数据库上的关键词查询技术研究进展与趋势

CCF 数据库专业委员会

摘　要

数据库是存储结构化和半结构化数据的主要方式。数据库上的关键词查询技术的出现，大大降低了用户使用数据库的门槛，提高了用户查询的效率和效果。用户不需要掌握复杂的结构化查询语言，也无需了解数据模式，只需要输入查询关键词就可以进行数据库的查询。针对数据库上的关键词查询问题，本文对该领域的国际和国内研究现状进行了归纳总结，主要介绍基于数据库模式的关系数据库上的关键词查询、基于数据图的关键词查询和 XML 数据库上的查询，并展望了未来的研究方向。

关键词：数据库，XML，关键词查询，信息检索，模式图，数据图

Abstract

Databases have become a dominant way of storing structured and semi-structured data. The introduction of keyword search over databases has made databases systems much easier to use and improved the effectiveness and efficiency of querying the databases. Users do not have to master the complex query languages; nor do they have to be aware of the database schema. They only need to enter the keywords to perform a database query. This article provides a survey of research on keyword search in databases, conducted both in China and abroad. We mainly focus on database schema-based keyword search over relational databases, graph-based keyword search, and keyword search over XML databases. We also discuss possible directions for future work.

Keywords: database, XML, keyword search, information retrieval, schema graph, data graph

1　引言

关系数据库、XML 数据库、数据仓库、空间数据库等各种数据库产品已经在社会生产、生活的各个领域得到广泛应用。经过四十多年的发展，关系数据库技术已臻成熟，成为电信、银行、保险等行业数据存储的首选方式。同时，在 XML 数据库和数据仓库等数据库中，也存储着大量的有价值的数据。如何快速有效地从这些数据中查询定位感兴趣的信息，是一个非常重要的问题。

现有数据库系统所支持的各种查询方式在灵活性、易用性及表达能力方面存在着一定的欠缺。对用户而言，目前的数据库产品仍然具有较高的使用门槛。为了查询数据库中的数据，用户必须学习和掌握有一定难度的结构化查询语言（如 SQL、MDX 等）。对

于一些复杂的查询，编写相应的 SQL 语句并不是一件简单的事情。此外，用户需要对数据模式有一定的了解，否则很难选出查询所需的那些表和属性。因此，必须研究更为方便易用的数据库查询方式。

为了满足这个需求，数据库领域的研究人员引入了信息检索（Information Retrieval，IR）领域的关键词查询概念。采用关键词进行查询，是信息检索领域取得成功和快速发展的一个重要因素。今天，当我们需要查询感兴趣的互联网信息时，只需要在互联网搜索引擎（如 Google、百度）中输入查询关键词，就可以迅速得到相关的网页内容。如果对数据库的查询也可以采用类似的方式，将极大降低用户的使用门槛，因为用户只需要输入查询关键词就可以从数据库中找到感兴趣的信息，而不需要掌握复杂的 SQL 语句。

但是，要真正让数据库支持高效的关键词查询，需要解决很多具有挑战性的问题。首先，由于数据库的规范化，信息的逻辑单元经常被分片存储到不同的物理表当中。对于一个给定的关键词集合，可能需要对多个关系表进行即席连接操作才能得到匹配的行集。因此，必须设计高效的算法发现关键词之间的语义关系，把来自不同数据库表中的元组“拼接”成与用户兴趣相关的元组集合。其次，包含查询关键词的元组集合可能非常多，但是，用户只希望得到一部分与自己兴趣高度相关的结果，因此，必须设计有效的查询结果评分和排序机制。此外，还存在着查询结果呈现和消除冗余结果等诸多问题。

数据库上的关键词查询问题，已经成为数据库的研究热点。近年来，许多研究者在 DB/IR 集成方面进行了大量的研究，在 SIGMOD、VLDB、ICDE 等数据库顶级会议上每年都有多篇相关论文发表。Amer-Yahia 等人[2]在 SIGMOD'05 上就此专题组织了分组讨论。Amer-Yahia 等人[3]、Chaudhuri 等人[6]、Chen 等人[8,9]在 VLDB'05、VLDB'09、SIGMOD'09、ICDE'11 上分别做了数据库上关键词查询的辅导报告（Tutorial）。ICDE'12 的 Influential Paper Award 也颁给了两篇关键词查询方面的论文[1,4]。这些都说明数据库上的关键词查询已成为当前数据库研究的一个主要方向。

本文对该问题的国际和国内研究现状进行了归纳总结，主要介绍针对关系数据库和 XML 数据库的相关研究，最后展望了未来的研究方向。

2 国际研究现状

数据库上的关键词查询，已经成为数据库领域的一个热门研究问题。下面将从四个方面分别介绍当前的国际研究现状：1）基于数据库模式的关系数据库上的关键词查询；2）基于数据图的关键词查询；3）XML 数据库上的关键词查询；4）其他方面的研究。

2.1 基于数据库模式的关系数据库上的关键词查询

基于数据库模式的方法是解决基于关系数据库的关键词查询的一种代表性方法。该方法利用数据库模式来枚举可能包含查询结果的所有连接表达式，然后在数据库上执行

这些表达式得到元组连接树[1]或元组连接网络[20]。

基于数据库模式的方法，主要包括三个步骤：1）利用数据库模式来枚举可能包含查询结果的所有连接表达式；2）根据一系列规则把连接表达式转换成 SQL 语句，并在数据库上执行，得到所有可能的候选结果；3）对结果进行排序并把相关内容返回给用户。

