Caché 数据库中数据的存储及其查询优化

牛彩云[1] 王建林[1] 光 奇[1] 樊 睿[1]
NIU Caiyun WANG Jianlin GUANG Qi FAN Rui

摘 要 Caché 数据库的多维数据模型可以存储丰富的数据，在处理复杂的医疗数据时减少了表连接等处理过程，从而使多维数组能更快地存取数据。与主流的 Oracle 和 SQL server 等关系型数库相比，Caché 主要在其存储结构上有很大的不同，Caché 主要是以 Global 的形式存储数据，依据 M 语言开发应用程序。首先，介绍了 Caché 数据库中数据的存储形式；然后，展示了在医院 HIS 系统应用过程中 Caché 数据库中数据查询的几种方式及应用场合；最后，总结 Caché 数据库中 SQL 优化的几种办法。结果表明，Caché 数据库具有更高的灵活性，适用于多种应用场合，而且在采用优化的查询方案后查询效率提高了很多倍。

关键词 Caché 数据库；多维数据模型；查询优化；SQL 语句；数据存储

doi：10.3969/j.issn.1672-9528.2024.01.003

0 引言

Caché 是一种面向对象的数据库，采用对象技术，快速的 Web 开发，增强的 SQL 和独一无二的数据库缓存技术，具有关系数据库所没有的高性能、可扩充和灵活性，并且，Caché 提供了非常方便和灵活的数据访问技术。它的多维数据模型可以存储丰富的数据，先进的对象编程能力使开发复杂的数据库应用变得方便快捷。

M 语言的特长是它有一个独特的多维数据库机制，这个数据库结构能更好表示真实世界里的数据关系。经过长期的运行考验和发展，现在已经发展成为在 M 技术基础上融合了面向对象、面向 Web 和优化 SQL 技术的 Caché 后关系型数据库，它具有突出的 OLTP 高速响应性能和高伸缩性，运行可靠性很高又便于维护，非常适合于像医院信息系统和集成医疗提供网络这样的场合使用。

Caché 数据库是 InterSystems 公司开发的 Caché 数据库管理系统产品，以其独特的存储形式备受医疗信息领域的青睐，Caché 数据库在 M 开发的程序方面具有速度快，性能高等的特性，但是在使用 SQL 查询工具 SQLDB 进行 SQL 语言查询时经常表现出性能低，速度慢等缺陷，有些复杂的 SQL 查询花费七八个小时，用户体验较差。基于此，本文探索 Caché 的查询过程中的几种方法，并和关系型数库的查询作比较，找到优化的查询方案。

Caché 支持所有传统建立 Web 应用的方法，并提供独一无二的技术：Caché Server Pages，即 CSP。这是一个最优化的快速面向对象数据库系统开发技术。CSP 具有先进的面向对象结构，采用动态服务器页面技术，可以和各种当前主流的 Web 开发工具连接。

1 Caché 数据库中数据的存储

Caché 数据库是以多维数据模型的存储形式代替传统关系型数据库的二维表，可以用树型结构表示。它降低了磁盘 I/O，提高了查询的效率 [1]。在 Caché 数据库中，不管是持久化对象还是表，都是以 GLobal 的形式存储在数据库中，只是在访问的时候通过不同机制将它们映射成为对象或是关系的形式 [2]。

Caché 数据库以高效的多维数据模型和稀疏存储技术来存储数据，支持多种形式的数据访问，这里首先用对象的形式来具体分析其数据的存储情况。

Caché 数据库中数据表的创建可以以类的形式来创建，类属性中用 SQLTableName 提供了对应的表名称，Storage-Strategy 选择 SQLStorage。类定义中，用 Relationship 指定字段对应的外键，Caché 数据库的优势在于使用了指针的形式，使得查询速度和 SQL 语言的书写更加简洁，可理解性很强，不需要额外的表连接条件，直接可以查到其指针指向表中的任意字段。数据的存储以 SQLStorage 的形式，字段在表中的位置用 Piece 的值来表示，字段之间是以上箭头的形式分割，

1. 兰州大学第一医院信息中心 甘肃兰州 730000

[基金项目] 甘肃省重点研发计划 - 新型肺炎疫情下基于视觉控制的医疗自助系统的应用研究（20YF8FA080）；甘肃省重点研发计划 - 工业类，甘科计〔2023〕17 号，基于可信区块链的数字医院电子病历共享应用研究（23YFGA0037）；基于计算机仿真模拟技术的临床实验室新冠相关检测风险度分析的应用研究（2020-XG-41）

