Atiitt **半结构化数据的处理**

目录

[1.1. ：结构化数据和非结构化数据。 半结构化数据 1](#_Toc23824)

[1.1.1. .半结构化数据 2](#_Toc32103)

[2. 常见半结构化数据 非结构化数据 3](#_Toc25949)

[2.1. Eml word pdf xml html office 3](#_Toc2943)

[2.2. Zip rar 3](#_Toc5181)

[2.3. ,xml json CSV,一些自定义格式) 3](#_Toc2307)

[2.4. Java obj 序列化瘀php序列化 3](#_Toc15458)

[2.5. Markdown doc ,url 3](#_Toc28965)

[2.6. 日志文件 3](#_Toc6765)

[2.7. 典型的人为非结构化数据包括： 3](#_Toc27469)

[2.8. 典型的机器生成的非结构化数据包括： 4](#_Toc26673)

[3. 优缺点 4](#_Toc25068)

[3.1. 自描述 灵活性 无模式”的。更准确地说，其数据是自描述的 4](#_Toc9910)

[3.2. 这种灵活性可能使查询处理更加困难 4](#_Toc30937)

[3.3. 半结构化数据可以通过灵活的键值调整获取相应信息，且数据的格式不固 5](#_Toc15777)

[4. 储存方式 5](#_Toc10148)

[4.1. 化解为结构化数据 5](#_Toc17082)

[4.2. 用XML格式来组织并保存到CLOB字段中 5](#_Toc8073)

[5. 对非结构化数据也即对全文数据的搜索主要有两种方法： 6](#_Toc4761)

[5.1. 一种是顺序扫描法(Serial Scanning)：所谓顺序扫描 6](#_Toc8676)

[5.2. 全文索引 6](#_Toc27587)

[5.3. 查询处理的主要操作包括选择操作、连接操作、投影操作、聚合函数、排序等。 6](#_Toc21331)

[5.4. 非结构化查询处理过程中除了结构化数据查询处理所包含的操作外，还有两个重要的操作相似性检索和相似性连接。 7](#_Toc1991)

## ：结构化数据和非结构化数据。 半结构化数据

* **结构化数据：**指具有固定格式或有限长度的数据，如[数据库](http://irfen.me/tag/%e6%95%b0%e6%8d%ae%e5%ba%93" \o "数据库" \t "http://irfen.me/lucene-learn-1/_blank)，元数据等。
* **非结构化数据：**指不定长或无固定格式的数据，如邮件，word文档等。

当然有的地方还会提到第三种，半结构化数据，如XML，HTML等，当根据需要可按结构化数据来处理，也可抽取出纯文本按非结构化数据来处理。

**非结构化数据又一种叫法叫全文数据。**

**非结构化数据**

像图片、声音、视频等等。这类信息我们通常无法直接知道他的内容，数据库也只能将它保存在一个BLOB字段中，对以后检索非常麻烦。一般的做法是，建立一个包含三个字段的表（编号 number、内容描述 varchar(1024)、内容 blob）。引用通过编号，检索通过内容描述。现在还有很多非结构化数据的处理工具，市面上常见的内容管理器就是其中的一种

**半结构化数据**

这样的数据和上面两种类别都不一样，它是结构化的数据，但是结构变化很大。因为我们要了解数据的细节所以不能将数据简单的组织成一个文件按照[非结构化数据](https://baike.baidu.com/item/%E9%9D%9E%E7%BB%93%E6%9E%84%E5%8C%96%E6%95%B0%E6%8D%AE" \t "https://baike.baidu.com/item/%E5%8D%8A%E7%BB%93%E6%9E%84%E5%8C%96%E6%95%B0%E6%8D%AE/_blank)处理，由于结构变化很大也不能够简单的建立一个表和他对应。本文主要讨论针对半结构化数据存储常用的两种方式。

先举一个半结构化的数据的例子，比如存储员工的简历。不像员工基本信息那样一致每个员工的简历大不相同。有的员工的简历很简单，比如只包括教育情况；有的员工的简历却很复杂，比如包括工作情况、婚姻情况、出入境情况、户口迁移情况、党籍情况、技术技能等等。还有可能有一些我们没有预料的信息。通常我们要完整的保存这些信息并不是很容易的，因为我们不会希望系统中的表的结构在系统的运行期间进行变更。

### **.半结构化数据**

定义：非关系模型的、有基本固定结构模式的数据，例如日志文件、XML文档、JSON文档、Email等

简析：区分半结构化与非结构化的意义在于，对两者的处理方法是不同的，非结构化数据大多采用内容管理方法，而半结构化数据基本没有有效的管理方法。

（1）结构化、半结构化、非结构化其实是按照数据格式分类。

（2）严格讲，结构化与半结构化数据都是有基本固定结构模式的数据

# ****常见半结构化数据 非结构化数据****

## ****Eml word pdf xml html office****

## ****Zip rar****

## ****,xml json**** CSV,一些自定义格式)

## Java obj 序列化瘀php序列化

## ****Markdown doc ,url****

## 日志文件

## ****典型的人为非结构化数据包括：****

* **文本文件：**文字处理、电子表格、演示文稿、电子邮件、日志。
* **电子邮件：**由于其元数据，电子邮件具有一些内部结构，人们有时将其称之为半结构化。但是，其消息字段是非结构化的，传统的分析工具无法解析它。
* **社交媒体：**来自Facebook、Twitter和LinkedIn的数据。
* **网站：Y**ouTube、Instagram、照片分享网站。
* **移动数据：**短信、地点。
* **通信：**聊天、即时消息、电话录音、协作软件。
* **媒体：**MP3、数码照片、音频文件、视频文件。
* **业务应用程序：**MS Office文档、生产力应用程序。

