张伟. 机读目录数据搜索引擎设计与实现[J]. 智能计算机与应用,2024,14(10):126-130. DOI:10.20169/j.issn.2095-2163.241017

机读目录数据搜索引擎设计与实现

张 伟
(国家图书馆, 北京 100081)

摘 要: 机读目录(机器可读目录)数据是一种表示与交流书目、规范、馆藏及相关信息的元数据。作为关键性生产要素,机读目录数据在图书情报界被广泛使用,并在跨机构的信息共享方面发挥着重要作用。为构建高性能、低成本的机读目录数据搜索环境,本文提出一种基于 Meilisearch 的全文搜索解决方案。该方案灵活、务实且高效,可轻松为任何机读目录数据集按需定制数据获取服务,并为开发人员和最终用户提供简单直观的搜索体验。

关键词: 机读目录; Meilisearch; 搜索引擎
中图分类号: TP391.3 **文献标志码**: A **文章编号**: 2095-2163(2024)10-0126-05

Design and implementation of machine-readable cataloging data search engine

ZHANG Wei
(National Library of China, Beijing 100081, China)

Abstract: MARC (MAchine-Readable Cataloging) data is a type of metadata for the representation and communication of bibliographic, authority, holdings and related information. As a key factor of production, MARC data is widely used in the library and information community. Subsequently, MARC data plays an important role in interagency information sharing. In order to build a high-performance and cost-effective MARC data search environment, this paper proposes a Meilisearch-based full-text search solution. This solution, which is flexible, pragmatic and efficient, makes it easy to tailor data access services for any MARC data set on demand and deliver a simple and intuitive search experience for both developers and end-users.

Key words: MARC; Meilisearch; search engine

0 引 言

数字化时代,日新月异的信息技术持续推动着文献资源数字化进程,使机读目录数据得以不断产生、积累和交换,进而为图书情报机构之间的信息资源共建与共享奠定基础。随着数据被纳入生产要素范畴,数据服务应用场景也逐渐丰富。

针对以多源异构的机读目录数据为原始数据,面向技术、编目和审校等业务相关者,灵活构建数据获取服务的场景,本文提出一种需求导向的基于搜索引擎新秀 Meilisearch 的解决方案。该方案可根据业务需求,在指定字段上建立索引,并通过后端接口和前端界面提供服务。其设计初衷旨在为某专题或类别的书目搭建搜索环境,但对任何平面文件数据源同样适用。

1 机读目录格式

1.1 MARC 格式

机读目录(MARC)是随着计算机自动化的发展而研制出来的一种计算机可识别和处理的数据格式[1]。一条 MARC 记录由记录头标、地址目次区、数据字段区、记录结束符四部分组成。这里将展开分述如下。

(1)记录头标。含有长度、状态、类型等记录特征相关信息,用于满足记录处理的需要。

(2)地址目次区。由若干目次项组成,用于标识记录中每个字段的字段号、字段长度、字段起始位置。目次区中的每个目次项描述一个字段,以字段分隔符结尾。

(3)数据字段区。包含控制字段和数据字段两种形式的数据,用于存储记录除头标、字段号以外的

作者简介: 张 伟(1981-),男,硕士,馆员,主要研究方向:数据分析。Email:zhv@nlc.cn

收稿日期: 2023-06-13

内容。控制字段由字段内容和字段分隔符组成，无字段指示符和子字段。数据字段由字段指示符、子字段标识符、子字段内容、字段分隔符组成。字段指示符在每个数据字段首个子字段分隔符之前，为字段提供附加信息。子字段标识符在子字段开头，由子字段分隔符和子字段代码组成，标识字段中不同的子字段。

(4)记录结束符。位于记录结尾，标识一条记录的终止。

1.2　MARCXML 格式

MARCXML 是通过 XML 表现机读目录的数据交换格式，使用各种元素、属性表示记录头标、字段和指示符等概念，对 MARC 保持语义兼容。MARCXML 使用树形结构组织数据，因为其中各种元素从根部开始延展，形成的结构类似于自然界中的树。