在 M 语言中，取字段的值就是通过 $p 函数来获取。

患者就诊表 User.PATADM 为例，创建的主要部分代码如下：

Class User.PATADM Extends (%Persistent, User.Abstract) [ClassType = persistent, Inheritance = right, Not ProcedureBlock, SqlTableName = PAT_ADM, StorageStrategy = SQLStorage]

{

// 定义 Caché 数据库的字段信息，外键信息，索引信息以及数据的存储信息。

}

其中，关系的定义类似于关系数据库中的外键，其定义的方式为：

Relationship ChildPATADMLoc As PATADMLoc [Cardinality = children, Inverse = LOCParRef];

表中字段定义形式为：

Property PATADMADMNo As %String(COLLATION = "AlphaUp", MAXLEN = 99, TRUNCATE = 1) [SqlColumnNumber = 2, SqlFieldName = PATADM_ADMNo]

PATADMADMNo 指定属性名，SqlColumnNumber 指定其列号，SqlFieldName 对应其二维表中的字段名，它定义了数据库查询语言所用的字段名，这样方便用户使用 SQL 来像查询二维表一样来查询 Caché 所存储的数据。

Property PATADMPAPMIDR As PATPatMas[Required, SqlColumnNumber = 3, SqlFieldName = PATADM_PAPMI_DR];

有些字段可以以指针的形式存储，定义中 PATADM_PAPMI_DR 为指针，它指向了患者基本信息表 PAT_PatMas，当我们要从就诊表里关联到患者基本信息时，可以不使用表之间的关联来获取，而是直接使用指针的形式。用具体的查询语言表示为 PATADM_PAPMI_DR->PAPMI_NAME 这种形式 , PATADM_PAPMI_DR 为 PAT_ADM 表中指向患者基本信息表 PAT_PatMas 的指针，PAPMI_NAME（患者姓名）为 PAT_PatMas 表中字段信息，这种表现形式简洁直观，省去了表之间的关联，尤其在复杂 SQL 的使用中给用户提供了很大的便利。

基本数据的存储形式定义：

```
Storage SQLStorage
{
  <ExtentSize>1000000</ExtentSize>
  <SqlIdExpression>$i(^PATADM(0))</SqlIdExpression>
  <SQLMap name="DataMasterMap">
    <Data name="PATADM_PAPMI_DR">
      <Delimiter>"^"</Delimiter>
      <Piece>1</Piece>
    </Data>
  <Global>^PATADM</Global>
  <RowIdSpec name=”1”>
    <Expression>{L1}</Expression>
    <Field>PATADM_RowID</Field>
  </RowIdSpec>
  <Structure>delimited</Structure>
  <Subscript name=”1”>
    <AccessType>sub</AccessType>
    <Expression>{PATADM_RowID}</Expression>
    <StartValue>1</StartValue>
  </Subscript>
  <Type>data</Type>
</SQLMap>
<SqlRowIdName>PATADM_RowID</SqlRowIdName>
<StreamLocation>^User.PATADMS</StreamLocation>
<Type>%CacheSQLStorage</Type>
}
```

Caché 中数据的存储是以 Global 的形式，在 Caché cube 提供的 Management portal 中可以查看 Global 的具体存储情况，它以节点的方式存储数据，0 节点代表根节点，在 M 中检索数据时一般以 0 节点开始检索，依次循环，直到检索到满足指定条件的记录，这种检索的速度相比较于传统关系型数据库来说速度更快，检索效率更高。它的表示形式为：

^PATADM(0)=12380740

^PATADM(1)="7^O^^261^^64148^36405^^2222^^^^N^^^^^^A^^^^^^N^^Y^^^^^^64148^36405^N^^^^^64148^36405^2740^^N^^^^^^^^^^^^^^^^1^^^^2^36540^^^^^^F^^^^^^^^^OP0000000001^"

^PATADM(1,1)="^^^1^^^1^^^^^^^^^N^^^^^^^^N^N^^^KXKYXKXWXN^^^^^^^^^^^^^N^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^"

从上面可以看出，Caché 数据库中数据的存储和关系型数据库中数据的存储形式存在很大的差异，但是，又有关系型数据库的影子，可以像操作关系型数据库一样通过 SqlDbx 等工具操作其中的数据，对数据进行增加、删除、修改等。需要说明的是，不是所有的 Global 数据都有对应的表信息，对于一些复杂的临床数据，Caché 中仅提供了 Global 的存储，对这类数据的操作只能通过对象技术访问。

