Paper Number 1070

Data Warehousing on a Shoestring Budget

Mark L. Strickland
American Water Heater Company
500 Princeton Road
Johnson City, TN 37601
423/283-8022

Data Warehousing - My Definition

Data warehousing has a variety of definitions and interpretations. In the "old days" before the term was coined the concept generally involved some type of historical data archive used for analysis such as a Sales Analysis System. Over the years Data Warehousing has taken on different meanings to different people. Today my definition of a data warehouse is: A database of related information that has been formatted into a normalized summary and/or detail form for ease of searching, sorting, and reporting. This definition corresponds to an ideal physical warehouse for storing paper records where similar related data is grouped, boxed, marked, racked, and indexed in such a way that retrieval is easy.

The Problem

At American Water Heater we were in the middle of a conversion to a packaged system that did not have a very flexible sales analysis system and very little time in the Applications Development Group to develop the custom reports needed. We had a mission critical task to provide a new Executive VP with sales data to help get a grip on a sales erosion problem and do it in a hurry. From previous experience the requirements and tools available were not adequate to accomplish the task in a timely manner. At that point we began looking for a solution to the problem.

The Search for a Solution

The first step to finding a solution was to look at the data available on the HP 9000 system. A summary of daily invoices existed in a Sales Statistics System but the format did not lend itself to the reporting requirements. It was also in a proprietary database format and the only tools available for reporting was a proprietary language. One third party report generator was available but it was expensive, about $35,000, and did not have the necessary flexibility needed to meet the reporting requirements. The budget alone made this an unattractive solution. The option of a new database on the HP 9000 exceeded $250,000, only a dream in the foreseeable future.

At that point we began looking at Personal Computer based solutions. We had a large LAN and frame relay WAN connecting nearly every location in the company as well as a Novel Server on the LAN to potentially use for a database server. Several OLAP (On Line Analytical Processing) tools were reviewed but the files sizes (half a million records) made these unattractive. A look at database products found that the best appeared to be Microsoft NT based not Novel based. After some quick research we decided on MS-SQL Server for NT. We then undertook a task to create an NT server on the network.

Again to keep costs down we took an existing PC and replaced the motherboard with a 100 MHz Pentium board and 64 megs of RAM. An ordinary 2 gigabyte IDE disk drive was transplanted into the system. The total cost for our NT server hardware was under $1,000. We purchased a copy of Microsoft NT server with a 5 user license and loaded it. After just a little work and a couple of days we had it integrated into our existing network although we had never used NT before this experience. The next step was to purchase a copy of Microsoft SQL Server with a 5 user license. This also installed with minimal effort. For only about $3,000 we had a platform for deploying our new data warehouse.

Warehouse Development and Reporting Tools

Concurrent with the building and setup of the server we began to look for a solution to the reporting tools. Since it was based on MS-SQL server the variety available was almost endless. We looked at Microsoft Visual BASIC and Borland Delphi. These were nice but the review of a tool I had used on another platform (HP3000) had distinct advantages. After looking at SpeedWare, although more expensive than Visual BASIC, we decided that it was the best tool for our environment. This was the only tool that had database prototyping as an integrated function. Report generation was also a strength of SpeedWare. The formatter was not exciting but the ability to control and add program code to all phases of the report program makes generating any report, no matter how complex, very easy. Most report "generators" do not allow customization of the generated code while maintaining the ability to return to the formatter. Because of existing experience with this tool and the ability to automatically do all of the SQL database administration activities transparently this was our best choice. We also began looking for an ad-hoc reporting tool but more about that later.

SpeedWare is not necessarily the best choice for all situations. You should consider several factors in selecting the data reporting tools for your needs.

Tool Selection Criteria:

* Current experience level with potential programming tools
* Current experience level with potential database administration tools
* Deployment requirements for the finished product
* Budget limitations

A critical factor in data warehouse development is the ability to model and repeatedly remodel the data tables as the warehouse data structures are created. We would all like to think that we have excellent Data Base Architect skills on staff but frequently this type of system evolves over time as the users define their needs. The fluency in the data administration tools can be the make or break factor in a data warehouse project. If remodeling data structures is a problem or cumbersome then the overall success can be effected. Data modeling and prototyping is a strength of SpeedWare.