格式良好的 MARCXML 文档的首行是 XML 声明，说明 XML 的版本和所使用的编码。为避免元素命名冲突，顶层元素通常带有命名空间。命名空间在顶层元素开始标签的 xmlns 属性中被定义，格式为：xmlns：前缀＝"URI"。带有相同前缀的子元素属于同一命名空间。省略前缀表示使用默认命名空间，使得子元素也不必带有前缀。URI 仅为命名空间赋予惟一标识，不会被解析器用于查找信息。文档中以 XSI(XML Schema Instance)为前缀的命名空间一般为固定写法，其属性 schemaLocation 定义了命名空间和对应 XSD 文档位置的关系。

MARCXML 记录集的根元素为 collection，即包含一条或多条记录的顶层元素，其中每条记录主要含有 5 种元素：

(1)record 为记录顶层元素，包含组成一条记录的记录头标和全部字段元素。

(2)leader 对应记录头标。MARC 记录头标中的记录长度和数据基地址对于此格式没有意义，但其位置仍被保留以保持兼容。

(3)controlfield 对应控制字段，包含 tag 属性，属性值为字段标识符。

(4)datafield 对应数据字段，包含 subfield 元素，以及 tag、ind1、ind2 三个属性，属性值分别为字段标识符、字段指示符 1、字段指示符 2。

(5)subfield 对应子字段，包含 code 属性，属性值为子字段代码。

MARCXML 记录中可能存在一些特殊字符组合：<、&、>、'、"。这些组合是 XML 中的预定义实体，分别表示<(小于号)、&(和号)、>(大于号)、'(单引号)、"(双引号)。严格来讲，在 XML 中只有<和 & 字符必须通过所对应的预定义实体表示，因为<表示元素开始，& 表示实体开始，但使用实体来代替这些特殊字符是较为规范的做法。在记录处理过程中，XML 解析器会将预定义实体转换为对应字符，恢复其原貌。

MACRXML 提供了一种独立于软件和硬件的机读目录数据存储、传输和共享方式。XML 的特点决定了 MARCXML 具有相对更好的描述性和扩展性。

2　Meilisearch 概述

2.1　Meilisearch 简介

Meilisearch 是由一家法国初创公司推出的高性能搜索引擎，旨在提供开箱即用的解决方案[2]。2023 年 2 月，开发团队发布了首个完全稳定、向前兼容和企业就绪的版本。与同类产品相比，使用 Rust 语言开发的 Meilisearch 具有完全开源、易于部署、便于维护和高度可定制等优势，其默认设置即可满足许多项目的需要。

作为一款以用户体验为导向的产品，Meilisearch 内置多种灵活实用的功能，使开发人员无需繁琐的配置即可为应用、网站和工作流集成搜索引擎，进而带给最终用户流畅的搜索体验，使其在极短的响应时间内即可获取预期结果。

2.2　基础术语与概念

Meilisearch 中以一个或多个字段形式存储和组织数据的对象称为文档。字段是由属性及其关联值组配而成的数据项。字段属性用于命名、描述和访问其关联值，字段值是任何有效的 JSON 格式数据。主字段是一种特殊字段，必须出现在所有文档中，用于标识、更新和删除文档。主字段属性是主键，值是文档唯一标识符。在默认设置下，文档中的所有字段均为可搜索字段，并且位置靠前的字段具有更高权重。

索引是一组具有关联设置的文档，由索引唯一标识符定义，包含一个主键、若干可定制的参数以及任意数量的文档。只有被添加到索引，文档才能用于搜索。索引数量和大小的上限取决于操作系统分配给单个进程的内存地址空间。

Meilisearch 在许多方面相当于一个数据库。其在底层采用 LMDB(Lightning Memory-mapped DataBase)作为存储引擎来保存已索引的文档以及与搜索结果相关的数据。LMDB 使用内存映射文件让整个数据库看起来就像在主存中一样[3]，既具有纯内存数据库的读取性能，也保留了基于磁盘数据库的持久性。得益于 LMDB 的优异特性，Meilisearch 在存储

