计算机科学
COMPUTER SCIENCE

基于嵌套集合模型的时态层次数据管理方法

杨振凯, 曹一冰, 赵鑫科, 郑景飚

引用本文

杨振凯, 曹一冰, 赵鑫科, 郑景飚.基于嵌套集合模型的时态层次数据管理方法[J]. 计算机科学, 2023, 50(6A): 220500290-5.

YANG Zhenkai, CAO Yibing, ZHAO Xinke, ZHENG Jingbiao. Temporal Hierarchical Data Management Based on Nested Intervals Scheme in Relational Database [J]. Computer Science, 2023, 50(6A): 220500290-5.

计算机科学 COMPUTER SCIENCE
http://www.jsjkx.com
DOI:10.11896/jsjkx.220500290

基于嵌套集合模型的时态层次数据管理方法

杨振凯　曹一冰　赵鑫科　郑景飚
信息工程大学地理空间信息学院　郑州 450052
(gisyzk@qq.com)

摘　要　时态层次数据是层次数据在时间维度的扩展,用于描述随时间变化的层次结构。相较于非时态层次数据,现有的时态层次数据管理方法仍存在存储方案复杂以及查询和更新效率低下等问题。针对上述问题,提出了一种基于嵌套集合模型的时态层次数据管理方法。首先从节点变化角度分析了层次数据变化的4种类型,在此基础上通过扩展时间标签字段实现了多版本节点在关系数据库中的存储和查询功能,最后提出了一种基于存量空间的嵌套集合模型(Abundantly Gapped Nested Intervals Scheme,AGNIS),用于解决主流嵌套集合模型插入数据记录效率较低的问题。基于我国2021—2022年行政区划及其调整数据的实验结果表明:提出的数据管理方法能够实现历史层次数据的存储和任意时刻层次结构快照的查询,且兼顾了时态层次数据查询和更新操作的高效性。

关键词:时态层次数据;嵌套集合模型;关系数据库;版本管理;行政区划
中图法分类号　TP392

Temporal Hierarchical Data Management Based on Nested Intervals Scheme in Relational Database

YANG Zhenkai,CAO Yibing,ZHAO Xinke and ZHENG Jingbiao
Institute of Geo-spatial Information,Information Engineering University,Zhengzhou 450052,China

Abstract　Temporal hierarchical data is a kind of hierarchical data characterized by time dimension description and is used to model the hierarchical structure that changes over time. Compared with management methods for common hierarchical data,there are still problems in temporal hierarchical data management such as the complexity of storage scheme design and inefficiency of query and update. To solve the above problems,a temporal hierarchical data management method based on nested intervals scheme is proposed. 4 types of change in hierarchical data are firstly analyzed from the perspective of the node change,based on which the storage and query capabilities of multi-version nodes in a rational database are then realized by extending the time labels. Finally,the abundantly gapped nested intervals scheme(AGNIS) is put forward to solve the problem of data insertion inefficiency in common nested intervals scheme. Experiments based on the data of Chinese administrative division and its adjustment from 2021 to 2022 show that the proposed method can implement the storage of historical hierarchical data and the query of hierarchical snapshot at any time,with a high efficiency in data query and update operation.

Keywords　Temporal hierarchical data,Nested intervals scheme,Relational database,Version management,Administrative division

1　引言

层次数据是最为常见的数据类型之一,其因具有严格的树状结构特征,在表达和描述组织架构、文件存储路径及行政区划分级等事物和场景时发挥着重要作用[1]。随着时间的推移,事物和场景的层次关系会发生变化,表现为不同时刻的树状结构不尽相同,需要借助时态层次数据进行描述[2]。时态层次数据是增加了时间维度的层次数据,其将层次结构单一时刻的静态描述扩展为任意时刻的动态描述,如以时态层次数据描述行政区划,不仅可以表达当前的行政区划隶属关系,还可以追溯历史上行政区划的调整情况。从这个角度而言,现有的非时态层次数据均可视作时态层次数据在某一时刻的快照。