实际上，可能的查询结果的数量可能很大，但并非所有的结果对用户而言都是有意义的。比如，如果两个元组之间通过太多的中间连接发生关系，那么，由于这两个元组之间的距离太遥远，对用户的实际应用价值就很小。因此，在第一步枚举连接表达式时，会对表达式尺寸进行限制，即限制表达式所包含连接的数目。在第二步中，执行 SQL 的方式可以分为两种[36]：一种是直接在 RDBMS（Relational Database Management System）上执行 SQL 语句；另一种是采用中间件方法，在 RDBMS 上面增加一个中间件层，然后在中间件上执行 SQL 语句。采用在 RDBMS 上面直接执行 SQL 语句的方式，可以充分利用商业数据库的强大功能，这方面的研究主要包括文献[1,20, 36]。但是，这种在 RDBMS 上面直接执行 SQL 语句的方式，需要处理大量的关系代数表达式，因此，许多研究[19, 33, 34]采用了基于中间件的方法，而没有充分利用 RDBMS 的自身能力。

下面介绍两个具有代表性的关键词查询系统的搜索过程，即 Agrawal 等人提出的 DBXplorer[1]和 Hristidis 等人提出的 DISCOVER[20]。

首先介绍 DBXplorer 系统的搜索过程。对于给定的查询关键词集合 K，DBXplorer 返回所有符合条件的元组连接树，树中的元组或者来自单个表，或者由多个表的连接操作得到，每个元组连接树都包含了所有查询关键词。DBXplorer 的关键词查询主要包含如下两个步骤：

1）发布：这是一个预处理步骤，主要任务是为数据库构建符号表和辅助结构，从而使其支持关键词查询。尤其需要指出的是，符号表在关键词查询过程中举足轻重。在查询时，使用符号表可以快速确定关键词在数据库中的位置（即关键词所在的表、行、列信息）。

2）搜索：这个步骤实现从发布的数据库中获得匹配的结果。首先，系统查找符号表，从而确定关键词所在的表、行、列等信息；其次，枚举连接树，每棵连接树都是数据库模式图中的一棵子树，连接树中的表在数据库模式图中都存在与之对应的节点，并且这些节点之间存在连接关系，对这些连接树中的表执行连接操作以后，就可能得到包含所有关键词的元组连接树；然后，为每个被枚举的连接树创建一个 SQL 语句，这个语句会对连接树中的表执行连接操作，由此可以得到许多元组连接树，系统会选择那些包含所有关键词的元组连接树；最后，对这些元组连接树进行排序，并且作为结果提交给用户。

现在介绍 DISCOVER 系统的搜索过程。首先，系统在接收到来自用户的关键词查询 $K = \{k_1, k_2, \cdots, k_m\}$ 后，利用关系数据库中的索引，对每个关系 R_i 都查找出包含这些关键词的元组集合 $R_i^{k_1}, \cdots, R_i^{k_m}$；其次，计算所有的候选网络；然后，对候选网络进行评估，并使用计划生成器生成一个执行计划，这个计划在评估候选网络过程中可以计算并使用中间结果；最后，为每一行执行计划都生成相应的 SQL 语句，并提交给 RDBMS

执行，RDBMS 返回元组连接网络，并进行排序，由此可以得到关键词查询的结果。

在枚举连接树过程当中，不同的方法采用不同的搜索策略。比如，DBXplorer、DISCOVER 和 DISCOVER-Ⅱ[19]等方法都使用宽度优先的图搜索策略，而 Simitsis 等人[42]提出 FIS（Find Initial Subgraphs）算法则采用最好优先的图搜索策略。

2.2 基于数据图的关键词查询

基于数据图的关键词查询的核心思想就是，把数据库转换成数据图，然后从数据图中找到包含关键词的连通子图。由于关系数据库和 XML 数据库都可以表示成数据图的形式，因此，基于数据图的方法既可以应用于关系数据库上的关键词查询，也可以用于 XML 数据库上的关键词查询。

基于数据图的方法可以分为：1）基于简化子树的方法；2）基于子图的方法。其中，基于简化子树的方法又包括：基于 Steiner 树的方法和基于相异根（Distinct Root）的方法；基于子图的方法又包括：基于 r-半径 Steiner 图的方法和基于多中心图的方法。

2.2.1 方法概述

一个数据库可以视为一个数据图 G，图 G 以元组和关键词作为节点。如果两个元组可以通过外键进行连接，那么二者之间就存在一条边。数据图的边具有方向性，可以反映不同方向上的连接的强弱，因为两个节点之间的连接在不同方向上的连接强度不是对称的，比如，在反映主外键关联的边中，外键到主键方向的边和其反方向的边是具有不同的重要性的。数据图中的节点和边都可以具有权重，这些权重都是预先被赋值的，从而可以更好地支持关键词查询。通过为数据图中的边和节点分配权重，就可以设计相应的评分机制，对从数据图中得到的查询结果进行评分和排序，为用户输出最相关的结果。比如，Bhalotia 等人[4]提出的 BANKS 方法就综合利用了节点权重和边权重来计算查询结果的相关性评分。此外，BANKS 还引入了 Google 在 PageRank 中采用的“美誉度”（Prestige）的概念，当某个节点具有更多的指针指向它时，它就具有更高的美誉度。BANKS 把节点的入度视为节点的美誉度，比如，如果有很多篇论文都指向（引用）某篇论文，那么，这篇论文就比较重要，因为就具有更高的美誉度。

基于数据图的方法主要包括两个步骤：首先，把数据库看成一个带权重的数据图，并且假设数据图已经被物化。然后，充分利用数据图中的节点（元组）和边（元组之间的主外键关联）的权重，来为关键词查询找到 top-k 个代价最小的、包含关键词的连通子图。

不同的研究采用不同结构的连通子图作为查询结果，主要包括两类结构：

1）简化子树：树中包含了所有查询关键词。

2）子图：比如 r-半径 Steiner 图[29]和多中心图[37]。相应地，基于数据图的方法可以分为基于简化子树的方法和基于子图的方法。

2.2.2 基于简化子树的方法

基于简化子树的方法可以进一步分为基于 Steiner 树的方法和基于相异根的方法。

2.2.2.1 基于 Steiner 树的方法

从数据图中找出包含查询关键词的 Steiner 树，作为查询结果。在许多文献当中，Steiner 树问题已经被证明是 NP-hard 问题，由于需要计算每棵树的整体相关性，就需要考虑节点的权重，这个问题就变得更加复杂了。