性能方面具有良好表现。

2.3 搜索相关性

搜索相关性是指搜索结果的准确性和有效性，确保最佳匹配结果被优先呈现。在各种调整与优化相关性的措施中，6 个内置排序规则最为重要。详述如下。

(1) Words：结果按照关键词的数量递减排列，即匹配更多关键词的文档优先。

(2) Typo：结果按照关键词的错别字数量递增排列，即匹配关键词且其中错别字更少的文档优先。

(3) Proximity：结果按照关键词之间的距离递增排列，即关键词依次出现且相距更近的文档优先。

(4) Attribute：结果按照可搜索字段的顺序排列，即关键词出现于更高权重字段的文档优先。

(5) Sort：结果按照用户在搜索时指定的字段顺序排列。

(6) Exactness：结果按照词语相似度排列，即包含与关键词完全相同词语的文档优先。

排序规则的权重逐条降低。首个规则适用于所有文档，而每个后续规则仅适用于被前一规则视为相等的文档。对于许多应用场景，默认配置足以适用，但用户仍可通过增减条目与调整顺序来满足个性化需求。

2.4 主密钥与 API 密钥

为防范违规操作、越权访问或人为错误等引发的数据安全风险，Meilisearch 采用 2 种类型的安全密钥作为加固措施：一个主密钥和若干 API 密钥。

主密钥是一种自定义安全凭证，用于保护实例免遭未经授权的使用，以及创建、列出、更新和删除具有细粒度权限的 API 密钥。在生产环境中，Meilisearch 无法脱离主密钥启动和运行。

API 密钥授予用户对指定索引、路由和端点的访问权，含有具体的权限范围和过期时间等。为便于应用，Meilisearch 在首次启动时会自动生成默认搜索和管理密钥。这 2 个默认密钥通常能够满足大多数应用程序的安全需求，确保只有授权用户才能执行相应任务。与主密钥相同，API 密钥在生产环境中也是必备项。

3 数据搜索引擎的构建

3.1 系统架构设计

机读目录数据搜索引擎架构如图 1 所示。由图 1 可知，主要由数据集成、Meilisearch 和前端搜索界面三个关联服务组成，并通过 Docker 进行容器化部署。Docker 是一个能够把开发的应用自动部署到容器的开源引擎[4]，有助于将一个复杂系统分解成一系列可组合的部分，让微服务架构成为可能[5]。

数据集成采用自动批处理方式，将 MARC 或 MARCXML 格式的记录通过一组 Python 脚本实现的机读目录数据管道转换为跨平台、跨语言的数据交换格式，然后将其作为文档添加到 Meilisearch 的预建索引。

在完成初始化和对文档的索引后，Meilisearch 通过 REST(REpresentational State Transfer)风格的接口提供数据服务。初始化即创建索引与配置参数的过程。通常，对于大多数用例，微调个别参数即可满足要求。可筛选字段、可排序字段和排序规则是其中最常用的索引参数，可帮助用户优化搜索结果。就中文书目(CNMARC)的索引而言，一种调参方案如下：

```
curl -X PATCH 'http://your-domain/indexes/cnmarc/settings' -H 'Content-Type: application/json' -H 'Authorization: Bearer API-KEY' --data-binary '{"filterableAttributes":["200b","210c","210d"],"sortableAttributes":["210d"],"rankingRules":["words","sort","typo","proximity","attribute","exactness"]}'
```

前端搜索界面负责人机交互与数据展示，基于开源用户界面库 InstantSearch.js[6] 及与之建立通信的 Meilisearch 插件 instant-meilisearch[7] 快速成型，并通过高性能 Web 服务器 nginx[8] 处理访问请求。