由于定位数据记录方法具有特殊性,层次数据的存储与检索优化一直是相关领域研究的重点内容之一,已有多种文件格式(如xml和json)和多种数据库类型(关系数据库、图数据库等)对非时态层次数据结构提供了支持[3-6]。就成熟度、可扩展性以及性能而言,关系数据库在层次数据存储方面具有一定优势,如新版本的Oracle关系数据库自带封装了高效的层次查询语句"CONNECT BY…START WITH…"[7];另有相关研究基于关系数据库提出了多种层次数据的存储模型,如邻接表模型(Adjacency List Scheme,ALS)、路径枚举模型(Path Enumeration Scheme,PES)、质数模型(Prime Number Scheme,PNS)和嵌套集合模型(Nested Intervals Scheme,

基金项目:国家重点研发计划(2021YFB3900900)
This work was supported by the National Key Research and Development Program of China(2021YFB3900900).
通信作者:曹一冰(cao_yibing@126.com)

NIS)等[8-10]，以及在此基础上的混合方法和标记索引化方法[11-12]。对于时态层次数据的研究也大多基于关系数据库展开，其存储管理方案可分为两类。1)基于变化量的存储方法[1]，其思路为基于一个原始层次数据版本和中间变化量集合实现时态层次数据的保存，其中变化量 δ 为一个将源版本节点编码空间转换为目标版本节点编码空间的函数，基于该变化量可完成相邻版本层次数据的层次关系转换。该方法有效减少了节点的冗余存储，缺点是只考虑了节点的增加、删除和移动对于编码的影响，未考虑节点本身的变化，且需额外建表来存储变化量 δ。2)基于节点版本的存储方法[2]，即对于节点的每次变化，同时保存变化前后的节点版本信息。该方法原理简单，其缺点是随着时间增加，节点版本数量不断增多，层次数据访问效率迅速降低。由此可知，上述研究在时态层次数据存储方面总体存在节点变化类型考虑不全面、数据存储方案设计复杂，以及难以兼顾查询和更新效率等问题。

本文提出了一种基于嵌套集合模型的时态层次数据管理方法，在分析时态层次数据变化类型的基础上，通过扩展时间标签实现节点版本存储，并提出了一种基于存量空间的嵌套集合模型用于改进时态层次数据的查询和更新效率。本文的主要贡献如下：

(1)较为全面地分析了时态层次数据的变化类型；

(2)提出了基于时间标签的节点版本管理方法，为任意时刻的层次结构快照查询和树状结构重构奠定了基础；

(3)提出了一种基于存量空间的嵌套集合模型，相较于常用的改进前序树遍历模型，所提模型在保证时态层次数据高效查询的同时提高了更新效率。

2 时态层次数据变化类型

在层次数据模型中，位于不同层级的现实世界的事物均被抽象为节点，事物之间的层次关系抽象为连边。即对于时态层次数据，任一时刻的快照均呈现为严格的树状结构，其特点为：

(1)有且仅有一个节点的入度为 0，称其为根节点，是其他所有节点的祖先节点。

(2)除了根节点的入度都为 1，则称这些节点为非根节点，非根节点至少具有 1 个祖先节点。

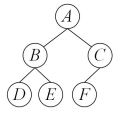
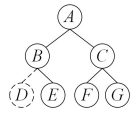
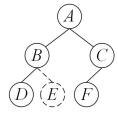
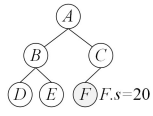
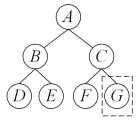
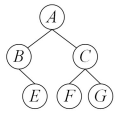
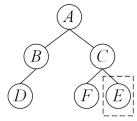
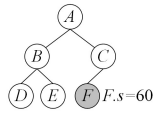
(3)非根节点中出度为 0 的节点为叶节点，其余为中间节点，叶节点的子孙节点数量为 0。

(4)与某个节点的直接相连的祖先节点和子孙节点(若存在)也被称为该节点的父节点和子节点，父节点仅有 1 个，子节点的数量没有限制。

以节点和连边表示层次树时，可以认为每个非根节点均绑定了一条指向父节点的连边，即节点数量始终比连边多 1 个。因此节点数量的变化和连边是同步的，节点和连边同时构建，也同时删除。下面以节点变化来分析层次数据随时间变化的类型，主要包含节点新增等 4 个类型，如表 1 所列。

表 1 层次数据变化类型

Table 1 Types of hierarchical data modification

Modification types	Node addition	Node deletion	Node moving	Node updating
Before modification				
After modification				

