Joins and Aggregates Are Critical for Achieving Faster Data Retrieval in a Dimensional Data Warehouse

By Craig Abramson and Thomas Mascoli

The processes discussed in this white paper can all be accomplished using Syncsort's DMExpress, the high-performance data transformation product for UNIX, Linux and Windows environments. DMExpress extracts data at very high speed from any source database or flat file, applies any kind of record level transformation and/or field level transformation, and then loads the data into any target database or flat file. With DMExpress, you can design, schedule, and control all data transformations from a simple graphical interface on a Windows desktop. It utilizes patented performance algorithms, state-of-the-art parallel processing technology, dynamic optimization and architecture exploitation to minimize the elapsed time of applications and minimize the computer resources needed to run those applications.

Data warehouses are becoming overloaded with information. From Web sites alone, a company can generate several gigabytes of data each day. Now combine this with such data as store inventory, monthly sales, and customer addresses, and you've got all the ingredients for a bottleneck. Maintaining aggregate tables and separating descriptive data from factual information are techniques that can speed the data query process in a dimensional data warehouse. Aggregates reduce the amount of source data that must be processed by certain queries, while joins allow you to combine descriptive

information with voluminous factual data at the latest possible moment during a processing sequence, thus minimizing storage and throughput requirements. However, in order for these techniques to be effective, you first have to prepare the data in the data warehouse.

Preparing Fact and Dimensional Tables

According to data warehousing expert Ralph Kimball, the only viable way to deliver data efficiently to end users is to employ dimensional modeling in the data warehouse. Applied properly, this model can enhance user understandability, query performance and flexibility by presenting data in a standard framework. (Kimball, 1998)

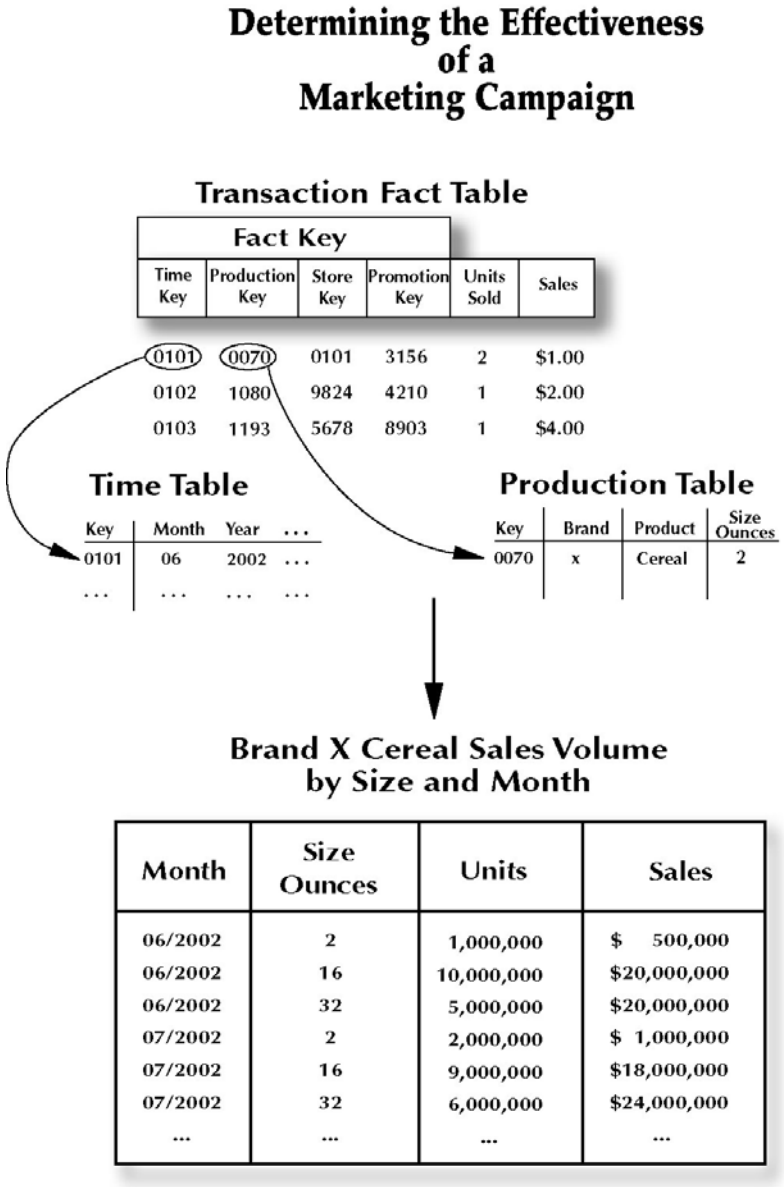
Fact and dimensional tables are the main components of the dimensional model. The fact table is the primary table in this model and it contains measurements of the business that are usually quantitative, such as “units sold.” Because fact tables accumulate large amounts of data resulting from individual transactions, they are designed to contain little or no descriptive information. Instead, a composite key that includes a subset of at least two or more component foreign keys identifies each transaction record. Each foreign key refers to a specific entry in a dimensional table. The dimensional table contains descriptive information about some aspect of the transaction, such as details about the product sold. This descriptive information is often the basis for constraining and grouping within data warehouse queries. For example, a product dimension table can include such information as product description, size, and category. (Kimball, 1998)