比较有代表性的图搜索算法是反向搜索（Backward Search）[4,18,23] 和动态编程方法[12,45]。这里主要介绍一下反向搜索算法，它最早在 BANKS 系统中提出，后来的研究诸如 Kacholia 等人[23]提出的 BANKS-Ⅱ和 He 等人[18]提出的 BLINKS 等扩展了该算法。概括地说，反向搜索算法的工作过程如下[18]：

1）在反向搜索的任何时刻，让 E_i 表示当前已经知道的可以到达关键词节点 k_i 的节点集合，其中，E_i 被称为关于 k_i 的簇。

2）在最初始阶段，E_i 被定义成直接包含 k_i 的节点集合，这个集合称为“原始簇”，它的成员节点为关键词节点。

3）在每一步搜索中，都从以前访问过的节点，比如说节点 v，选择一条入射边，然后沿着这条边反向访问它的源节点，比如说节点 u，任何包含节点 v 的 E_i，现在都被扩展到节点 u。一旦一个节点已经被访问，那么，搜索算法就可以获知它的所有入射边的信息，搜索就可以在以后步骤中访问这些边。

4）如果对于每个簇 E_i，都有或者节点 x 属于 E_i，或者存在一条从 x 到 E_i 中某个节点的边，就意味着已经发现了一个答案的根节点 x。

BANKS 采用了反向搜索算法在数据图中搜索满足条件的子树。同时，BANKS 系统也提供了丰富的界面来对数据库的内容进行浏览。浏览系统会自动为数据库关系和查询结果生成可浏览的视图，不需要编程和用户干预。BANKS 还提供了模板功能，允许用户把数据显示方式预先定义成模板，以便以后多次随时使用。

Ding 等人[12]提出了一个动态编程算法，可以生成 top-1 个结果，然后对此进行扩展继而生成其他结果。但是，它不能保证接下来生成的 $k-1$ 个结果就一定属于 top-k 中的结果，许多结果根本就不会被生成，即使算法已经运行到不会产生任何结果的时候。这是因为，为了提高算法效率，作者在枚举工作中采用了启发的顺序，也就是采用一个与排序顺序比较相似的顺序（无法保证是排序顺序）。另外，此算法不能产生一些高度相关的结果。

2.2.2.2 基于相异根的方法

由于 Steiner 树问题已经被证明是 NP-hard 问题，因此，一些研究工作开始寻求更简易的方法，基于相异根的方法就是一种更加简易的方法。在该方法中，对于数据图 G 和节点 v，以节点 v 为根的可能的子树，是从 v 到其他各个关键词节点的最短路径的并集。与基于 Steiner 树的方法不同的是，基于相异根的方法采用了不同的子树权重计算方法，在该方法中，子树的权重是从根到每个关键词节点的最短距离之和。此外，对于一个给

定的关键词查询，两种不同的方法所得到的简化子树的数量也是不同的。对于基于 Steiner 树的方法，将会生成指数级数量的子树，即 $O(2^m)$，其中，m 是数据图 G 中的边的数量。但是，对于基于相异根的方法，最多只会生成 n 棵子树，其中，n 是数据图 G 中的节点的数量，也就是说，只存在 0 或 1 棵以节点 v 为根的子树。

在基于相异根的方法方面，具有代表性的研究包括双向搜索[23]、双层索引[18]和外部存储数据图[11]。

1）双向搜索。BANKS 采用反向扩展搜索（Backward Expanding Search）算法，直接对数据图进行搜索。但是，BANKS 算法也存在一个明显的缺点，那就是当一个查询关键词匹配非常大量的节点时，或者当算法遇到一个入度非常大的节点时，反向扩展算法的性能就变得很差。针对这个问题，Kacholia 等人[23]提出的 BANKS-Ⅱ采用了一种新的搜索方法——双向搜索，它允许从可能的根节点出发朝着叶子节点进行前向搜索，从而改进了后向搜索的性能。BANKS-Ⅱ把查询结果树看成是路径的集合，每一个关键词对应一条路径，每条路径从根出发到达包含关键词的节点，结果树的边积分就是路径长度的总和。BANKS-Ⅱ可以在多项式时间延迟内回答查询，同时可以避免生成大量具有同一个根的相似结果。

2）双层索引。He 等人[18]提出的 BLINKS 采用了一个基于代价均衡扩展（Cost-Balanced Expansion）的反向搜索策略，并且使用了额外的双层索引来加快搜索速度。双层索引明显减少了运行最优反向搜索算法的开销，并且可以支持前向搜索，从而有效地实现类似 BANKS-Ⅱ中的双向搜索。和 BANKS-Ⅱ双向搜索方法相比，BLINKS 在搜索过程当中，可以实现更大的前向跳跃，加快了搜索速度。BLINKS 是一个基于内存的方法，当双层索引在内存中时，BLINKS 方法表现得最好。

3）外部存储数据图。基于数据图的方法大多假设数据图可以一次性放入内存。但是，现在的企业数据库中存储了越来越多的数据，并不是所有数据库的数据图都可以一次性放入内存，尤其是一些存储了海量数据的数据库，更是无法放入有限的内存，有时候外部数据必须多次反复进出内存才能得到查询结果，这带来了大量的 I/O 开销。Dalvi 等人[11]研究了在相异根语义下，数据图驻留在外存时的关键词查询策略。作者提出了一种新的图表示技术，它融合了超节点图（图的浓缩版本，总是可以放入内存）和缓存中的细节图，从而形成多粒度的图表示方法。超节点图中的节点是通过对完整图中的节点进行聚类而得到的，如果两个超节点中包含了互相连接的节点，那么，就在这两个超级节点之间建立一条边。多粒度图表示了可以获得的关于完整图的存放在当前内存中那部分图的信息。作者还对现有的基于数据图的搜索算法进行扩展，提出了可以利用多粒度图的“反复扩展”（Iterative Expansion）和“增量扩展”（Incremental Expansion）搜索算法，这两种算法可以通过把搜索过程引导到图中可能产生结果的区域，来最小化 I/O 开销。