Data Sources

Once the platform and tools have been selected then a data source has to be found. Several possibilities usually exist. In our case summary files of historical sales information did exist on our HP9000 that met the basic reporting requirements. If a summary file does not exist then some type of summarizing process must be developed. This processing may be done directly into the warehouse tables by capturing the detail data needed such as invoices and creating an archiving program to summarize it. After a data source is selected then a bridge to the warehouse must be built. In our case we exported the summary sales data table and all supporting tables such as customer master and item master in ASCII delimited format on the HP9000 and moved them to the NT server with an FTP file transfer at 10 megabit Ethernet speeds. Our largest table is about 150 megabytes in ASCII format but since it is only moved once per month it is not a problem. We actually empty out the warehouse each month and completely refresh the detail tables and then regenerate the summary tables. This technique insures that the current system tables are always in sync with the warehouse tables and avoids complex partial update logic. This complete refresh technique insures the data in the external warehouse stays in synch with the source.

In some cases the exported tables are loaded into the warehouse in extracted format but others require either reformatting for a variety of reasons. Reformatting includes building concatenated key fields from separate fields and data normalization. We accomplished this by writing little custom applets in Power BASIC since it was fast and had the parsing functions to deal with the delimited data easily.

Other off the shelf programs exist such as DataJunction that would do the same job by creating parameter files. A final option would be to load the table intact into the database and reformat it by copying from one table to another. This may not be the best option due to the speed of processing tables with multiple keys. We found that the Bulk Copy Program supplied with Microsoft SQL Server operated with amazing speed, actually faster than the database tools on our HP9000 K410 brute with gigabyte of RAM. On a Pentium 100 PC with a single ordinary IDE disk we found we could load a table with over half a million records at 1,600 records per second if indexes were dropped! The same table with three keys could be reindexed in about 10 minutes with several simple SQL statements. This yielded a total load time for over 500,000 records of about 15 minutes. Pretty amazing for an "ordinary" Personal Computer used as a server. Using the bulk copy program supplied with MS-SQL Server with the proper options set provides excellent performance loading ASCII delimited files. If you use this technique turn off transaction logging for bulk loads and drop all the index files before the load using I/SQL (Interactive SQL). After the load is complete recreate the index files with I/SQL which only requires one statement per index. The drop and create I/SQL statements can be generated by the server itself in the Enterprise Manager program.

Data Normalization

Data normalization is an important step in creating data that can be processed easily for a variety of report requirements. We elected to do normalization at the ASCII record copy step. The tables extracted from ManMan/X contained arrays for 12 months of data for a specified fiscal year. This was good for saving disk space but reporting calendar periods that spanned multiple records would have been a programming nightmare. We normalized the data into a third normal format reducing each element of the array to a single record with all the keys necessary to retrieve the record for reporting. The output of zero value element array items was suppressed to save space. Don't overlook this very important normalization step. Extra time in analyzing and creating the proper formatted data will pay off big when it is time to program reports.

Pre-summarized Data

When analyzing the reporting requirements we found that several summary formats were needed. Although the server was reasonably fast, selecting, sorting, and summarizing the same records repeatedly was not a particularly fast task on the network client PC's. A PC is just not suited to sorting work files with hundreds of thousands of records. We developed several summary format tables that were built with utility programs from the main detail table. The single detail table that was the source of all data for the "warehouse" contained sales by customer and part number with over 500,000 records. A "pre-built" summary by sales agent and product category was only about 25,000 records. The table of customer totals by sales agent was only 50,000 records. These pre-summarized formats, built from the same source data, allowed quick reporting for these summary reports. The detail table by customer and product was used for customer specific reports. The utility to create each of the needed summary formats takes about four to six hours to populate one of the summary tables but since it is only done once per month it is worth the CPU time. These utilities are run directly on the server PC to eliminate the network activity - a handy feature of NT that cannot be done on a Novel server.

Reports - Things to Think About

You should pay special attention to details when creating reports. Work very hard at extracting the reporting requirements from your "customers". Frequently you need to create a left brain (analytical) view of their right brain (emotional) requirements. Little things like helping sales management understand how the reporting field relationships work will help immensely since you cannot create data that does not exist if you are simply moving existing data to a new platform or format for reporting purposes. If certain key fields do not exist in the source you cannot report on them. Other simple things such as placing page breaks on reports so they can be separated easily and distributed to the responsible party such as sales management reports paged by sales rep. Spend some time creating clean looking layouts for optimum report readability. This is very important if you ultimately want to distribute reports in an on-line paperless format.

