Vertical Database Design for Scalable Data Mining

William Perrizo
*North Dakota State University, USA*

Qiang Ding
*Concordia College, USA*

Masum Serazi
*North Dakota State University, USA*

Taufik Abidin
*North Dakota State University, USA*

Baoying Wang
*North Dakota State University, USA*

INTRODUCTION

For several decades and especially with the preeminence of relational database systems, data is almost always formed into horizontal record structures and then processed vertically (vertical scans of files of horizontal records). This makes good sense when the requested result is a set of horizontal records. In knowledge discovery and data mining, however, researchers are typically interested in collective properties or predictions that can be expressed very briefly. Therefore, the approaches for scan-based processing of horizontal records are known to be inadequate for data mining in very large data repositories (Han & Kamber, 2001; Han, Pei, & Yin, 2000; Shafer, Agrawal, & Mehta, 1996).

On the contrary, more and more advantages of using vertical data organization have been realized. For example, it makes hardware caching work well, it makes compression easy to do, and it may greatly increase the effectiveness of the I/O device since only participating fields are retrieved instead of the whole record. The vertical decomposition of a relation also permits a number of transactions to execute concurrently. Recently, much effort has been focused on subsampling and indexing to address problems of scalability. However, subsampling requires that the subsampler knows enough about the large data set in the first place, to subsample "representatively." That is, subsampling representatively presupposes considerable knowledge about the data. For many large data sets, such knowledge may be inadequate or nonexistent.

Index files are vertical structures and they are vertical access paths to sets of horizontal records. Some indices, such as the bit-sliced index (BSI; Chan & Ioannidis, 1998; O’Neil & Quass, 1997; Rinfret, O’Neil, & O’Neil, 2001) and encoded bitmap index (EBI; Wu, 1998; Wu & Buchmann, 1998), do address the scalability problem in many cases, but they do so at the cost of creating and maintaining additional index files separate from the data files.

Another approach, which is different from the above conceptually, is to build the whole database vertically. Such a database can be used not only for routine data management, but also for data mining. Unlike the horizontal databases, which are stored horizontally and processed vertically, vertical databases are stored vertically and processed horizontally. With other characteristics, vertical databases are shown to address the scalability issues.

BACKGROUND

The concept of vertical data files, in fact, is not new at all. Copeland and Khoshafian (1985) presented an attribute-level decomposition storage model called DSM, similar to the attribute transposed file model (ATF; Batory, 1979) that stores each column of a relational table into a separate table. However, DSM was shown to perform well. It utilizes surrogate keys to map individual attributes together, hence requiring a surrogate key to be associated with each attribute of each record in the database. Attribute-level vertical decomposition is also
3 more pages are available in the full version of this document, which may be purchased using the "Add to Cart" button on the product's webpage:
www.igi-global.com/chapter/vertical-database-design-scalable-data/11232?camid=4v1

www.igi-global.com/e-resources/library-recommendation/?id=1

Related Content

Concurrency Control for Replicated Data in Distributed Real-Time Systems
www.igi-global.com/article/concurrency-control-replicated-data-distributed/51162?camid=4v1a

The Expert’s Opinion
www.igi-global.com/article/expert-opinion/51116?camid=4v1a

Set Comparison in Relational Query Languages
www.igi-global.com/chapter/set-comparison-relational-query-languages/11215?camid=4v1a

Evaluating the Dynamic Behavior of Database Applications
Zhen He and Jérôme Darmont (2005). *Journal of Database Management* (pp. 21-45).
www.igi-global.com/article/evaluating-dynamic-behavior-database-applications/3330?camid=4v1a