Combining Fact and Dimensional Tables Using a Join

Joins are used to look up descriptive dimensional information about the factual data only when this descriptive information is needed in response to a query. The primary key for each fact record consists of the keys for each of the dimensional records associated with it. In the fact record, each of the dimensional component keys are known as foreign keys because they are a primary key in another table. The dimensional component foreign key of the composite fact record key matches the fact record with the corresponding dimensional record, allowing for the retrieval of the descriptive information.

Suppose a grocery store chain wanted to determine the effectiveness of a marketing campaign that promoted the convenience of individual serving size packages of brand X cereals. They want to see monthly sales over the past year and compare sales of the product in the larger package sizes. Each factual record's composite primary key consists of all of the foreign keys that refer to associated dimension tables including the time dimension table primary key and the product dimension table primary key. The time and product table keys by themselves do not reveal in which month the sale occurred, or the brand and package size of the product sold. To obtain this information, it is necessary to join each factual record with its corresponding time and product dimension record using these keys. Once joined, the factual information is now complemented by the needed time and product details to complete the query, namely the month in which the sale occurred, the product brand name, and the product

package size. Now the factual and descriptive information may be summarized in a meaningful form.



Preprocessing Fact and Dimension Tables

Prior to loading the fact and dimension tables into the data warehouse, it is critical to preprocess the data to get it into the proper form for the dimensional data warehouse,

which will ultimately help you to reduce the elapsed time needed to retrieve the information. The data staging area is used to perform the necessary preprocessing, such as cleaning and merging records. The dimensions are usually processed before the facts in order to maintain referential integrity. Dimensional data processing may include generation of more compact surrogate keys by a look up (join) operation using the existing production key in the dimensional record. A surrogate key will be generated for new dimensional entries. Dimensional data from different sources is often merged, converted to a uniform format and summarized to eliminate duplicates before it is loaded. Cleansing operations may also be performed to remove erroneous data before loading. For example, sorting dimensional records on a textual attribute may reveal variations in spellings. In this way, erroneous records may be detected and deleted. (Kimball, April 1998)