2.2.3 基于子图的方法

基于子图的方法可以进一步分为基于 r - 半径 Steiner 图的方法和基于多中心图的

方法。

2.2.3.1 基于 r - 半径 Steiner 图的方法

给定一个图 G 和图中的任何节点 v，节点 v 的中心距离表示成 $CD(v)$，是节点 v 和图 G 中的任意节点 u 之间的距离的最大值，即 $CD(v) = \max_{u \in G}\{D(v, u)\}$，其中，$D(v, u)$ 表示节点 u 和 v 之间的距离，也就是节点 u 和 v 之间的最短路径的长度。图 G 的半径用 $R(G)$ 表示，是图 G 中的每个节点的中心距离的最小值，即 $R(G) = \min_{v \in G}\{CD(v)\}$。如果图 G 的半径正好是 r，那么图 G 就称为“r - 半径图”。给定一个 r - 半径图 G 和一个关键词集合 K，节点 w 称为内容节点，如果节点 w 直接包含了集合 K 中的关键词。节点 s 被称为 Steiner 节点，如果存在两个内容节点 u 和 v，并且 s 在 u 和 v 的路径上（特殊情形下，s 可能是 u 或者 v）。图 G 中由 Steiner 节点和相关边构成的子图称为 r - 半径 Steiner 图。一个 r - 半径 Steiner 图的半径可以比 r 小，但是不能大于 r。

很显然，可以把包含全部或部分关键词的 r - 半径图作为查询的答案。因为 r - 半径图是非常简洁和有意义的，并且包含了一些相关和互补的节点来扩展可回答性。而且，r - 半径 Steiner 图更加精确，因为非 Steiner 节点已经被排除了。

Li 等人[29]提出的 EASE 方法不是搜索 Steiner 树，而是搜索 r - 半径图（或称为“r - 半径 Steiner 树”）。为了加速寻找 r - 半径图，它对图进行了分区，具体方法是：首先对 r - 半径图进行聚类，从而得到多个簇；然后，基于簇对整个图进行分区，每个簇都对应图的一部分。为了加快搜索速度，EASE 为 r - 半径图设计了一个有效的图索引 EI-Index（Extended Inverted Index），而不是采用传统的倒排索引。因为传统的倒排索引对于基于文本和文档的搜索而言是很有效的，但是，对基于结构化和半结构化数据的关键词查询而言，传统的倒排索引无法发现最好的答案，尤其是无法发现最好的 r - 半径图。在传统倒排索引中，每个索引项是单个关键词，而在图索引 EI-Index 中，每个索引项是“关键词对”（即两个关键词的组合），该索引项对应的值是包含该“关键词对”的 r - 半径图及其评分。实验证明，这种图索引在确定图结构化信息方面是非常高效的。

2.2.3.2 基于多中心图的方法

基于多中心图的方法也被称为基于“社区”（Community）的方法。Qin 等人[37]认为，Steiner 树并不能很好地代表用户的查询兴趣，因此，他们提出了“社区”的概念。它是一个数据图中的多中心有向图，可以更好地反映用户查询兴趣。同时，Qin 等人也提出了一个算法，它可以在多项式时间延迟内枚举所有社区，而且算法所寻找到的社区是完整和无重复的。所谓完整性，就是指可以找到所有的社区。作者提出了一个弱的无重复概念，并在此基础上设计了一个多项式延迟算法，时间复杂度是 $O(l.(n.\log n + m))$，空间复杂度是 $O(l.n + m)$，其中，n 和 m 是数据图中的节点和边的数量，l 是用户查询关键词的数量。多项式延迟枚举算法是目前公认的最好的枚举算法。作者还提出了一个多项式延迟算法，时间复杂度是 $O(l.(n.\log n + m))$，空间复杂度是 $O(l^2.k + l.n + m)$，它可以采用排序的顺序精确地枚举 top - k 个社区，该算法的一个鲜明特色是，可以在查询过程中，随时让用户改变 k 的值，而不需要增加额外代价。此外，作者提出了一

个高效的索引方法来索引数据库，利用这个索引，可以为查询构建一个更小的数据图，并且可以寻找到同样的社区，从而大大减少搜索空间。

2.3 XML 数据库上的关键词查询

一个 XML 文档可以被看做是一棵有根标记树。树中的每个节点 v 对应着 XML 文档中一个 XML 元素，称为元素节点；树中的每个叶子节点对应着一个数值，称为值节点。一棵 XML 树中的每个节点，都采用唯一的 Dewey 编码进行标记。

XML 数据库上的关键词查询问题定义为：给定一个关键词查询 $Q = \{k_1, k_2, \cdots, k_l\}$ 和一棵 XML 树 T，找到一个有意义的子树集合 $\{(t, M)\}$，对于每棵子树 (t, M)，t 代表这棵子树的根节点，M 代表匹配节点（即包含一个查询关键词的节点），对于每个查询关键词而言，必须包含至少一个匹配节点。假设存在 m 个匹配节点，即 $M = <v_1, \cdots, v_m>$，则必须满足 $t = lca(v_1, v_2, \cdots, v_m)$，其中，$lca(v_1, v_2, \cdots, v_m)$ 表示 $v_1, v_2, \cdots, v_m$ 的 LCA（Lowest Common Ancestor，最近公共祖先）。

目前，研究人员已经提出了不同的方法来确定一个有意义的子树集合，主要包括基于 SLCA（Smallest LCA）的方法、基于 ELCA（Exclusive LCA）的方法和其他方法[54]。

