Managing Virtual Data Marts with Metapointer Tables

Leslie K. Hodge
Storage Technology Corporation
leslie_hodge@storagetek.com

Charles Milligan
Storage Technology Corporation
charles_milligan@storagetek.com

Abstract

Although Internet/Intranet topology has radically evolved communications for information workers, the access and transfer of large volumes of data through this topology is still inefficient and is not inherently conducive to point-in-time recall. Because information managers make real time decisions based on relevant subsets of an enterprise data warehouse, data marts are required for optimized data accessibility and use. Storage virtualization, based on pointer systems, has introduced new instant copy functions which do not require that data be actually stored outside the data warehouse even though it is referenced in a variety of contexts (e.g., multiple data marts utilizing overlapping subsets of the data warehouse). The purpose of this paper is to describe a method enabling users to access relevant subsets of large data populations residing in different data warehouses. This method enables the creation of data marts for use by local decision support software without requiring the data mart data population to be stored locally. This will save storage space and data transmission resources, and simplify synchronization of the data mart and the warehouse when updates occur.

1. Introduction

The concept of the data warehouse was developed to enable a company to organize enterprise wide management of their data assets and try to invoke a level of standardization while minimizing duplication. The data mart arose from the size and cost complexity of generating full data warehouses, and is essential for enabling end users to efficiently access the data they seek. The variations of these concepts (and their corresponding definitions) have yielded a plethora of locally applicable definitions and relationships of the data warehouse and the data mart. A common element of these definitions is that the value of a data mart is measured by the efficiency of delivering a relevant and complete subset of data residing in a data warehouse to perform decision support. In other words, the rules governing the creation of a data mart are essentially the rules for identifying data that is rich in a specific class of information required for decision support. Theoretically, a self-contained enterprise could host a data warehouse that provided the data population needed to deliver maximum value to every data mart user. However, a current trend in organizations seeking to 'right-size' is to out-source services to independent organizations outside the enterprise. In this case, subsets of the data warehouse, and instances of data marts that resided within the enterprise, are now managed by external organizations and perhaps are even hosted elsewhere. This raises a set of data security and data alignment issues that are compounded by multiple instances of data residency.

This phenomenon of data warehouses residing outside the enterprise is becoming more commonplace as data mart users require greater quantities of relevant data. For example, the Internet has developed as a ubiquitous although chaotic data warehouse where users are constantly acquiring data from the chaotic world wide warehouse into some measure of control within the enterprise. However such acquisition although important to effective decision making, may result in a compendium of data which is difficult to support accurately and consistently within the data marts data structures. From this trend emerging issues of added cost and data management complexity arise. The purpose of this paper is to promote a metadata-centric method that manages the location of data elements and facilitates data mart-based decision support based upon the evaluation of metadata without requiring the entire data population supporting the data mart to be stored locally.

2. Are we working too hard and storing too much?

Data elements generally reside in a data warehouse and also reside within the related data marts. They are traditionally referenced from within a data mart using access calls created by an application or system service

Proceedings of the 36th Hawaii International Conference on System Sciences (HICSS'03)
0-7695-1874-5/03 $17.00 © 2002 IEEE

executing within the data mart. The response to such access calls is that the data is fetched from the data mart and introduced into the decision support environment. Decision support is then executed and is influenced by the value and relevancy of that data element.

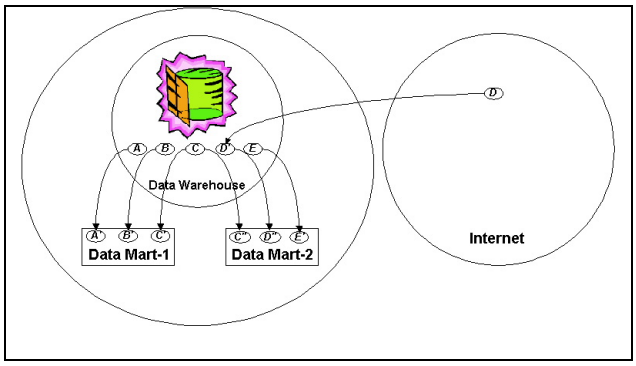

Figure 1. Traditional relationship of data elements as they reside in a data warehouse, its associated data marts, and the Internet