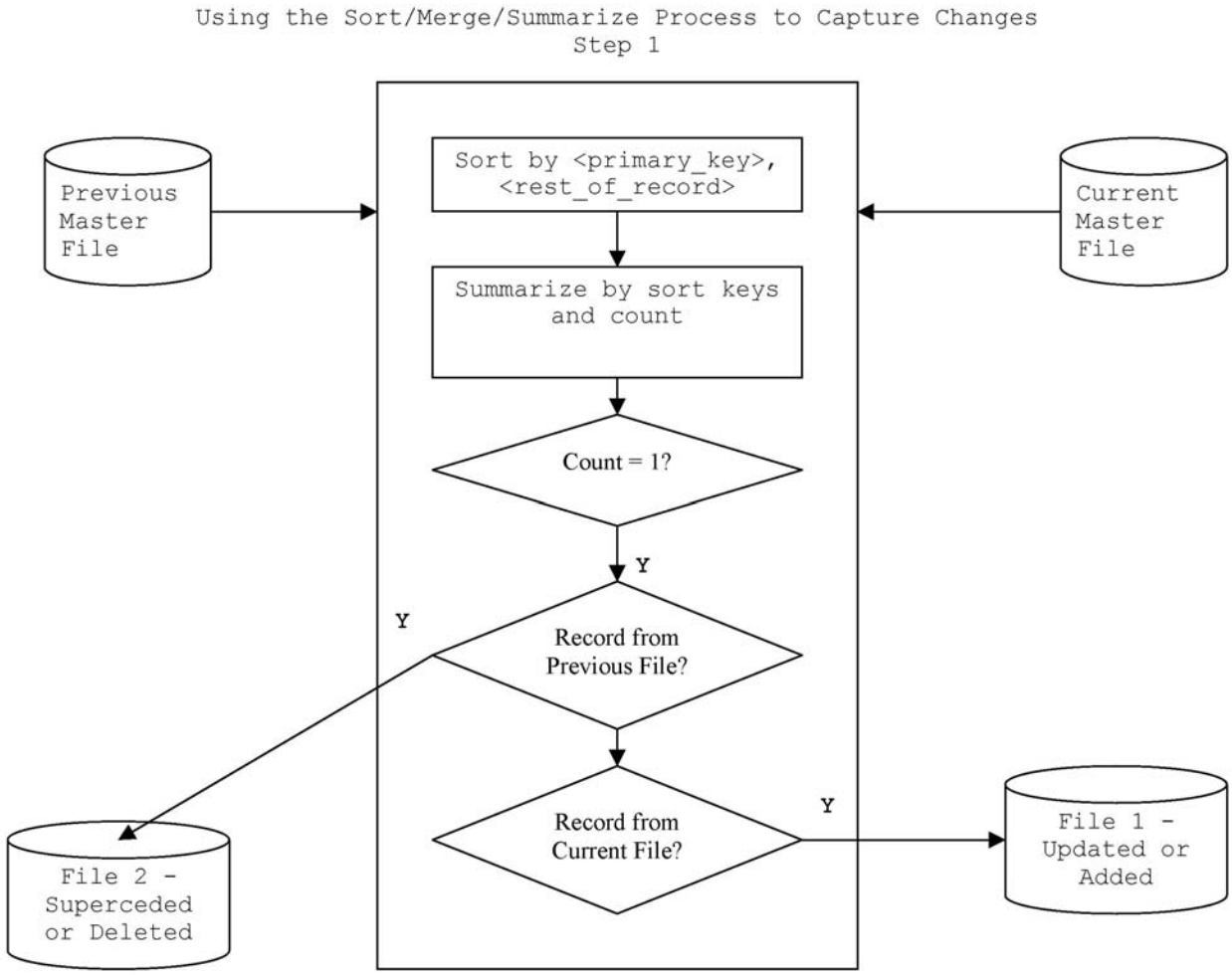
Once dimensional processing is complete, that attention shifts to fact processing. Factual records may also contain a production key that corresponds to a more compact data warehouse surrogate key. A lookup operation will be required to obtain the correct data warehouse surrogate key. Factual data should be summarized to the minimal level of granularity required by the data warehouse. For example, if data warehouse queries refer to time increments of no less than one day, and the raw factual data consisted of individual transactions that occurred during the day, it would be appropriate to aggregate the data by day before loading it. (Kimball, April 1998) When factual information entered into the warehouse consists of a period snapshot of data from online transaction processing (such as membership records), and if a relatively small

percentage of these records are added, changed or deleted between data warehouse load times, it may be more efficient to compare the current and previous data snapshot to identify and load only added and changed records.

Joins and Merges Are Effective for Capturing Changes to Large Operational Data Stores

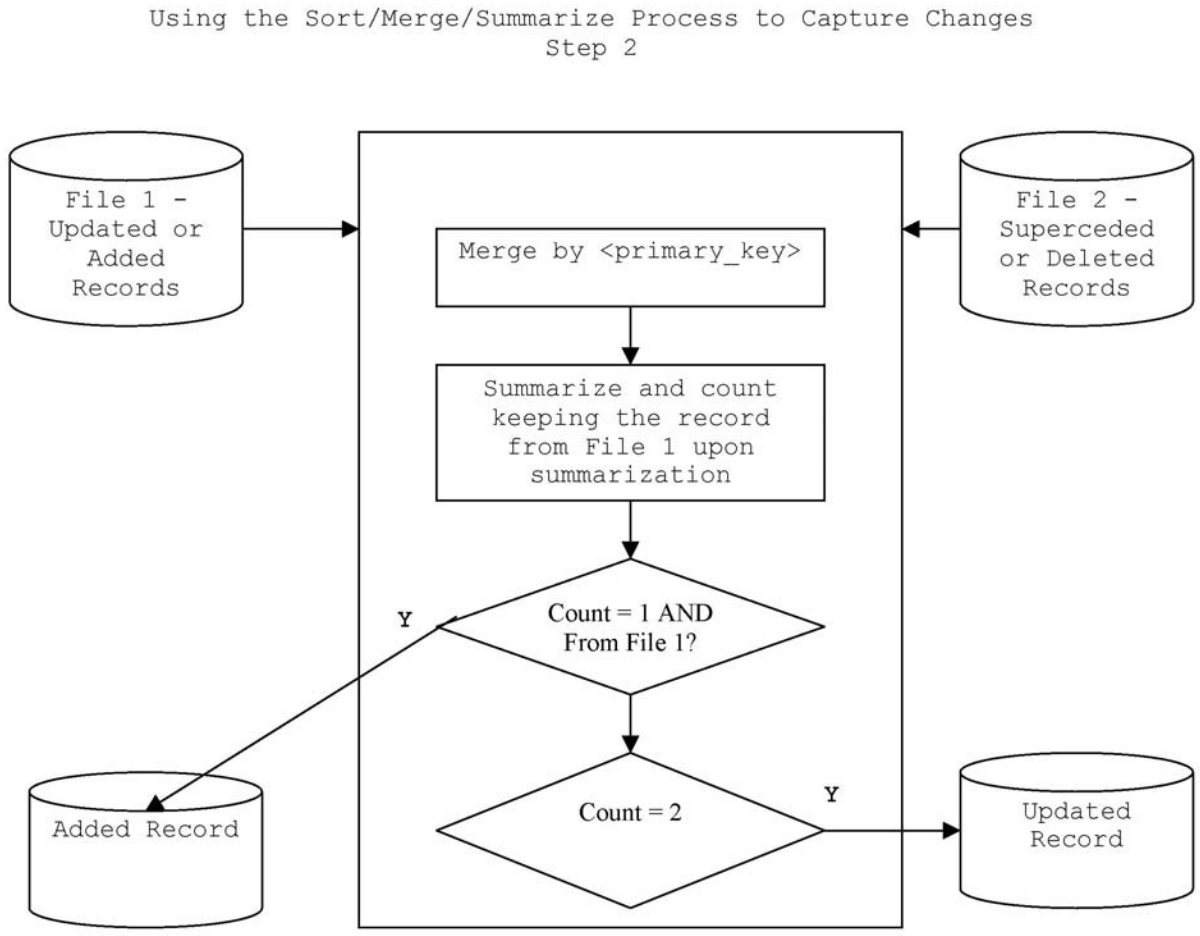
Data warehouses are updated at scheduled times with new data from the online transactional database. One way to perform this update is to replace the information in the data warehouse with all of the data in the online transactional database. However, if only a small fraction of the records in the database change during the scheduled updates, then it would be much more efficient to modify the data warehouse information with only the changed or added records.

