
| alldatesusetime | • | • | – | – |
| alltimes | • | • | – | – |
| arxiv | • | • | – | – |
| autocite | • | • | – | – |
| autopunct | • | • | – | – |
| autolang | • | • | – | – |
| backend | • | – | – | – |
| backref | • | • | – | – |
| backrefsetstyle | • | • | – | – |
| backrefstyle | • | • | – | – |
| bibencoding | • | • | – | – |
| bibstyle | • | – | – | – |
| bibwarn | • | • | – | – |
| block | • | • | – | – |
| citecounter | • | • | – | – |
| citereset | • | • | – | – |
| citestyle | • | – | – | – |
| citetracker | • | • | – | – |
| clearlang | • | • | – | – |
| datamodel | • | – | – | – |
| dataonly | – | – | • | • |
| date | • | • | – | – |
| labeldate | • | • | – | – |
| <datetype>date | • | • | – | – |
| dateabbrev | • | • | – | – |
| datecirca | • | • | – | – |
| dateera | • | • | – | – |
| dateerauto | • | • | – | – |
| dateuncertain | • | • | – | – |
| datezeros | • | • | – | – |
| defernumbers | • | • | – | – |
| doi | • | • | – | – |
| eprint | • | • | – | – |
| <namepart>inits | • | • | – | – |
| gregorianstart | • | • | – | – |
| hyperref | • | • | – | – |
| ibidtracker | • | • | – | – |
| idemtracker | • | • | – | – |
| indexing | • | • | • | • |
| isbn | • | • | – | – |
| julian | • | • | – | – |
| labelalpha | • | • | • | – |
| labelnamefield | – | – | – | • |
| labelnumber | • | • | • | – |
| labeltitle | • | • | • | – |
| labeltitlefield | – | – | – | • |
| labeltitleyear | • | • | • | – |
| labeldateparts | • | • | • | – |
| labeltime | • | • | – | – |
| labeldateusetime | • | • | – | – |
| <datetype>time | • | • | – | – |
| <datetype>dateusetime | • | • | – | – |
| language | • | • | – | – |
| loadfiles | • | • | – | – |

| Option | Scope | | | |
|---|---|---|---|---|
| | Load-time | Global | Per-type | Per-entry |
| loccittracker | • | • | – | – |
| maxalphanames | • | • | • | • |
| maxbibnames | • | • | • | • |
| maxcitenames | • | • | • | • |
| maxitems | • | • | • | • |
| maxnames | • | • | • | • |
| maxparens | • | • | – | – |
| mcite | • | – | – | – |
| minalphanames | • | • | • | • |
| minbibnames | • | • | • | • |
| mincitenames | • | • | • | • |
| mincrossrefs | • | • | – | – |
| minxrefs | • | • | – | – |
| minitems | • | • | • | • |
| minnames | • | • | • | • |
| natbib | • | – | – | – |
| noinherit | – | – | – | • |
| notetype | • | • | – | – |
| opcittracker | • | • | – | – |
| openbib | • | • | – | – |
| pagetracker | • | • | – | – |
| parentracker | • | • | – | – |
| punctfont | • | • | – | – |
| refsection | • | • | – | – |
| refsegment | • | • | – | – |
| safeinputenc | • | • | – | – |
| seconds | • | • | – | – |
| singletitle | • | • | • | – |
| skipbib | – | – | • | • |
| skipbiblist | – | – | • | • |
| skiplab | – | – | • | • |
| sortcase | • | • | – | – |
| sortcites | • | • | – | – |
| sorting | • | • | – | – |
| sortnamekeyscheme | – | – | – | • |
| sortlocale | • | • | – | – |
| sortlos | • | • | – | – |
| sortupper | • | • | – | – |
| style | • | – | – | – |
| terseinits | • | • | – | – |
| texencoding | • | • | – | – |
| timezeros | • | • | – | – |
| timezones | • | • | – | – |
| uniquelist | • | • | • | • |
| uniquename | • | • | • | • |
| uniquetitle | • | • | • | – |
| uniquebaretitle | • | • | • | – |
| uniquework | • | • | • | – |
| uniqueprimaryauthor | • | • | – | – |
| url | • | • | – | – |
| useprefix | • | • | • | • |
| use<name> | • | • | • | • |

# F 更新历史

This revision history is a list of changes relevant to users of this package. Changes of a more technical nature which do not affect the user interface or the behavior of the package are not included in the list. More technical details are to be found in the `CHANGES.org` file. The numbers on the right indicate the relevant section of this manual.