2.1 节点新增

节点新增是指为指定节点添加后代节点，包含两种情况：1)仅增加叶节点，即增加的节点其本身不含子节点；2)增加分支结构，即增加的子节点本身是一个多层树状结构。由于父节点的创建早于子节点，在分支结构增加至主结构时，层级越高的节点应优先插入，因此可将分支结构的增加转化为从高层级节点到低层级节点的顺序叶节点增加过程。节点新增只涉及新记录的增加，对已有数据影响不大。

2.2 节点删除

节点删除是指删除指定节点的后代节点，同样包含删除叶节点和删除分支结构。其中，分支结构删除意味着所涉及的节点均做删除处理，可将其转化为由低层级节点到高层级节点的顺序叶节点删除过程。为了实现时间回溯，不能清除已有数据记录，而应对记录添加删除标记。

2.3 节点移位

节点移位是指删除指定节点与原父节点的关系并选择一个新节点作为父节点。当该节点为中间节点时，移位前后依旧保持其与后代节点的关系。节点移位同样改变了已有记录，通常采用节点删除与节点新增的组合操作来实现。

2.4 节点更新

节点更新是指节点自身的描述信息发生了变化，而其本质信息(如唯一标识)和连边信息未发生变化。如节点代表一个公司职员，其住址和电话等信息发生了变化，但其所属部门不变。此时需要增加记录用于保存节点更新前后的信息。连边属性信息(如联系的强弱)的变化也可以归入此类。

由上可知，时态层次数据是时态图数据的一种特殊类型。相较于一般的时态数据(非图数据)，时态层次数据(图数据)不仅要记录节点自身的信息变化，还要记录父子节点关系的变化。相较于时态图数据，时态层次数据的连边数不会多于节点数，其数据量和复杂度相对较小，因此可采用关系数据库进行管理。

3 基于嵌套集合模型的时态层次数据存储方案设计

本节将介绍如何在存储时态层次数据的基础上，实现

任意时刻层次数据的高效查询和更新。其主要思路为：采用单表存储层次数据，通过扩展列字段的方式辅助实现版本存储，并提出了基于存量空间的嵌套集合模型用于提高查询和更新效率，从而确保该方案在任意关系数据库的适用性。

3.1 基于时间标签的节点版本管理

为了表达时态层次数据不同时刻的层次结构特征，需要记录所有节点的生命周期。本文采用创建时间(create_time)和删除时间(remove_time)来描述节点的生命周期，这两种时间均为有效时间，即反映了节点所代表的事物在现实世界的存在时间范围。在添加时间项之后，每 1 行记录由原先的表达节点本身变为表达节点的 1 个版本，节点之间的层级关系由版本存储。版本表示节点在某一时段相对稳定的存在状态，1 个节点至少对应 1 个版本，二者呈“1∶n”的关系。对应于时态层次数据的变化类型，分别对版本记录的具体操作进行说明。

(1)节点新增。执行插入“Insert”操作新增一条版本记录，根据节点所代表的事物实际产生或建立的时间设置其创建时间，该时间不早于父节点版本的创建时间，其删除时间默认为所用日期格式所能表示的最大值(如 yyyy-MM-dd 对应的 9999-12-31)。

(2)节点删除。对于要删除的节点版本，执行更新“Update”操作而不执行删除“Delete”操作，只需更新该记录的删除时间，删除时间不早于创建时间。当删除分支结构时，为分支结构中的所有节点版本设置相同的删除时间。

(3)节点移位。对于发生移位的所有节点，按照由低层级到高层级节点的顺序先执行(2)中的版本删除操作，然后按照相反的顺序执行(1)中的版本插入操作。其中，执行删除操作的版本删除时间与执行插入操作的版本创建时间相同。

(4)节点更新。首先对节点的原版本更新删除时间，而后插入更新后的版本记录。

基于时间标签对节点数据进行版本管理，将所有节点的版本数据存储于一个表中，简化了表的设计；此外，该方法可以实现任意时刻的层次结构快照查询和树状结构重构，屏蔽了复杂关联的时态层次数据演变过程。

3.2 基于存量空间的嵌套集合模型生成标记

相较于邻接表模型以及其他层次数据存储方案，嵌套集合模型以数字区间的包含关系处理层次结构，在占据数据空间、后代搜索效率等多个评价指标中均表现优秀[10]。因此，为了保证多版本层次数据的查询效率，本文采用嵌套集合模型的思路对每个版本记录进行标记。