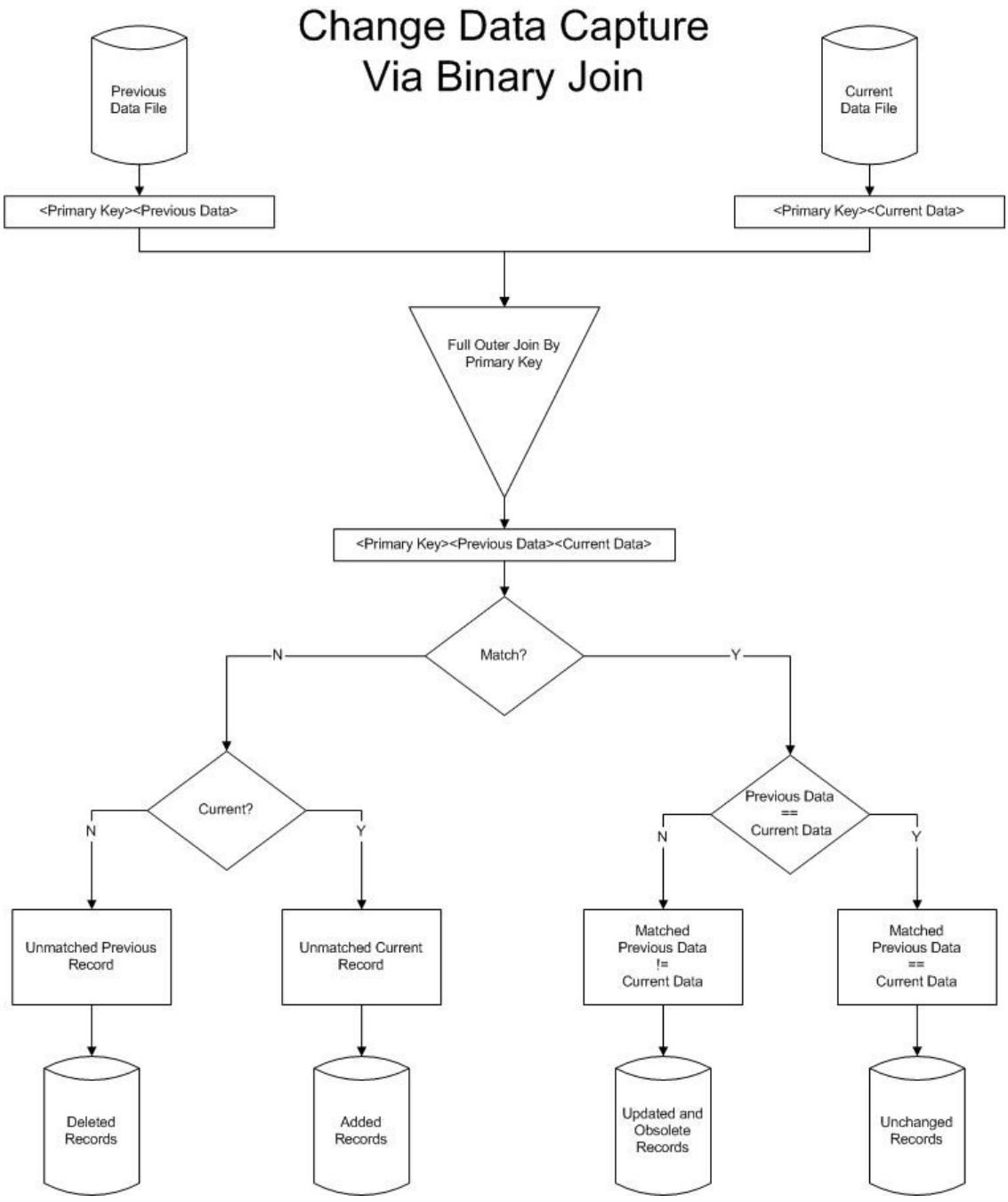
Using a join is an effective technique for comparing the previously loaded database with the current database to capture deleted, updated or changed records. A join will match the primary key of the previously loaded record with its corresponding new record. The data portions of the record can be compared to determine if they have changed. For instance, if a previously loaded record has no matching record in the new file, then it is deleted. If a new record has no match in the previously loaded data, then it is added as a new record. If a previous record matches with a new record but the corresponding data has changed, then the record is updated in the data warehouse. This technique will help reduce significant amounts of elapsed time when performing change data capture or delta processing.