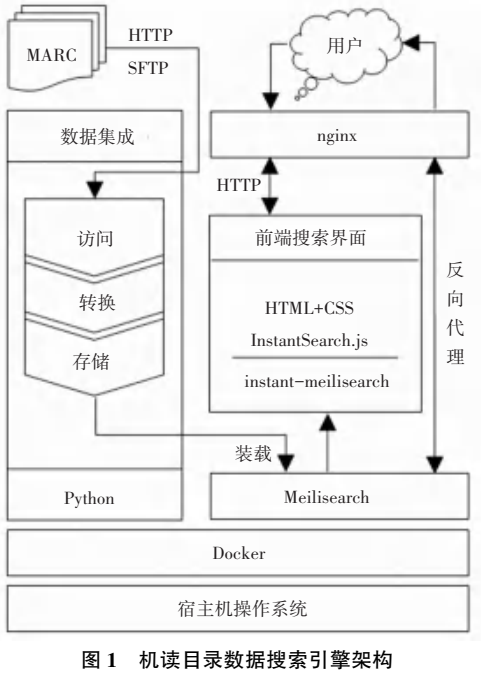

图 1 机读目录数据搜索引擎架构

Fig. 1 MARC data search engine architecture

3.2　数据集成工序

（1）数据访问：从内部或外部数据源读取机读目录数据文件，并按照数据格式（MARC 或 MARCXML）将记录分配给对应解析器。

（2）数据转换：按照既定条件筛选记录，并从匹配记录中抽取、聚合特征字段，然后将其序列化为 JSON（JavaScript Object Notation）格式。JSON 是一种数据交换格式[9]，也是一项技术标准，不局限于某项技术，本身非私有，且可移植[10]。其简明紧凑的设计可为机读目录数据的组织和交换提供理想的数据结构支持。

（3）数据存储：将已转换数据以 NDJSON 格式写入服务器上暂存区的一个或多个文件中。NDJSON（Newline Delimited JSON）是一种采用 UTF-8 编码，并以换行符分隔每个 JSON 文档的格式，用于存储和流式传输可以一次处理一条记录的结构化数据[11]。

以中文书目（CNMARC）为例，记录控制号、标准编号或代码（ISBN、ISSN、ISRC 等）、题名与责任说明（正题名、并列题名、分卷册题名等）、出版发行项（出版者名称、出版年等）经过数据管道将被转换为如下格式：

{"001": "001571197", "01_a":["7-121-01765-2"], "200a": ["程序员修炼三部曲"], "200h": ["第三部", "Volume III"], "200d": ["The pragmatic starter kit"], "200e": ["如何建构、部署、监控 Java 应用", "how to build, deploy, and monitor java applications"], "200i": ["项目自动化之道", "Pragmatic project automation"], "200b": ["专著"], "200f": ["(美) Mike Clark 著"], "200g": ["张菲译"], "2009": ["cheng xu yuan xiu lian san bu qu"], "210c": ["电子工业出版社"], "210d": ["2005"]}

……

{"001": "007178734", "01_a": ["978-7-5444-4663-1", "978-7-5444-4658-7"], "200a": ["约翰·塞巴斯蒂安·巴赫平均律钢琴曲"], "200h": ["第一卷", "I"], "200d": ["Johann Sebastian Bach Das Wohltemprierte Klavier"], "200e": ["维也纳原始版"], "200i": ["BWV 846-869"], "200b": ["专著"], "200f": ["瓦尔特·登哈特（Walther Dehnhard），德特勒夫·克劳斯（Detlef Kraus）[编订]"], "200g": ["李曦微译"], "2009": ["yue han · sai ba si di an · ba he ping jun lv gang qin qu ji"], "210c": ["上海教育出版社"], "210d":["2013"]}

（4）数据装载：将 NDJSON 格式的数据集导入 Meilisearch，以供下游应用程序使用。暂存区中的文件通常在此后会被删除，但出于故障排除或数据分析等目的也可能会被保留更长时间。

3.3　后端接口功能

后端接口面向开发人员提供搜索与获取 JSON 格式文档的功能。

（1）全文搜索：通过关键词、过滤器和排序等参数查询文档，并返回匹配文档的全部或部分字段，例如：

```
curl -X POST 'http://your-domain/indexes/cnmarc/search' -H 'Content-Type: application/json' -H 'Authorization: Bearer API-KEY' --data-binary '{"q": "秘密", "filter": "200b IN [专著, 电子资源] AND 210c = 电子工业出版社", "sort": ["210d:desc"], "attributesToRetrieve": ["200a", "200b"]'}
```