在众多基于嵌套集合模型的层次数据管理方法中，改进前序树遍历模型(Modified Preorder Tree Traversal Scheme，MPTTS)是最常用的方法。为了解决 MPTTS 在记录插入或删除时需要对所有后续节点(标记值大于当前节点的节点)记录执行更新操作而导致的更新效率较低的问题，此处采用预先留空的思路，提出了一种基于存量空间的嵌套集合模型(Abundantly Gapped Nested Intervals Scheme，AGNIS)，用于时态层次数据的存储和访问。

区别于 MPTTS 父子节点及兄弟节点连贯的左右区间值标记，AGNIS 采用左值 l 和区间容量 s 对数据记录进行标记。其中，l 表示该节点的序号，s 表示可容纳的后代节点数(含该节点本身)。任意节点的区间可表示为$[l,l+s)$，其后代节点的序号均位于该区间之内。相邻节点以及父子节点的区间边界值无需连续，以便于新节点的快速插入。因此，对于具有祖先/后代关系的一对节点，其左值和容量满足式(1)。

$$l_a<l_d<l_d+s_d\leqslant l_a+s_a \tag{1}$$

其中，l_a 和 l_d 分别表示祖先和后代节点的左值，s_a 和 s_d 分别表示祖先和后代的容量。在此基础上，本文提出的 AGNIS 的具体方法如下：

(1)依据层级数 L 确定每层的节点初始化容量 S_i，其中 $i\in[1,L]$。区别于节点的区间容量 s，S_i 数值是固定的，用于初始化 s 以及对 s 动态扩容。对于每一层级的节点，确定一个不小于任意节点子节点数的数值 C_i，用于初始化节点的区间容量。设最底层的层号为 1，根节点所在层的层号为 L，则任意层级节点的初始化容量 S_i 的计算方法如式(2)所示：

$$\begin{cases} S_i=1, & i=1 \\ S_i=S_{i-1}C_i+1, & i\neq 1 \end{cases} \tag{2}$$

(2)依据前序遍历法确定节点序号。设节点 v_c 位于第 i 层，是其父节点 v_p 的第 j 个子节点。由于同一级节点之间的序号初始差值为 S_i，因此采用递归的方法依据父节点的序号 l_p 确定子节点序号 l_c 如式(3)所示：

$$l_c=l_p+S_i(j-1)+1 \tag{3}$$

(3)插入节点时，依据插入节点的层级确定该节点的容量 s_n，即 $s_n=S_i$，依据当前兄弟节点区间右值的最大值确定 l_n。若 l_n 不小于父节点的区间右值，则首先统一为该节点的祖先节点扩容 S_{i+1}，即区间上限扩大 S_{i+1}；然后对所有左值不小于 l_n 的节点的左值增加 S_{i+1}，即区间向数轴正轴平移 S_{i+1}；最后执行节点插入操作。

(4)删除节点时，为提高更新速度，仅更新节点的删除时间，不更新左值和容量。

综上可知，AGNIS 与 MPTTS 均依靠区间包含关系构建上下层次结构，因此在查询祖先和后代节点时原理是一致的。在插入节点时，首先判断待插入节点左值与父节点区间右值的大小关系，若父节点的区间还有容量则可以直接插入，否则需要在插入之前更新后续节点的左值 l 和祖先节点的区间容量 s。相较于 MPTTS，AGNIS 无需每次插入都更新其他节点的标记值；当需要为父节点扩容时，涉及的记录为后续节点和祖先节点。由于层级数一般远少于节点数，因此两种方法需要更新的记录数相差不大。

4 实验分析

本文实验的软硬件环境为：联想昭阳 E5，处理器 Core i7-10510U @1.80 GHz，内存 24 GB，系统版本为 Windows10 19044。使用 C++作为基础开发语言，结合 Qt 开发框架实现 ODBC，数据存储选择轻量级关系数据库 SQLite。

实验数据选择我国 2021－2022 年的县区级以上行政区划及变更数据。其中，以 2021 年末的国家级-省级-地级-县级 4 级行政区划数据为原始层级数据，共计 3 200 余条；将 2021 年发生的行政区划调整事件转换为数据变更操作，从而模拟时态层次数据的管理。实验中用于存储时态行政区划数据的