A sort, merge and summarize process is also effective in capturing changes. Previous snapshot data is summarized with current snapshot data using the entire record as the summary key. A counter is initialized to ‘1’ and appended to each record before the summarization, and totaled during the summarization process. Summarized records with a count of ‘2’ are unchanged between the previous and current snapshots and are usually eliminated from further processing. Summarized records originating from the current file with a count of ‘1’ are updated or added records. Summarized records with a

count of ‘1’ originating from the previous snapshot are either deleted records or obsolete records (records from the previous snapshot with a corresponding updated record from the current snapshot). A final processing step can partition the added, changed and deleted record types to individual targets.





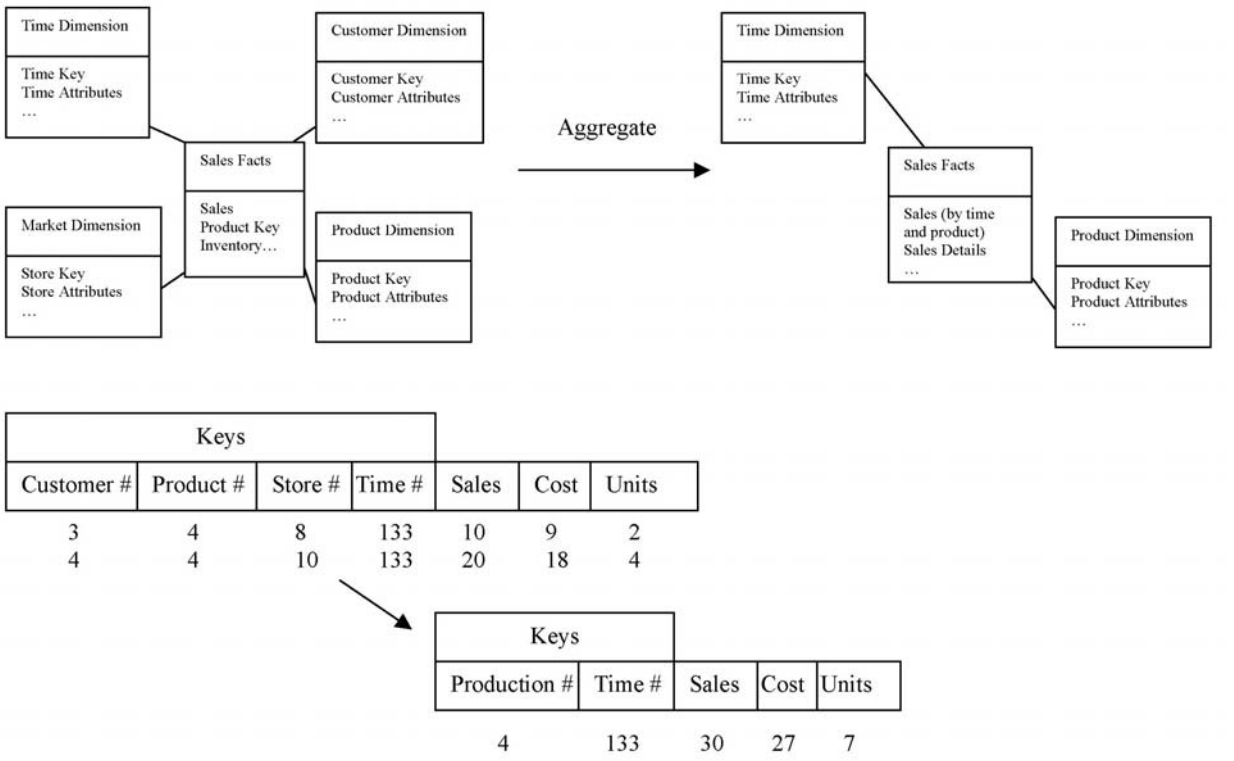
Building Aggregates in the Dimensional Model