表 1 以 ^PATADM(1) 存储数据为例（略去空值属性），说明数据代表含义。

表 1 Global 数据存储示例

^PATADM	node	Piece	值	含义
PATADM_PAPMI_DR		1	7	病人信息，指向 PAT_PatMas
PATADM_Type		2	O	患者就诊类型（O 代表门诊）
PATADM_DepCode_DR		4	261	就诊科室，指向 CT_Locs 表，261 为 CT_Locs 的主键值
PATADM_AdmDate		6	64 148	就诊日期，为数值型数据
PATADM_AdmTime		7	36 405	就诊时间，为数值型数据
PATADM_AdmDocCodeDR		9	2222	就诊医生，指向医护人员表 CT_CareProve
PATADM_ADMNo		81	OP0000000001	就诊号，患者就诊的标识
PATADM_CurrentProcess_DR	1	4	1	处理流，指向 PA_Process
PATADM_AdmReason_DR	1	7	1	医保类型，指向 PAC_AdmReason
.....				

2 Caché 与关系型数据库

Caché 与关系数据库之间尽管存在很多的差异，但也有共性，表 2 可以比较清楚地帮助用户认识 Caché 数据库。

表 2 Caché 与关系数据库概念对照

Caché 中概念	关系型数据库概念
包（package）	SQL 模式（schema）
类（class）	表（Table）
实例（Instance）	记录（row）
对象 ID（RowId）	主键 (Primary key)
普通文字属性（literal property）	字段（column）
持久对象引用	外键（foreign key）
嵌入对象	每个属性一个字段
关系（relationship）	外键，依赖关系
列表集合	一个字段
数组集合	子表
流（stream）	Blob
索引（Index）	索引（Index）
查询（query）	存储过程或试图
类方法（class method）	存储过程

Caché 作为后关系型数据库，保留了关系型数据库的某些特性，同时，它又对关系型数据库做了一些改进和优化，在进行标准化语言查询时，依然可以使用 SQL 语句来操作数据库中的数据。作为面向对象的数据库，它是一种适应多维数据服务和多种应用需求的新一代数据库技术。Caché 采用了先进的对象技术、快速的 Web 开发、增强的 SQL 和独一无二的数据缓存技术，它提供了关系技术所没有的高性能、可扩充性和灵活性。

3 Caché 数据库查询及优化

鉴于 Caché 数据库的存储形式的不同，Caché 数据的查询方式具有多样化，可以用 SQL 语句、M 语言等多种形式查询和分析数据，可以根据用户的喜好选择适合自己的方式。它既可以通过对象访问技术访问数据，也可以通过 SQL 语句的方式查询数据。在应用开发方面，Caché 是用 M 语言来开发，由于 M 语言不同于主流的开发语言，在编写程序时如果用户不熟悉 M，可以用 ODBC 连接到 Caché，然后用自己熟悉的语言来开发应用程序。

3.1 Caché 数据库中通过 Query 快速的查询数据

Caché 使用 M 语言开发应用程序，程序员可以通过写 Query 的方式很快获取数据，具有高效快捷的特性，它通过对象技术访问存储在 Global 中的数据，以医院需求为例，需查询医疗机构各科室的基本药物使用情况，Query 的定义中包含以下 4 个部分：Query、Execute、Close、Fetch 四个方法。

Query 定义查询入口，定义了相应的入参信息以及需要返回的出参信息。Execute 方法中定义了 query 需要执行的业务代码，用 M 语言编写，程序中可以选择满足条件的索引从 0 节点开始循环查找 Global 中存储的数据，将满足条件的数据记录输出。用 Query 写的查询程序接下来需要展示在前台界面，这里采用润乾报表设计器设计个性化的界面，最终在应用程序中展示给用户使用。

这种查询方法以高效的索引为入口，检索多维存储结构中的数据，其查询速度快、性能高、响应及时，备受用户青睐，这种查询方法广泛应用在 Caché 数据库的使用中。

3.2 指标访问技术

指标是另一种在 Caché 数据库中广泛使用的数据检索方法，指标的定义中主要包含指标 ID、编码、指标名称、指标描述、执行代码、数据节点、维度、类型、区间等信息。其中执行代码中定义了计算指标时需要执行的 M 语言代码，它是一个创建的 Caché ObjectScript Routine，其中定义了执行指标对应的 M 书写的方法。由于指标中增加了任务设置，可以按照日、月、季、年定期抽取，在查询时速度更快，性能更高，这种方法的另一个优点是如果业务表中数据变动，不会影响到指标中数据的变动，适用于对已产生数据不能改变的场景应用中。