For example, consider a traditional relationship of a data warehouse and two data marts as indicated in Figure 1. Data Mart-1 uses data elements "A", "B", and "C" in its decision support. These data elements are copied into this data mart during an asynchronous copy process. The "A'", "B'", and "C'" notations indicate that once these elements are copied into the data mart they are subject to local updates within the data warehouse without the original data elements also being updated in the data mart, and vice versa. With two copies of the same data element residing in two locations, twice the storage required to host the same data element is being consumed.

For Data Mart-2 (as referenced in Figure 1), data elements "C", "D" and "E" are used its decision support. Because the data element "C" is the basis for decision support for Data Mart-1 (and therefore the subject of asynchronous copy processes prerequisite to decision support in both Data Mart-1 and Data Mart-2) this data element is referenced as "C'" in Data Mart-2. The significance of this notation is that "C" could be updated in the data warehouse, but without a synchronous copy process initiated in the Data Mart (pull) or in the associated Data Warehouse (push) the data used for decision support may not be the most accurate or relevant data. The data element "D" originally resides in the Internet, and was asynchronously copied into the Data Warehouse. Because this data element is also used for decision support in Data Mart-2, it is again asynchronously copied into Data Mart-2. In this case, three copies of the same data element now reside on storage somewhere. And these copies will be subject to content inconsistency if they are not synchronously propagated when one version is updated. Decision support using inconsistent data elements can be very counter-productive.

To solve the data consistency challenge in a traditional data warehouse, the asynchronous copying of data elements would need to become a synchronous copy process. This process would be initiated when the data element is updated in one location, then the updated data element would need to be propagated to (1) all applicable data warehouses and (2) their associated data marts. The data elements propagated in this manner would impose I/O transfer overhead for the entire data element without regard whether that updated data element will be used in decision support prior to the next update. In addition, that decision support will rely upon a static 'one size fits all' representation of the data element that is assumed to be relevant and accurate for all data marts.

3. A solution: just point

Several issues associated with the management and accuracy of traditional data marts have been identified, including:

- The additional storage resources of preserving copies the same data element in multiple locations
- The lack of data element consistency used for decision support in multiple data marts (due to the asynchronous nature of data transfers between a data warehouse and its associated data marts) directly influences decision support accuracy
- Any attempt to adopt synchronous updated data element propagation will result in higher I/O workloads

The solution begins with the concept of a Virtual Data Mart. While a Traditional Data Mart uses local storage resources to host copies of a data element, a Virtual Data Mart uses a metadata table structure to reference data elements within the data warehouse or reference another metadata table structure residing in another data warehouse hosting the target data element. This metadata table structure, called a Metapointer Table, is where metadata associated with data elements is recorded. This method has some unique qualities:

- The target data element is stored in just one location
- No local disk storage is required for the data mart

- No data elements are actually transferred unless the pre-access decision support outcome specifies a full data transfer
- The data transfer would be facilitated in a manner where the environmental-dependent transfer method is executed in a manner that is transparent to the requesting data mart
- If requested, a condition test can be made against several metadata definitions to determine which of several data elements should be transferred in whole.
- Data elements can reside outside the enterprise data warehouse in an external data warehouse
- Synchronization and revision journaling functions would use this metadata table structure as its information hub