Another way to improve performance when processing large amounts of data in a data warehouse is to build aggregates. A query answered from base-level data can take

hours and involve millions of data records and millions of calculations. With pre-calculated aggregates, the same query can be answered in seconds with just a few records and calculations.

Building aggregates from dimensional models is a simple approach because each large fact table can provide numerous aggregates with a predictable structure and a predictable relationship to the base fact table. There are three basic ways to build aggregates from the fact table in this type of model. The first approach is to exclude one or more dimensions when summarizing a fact table. This is the easiest type of aggregate to create because the dimensional data doesn't need to be accessed. The aggregate can also provide significant performance advantages over the base fact table. (Corr, 2002)

Figure 5: Creating Aggregates by Excluding One or More Dimensions When Summarizing a Fact Table



The second approach is to have one or more dimensions replaced by rolled up versions of themselves. For instance, you can use this approach to create a monthly-product-sales-by-store summary of the original fact table. You're aggregating all of the daily sales records into monthly records and reducing the size of the fact table, but you're still able to do dimensional analysis by time at the month, quarter, and year levels. While this helps achieve a balance between performance and usability, it will require maintenance of the physical dimensions and keys. The third approach eliminates this maintenance, replacing dimensional keys with high-level dimensional attributes. This type of aggregate should only be used for high-level summaries where the dimensional attributes are limited and short .An example of this would be a quarterly-product line-unit

sales summary. This approach also accelerates query performance by eliminating the need to create joins for query processing. (Corr, 2002)

Figure 6: Creating Aggregates by Replacing Dimensions with Rolled Up Versions of Themselves