### 3.7 2016-12-08

Corrected default for `\bibdateeraprefix` . . . . . 4.10.2

Added `\DeclareSortInclusion` . . . . . 4.5.6

Added `\relateddelim<relatedtype>` . . . . . 3.10.1

### 3.6 2016-09-15

Corrected some documentation and fixed a bug with labeldate localisation strings.

### 3.5 2016-09-10

Added `\ifuniquebaretitle` test . . . . . 4.6.2

Documented `\labelnamesource` and `\labeltitlesource` . . . . . 4.2.4.1

Added `\bibdaterangesep` . . . . . 3.10.2

Added `refsection` option to `\DeclareSourcemap` . . . . . 4.5.3

Added `suppress` option to inheritance specifications . . . . . 4.5.11

Added `\ifuniquework` . . . . . 4.6.2

Changed `\DeclareStyleSourcemap` so that it can be used multiple times . . . 4.5.3

Added `\forcezerosy` and `\forcezerosmdt` . . . . . 4.10.4

Changed `\mkdatezeros` to `\mkyearzeros`, `\mkmonthszeros` and `\mkdayzeros` . 4.10.4

Added `namehash` and `fullhash` for all name list fields . . . . . 4.2.4.1

Generalised `giveninits` option to all nameparts . . . . . 3.1.2.3

Added `inits` option to `\DeclareSortingNamekeyScheme` . . . . . 4.5.6

Added `\DeclareLabelalphaNameTemplate` . . . . . 4.5.5

Added full EDTF Levels 0 and 1 compliance for parsing and printing times . . 2.3.8

Changed dates to be fully EDTF Levels 0 and 1 compliant. Associated tests and localisation strings . . . . . 2.3.8

Added `timezeros` . . . . . 3.1.2.1

Added `mktimezeros` . . . . . 4.10.4

Changed `iso8601` to `edtf` . . . . . 3.1.2.1

Added `\DeclareUniquenameTemplate` . . . . . 4.11.4

Removed experimental RIS support

sortnamekeyscheme and useprefix can be now be set per-namelist and per-name for BibTeX datasources . . . . . 4.5.6

Added \DeclareDelimcontextAlias . . . . . ??

Added Estonian localisation (Benson Muite)

Reference contexts may now be named . . . . . 3.7.10

Added notfield step in Sourcemaps . . . . . 4.5.3

### 3.4 2016-05-10

Added \ifcrossrefsource and \ifxrefsource . . . . . 4.6.2

Added data annotation feature . . . . . 3.6

Added package option minxrefs . . . . . 3.1.2.1

Added \ifuniqueprimaryauthor and associated global option . . . . . 4.6.2

Added \DeprecateField, \DeprecateList and \DeprecateName . . . . . 4.4.1

Added \ifcaselang . . . . . 4.6.2

Added \DeclareSortTranslit . . . . . 4.5.6

Added uniquetitle test . . . . . 4.6.2

Added \namelabeldelim . . . . . 3.10.1

New starred variants of the \assignrefcontext* macros . . . . . 3.7.10

New context-sensitive delimiter interface . . . . . ??