2.3.1 基于 SLCA 的方法

采用基于 SLCA 的方法进行关键词查询时，一棵 XML 树 T 中的每个节点都被看成现实世界中的一个实体。如果节点 u 是节点 v 的祖先，那么，就可以认为节点 v 所代表的实体属于节点 u 所代表的实体。对于一个关键词查询，应该返回那些包含了查询关键词的最小的 LCA，即 SLCA（Smallest LCA）。

研究人员提出了一些高效的算法来获得 SLCA，主要包括 Guo 等人[17]提出的 StackAlgorithm、IndexedLookupEager 和 Xu 等人[51]提出的 ScanEager。不同的算法都有各自不同的特性，在不同的应用场景下具有不同的效率。

StackAlgorithm 的核心思想是，使用一个堆栈来模拟一个虚拟 XML 树的后序遍历，这棵 XML 树是由根节点到 Dewey 列表 $S_1, S_2, \cdots, S_l$ 中的每个节点的所有路径构成的并集。但是，StackAlgorithm 算法会平等地对待所有 Dewey 列表 $S_1, S_2, \cdots, S_l$，没有考虑到这些列表的大小的差别。实际上，$S_1, S_2, \cdots, S_l$ 的大小可能存在很大的差别。针对这种差别较大的情形，文献[51]提出了 IndexedLookupEager 算法来生成所有的 SLCA 节点。当这些列表的大小相差不大时，作者又进一步对 IndexedLookupEager 算法进行了改进，得到了 ScanEager 算法，它可以充分利用 IndexedLookupEager 算法严格按照升序访问列表 $S_1, S_2, \cdots, S_l$ 的特性，提高算法效率。

2.3.2 基于 ELCA 的方法

ELCA（Exclusive LCA）是 SLCA 的一个超集，它可以发现一些 SLCA 无法发现的相关信息。Guo 等人[17]最先提出了基于 ELCA 的方法，即 XRank。XRank 为 XML 结果树提

出了一个排序函数，它综合考虑了树中各个节点的积分情况。这些树节点会被以离线的方式赋予 Page-Rank 类型的积分。XRank 采用了一个门限算法（Threshold Algorithm）[13] 的变种来搜索 top-k 个子树。但是，它无法保证在任何情况下都具备很好的性能，在某些情形下，算法效率会变得很差。而且 XRank 也没有考虑关键词之间的相关性。

2.3.3 其他方法

其他 XML 数据库上的关键词查询方法包括 MLCA（Meaningful LCA）[31]、互连法[7]、CVLCA（Compact Valuable LCA）[25] 和基于相关性的排序[5]。MLCA 是在 SLCA 方法基础上发展起来的，而互连法得到的查询结果子树的根节点可能不是一个 SLCA 节点。CVLCA 方法融合了 ELCA 和互连法。

此外，Florescu 等人[16] 扩展了 XML 查询语言，从而支持 XML 元素粒度的关键词查询，但是，作者没有考虑关键词接近度。Hristidis 等人[21] 把一个 XML 数据库视为一个由最小 XML 片段组成的图，然后从中查找那些包含所有关键词的、由 XML 片段构成的最小连接树，作者主要关注查询结果的呈现，并且使用实视图技术来缩短查询响应时间。Shao 等人[40] 研究了对虚拟 XML 视图进行关键词查询的方法。Liu 等人[30] 研究了查询结果之间的区分度问题，从而可以使用有限的特征集合来最大程度地把一个查询结果和其他查询结果区分开来。

2.4 其他方面的研究

现有的工作都是研究针对单个 DBMS 的查询。但是，在传统和新兴应用中，分布式数据库广泛存在，这就需要面向多个数据库实现基于关键词的数据共享和查询。为了避免由查询大量无关数据库而带来的开销，Vu 等人[44] 提出了 G-KS。这是一种新的方法，可以根据候选者包含给定查询答案的可能性的大小来选择 top-k 个候选数据库。G-KS 用一个关键词关系图来表示一个数据库，其中，节点代表了关键词，边表示它们之间的关系。可以利用关键词关系图来计算每个数据库和关键词查询之间的相似性，从而使得在查询过程中，只对最有希望得到查询结果的数据库进行查询。

关键词查询研究的范围也越来越广，扩展到了包括关系数据库、XML[16, 17, 21]、数据流[34, 38]、分布式数据库[39, 44, 53]、空间数据库[14, 57]、OLAP[49, 58]、工作流[41] 和不确定数据[43] 等在内的各种环境。

3 国内研究进展

国内的研究目前主要关注关系数据库和 XML 数据库上的关键词查询。

3.1 关系数据库上的关键词查询

目前，国内的很多研究都集中在针对关系数据库的关键词查询。一些研究人员根据对该领域相关知识的调研，并结合自己的研究经验，撰写了具有指导意义的综述文章，比如王珊等人[47]、林子雨等人[27,28]的综述，为介绍和指导该领域的研究起到了积极的作用。

国内研究人员对一些关键问题也提出了创新性的解决方案。基于关系数据库的关键词查询面临的一个很大的挑战就是，如何改进查询结果的呈现方式，从而帮助用户快速浏览查询结果。彭朝晖等人[35]提出了一种基于数据库模式的结果展现方法 S-CBR（Schema-based Classification，Browsing and Retrieving），它结合了按结构聚类和按内容聚类这两种方法，将查询结果组织成两级类别，从而帮助用户快速浏览结果。文继军等人[48]提出的 SEEKER 是一个基于关键词的关系数据库信息检索系统，它不仅可以检索关系数据库里的文本属性，还可以检索数据库的元数据以及数字属性。蔡宏艳等人[10]在离线系统 DETECTOR 的基础上，设计并实现了增量更新方案，在不影响查询效率的前提下，可以最大程度地保证查询结果的准确性。王斌等人[46]为了增强关系数据库中的关键词查询结果，考虑了多表之间以及元组之间的语义关系，提出了一种语义评分函数，该语义评分函数不仅涵盖了当前的评分思想，并且加入新指标来衡量查询结果与查询关键字之间的相关性，保证了搜索结果的高查准率和查全率；基于该评分函数，作者提出了两种以数据块为处理单位的 top-k 搜索算法，改善了现有方法的查询性能。禹晓辉等人[55]提出了一种对查询结果进行评估的新方法 CI-Rank，不但考虑查询结果树中每个节点的重要性，而且考虑到整个结果树的连接紧密程度，从而避免了原有评价函数存在的许多问题，改善了查询效果。