Where possible always design reports that support the theory of "directional analysis". This type of reporting starts at a high level summary format and works toward a more detailed format on various reports. As summary level information is analyzed the companion detailed reports are generated for a smaller sub-set of data that create detail reports in a manageable size. Look at the techniques of ranking style reports that put the best at the front of a list and the worst at the end such as a customer ranking analysis. This type of ranking is much easier to generate from the pre-summarized data as outlined previously. The SpeedWare tools we choose would easily generate a work file record at report sub-total lines that could be used as the input for another report to create some very interesting rank analysis reports.

Use flexible report request parameters. If your data is stored in fiscal periods different from calendar periods create a method to request reports on both fiscal and calendar period ranges. This allows your "customers" to work with information in a format that is familiar and flexible. If the true third normal format is used with one reporting period per data record then combining periods over a specified range is very easy to program.

As a rule try and create reporting periods that allow comparison ranges such as current month this year versus the same month last year, current year to date versus last year, or any other useful comparison. Comparison of reporting periods allows easy "directional analysis" to find where things have changed. If you can find where problems have occurred using summary information then reporting the detail to figure out why is much easier. One frequently ignored measurement period is a rolling 12 month total. This can show a full year's data ending in the current month on a this year versus last year to date report. This will allow two different views of the same data and the rolling 12 month view is like a year end view at the end of each month. Include a comparison in both units and percent where possible as well as average calculations such as average selling price or other similar measurement. Frequently these can be mentally calculated from the reported data but the whole purpose is decision making and displaying the proper measurement metrics that the target "customer" relate to is very important.

When designing reports work hard to create this concept of "directional analysis" or "drill down" as it called today. Ask leading questions that help brainstorm how the reports will be used.

Ideas for report design criteria:

* How can reporting period comparisons be used?
* Is rolling 12 month data useful?
* Is any type of average calculation useful?
* Can ranking be used effectively?
* Design optimum indexed keys to speed report run time

Credibility of the Warehouse

Make sure that your "customers" have confidence in the accuracy of the data stored in the Warehouse. Work hard to establish this credibility by creating reports that match other existing reports in your system. Without this important step your new warehouse, no matter how useful, will be suspect and won't be used.

End User Access to the Data

I feel that it is very important to ultimately put the "customers" in control of their own environment. The reporting tools, although many will be programmer built, should be accessible to the user directly. To do this, "friendly" report parameter input to run reports is necessary. Restrictions that do not allow illogical input need to be created and built in connivance features such as customer number look-up from names allows easy and accurate report generation. Be sure to include the selection criteria used on the report to eliminate confusion.

Ad-hoc Reporting Tools

Ad-hoc reporting tool selection can be another element to put control in the hands of your "customers". This is a potentially dangerous area that can lead to frustration and erroneous output. The "programmerless" environment has not yet been created although some believe differently. It is possible to create SQL statements that are syntactically correct but yield incorrect results. The untrained user using an "unfriendly" tool can get bad answers and not even know it. Be sure to deploy these tools carefully and provide the necessary training to be successful in their use. We are exploring several tools but have yet to deploy any for widespread use.

Some final thoughts

* Know your "customer's" requirements
* Pick a platform you are or can become proficient with quickly
* Pick reporting tools based upon fluency and flexibility
* Pick reliable data sources to populate the warehouse
* Design normalized data structures
* Design tables with plenty of index keys for rapid access of subset data
* Design reports with directional analysis in mind
* Build confidence with reports that match other standard reports in your system
* Give your "customers" direct access to their own reports
* Deploy ad-hoc reporting tools carefully

Conclusion

The price to create a data warehouse can vary dramatically. Some off the off the shelf products for larger platforms can be in the hundreds of thousands of dollars any you still must create useful output. High end programs come with a wide variety of reporting tools but but many times the same or similar tools are available on PC based platforms very inexpensively. The two main keys for success are proper data structures and the right reporting tools for the job. Be sure to follow the tips above and with a little creativity you can build a very useful data warehouse on a tight budget.