（2）批量获取：通过配置和调整参数分批取得文档。默认返回前 20 个文档的全部字段。可选参数 offset 指定起始位置，limit 指定返回数量，fields 指定显示字段，filter 指定筛选条件。例如：

```
curl -X POST http://your-domain/indexes/cnmarc/documents/fetch -H 'Content-Type: application/json' -H 'Authorization: Bearer API-KEY' --data-binary '{"offset": 10, "limit": 10, "fields": ["200a", "210d"], "filter": "(210d = 2010 AND (200b = 专著 OR 200b = 期刊))"}'
```

（3）逐条获取：通过文档唯一标识符取得对应文档。默认返回该文档的全部字段。可选参数 fields 指定显示字段。例如：

```
curl -X GET 'http://your-domain/indexes/cnmarc/documents/000008112? fields = 001, 200a, 210c' -H 'Authorization: Bearer API-KEY'
```

3.4　前端界面功能

前端界面设计如图 2 所示。由图 2 可知，前端界面是以查询、筛选和列表相结合的方式为最终用户提供直观的搜索体验。具体阐释如下。

（1）即时搜索：根据用户输入的关键词动态交付与更新结果，用户可参照实时结果来调整和完善关键词。

（2）容错搜索：识别和纠正关键词中的拼写错误，自动推断正确词汇并给出相应结果。

(3)文本高亮:突出显示文档中的关键词,帮助用户在搜索结果中快速定位所匹配的文本。

(4)排序:将搜索结果按照相关度或预设的可排序字段排列,让用户自由选择结果显示顺序。

(5)分面搜索:将搜索结果分类,让用户通过简洁的导航选项来扩大或缩小结果范围。

图 2 前端搜索界面

Fig. 2 Front-end search interface

4 结束语

本文在介绍机读目录与 Meilisearch 相关概念的基础上,提出一种灵活构建机读目录数据搜索引擎的技术方案。虽然 Meilisearch 不是同类产品中功能最丰富的,但可在开发效率、性能、成本和易用性之间取得平衡,以满足不同受众的需求和偏好。

实际应用表明,该方案对千万级数据搜索可实现毫秒级响应,在总体上已经达到业技融合助力敏捷交付数据服务的预期效果。

为满足用户多元化需求,跨索引搜索和用户体验优化将成为机读目录数据搜索引擎后续开发重点。

参考文献

[1] 中华人民共和国文化部. 中国机读书目格式:GB/T 33286-2016[S]. 北京:中国标准出版社,2016.

[2] SULEMANI M, MACHIAVELLI G. What is Meilisearch? [EB/OL]. [2023-06-08]. https://www.meilisearch.com/docs/learn/what_is_meilisearch/overview.

[3] HARDIN M. Understanding LMDB database file sizes and memory utilization[EB/OL]. [2023-06-08]. https://www.symas.com/post/understanding-lmdb-database-file-sizes-and-memory-utilization.

[4] TURNBULL J. 第一本 Docker 书[M]. 李兆海,刘斌,巨震,译. 北京:人民邮电出版社,2015.

[5] 伊恩·米尔. Docker 实践[M]. 吴佳兴,梁晓勇,黄博文,等,译. 北京:人民邮电出版社,2018.

[6] ALGOLI A. What is InstantSearch. js? [EB/OL]. [2023-06-08]. https://www.algolia.com/doc/guides/building-search-ui/what-is-instantsearch/js/.

[7] FERREIRA C. Instant Meilisearch: recipe to a great front-end search[EB/OL]. [2023-06-08]. https://blog.meilisearch.com/instant-meilisearch/.

[8] 陶辉. 深入理解 Nginx:模块开发与架构解析[M]. 北京:机械工业出版社,2016.

[9] BASSETT L. JSON 必知必会[M]. 魏嘉汛,译. 北京:人民邮电出版社,2016.

[10] 汤姆·马尔斯. JSON 实战[M]. 邵钏,译. 北京:人民邮电出版社,2018.

[11] ndjson.org. Newline delimited JSON[EB/OL]. [2023-06-08]. http://ndjson.org.