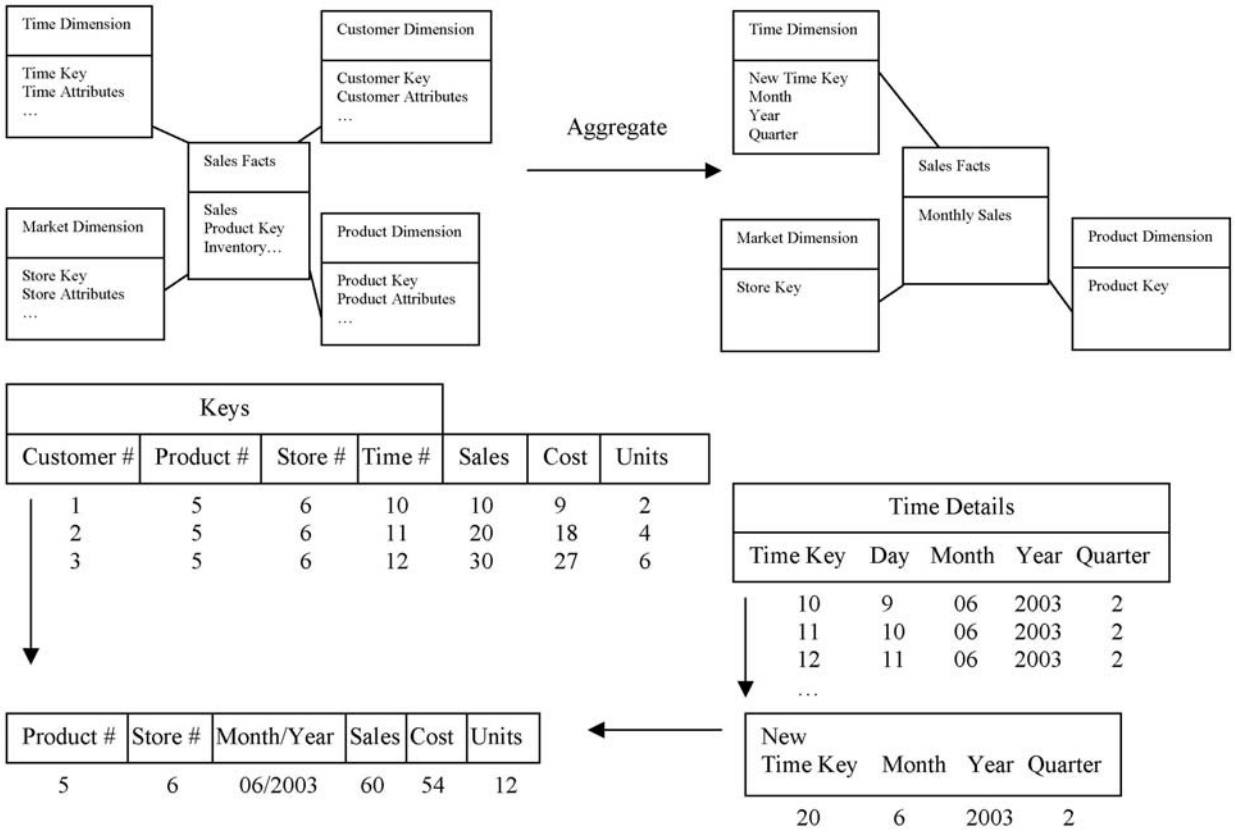
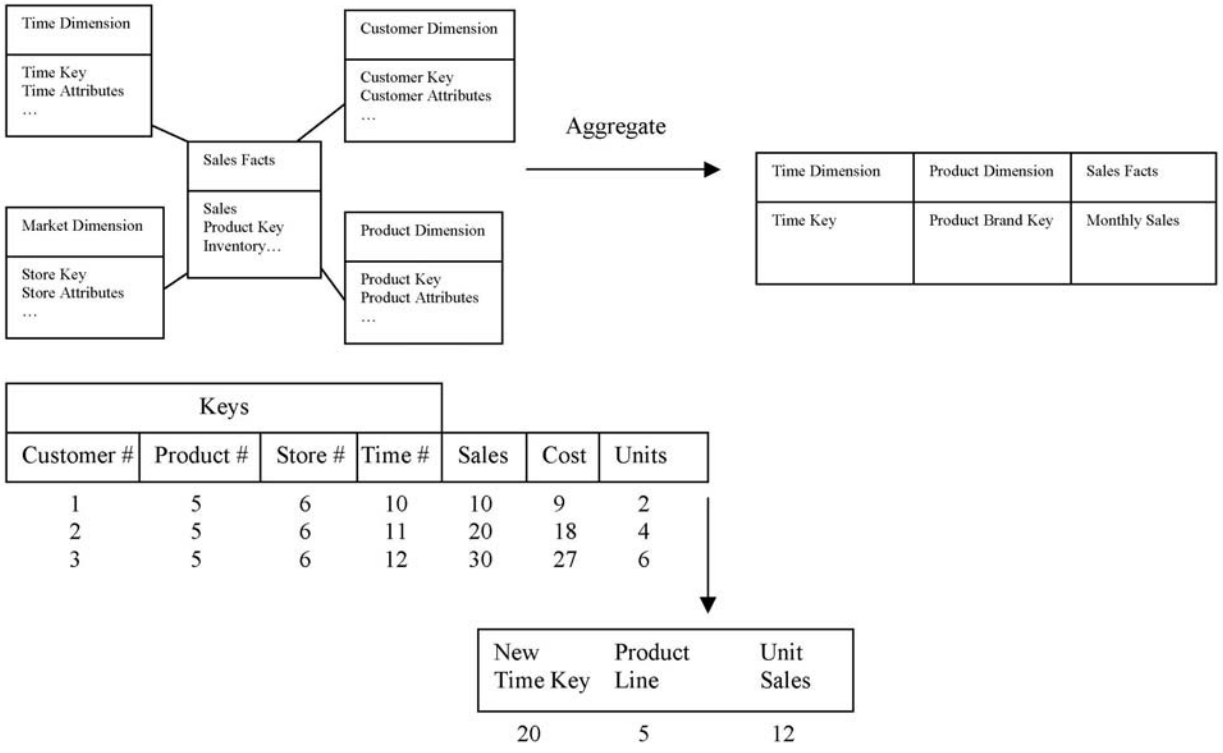


Figure 7: Creating Aggregates by Replacing Dimensional Keys with High-Level Dimensional Attributes.



ACNielsen Builds Aggregates for the Analysis of Sales Data

ACNielsen, a VNU business, is in the process of building its New Factory for Europe, the largest retail sales data factory. New Factory will be used to analyze sales data from different retailer channels and countries in Europe. The analysis will provide insight into how much impact a specific promotion has had on the sale of a product, how well brands perform in comparison with other brands, and how successful the launch of a new product was, as well as a variety of other measurements.