3.3 索引的创建

和其它关系型数据库相似，查询优化主要就是索引的优化，众所周知，正确合理使用索引可以极大地提高查询效率，

相反，如果索引建立不当或者使用不当，不但不能提高效率，反而会降低查询性能。

索引的创建规则和其它关系型数据库一样，尽量选择经常使用的、必要的字段来创建索引，建立索引时要考虑到各字段数据的分布情况、表的主键、指针字段、经常与其它表做连接的字段、是否经常出现在 where 子句中等情况 [3]。满足上述这些条件的字段上一般会考虑创建索引，来提高数据查询的效率，增强应用程序的性能。

索引的创建一般是在医院信息系统数据库建设之初进行的，一旦系统上线运行，尤其在数量较大的情况下就要慎重建立。

在 Caché 数据库中索引的一种定义形式为：

```
<SQLMap name="IndexAdmissionDate">
  <Global>^PATADMi</Global>
  <PopulationType>nonnull</PopulationType>
  <RowIdSpec name="1">
    <Expression>{L3}</Expression>
    <Field>PATADM_RowID</Field>
  </RowIdSpec>
  <Structure>delimited</Structure>
  <Subscript name="1">
    <Expression>"PATADM_AdmDate"</Expression>
  </Subscript>
  <Subscript name="2">
    <Expression>{PATADM_AdmDate}</Expression>
  </Subscript>
  <Subscript name="3">
    <Expression>{PATADM_RowID}</Expression>
  </Subscript>
  <Type>index</Type>
</SQLMap>
```

3.4 SQL 语句的优化

SQL 语句优化是开发者比较容易忽略的一种提高查询效率的方法，SQL 书写不规范会导致笛卡尔积操作、全表扫描、索引跳扫、索引全扫、 Filter 低效过滤等低效操作 [4]，为了改进查询性能，提高应用程序效率，增强用户体验，对 SQL 进行优化是提高信息技术业务水平的一个关键因素。

Caché 在使用 SqlDbx 查询时，DBMS Type 这里选取 InterSystems CACHE，通过指定的命名空间，建立数据库连接。

Caché 数据库中，大多数的数据可以映射到关系表，如患者就诊信息表 PATADM，Global 名称为 ^PATADM，它所映射的关系表为 PAT_ADM，SQL 查询中可以直接用 SQL 语言查询其存储的数据。但是其查询速度没有用 M 语言编写的 QUERY 快，为了提高查询速度，必须要对 SQL 做好优化，这样才能提高效率，改进用户体验。经过长时间的应用积累，积累了以下几种优化策略。

（1）FIRSTTABLE 的应用

下面的查询语句是没有优化前的查询，数据查询时长为 1 h 48 min 31.641 s，用 FIRSTTABLE 优化后（在 FROM 后面增加了 FIRSTTABLE a ），同样的查询环境下，查询速度为 5.781 s，查询效率提高了 1000 多倍，极大改进了查询性能。

```
SELECT a.PATADM_PAPMI_DR->PAPMI_Medicare 住院号 ,
a.PATADM_AdmDate 入院日期 ,a.PATADM_DischgDate 出院日期 , a.PATADM_DepCode_DR->CTLOCS_Desc 科室 ,
p.MRCID_Desc 诊断
FROM PAT_ADM a ,
(SELECT w.TYP_MRCDiagTyp->DTYP_Desc DTYP_Desc,
g.MRDIA_MRADM_ParRef AS MRDIA_MRADM_ParRef,
g.MRDIA_ICDCode_DR AS MRDIA_ICDCode_DR,
h.MRCID_RowId AS MRCID_RowId,
h.MRCID_Desc AS MRCID_Desc,
h.mrcid_icd9cm_code AS mrcid_icd9cm_code ,
g.MRDIA_MainDiagFlag AS MRDIA_MainDiagFlag
FROM mr_diagnose g,mrc_icdx h,MR_DiagTyp w
WHERE g.MRDIA_ICDCode_DR=h.MRCID_RowId AND w.TYP_ParRef = g.MRDIA_RowId
AND w.TYP_MRCDiagTyp=4 AND (h.MRCID_Desc LIKE '% 肺栓塞 %' OR h.MRCID_Desc LIKE '% 主动脉夹层 %' )) p
WHERE a.PATADM_AdmDate>='2023-06-02' AND
a.PATADM_AdmDate<='2023-06-12' AND
a.PATADM_mainmradm_dr =p.MRDIA_MRADM_ParRef
AND a.PATADM_Type='I'
```

（2）索引的优化

查询中为了合理使用索引，某些表尽管对查询所需的数据没有用处，但是刻意增加相关联的表，让整个查询满足索引的最左匹配法则，这样同样会很大程度上改进查询性能。下面的语句优化前需要的时间为 1 h 6 min 49.640 s ，优化后仅需要 7 min 4.485 s。