表字段设计如表 2 所列。

表 2 时态行政区划数据库的表字段

Table 2 Database table fields of temporal administrative division

Field name	Field Type
id	Int, primary key
name	string
ad_level	int
code	int
create_time	long
remove_time	long
left	long
size	long

4.1 时态层次数据版本管理效果验证

2021 年发生县区以上行政区划调整事件的省份共有 12 个，且均为县区级行政区划调整，对应于层次数据集中叶节点的增删变化，即此处层次数据的更新主要体现在节点的插入与删除。为了验证所提方案对于层次数据更新和查询的支持，实验首先将行政区划调整事件进行分类并转换为可执行的代码语句，如表 3 所列。

表 3 行政区划调整事件及对应操作

Table 3 Administrative division adjustment and corresponding operation in database

Adjustment events	Operations	Codes
2021 年 1 月 29 日 陕西省宝鸡市 撤销 凤翔县 610322	节点删除： 更新记录 删除时间	UPDATEt_admin SET remove_time=1611849600 WHERE id=XXX
2021 年 1 月 29 日 陕西省宝鸡市 设立 凤翔区 610305	节点新增： 插入记录	INSERT INTOt_admin(name, code, create_time, …) VALUES('凤翔区','610305', 1611849600,…)

在所有行政区划变更事件对应的操作执行完毕后，分别查询 2021 年 1 月 1 日和 2022 年 1 月 1 日杭州市的县区级行政区划构成情况，其结果如图 1 所示。由此可知基于时间标签的版本管理方案能够实现对历史层次数据的更新与查询。

图 1 时态层次数据快照查询(以杭州市为例)

Fig. 1 Snapshot query of temporal hierarchical data(taking Hangzhou as an example)

4.2 层次数据查询与更新效率验证

为了对层次数据的查询和更新效率进行测试，实验分别采用邻接表模型 ALS、改进前序树遍历模型 MPTTS 和本文提出的基于存量空间的嵌套集合模型 AGNIS 进行对比实验。以中国、河南省和郑州市为例分别查询国家级、省级和市级节点的后代节点并在内存数据中构造层次树，所用平均时间如图 2 所示。

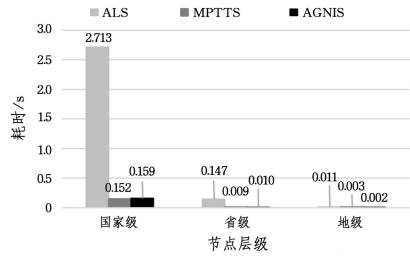


图 2 不同等级节点的查询效率对比

Fig. 2 Query efficiency of 3 schemes for nodes of different levels

本实验所构造的层次树中，每个节点分别存储父节点指针和子节点指针容器，用于记录父子节点关系。由图 2 可知，本文提出的 AGNIS 模型与 MPTTS 用时基本相同，且远少于 ALS，当层次树的层级越多、数据量越大时，二者的查询效率优势愈发明显。在已知节点 l 和 s 数值的情况下，本文模型在查询时只访问一次数据库便将所有后代节点的记录按照 l 的降序返回，通过堆栈构造层次树，效率远高于需要多次访问数据库的邻接表模型。

为了对比层次数据的更新效率，基于上述 3 种模型分别模拟不同级别行政区划的设立和撤销，即对节点进行插入和节点删除操作，所用平均时间如图 3 所示。

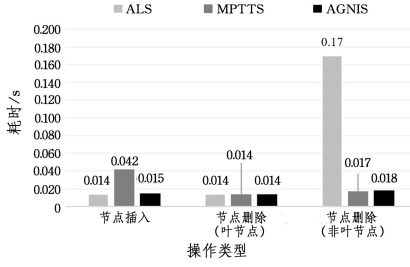


图 3 节点更新效率对比

Fig. 3 Node update efficiency comparison of 3 schemes

在模拟县级行政区划的撤销时，3 种方法均只对叶节点的删除时间进行更新，故所用时间相同。在模拟省级行政区划(本文以河北省为例)撤销时，ALS 所需时间远大于 MPTTS 和 AGNIS。当删除非叶节点时，ALS 需要迭代搜索子节点，而 MPTTS 和 AGNIS 可依据标记范围确定需要进行删除操作的子节点，因此前者的执行效率要远低于后两者。对于单个节点的插入操作，由于 AGNIS 使用了存量空间，其执行效率要优于 MPTTS，与 ALS 基本相同。当且仅当父节点的存量空间用完时，AGNIS 才对整个层次树进行更新，此时执行效率与改进前序树遍历模型基本相同。由此可知，对于时态层次数据，基于存量空间的嵌套集合模型能够在保证较高查询效率的基础上，有效提高其更新效率。