It is through innovations like New Factory that ACNielsen has become the world's leading marketing information provider. Offering services in more than 100 countries, the unit provides measurement and analysis of marketplace dynamics and consumer

attitudes and behavior. Clients rely on ACNielsen's market research, proprietary products, analytical tools and professional service to understand competitive performance, to uncover new opportunities and to raise the profitability of their marketing and sales campaigns.

Due to the necessary computation of non-additive distribution facts in New Factory, simple roll-up or cube functions would not be sufficient. But the need to complete a large number of complex aggregations represented a potential performance bottleneck for the company. ACNielsen began a thorough search to find a powerful, high-performance ETL solution that could complete the aggregations. DMExpress proved to be that solution.

DMExpress was installed by ACNielsen in a proof of concept to aggregate an initial 2.7 billion facts over 4 different dimensions, varying in hierarchy depth from 2 to 9 levels. According to Technical Director Michael Benillouche, “When we started developing our data factory application, called New Factory, we knew that performance was going to be an issue. We searched for a solution that could handle the high volume of data we were processing in the shortest amount of time. After considering ETL software from major vendors, we selected DMExpress. DMExpress easily integrated into New Factory’s distributed computing framework and provided us with the outstanding results we needed.”

ACNielsen tested DMExpress in New Factory, running it on a large-scale UNIX server with 16 CPUs, 32 gigabytes of memory, and terabytes of disk arrays. The server is capable of delivering data at a sustained rate of 600 MB/sec. Once in production, data will be constantly processed in this carefully designed factory. It is estimated that 12 billion sales facts will be aggregated along four different dimensions each week in order to aggregate the thousands of datascopes accessed through the New Factory Web applications.

ACNielsen discovered that as data volumes grow, so do the performance advantages of DMExpress. With the ability to process in parallel, DMExpress speeds through data-intensive applications. Application development is also much faster utilizing the advanced, easy-to-use graphical user interface (GUI). Instead of focusing on processing the data, you can use the time to create what you need.

Discussing ACNielsen's use of DMExpress for aggregation, Andrew Coleman, Syncsort's Director of Software Engineering added, "More and more, we see the aggregation step being the critical performance issue in our customer's data warehouse applications. The hardware capacity is typically available, provided that the software can fully exploit it. Our combination of proprietary aggregation algorithms and relentless pursuit of parallelization across multiple processors and multiple servers allows DMExpress to achieve the maximum from the hardware."

Getting the Speed You Need

The process of preparing data for a dimensional data warehouse can be significantly reduced by the effective use of aggregation and join processing performed by a tool optimized for fast sequential processing. Properly prepared factual and dimensional data can improve query performance in a data warehouse by minimizing the amount of data that must be processed in response to a query. In many cases, processing performed in the data warehouse staging area is sequential. DMExpress is designed to handle sequential processing and can help minimize the elapsed time needed to process large amounts of data, which in turn can dramatically improve the performance of the database.

DMExpress simplifies the creation, administration, and execution of aggregation jobs. It summarizes data much faster than other aggregation methods such as C programs, SQL statements, or third party multi-purpose data warehouse packages. With DMExpress, you'll gain the flexibility needed to select the best aggregates to optimize query performance at a site. It offers advanced aggregation functions including calculating maximums, minimums, averages and more. You'll also be able to combine fact and dimensional tables as well as complete delta processing using DMExpress. This can be accomplished by following the steps discussed in the white paper. For more information about DMExpress, visit www.syncsort.com.

References

Corr, Lawrence. “Lost, Shrunken, And Collapsed. Why and how to create the three types of aggregates in a dimensional data warehouse,” www.intelligententerprise.com (2002), 1-2

Kimball, Ralph, Laura Reeves, Margy Ross, and Warren Thornthwaite, “The Data Warehouse Lifecycle Toolkit: Expert Methods for Designing, Developing, and Deploying Data Warehouses,” John Wiley & Sons, (1998), 17, 140

Kimball, Ralph, “Is Data Staging Relational? Or Does It Have More to Do with Sequential Processing?,” DBMS, (April, 1998)

Biographies

Craig Abramson is a technical analyst at Syncsort Incorporated, focusing on the latest data warehouse performance solutions. He has over 7 years experience in the field working on projects dealing with data warehousing, database management and Web log processing.

Craig Abramson

Technical Analyst

Syncsort Incorporated

201-930-9700 ext. 4308

Email: cabramson@syncsort.com

Thomas Mascoli is a Senior Software Development Engineer in Technical Support at Syncsort Incorporated. His responsibilities include implementing product fixes and enhancements, and assisting customers with high performance data processing solutions utilizing Syncsort's suite of products for UNIX and Windows.

Thomas Mascoli

Senior Software Development Engineer

Syncsort Incorporated

201-930-9700 ext. 4296

Email: tmascoli@syncsort.com