## 典型的机器生成的非结构化数据包括：

* 卫星图像：天气数据、地形、军事活动。
* 科学数据：石油和天然气勘探、空间勘探、地震图像、大气数据。
* 数字监控：监控照片和视频。
* 传感器数据：交通、天气、海洋传感器。

# ****优缺点****

## 自描述 灵活性 无模式”的。更准确地说，其数据是自描述的

另一方面，对半结构化数据模型感兴趣的动机主要是它的灵活性。

特别的，半结构化数据是“无模式”的。更准确地说，其数据是自描述的。它携带了关于其模式的信息，并且这样的模式可以随时间在单一数据库内任意改变。 [1] 

人们可能很自然地想知道无模式地创建[数据库](https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%BA%93" \t "https://baike.baidu.com/item/%E5%8D%8A%E7%BB%93%E6%9E%84%E5%8C%96%E6%95%B0%E6%8D%AE/_blank)是否存在优点，在这样的数据库中，可以随意的输入数据，并且访问该数据时你感觉到的模式信息就是适合它的模式。实际上有一些小规模的信息系统，如Lotos Notes，它们就采用了自描述数据的方法。这种灵活性可能使查询处理更加困难，但它给用户提供了显著地优势。

## 这种灵活性可能使查询处理更加困难

## 半结构化数据可以通过灵活的键值调整获取相应信息，且数据的格式不固

# 储存方式

[编辑](https://baike.baidu.com/item/%E5%8D%8A%E7%BB%93%E6%9E%84%E5%8C%96%E6%95%B0%E6%8D%AE/javascript:;)

## 化解为结构化数据

这种方法通常是对现有的简历中的信息进行粗略的统计整理，总结出简历中信息所有的类别同时考虑系统真正关心的信息。对每一类别建立一个子表，比如上例中我们可以建立教育情况子表、工作情况子表、党籍情况子表等等，并在主表中加入一个备注字段，将其它系统不关心的信息和一开始没有考虑到的信息保存在备注中。

优点：查询统计比较方便。

缺点：不能适应数据的扩展，不能对扩展的信息进行检索，对项目设计阶段没有考虑到的同时又是系统关心的信息的存储不能很好的处理。

## 用XML格式来组织并保存到CLOB字段中

XML可能是最适合存储半结构化的数据了。将不同类别的信息保存在XML的不同的节点中就可以了。

优点：能够灵活的进行扩展，信息进行扩展式只要更改对应的DTD或者XSD就可以了。

缺点：查询效率比较低，要借助[XPATH](https://baike.baidu.com/item/XPATH" \t "https://baike.baidu.com/item/%E5%8D%8A%E7%BB%93%E6%9E%84%E5%8C%96%E6%95%B0%E6%8D%AE/_blank)来完成查询统计，随着数据库对XML的支持的提升性能问题有望能够很好的解决。

# 对非结构化数据也即对全文数据的搜索主要有两种方法：

## 一种是****顺序扫描法(Serial Scanning)：****所谓顺序扫描

，比如要找内容包含某一个字符串的文件，就是一个文档一个文档的看，对于每一个文档，从头看到尾

一种是**顺序扫描法(Serial Scanning)：**所谓顺序扫描，比如要找内容包含某一个字符串的文件，就是一个文档一个文档的看，对于每一个文档，从头看到尾，如果此文档包含此字符串，则此文档为我们要找的文件，接着看下一个文件，直到扫描完所有的文件。如利用windows的搜索也可以搜索文件内容，只是相当的慢。如果你有一个80G硬盘，如果想在上面找到一个内容包含某字符串的文件，不花他几个小时，怕是做不到。Linux下的grep命令也是这一种方式。大家可能觉得这种方法比较原始，但对于小数据量的文件，这种方法还是最直接，最方便的。但是对于大量的文件，这种方法就很慢了

## 全文索引

传统的结构化查询处理过程是:首先翻译器翻译查询请求生成查询表达式，然后由优化器优化查询表达式，得到优化过的查询计划，最后由执行器选择最优的查询计划执行，得到查询结果。

## 查询处理的主要操作包括选择操作、连接操作、投影操作、聚合函数、排序等。

查询优化的方法包括基于代价估算的优化和基于启发式规则的优化等。

## 非结构化查询处理过程中除了结构化数据查询处理所包含的操作外，还有两个重要的操作相似性检索和相似性连接。

相似性检索是指给定一个元素，在由该种类元素组成的集合中寻找与之相似的元素。例如论文查重系统用到文本的相似性检索，谷歌的以图搜图的功能用到图像的相似性检索，手机上根据哼唱匹配音乐是音频的相似性检索等。相似性连接是数据库连接操作在非结构化数据上的一种扩展，它寻找两个元素种类相同的集合之间满足相似性约束的元素对，在数据清洗、数据查重、抄袭检测等领域有着重要的作用。非结构化查询处理框架要针对这两种非结构化数据特有的查询操作对结构化查询处理框架进行改进。

**结构化数据 vs. 非结构化数据 - 51CTO.COM.html**