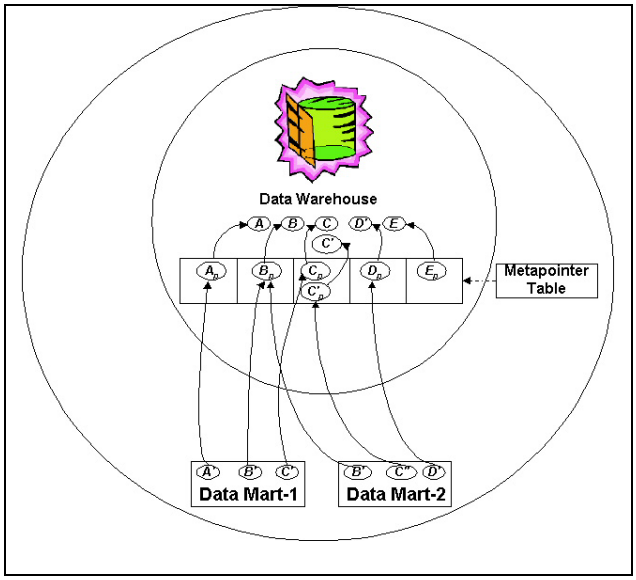

Figure 2. Basic relationship of virtual data marts with a Metapointer table residing in a data warehouse

In Figure 2, an example of two data marts and their relationship with the Metapointer Table is displayed. Data Mart-1 references data elements "A", "B", and "C". But to access these elements, the access call would actually reference the metapointer table with a call targeting the data element by symbolic name. Metadata associated with data elements "A", "B" and "C" resides within the Metapointer table. A component of this metadata is a pointer to the physical location of the data element in the data warehouse span of control. In some instances, it may actually be desirable to point beyond the warehouse span of control to areas outside the warehouse where data is located. This is true when the data is owned and maintained by a trusted "other" but allowed to be used in the business operations. Some internet based locations would qualify in this usage while others would not. This information is used to facilitate an access of the data element.

Another function of the Metapointer Table is to be a location where metadata associated with data element updates can be recorded. For example in Figure 2, data element "C" has been updated, with the updated version referenced as "C'". The Metapointer Table can support pointers to both versions of data element "C". In this example, Data Mart-2 references data elements "B", "C'" and "D".

4. A closer look at the metapointer table structure

The focal point of this method is a protected, open-ended metapointer table. A metapointer table resides within the data warehouse where the target data mart(s) reside(s).

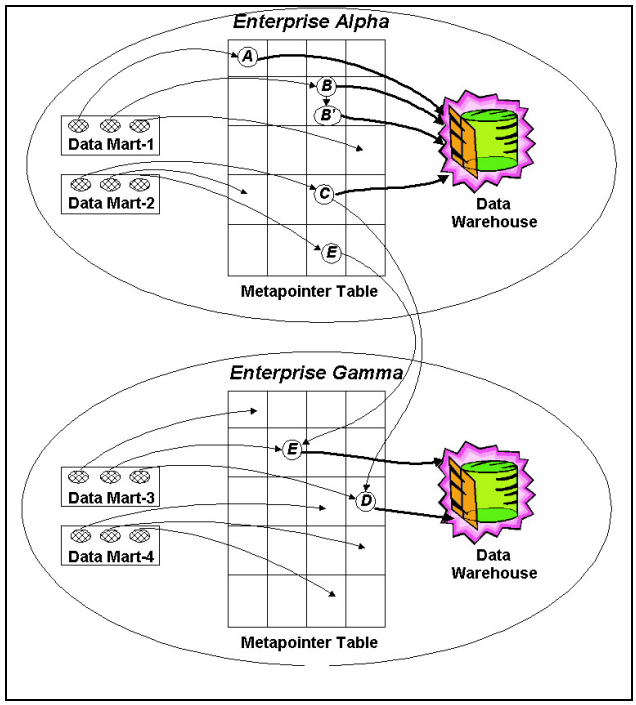

Figure 3. Basic relationship of virtual data marts with a Metapointer Table residing in a data warehouse

The Metapointer Table contains metadata that facilitates the I/O access and a fast-look evaluation of

defined data elements. In Figure 3, several use scenarios are displayed:

- 'A' represents a data element referenced by Data Mart-1 in the 'Alpha' Enterprise. A single version of this data element resides in this enterprise's data warehouse.
- 'B represents a data element referenced by Data Mart-1 in the 'Alpha' Enterprise. Two versions of this data element reside in this enterprise's data warehouse.
- 'C' represents a data element referenced by Data Mart-2 in the 'Alpha' Enterprise. A version of this data resides in the 'Alpha' Enterprise, and another version of this data element resides in the 'Gamma' Enterprise.
- 'D' represents a data element referenced by Data Mart-3 in the 'Gamma' Enterprise. A version of this data resides in the 'Gamma' Enterprise, but is also referenced via a Metapointer Table entry in the 'Alpha' Enterprise.
- 'E' represents a data element referenced by Data Mart-2 in the 'Alpha' Enterprise. The metadata for this data element, as it resides in the 'Alpha Enterprise', indicates that instead of a pointer to the physical location of this data element within the 'Alpha' Enterprise, the metadata pointer instead points to the Metapointer Table residing in the 'Gamma' Enterprise. It is within this enterprise. Data Mart-3 also references this data element within the 'Gamma' Enterprise's Metapointer Table, which in turn points to the location of data element 'E' as it resides within the 'Gamma' Enterprise data warehouse.

5. What a metapointer table may contain

The metadata, associated with data elements, may be represented in the Metapointer Table by variable-length pointer table entries using an Anchor/Point in Time (PIT) Entry metadata structure. The anchor element of a pointer table entry may include, but not be limited to, the following:

- symbolic name of the data element
- number of PIT entries for this data element
- pointers to next and previous Anchor Element, or end-of-sequence flag
- offset to first PIT Entry in this pointer structure

A PIT entry is created when a data element is defined to the data mart, or when the data element is updated based upon a local policy defined the number of revisions of the data element that should be tracked. A PIT Entry element of a pointer table entry may include, but not be limited to, the following:

- status bits for data element (i.e. if data element is shared with another data mart, if the data element has a dependent relationship with another data element, etc.)
- class of the data element (to limit access to specific class data marts)
- physical location of the data element (data warehouse, storage subsystem, volume/LUN address, starting block offset)
- data element size
- generation number of this data element
- sequence number of this data element (if part of a historical update journal)
- creation/revision timestamp
- read/write capability for resident data element
- a DIF characterization between this data element and the previous version of same data element (represented as a correlation factor)
- Pointers to previous and next data element in pointer sequence, or end-of-sequence flag

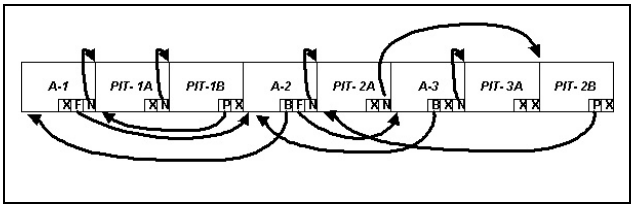

Figure 4. Sample Metapointer Table entry structure

In Figure 4, three different data elements are represented. Each data element has an anchor element (referenced as 'A-1', 'A-2', and 'A-3'. Each anchor element is forward chained to the next anchor element (referenced as 'F' in Figure 4) and backward chained to the previous anchor element (referenced as 'B'). For Data Element #1 (anchor element 'A-1'), the metadata for two versions (PIT-1A and PIT-1B) of that data element are referenced in the Metapointer Table. PIT elements referencing the same anchor are forward chained (referenced as 'N'), and backward chained (referenced as 'P' in Figure 2). For the Data Element #2 (anchor element 'A-2'), the metadata for an initial version of this data element is referenced as "PIT-2A". A second version of this data element was introduced to the Metapointer Table after the definition of Data Element #3 (anchor element 'A-3'). In this scenario, the second version of Data Element #2 (PIT element 'PIT-2B') is chained to the first versions PIT element, but physically resides in the Metapointer Table away from related PIT entries for the same data element.

Serially recording Metapointer Table entries supports a most efficient use of this resources storage, and illustrates a benefit of using forward and backward pointers.

6. Decision support without accessing entire data elements

For data elements with multiple versions, the use of a Metapointer Table structure facilitates an automated decision support environment that performs intelligent selection from multiple generations of the same data element. Many current processes that facilitate decision support require a user to perform multiple accesses of related data elements to determine the data element to be selected for use. The access of these multiple data elements imposes I/O overhead within the enterprise, and possibly between enterprises (including the Internet). Add to this the user overhead of determining which version of a data element to use (assuming all eligible data elements have been fetched), and the result is a decision support environment with a significant decision support process inefficiency.