很多研究方法大都采用即席的方式确定元组之间的关系，由于元组之间存在大量关系，因此这种方式通常效率较低。为此，李国良等人[26]提出了元组单元（Tuple Unit）的概念，并为关键词查询增加了数据预处理阶段，在这个离线处理阶段，需要搜索元组单元并对这些单元进行物化，在关键词查询时，就可以直接利用这些元组单元加速关键词查询的在线处理。但是，该研究只是使用单个元组单元来回答关键词查询，因此，冯建华等人[15]研究了如何使用多个元组单元来回答查询，并设计了新的索引结果来快速找到与查询相关的元组单元；作者还采用了新的排序技术和算法，实现了以渐进的方式生成 top-k 个查询结果。

由于排版错误或者缩写问题，数据库中会存在一些重复数据。针对这个问题，杨晓春等人[52]设计了一个名为 RESEARCH 的系统，提出了确定重复数据的有效方法，可以高效生成相关查询结果。

禹晓辉等人[56]研究了数据图较大无法放入内存的情形，提出了级联 top-k 关键词查询算法，与现有的其他方法不同的是，在查询的每个步骤，它并不是生成 Steiner 树，而是生成“超节点”，因此需要的内存空间更少，同时也加快了查询响应时间。

现有的研究大都在静态的数据库上进行关键词查询。以基于数据图的方法为例，从数据库转换得到的数据图，是数据库在某个时刻的一个快照，不会发生更新。但是，在实际应用中，数据库会经常更新，而用户对于某个特定的主题会表现出持久的兴趣。因此，Xu 等人[50]提出了高效的方法来支持针对关系数据库的、连续的 top-k 关键词查询，该方法以现有的面向关系数据流的关键词查询机制为基础，融合了评分排序机制，可以快速计算出静态数据库中的 top-k 个结果，并在数据库发生更新时，及时更新查询结果。

3.2 XML 数据库上的关键词查询

李国良等人[24]设计了针对 XML 关键词查询的 SAILER 系统，作者把网页和 XML 文档都建模成图的形式，并提出了“关键树”（Pivotal Tree）的概念，设计了高效的算法从图中快速找到 top-k 个评分最高的关键树。

李国良等人[29]提出的 EASE 可以普遍适用于非结构化数据（文本数据）、半结构化数据（XML 文档）、结构化数据（关系数据库）。EASE 首先把无结构、半结构化和结构化数据建模成图，然后对图进行汇总，并构建图索引，而不是传统的倒排索引，来支持关键词查询，作者还提出了一个新的排序机制来提高查询结果的有效性。

吉聪睿等人[22]以树的 Dewey 编码为基础，分析并证明了 XML 关键词检索中的核心概念 SLCA 的两个重要性质，并在其基础上提出了 Nearest Pair 算法。该算法采用二分迭代查找技术寻找最邻近点，将求解中间结果的次数降低了一个量级。

Lou 等人[32]的研究指出，现有的方法或者没有考虑相关性排序，或者在排序时采用了传统的基于文本的 IR 技术，而没有考虑 XML 片段的语义信息。因此，作者提出了 XML 片段之间的语义相似性的概念，并定义了语义相似性的度量方法，设计了新的 XML 关键词查询结果的排序机制。

4 结论与展望

关系数据库发展到今天，已经成为一种非常成熟的数据存储和管理技术，数据库厂商提供了高性能的、稳定的关系数据库产品，可以满足用户的多种数据管理需求。随着信息时代的发展，关系数据库中会存储越来越多的用户数据。与此同时，XML 数据库的应用也在迅速增加。以关系数据库和 XML 数据库为代表的数据库产品，只有提供简洁友好的信息查询方式，才能够为社会生产和日常生活提供最大的经济价值。数据库上的关键词查询技术，很好地迎合了市场对数据库产品的最新功能需求，将发挥越来越重要的作用。

结合国内外的研究现状，我们认为数据库上的关键词查询技术在未来的发展趋势将主要体现在以下几个方面：

1）聚合关键词查询：现有方法都利用关系数据库的主外键关联，输出与关键词查询

相关的、具有一定结构的元组集合。然而，在实际的企业应用中，元组集合中包含大量的结果，让用户自己从中寻找有用的信息是非常困难的。此外，Zhou 等人[59]通过大量的实际应用调研表明，元组集合的聚合结果对用户更有价值。因此，研究聚合关键词查询具有重要的应用价值。

2）适应数据库频繁更新的关键词查询：现有的关键词查询方法大都基于静态的数据库进行查询，无法应用于数据频繁更新的数据库环境。一些研究（比如 Xu 等人的研究[50]）已经开始关注数据库频繁更新问题，并给出相应解决方案。在未来，这个方面需要更多的研究。

3）多种数据库环境下的关键词查询：数据仓库、数据流、空间数据库和分布式数据库等都存储了大量的企业数据，为了让企业用户能够方便地访问这些数据，需要引入关键词查询。针对上述多种数据库环境下的关键词查询技术，将吸引越来越多研究人员的关注。

4）融合关键词查询技术的数据库管理系统：目前市场上还没有出现可以支持关键词查询技术的数据库产品。一些研究人员设计的原型系统，不仅功能和界面仍显粗糙，而且无法和具体的数据库产品实现内部的技术融合。未来支持关键词查询的数据库产品，应该把关键词查询技术与自己产品的内部实现技术进行紧密结合，以期获得最大的性能优化。

参考文献

[1] Sanjay Agrawal, Surajit Chaudhuri, Gautam Das. DBXplorer: A System for Keyword-Based Search over Relational Databases. ICDE 2002: 5-16.