Moved prefixnumbers option to \newrefcontext and renamed to labelprefix 3.7.10

Added \DeclareDatafieldSet . . . . . 4.5.2

### 3.3 2016-03-01

New macros for auto-assignment of refcontexts . . . . . 3.7.10

Schema documentation for biblatexml . . . . . D

Sourcemapping documentation and examples for biblatexml . . . . . 4.5.3

Changes for name formats to generalise available name parts . . . . . 4.4.2

useprefix can now be specified per-namelist and per-name in biblatexml datasources

New sourcemapping options for creating new entries dynamically and looping over map steps . . . . . 4.5.3

Added noalphaothers and enhanced name range selection in \DeclareLabelalphaTemplate . . . . . 4.5.5

Added \DeclareDatamodelConstant . . . . . 4.5.4

Renamed firstinits and sortfirstinits

Added \DeclareSortingNamekeyScheme . . . . . 4.5.6

Removed messy experimental endnote and zoterordf support for Biber

Added `\nonameyeardelim` . . . . . 3.10.1

Added `\extpostnotedelim` . . . . . 3.10.1

### 3.2 2015-12-28

Added `pstrwidth` and `pcompound` to `\DeclareLabelalphaTemplate` . . . . . 4.5.5

Added `\AtEachCitekey` . . . . . 4.10.6

### 3.1 2015-09

Added `\DeclareNolabel` . . . . . 4.5.5

Added `\DeclareNolabelwidthcount` . . . . . 4.5.5

### 3.0 2015-04-20

Improved Danish (Jonas Nyrup) and Spanish (ludenticus) translations

`labelname` and `labeltitle` are now resolved by Biblatex instead of Biber for more flexibility and future extensibility

New `\entryclone` sourcemap verb for cloning entries during sourcemapping 4.5.3

New `\pernottype` negated per-type sourcemap verb . . . . . 4.5.3

New range calculation command `\frangelen` . . . . . 4.6.4

New bibliography context functionality . . . . . 3.7.10

Name lists in the data model now automatically create internals for `\ifuse<name>` tests and booleans . . . . . 3.1.3.1 and . . . . . 4.6.2

### 2.9a 2014-06-25

`resetnumbers` now allows passing a number to reset to . . . . . 3.7.2

### 2.9 2014-02-25

Generalised shorthands facility . . . . . 3.7.3

Sorting locales can now be defined as part of a sorting scheme . . . . . 4.5.6

Added `sortinithash` . . . . . 4.2.4.1

Added Slovene localisation (Tea Tušar and Bogdan Filipič)

Added `\mkbibitalic` . . . . . 4.10.4

Recommend `begentry` and `finentry` bibliography macros . . . . . 4.2.3

### 2.8a 2013-11-25

Split option `language=auto` into `language=autocite` and `language=autobib` . 3.1.2.1

### 2.8 2013-10-21

New `langidopts` . . . . . 2.2.3

`hyphenation` field renamed to `langid` . . . . . 2.2.3

`polyglossia` support

Renamed `babel` option to `autolang` . . . . . 3.1.2.1

Corrected Dutch localisation

Added `datelabel=year` option . . . . . 3.1.2.1

Added `datelabelsource` field . . . . . 4.2.4.1

### 2.7a 2013-07-14

Bugfix - respect maxnames and uniquelist in `\finalandsemicolon`

Corrected French localisation

### 2.7 2013-07-07

Added field `eventtitleaddon` to default datamodel and styles . . . . . 2.2.2

Added `\ifentryinbib`, `\iffirstcitekey` and `\iflastcitekey` . . . . . 4.6.2

Added `postpunct` special field, documented `multiprenote` and `multipostnote` special fields . . . . . 4.3.2

Added `\UseBibitemHook`, `\AtEveryMultiCite`, `\AtNextMultiCite`, `\UseEveryCiteHook`, `\UseEveryCitekeyHook`, `\UseEveryMultiCiteHook`, `\UseNextCiteHook`, `\UseNextCitekeyHook`, `\UseNextMultiCiteHook`, `\DeferNextCitekeyHook` . . . . . 4.10.6

Fixed `\textcite` and related commands in the numeric and verbose styles . . 3.8.2

Added multicite variants of `\volcite` and related commands . . . . . 3.8.6

Added `\finalandsemicolon` . . . . . 3.10.2

Added citation delimiter `\textcitedelim` for `\textcite` and related commands to styles . . . . . 4.10.1

Updated Russian localisation (Oleg Domanov)