结束语 本文设计了一种基于嵌套集合模型的时态层次数据管理方法，基于时间标签实现了多版本节点在关系数据库中的存储和查询功能，并提出了一种基于存量空间的嵌套集合模型用于改进时态层次数据的更新效率。当前学术界对于时态层次数据管理方法的研究较少，且对于层次数据变化

类型的分析还不全面。而本文提出的管理方法能够兼顾数据查询和更新效率，采用单表存储方案且无需建立其他索引，适用于任意关系数据库，具有一定的独创性。本文方法也存在一定的不足，如区间容量的初始化需要在预先确定层次数据层级的基础上才能完成，因此后续研究将围绕层级与区间容量的关系展开，针对不同层级和分布特征的层次数据探索最佳的区间生成方案。

参 考 文 献

[1] FINIS J，BRUNEL R，KEMPER A，et al. DeltaNI：an efficient labeling scheme for versioned hierarchical data[C]// Proceedings of the ACM SIGMOD International Conference on Management of Data. New York：ACM Press，2013：905-916.

[2] JOMPHROM C，PORKAEW K. Version management of hierarchical data in relational database[C]// Recent Advances in Information and Communication Technology 2015. Cham：Springer，2015：275-284.

[3] YUN J，CHUNG C. Dynamic interval-based labeling scheme for efficient XML query and update processing[J]. Journal of Systems and Software，2008，81(1)：56-70.

[4] LI Q，MOON B. Indexing and querying XML data for regular path expressions[C]// Proceedings of the 27th International Conference on Very Large Data Bases. San Francisco：Morgan Kaufmann Publishers Inc.，2001：361-370.

[5] ZHAO A，CHEN H，XIONG J. Hierarchical tree queries mechanism in relation database system[J]. Computer Engineering and Design，2006，27(18)：3454-3456.

[6] ZHANG Z. Research on key technology ofspatio-temporal object associative relationship generation，management，and visualization[D]. Zhengzhou：PLA Strategic Support Force Information Engineering University，2020.

[7] Oracle. SQL Language Reference [EB/OL]. (2022-03-01) [2022-05-01]. https://docs. oracle. com/en/database/oracle/oracle-database/21/sqlrf/Hierarchical-Queries. html # GUID-0118D-F1DB9A9-41EB-8556-C6E7D6A5A84E.

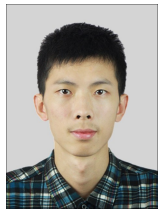
[8] TULDER G V. Storing Hierarchical Data in a Database [EB/OL]. (2003-04-30) [2022-05-01]. https://www. sitepoint. com/hierarchical-data-database.

[9] KUANG L，XIONG F，HAN X. On generation algorithm of prefix code-based preorder traversal tree and its application[J]. Computer Applications and Software，2011，28(4)：67-70，94.

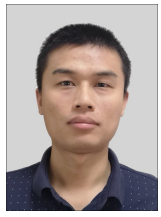
[10] MOROZOV S，SAIEDIAN H，WANG H. Reusable prime number labeling scheme for hierarchical data representation in relational databases[J]. Journal of Computing and Information Technology，2014，22(1)：31-44.

[11] WELLENZOHN K，BOHLEN M H，HELMER S. Dynamic interleaving of content and structure for robust indexing of semi-structured hierarchical data[C]// Proceedings of the VLDB Endowment. 2020：1641-1653.

[12] FINIS J，BRUNEL R，KEMPER A，et al. Order Indexes：supporting highly dynamic hierarchical data in relational main-memory database systems[J]. The VLDB Journal，2017，26(1)：55-80.



YANG Zhenkai，born in 1993，Ph.D candidate. His main research interests include analysis and visualization of hierarchical data.



CAO Yibing，born in 1986，Ph. D. His main research interests include core technology and software system of geospatial intelligence.