[2] Sihem Amer-Yahia, Pat Case, Thomas Rölleke, Jayavel Shanmugasundaram, and Gerhard Weikum. Report on the DB/IR Panel at SIGMOD 2005. SIGMOD Record, 2005, 34(4): 71-74.

[3] Sihem Amer-Yahia, Jayavel Shanmugasundaram. XML Full-Text Search: Challenges and Opportunities. VLDB 2005: 1368, 2005.

[4] Gaurav Bhalotia, Arvind Hulgeri, Charuta Nakhe, Soumen Chakrabarti, S. Sudarshan. Keyword Searching and Browsing in Databases using BANKS. ICDE 2002: 431-440.

[5] Zhifeng Bao, Tok Wang Ling, Bo Chen, and Jiaheng Lu. Effective XML Keyword Search with Relevance Oriented Ranking. In Proc. 25th Int. Conf. on Data Engineering, 2009, pages 517-528.

[6] Surajit Chaudhuri, Gautam Das. Keyword Querying and Ranking in Databases. PVLDB 2(2): 1658-1659.

[7] Sara Cohen, Jonathan Mamou, Yaron Kanza, Yehoshua Sagiv. XSEarch: A Semantic Search Engine for XML. VLDB 2003: 45-56.

[8] Yi Chen, Wei Wang, Ziyang Liu. Keyword-based Search and Exploration on Databases. ICDE 2011: 1380-1383.

[9] Yi Chen, Wei Wang, Ziyang Liu, Xuemin Lin. Keyword Search on Structured and Semi-Structured Data. SIGMOD 2009: 1005-1010.

[10] 蔡宏艳，姚佳丽，王珊. DETECTOR：基于关系数据库通用的在线关键词查询系统. 软件学报. 2007，44(1)：119-125.

[11] Bhavana Bharat Dalvi, Meghana Kshirsagar, S. Sudarshan：Keyword Search on External Memory Data Graphs. PVLDB 1(1)：1189-1204 (2008).

[12] Bolin Ding, Jeffrey Xu Yu, Shan Wang, Lu Qin, Xiao Zhang, Xuemin Lin. Finding Top-k Min-Cost Connected Trees in Databases. ICDE 2007：836-845.

[13] Ronald Fagin, Amnon Lotem, Moni Naor. Optimal Aggregation Algorithms for Middleware. J. Comput. Syst. Sci. (JCSS), 2003, 66(4)：614-656.

[14] Ian De Felipe, Vagelis Hristidis, Naphtali Rishe. Keyword Search on Spatial Databases. ICDE 2008：656-665.

[15] Jianhua Feng, Guoliang Li, Jianyong Wang. Finding Top-k Answers in Keyword Search over Relational Databases Using Tuple Units. IEEE Trans. Knowl. Data Eng. (TKDE), 2011, 23(12)：1781-1794.

[16] Daniela Florescu, Donald Kossmann, Ioana Manolescu. Integrating Keyword Search into XML Query Processing. Computer Networks (CN), 2000, 33(1-6)：119-135.

[17] Lin Guo, Feng Shao, Chavdar Botev, Jayavel Shanmugasundaram. XRANK：Ranked Keyword Search over XML Documents. SIGMOD 2003：16-27.

[18] Hao He, Haixun Wang, Jun Yang, Philip S Yu. BLINKS：Ranked Keyword Searches on Graphs. SIGMOD 2007：305-316.

[19] Vagelis Hristidis, Luis Gravano, Yannis Papakonstantinou. Efficient IR-Style Keyword Search over Relational Databases. VLDB 2003：850-861.

[20] Vagelis Hristidis, Yannis Papakonstantinou. DISCOVER：Keyword Search in Relational Databases. VLDB 2002：670-681.

[21] Vagelis Hristidis, Yannis Papakonstantinou, Andrey Balmin. Keyword Proximity Search on XML Graphs. ICDE 2003：367-378.

[22] 吉聪睿，邓志鸿，唐世渭. 基于 Nearest Pair 的 XML 关键词检索算法[J]. 软件学报，2009，20(4)：910-917.

[23] Varun Kacholia, Shashank Pandit, Soumen Chakrabarti, S Sudarshan, Rushi Desai, Hrishikesh Karambelkar. Bidirectional Expansion For Keyword Search on Graph Databases. VLDB 2005：505-516.

[24] Guoliang Li, Jianhua Feng, Jianyong Wang, Xiaoming Song, Lizhu Zhou. Sailer：An Effective Search Engine for Unified Retrieval of Heterogeneous XML and Web Documents. WWW 2008：1061-1062.

[25] Guoliang Li, Jianhua Feng, JianyongWang, and Lizhu Zhou. Effective Keyword Search for Valuable LCAS over XML Documents. In Proc. 16th ACM Conf. on Information and Knowledge Management, pages 31-40, 2007.

[26] Guoliang Li, Jianhua Feng, Lizhu Zhou. Retune：Retrieving and Materializing Tuple Units for Effective Keyword Search over Relational Databases. ER 2008：469-483.

[27] 林子雨，杨冬青，王腾蛟，张东站. 基于关系数据库的关键词查询[J]. 软件学报. 2010，21(10)，2454-2476.

[28] 林子雨，左思强，赖永炫，张东站. DB&IR 系统研究综述[J]. 计算机研究与发展，2010，47(Suppl.)：176-180.

[29] Guoliang Li, Beng Chin Ooi, Jianhua Feng, Jianyong Wang, Lizhu Zhou. EASE：An Effective 3-in-1 Keyword Search Method for Unstructured, Semi-Structured and Structured Data. SIGMOD Conference 2008：

903-914.

[30] Ziyang Liu, Peng Sun, Yi Chen. Structured Search Result Differentiation. PVLDB 2(1): 313-324 (2009).

[31] Yunyao Li, Cong Yu, H. V. Jagadish. Enabling Schema-Free XQuery with Meaningful Query Focus. VLDB J. 2008, 17(3): 355-377.