Fixed Brazilian and Finnish localisation

### 2.6 2013-04-30

Added `\printunit` . . . . . 4.7.1

Added field `clonesourcekey` . . . . . 4.2.4.1

New options for `\DeclareLabelalphaTemplate` . . . . . 4.5.5

Added `\DeclareLabeldate` and retired `\DeclareLabelyear` . . . . . 4.5.10

Added `nodate` localisation string . . . . . 4.9.2.14

Added `\rangelen` . . . . . 4.6.4

Added starred variants of `\citeauthor` and `\Citeauthor` . . . . . 3.8.5

Restored original `url` format. Added `urlfrom` localisation key . . . . . 4.9.2.15

Added `\AtNextBibliography` . . . . . 4.10.6

Fixed related entry processing to allow nested and cyclic related entries

Added Croatian localisation (Ivo Pletikosić)

Added Polish localisation (Anastasia Kandulina, Yuriy Chernyshov)

Fixed Catalan localisation

Added smart “of” for titles to Catalan and French localisation

Misc bug fixes

### 2.5 2013-01-10

Made `url` work as a localisation string, defaulting to previously hard-coded value ‘URL’.

Changed some Biber option names to cohere with Biber 1.5.

New sourcemap step for conditionally removing entire entries . . . . . 4.5.3

Updated Catalan localisation (Sebastià Vila-Marta)

### 2.4 2012-11-28

Added `relatedoptions` field . . . . . 4.5.1

Added `\DeclareStyleSourcemap` . . . . . 4.5.3

Renamed `\DeclareDefaultSourcemap` to `\DeclareDriverSourcemap` . . . . . 4.5.3

Documented `\DeclareFieldInputHandler`, `\DeclareListInputHandler` and `\DeclareNameInputHandler`.

Added Czech localisation (Michal Hoftich)

Updated Catalan localisation (Sebastià Vila-Marta)

### 2.3 2012-11-01

Better detection of situations which require a Biber or LaTeX re-run

New append mode for `\DeclareSourcemap` so that fields can be combined . . 4.5.3

Extended auxiliary indexing macros

Added support for plural localisation strings with `relatedtype` . . . . . 4.5.1

Added `\csfield` and `\usefield` . . . . . 4.6.1

Added starred variant of `\usebibmacro` . . . . . 4.6.4

Added `\ifbibmacroundef`, `\iffieldformatundef`, `\iflistformatundef` and `\ifnameformatundef` . . . . . 4.6.4

Added Catalan localisation (Sebastià Vila-Marta)

Misc bug fixes

### 2.2 2012-08-17

Misc bug fixes

Added `\revsdnamepunct` . . . . . 3.10.1

Added `\ifterseinits` . . . . . 4.6.2

### 2.1 2012-08-01

Misc bug fixes

Updated Norwegian localisation (Håkon Malmedal)

Increased data model auto-loading possibilities . . . . . 4.5.4

### 2.0 2012-07-01

Misc bug fixes

Generalised `singletitle` test a little . . . . . 4.6.2

Added new special field `extratitleyear` . . . . . 4.2.4

Customisable data model . . . . . 4.5.4

Added `\DeclareDefaultSourcemap` . . . . . 4.5.3

Added `labeltitle` option . . . . . 3.1.2.3

Added new special field `extratitle` . . . . . 4.2.4

Made special field `labeltitle` customisable . . . . . 4.2.4

Removed field `reprinttitle` . . . . . 3.4

Added related entry feature . . . . . 3.4

Added `\DeclareNoinit` . . . . . 4.5.8

Added `\DeclareNosort` . . . . . 4.5.9

Added `sorting` option for `\printbibliography` and `\printshorthands` . . . 3.7.2

Added `ids` field for citekey aliasing . . . . . 2.2

Added `sortfirstinits` option . . . . . 3.1.2.3

Added data stream modification feature . . . . . 4.5.3

Added customisable labels feature . . . . . 4.5.5

Added `\citeyear*` and `\citedate*` . . . . . 3.8.5