优化前：

```
SELECT COUNT(a.ICUO_RowId)
```

FROM DHC_ICU_Ord a

WHERE a.ICUO_StDate BETWEEN '2023-03-01' AND '2023-03-31'

AND a.ICUO_ComOrd_Dr->ICUCRI_Type='V';

优化后：

SELECT COUNT(a.ICUO_RowId)

FROM DHC_ICU_Ord a,DHC_ICU_Arrange b

WHERE a.ICUO_StDate BETWEEN '2023-03-01' AND '2023-03-31'

AND b.ICUA_RowId=a.ICUO_ICUA_Dr

AND a.ICUO_ComOrd_Dr->ICUCRI_Type='V';

（3）ignore 的使用

合理使用 IGNOREINDEX，下面的语句中，EMRinstance.InstanceData 表上的可用索引有：

Idx List No Template ID Version ^DHCEMRI. Instance DataI

("IdxListNoTemplateIDVersion", TemplateID, Template Version, ListNo,ID)

在优化前，由于 EMRinstance.InstanceData 表上存在该索引，查询器会自己判断，可能会选取此索引来查询数据，导致数据查询速度很慢，查询时间为 7 min 47.813 s，但是经过优化后，指定不用该索引，这样查询时长为 5.781 s。

优化前：

SELECT COUNT(a.ID)

FROM %FIRSTTABLE c EMRinstance. Instance Data a, PA_Adm c

WHERE a.EpisodeID = c.PAADM_RowID

AND a.Status = 'Save'

AND c.PAADM_VisitStatus = 'D'

AND c.PAADM_DischgDate BETWEEN "2022-11-01" AND "2023-01-31"

AND a.TemplateID = 38;

优化后：

SELECTCOUNT(a.ID)

FROM %FIRSTTABLE c %IGNOREINDEX

IdxListNoTemplateIDVersion EMRinstance.InstanceData ,

PA_Adm c

WHERE a.EpisodeID = c.PAADM_RowID

AND a.Status = 'Save'

AND c.PAADM_VisitStatus = 'D'

AND c.PAADM_DischgDate BETWEEN "2022-11-01" AND "2023-01-31"

AND a.TemplateID = 38;

3.5 服务器性能优化

服务器的性能也是影响查询速度的重要因素之一，这表现在很多方面：

CPU 性能：CPU 是服务器的一项重要资源，合理规划系统 CPU 资源，使得 CPU 在高峰期仍能将使用率保持在合理的范围内，从而提高其性能。

操作系统：服务器的操作系统性能直接影响到数据库的性能。

内存空间：内存不足将减少数据库用于存放最近访问过的数据的缓冲区空间，并导致操作系统频繁进行页面交换，从而导致计算机系统额外的 I/O 开销。

负载均衡：负载均衡服务器将用户的请求转发到多个后端服务器，当一个后端无法处理更多的请求时，其他可用的服务器会接替它的工作，从而增加容错性和可靠性。负载均衡算法的选择对服务器的性能有重要的影响，从而会影响到客户端查询的效率。

4 结语

Caché 数据库在处理医疗领域数据方面具有较大的优势，合理利用它不仅可以发挥出数据库本身的优势，还可以提高应用程序的性能。对 SQL 进行性能调优，采用合理的策略方法，对准确高效提升业务水平有重要意义。在长期的使用过程中，可以发现，Caché 数据在查询方面，Query 等查询方案的效率要明显高于 SQL 查询，这种优势主要得利于其多维存储形式，它的树型存储形式在进行查找操作时具有速度快性能高等特点。

参考文献：

[1] 康世英.“后关系型”数据库 Caché 在 HIS 中的应用研究 [J]. 微处理机 , 2014.

[2] InterSystems Corporation. Caché 技术手册 [EB/OL].(2010-01-11)[2023-04-05].http://www.intersystems.cn/Caché/technology/techguide/index.html.

[3] 龚维荣 , 周顺平 , 万波 . 浅谈 Oracle 数据库基于索引的 SQL 语句优化方法 [J]. 计算机工程与应用 , 2003(5):196-199.

[4] 陆家俊 , 顾梅 .Oracle 查询优化的研究与应用 [J]. 信息技术与信息化 , 2022(1):57-60.

【作者简介】

牛彩云（1981—），女，甘肃天水人，硕士研究生，工程师，研究方向：数据挖掘与粗糙集理论。

（收稿日期：2023-06-12）