[32] Ying Lou, Zhanhuai Li, Qun Chen. Semantic Relevance Ranking for XML Keyword Search. Inf. Sci. (ISCI) 190: 127-143 (2012).

[33] Yi Luo, Xuemin Lin, Wei Wang, Xiaofang Zhou. Spark: Top-k Keyword Query in Relational Databases. SIGMOD 2007: 115-126.

[34] Alexander Markowetz, Yin Yang, Dimitris Papadias. Keyword Search on Relational Data Streams. SIGMOD 2007: 605-616.

[35] 彭朝晖, 张俊, 王珊. S-CBR: 基于数据库模式展现数据库关键词检索结果[J]. 软件学报, 2008, 19(2): 323-337.

[36] Lu Qin, Jeffrey Xu Yu, Lijun Chang. Keyword Search in Databases: The Power of RDBMS. SIGMOD Conference 2009: 681-694.

[37] Lu Qin, Jeffrey Xu Yu, Lijun Chang, Yufei Tao. Querying Communities in Relational Databases. ICDE 2009: 724-735.

[38] Lu Qin, Jeffrey Xu Yu, Lijun Chang, Yufei Tao. Scalable Keyword Search on Large Data Streams. ICDE 2009: 1199-1202.

[39] Mayssam Sayyadian, Hieu LeKhac, AnHai Doan, Luis Gravano. Efficient Keyword Search Across Heterogeneous Relational Databases. ICDE 2007: 346-355.

[40] Feng Shao, Lin Guo, Chavdar Botev, Anand Bhaskar, Muthiah Chettiar, Fan Yang, and Jayavel Shanmugasundaram. Efficient Keyword Search over Virtual XML Views. VLDB J., 2009, 18(2): 543-570.

[41] Qihong Shao, Peng Sun, Yi Chen. WISE: A Workflow Information Search Engine. ICDE 2009: 1491-1494.

[42] Alkis Simitsis, Georgia Koutrika, Yannis E. Ioannidis. Précis: From Unstructured Keywords as Queries to Structured Databases as Answers. VLDB J. 2008, 17(1): 117-149.

[43] Xiaoming Song, Guoliang Li, Jianhua Feng, Lizhu Zhou: Effective Fuzzy Keyword Search over Uncertain Data. DASFAA 2009: 66-70.

[44] Quang Hieu Vu, Beng Chin Ooi, Dimitris Papadias, Anthony K. H. Tung. A graph Method for Keyword-based Selection of The Top-K Databases. SIGMOD 2008: 915-926.

[45] Shan Wang, Zhaohui Peng, Jun Zhang, Lu Qin, Sheng Wang, Jeffrey Xu Yu, Bolin Ding. NUITS: A Novel User Interface for Efficient Keyword Search over Databases. VLDB 2006: 1143-1146.

[46] 王斌, 杨晓春, 王国仁. 关系数据库中支持语义的 Top-K 关键字搜索[J]. 软件学报, 2008, 19(9): 2362-2375.

[47] Shan Wang, Kunlong Zhang. Searching Databases with Keywords. J. Comput. Sci. Technol. (JCST), 2005, 20(1): 55-62.

[48] 文继军, 王珊. SEEKER : 基于关键词的关系数据库信息检索[J]. 软件学报, 2005, 16(7): 1270-1281.

[49] Ping Wu, Yannis Sismanis, Berthold Reinwald. Towards Keyword-Driven Analytical Processing. SIGMOD 2007: 617-628.

[50] Yanwei Xu. Scalable Continual Top-k Keyword Search in Relational Databases CoRR abs/1108.4516 (2011).

[51] Yu Xu and Yannis Papakonstantinou. Efficient Keyword Search for Smallest LCAs in XML Databases. SIGMOD 2005: 537-538.

[52] Xiaochun Yang, Bin Wang, Guoren Wang, Ge Yu. Enhancing Keyword Search in Relational Databases Using Nearly Duplicate Records. IEEE Data Eng. Bull. (DEBU), 2010, 33(1): 60-66.

[53] Bei Yu, Guoliang Li, Karen R Sollins, Anthony K. H. Tung. Effective Keyword-Based Selection of Relational Databases. SIGMOD 2007: 139-150.

[54] Jeffrey Xu Yu, Lu Qin, Lijun Chang. Keyword Search in Databases. Morgan & Claypool Publishers 2010.

[55] Xiaohui Yu, Huxia Shi. CI-Rank. Ranking Keyword Search Results Based on Collective Importance. ICDE 2012: 78-89.

[56] Ziqiang Yu, Xiaohui Yu, Yang Liu. Cascading Top-k Keyword Search over Relational Databases. DOLAP 2011: 95-100.

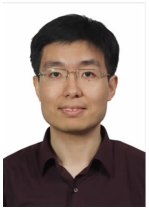
[57] Dongxiang Zhang, Yeow Meng Chee, Anirban Mondal, Anthony K. H. Tung, Masaru Kitsuregawa. Keyword Search in Spatial Databases: Towards Searching by Document. ICDE 2009: 688-699.

[58] Bin Zhou, Jian Pei. Answering Aggregate Keyword Queries on Relational Databases Using Minimal Group-bys. EDBT 2009: 108-119.

[59] Bin Zhou, Jian Pei. Aggregate Keyword Search on Large Relational Databases. Knowl. Inf. Syst. (KAIS), 2012, 30(2): 283-318.

作者简介

禹晓辉 博士，山东大学计算机学院特聘教授。CCF 数据库专业委员会委员。主要研究方向为大数据管理、关键词查询、时空数据管理、查询优化、意见挖掘等。



林子雨 博士，厦门大学计算机科学系助理教授。CCF 会员。主要研究方向是数据库、数据仓库和数据挖掘。受厦门大学基础创新科研基金（中央高校基本科研业务费专项资金）资助（NO. 2011121049）。