The use of a Metapointer Table structure reduces that inefficiency because the metadata for all the eligible data elements are located and presented for a software-based or user-driven intelligent pre-access selection. Assuming the Metapointer Table infrastructure reinforces data consistency, the selection process can yield the most relevant and accurate version of a data element for a given purpose facilitated by the data mart. While the reduced I/O operations implied by this structure should reduce data transfer overhead, the primary benefit of this structure lies within its ability to promote robust decision support.

7. How is data accessed?

At the time a process facilitating decision support is initiated, an inline access query is made for a data element. This query, targeting the data element by its symbolic name, can initiate an immediate access of the data element or support pre-access processing of eligible versions of a targeted data element. Immediate access processing parameters may include, but not be limited to, the following:

- Fetch newest or oldest version of data element
- Fetch data element by element size
- Fetch data element by data warehouse or storage subsystem

Pre-access processing reads the metadata associated with a targeted data element to populate a data mart buffer with a sequential metadata representation of the data elements conforming with the pre-access processing parameters. These parameters may include, but not be limited to, the following:

- Fetch metadata for all data elements by range of sizes
- Fetch metadata for all data elements by DIF characterization
- Fetch metadata for all data elements by specific timestamp range
- Fetch metadata for all data elements by status or class bits

The metadata written to this data mart buffer can then be targeted by a local application that presents options to select the optimal version of the data element as decision support input. This intelligent selection of the optimal version of the data element is achieved without physically accessing all the versions of the data element. Only after the selection of the optimal version of the data element is an access performed against the disk-resident data element.

8. Data consistency within a query context

To this point, the discussion about the Virtual Data Mart solution using a Metapointer Table has focused on issues of efficient data accessibility and optimized resource utilization. But an important underlying principle of data synchronization must be addressed as a component of this solution for Virtual Data Marts to become a reality.

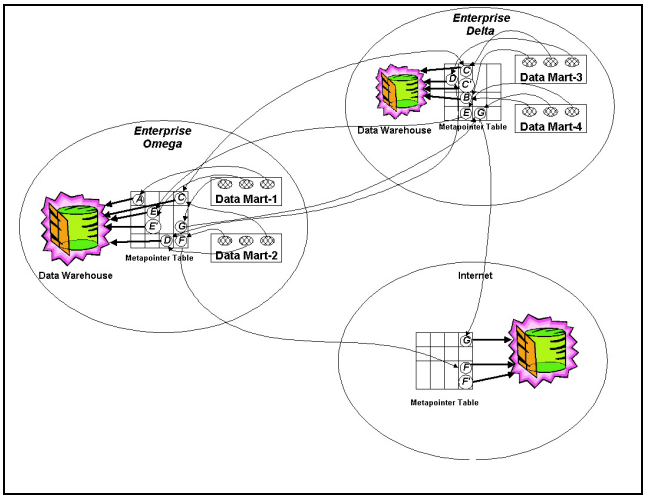

Figure 5. Relationship of Data Elements residing in three data warehouses

The use of linked Metapointer Table structures between multiple enterprises yield several synchronization and accessibility scenarios. In Figure 5, the following scenarios are displayed:

- 'A' represents a data element referenced by Data Mart-1 in the 'Omega' Enterprise. Because a single version of this data element resides in this enterprise's data warehouse, and this element is not referenced from another enterprise, the metadata for data element 'A' is updated from within the 'Omega' Enterprise.
- 'B' represents a data element referenced by Data Mart-3 and Data Mart-4 in the 'Delta' Enterprise. A single version of this data element resides in this enterprise's data warehouse. In this scenario a benefit of the Metapointer table structure is realized when this data element is updated. Metadata reflecting this update is recorded in a single Metapointer Table location, which is referenced by these data marts. Both these data marts will pick up the updated data element version without any change to the access calls within these data marts.
- 'C' represents a data element with three versions residing in two enterprises. Two data marts residing in these enterprises reference data element 'C'. Two versions of this data element reside in the 'Delta' enterprise, and the third version of data element 'C' resides in the 'Omega' enterprise. With a linked Metapointer Table structure, Data Mart-1 not only can reference the metadata (and therefore the data element itself) associated with the version residing in this enterprise, but it also can reference the two versions of data element 'C' residing in the 'Delta' enterprise. Likewise, Data Mart-3 in the 'Delta' enterprise not only can reference the two versions of data element 'C' residing within its enterprise, but in addition can reference the version of this data element residing in the 'Omega' enterprise. This means that Data Mart-1 and Data Mart-3 can perform pre-access processing against the metadata for versions of data element 'C' that do not reside in that data mart's enterprise. The Metapointer Table entries for this data element would include pointers to linked Metapointer Tables residing within other enterprises to support this cross-enterprise accessibility.
- 'D' represents a data element with two versions residing in two enterprises. Two data marts residing in these enterprises reference data element 'D'. One version of this data element resides in the 'Delta' enterprise, and the second version of data element 'D' resides in the 'Omega' enterprise. With a linked Metapointer Table structure, Data Mart-2 not only can reference the metadata (and therefore the data element itself) associated with the version residing in this enterprise, but it also can reference the version of data element 'D' residing in the 'Delta' enterprise. Likewise, Data Mart-3 in the 'Delta' enterprise not only can reference the version of data element 'D' residing within its enterprise, but in addition can reference the version of this data element residing in the 'Omega' enterprise. This means that Data Mart-2 and Data Mart-3 can perform pre-access processing against the metadata for versions of data element 'D' that do not reside in that data mart's enterprise. The Metapointer Table entries for this data element would include pointers to linked Metapointer Tables residing within other enterprises to support this cross-enterprise accessibility.
- 'E' represents a data element referenced by Data Mart-1 with two versions residing in the 'Omega' Enterprise. Because there are two versions associated with this data element, two sets of metadata can be used by Data Mart-1 for pre-access processing to determine the optimal version of this data element to access.
- 'F' represents a data element with two versions residing on the Internet. This data element is referenced by Data Mart-1 in the 'Omega' enterprise. A Metapointer Table entry on the Omega Enterprise contains a link to the Internet Metapointer Table where the metadata associated with the two versions of data element 'F' reside. This metadata can be brought into the Data Mart-2 environment for pre-access processing to determine the optimal version of this data element to access from the Internet.
- 'G' represents a data element residing on the Internet. This data element is referenced by Data Mart-1 in the 'Omega' Enterprise and also by Data Mart-4 in the 'Delta' Enterprise. A Metapointer Table entry in the 'Omega' Enterprise contains a link to a Metapointer Table entry in the 'Delta' Enterprise, which in turn has a link to the Internet Metapointer Table where the metadata associated with data element 'G' resides.

9. Conclusion

The Internet is emerging as a chaotic worldwide warehouse. Many businesses never quite manage to overcome the complexity of building a contained enterprise data warehouse. The business world is managed by a highly segmented user base of decision-makers that require subsets of warehoused data. Individually, these are three highly demanding conditions. But when these conditions converge their

combined impact is overwhelming. The concept of the Virtual Data Mart using a Metapointer Table addresses this convergence by providing an infrastructure that reinforces data consistency while supporting multiple versions of a data element. Decision support is vastly improved by an infrastructure that delivers the right data while improving the availability and accessibility of that data. Data mart users will aggressively demand innovations that improve the speed, relevancy, and accessibility of data presented in data marts. As wireless technology continues to evolve and gain popularity, the use of diskless devices to access the Internet and other data sources will become more widespread. This is yet another factor that will influence the use of metadata processing to access centrally resident data elements. The solution presented in this paper will require a significant amount of development and research to be realized economically.

Proceedings of the 36th Hawaii International Conference on System Sciences (HICSS'03)
0-7695-1874-5/03 $17.00 © 2002 IEEE