

**Paper Number: 1040**  
**Title: Building the Data Warehouse - Enterprise Wide Data Access**

**Ralph Bagen**  
**PowerCrew Incorporated**  
**382 Springfield Ave Suite 310**  
**Summit, NJ 07901**  
**(908) 522-3040**  
**rbagen@powercrew.com**

**Perspective:**

It is the purpose of this paper to present, and hopefully enlighten, anyone who might be about to embark on the creation of a data warehouse / data mart project. The size of such an undertaking can be huge, and it must be given some very heavily thought out design considerations. Many data warehousing projects are doomed to failure from inception, right on the drawing board. It is my hope that by sharing some of my experiences, I can at least raise an awareness to potential downfalls, hurdles and myths.

**What is a Data Warehouse anyway... and why should I want one?**

The person generally accredited as being the “father of data warehousing”, Bill Inmon uses the following definition in his book “Building The Data Warehouse”:

“A Data Warehouse is a collection of *subject-oriented, integrated, non-volatile, time variant* information in support of management’s decision making process.”

Let’s look at this more closely.

- Data warehouses by design are subject-oriented, that is to say they are collections of data about a subject. This differs in principle from traditional, “classic” application stores of information which are function-oriented.
- Integrated data refers to the fact that information is being acquired from multiple sources and is being integrated using consistent naming conventions and coding of attributes.
- Non-volatile means that after the initial population, the data in the warehouse is never changed, i.e. updated.
- Time variant infers that data is as of some moment in time and usually has a time component enclosed, such as a date / timestamp.

But what does this mean to the project’s success? It must be observed that the data must be stored in a fundamentally different structure than it is in a typical business application package. What this obviously means is a complete design of underlying data structures to support the warehousing environment. I strongly recommend building a team of key individuals who understand the various sources of existing data and will be able to explain the strengths and limitations of its current confines. The data warehouse is always going to be populated from an operational environment, but as such must become a separate store of data, transformed from the application data found in the operational environment. If this is designed and performed correctly, a data warehousing solution will provide end-users **easy** and **timely** access to **accurate** and **consistent** information, thereby enhancing the ability for superior business decision making.

It may be obvious, but in order to manage a business, decision makers need access to information. Historically, even although legacy applications did a tremendous job capturing and accumulating transaction data, the mechanisms were not in place to put this information at the fingertips of a knowledge worker. Oftentimes, this condition has led to long MIS report request turnaround times, or worse, flat-out frustration in settling for canned information that would have to suffice. Newer technologies permit closer scrutiny of data on multitudes of selection criteria. Areas of typical benefit for example could be customers buying habits, marketing and sales planning, product profitability and inventory control.

### **Advantages of a Data Warehouse**

Since the approach has been taken to receive information from multiple data sources, the end result should be a new set of consistent and integrated data. There should be a corresponding drop on the load on the production systems, directly proportional to the acceptance of the Data Warehouse. Clearly, as we are crafting these new solutions we are unencumbered by "legacy technology baggage". The data warehouse will just accumulate information as it happens, meaning the operational systems can be kept lean and trim. Both denormalized detail and summary data should be stored in the warehouse structure, meaning continuous history will always be on-line. This is particularly of importance if trend analysis is important in your company. Keep in mind that procedures will be required to synchronize the data mart with the enterprise data warehouse. The following are examples of the data life cycles in a warehouse environment:

- Highly summarized - monthly sales by product '87 - '95
- Lightly summarized - weekly sales by subproduct '93 - '95
- Operational data - sales detail '94 - '95
- Archived data - sales detail '90 - '93
- Purged data - sales detail '87 - 89

### **Data Warehousing & Client/Server**

Initial appearances would seem that there is not much in common with the two. Upon further inspection, however, it should be noted that they are actually very closely aligned. Both technologies were brought about by improved processing power, GUI development and access tools. Couple this with lower hardware costs and scaleable servers, and it can be seen that a data warehousing environment is almost a 3-tiered client/server model. The warehouse may reside on an enterprise server, decentralized data mart servers or both. A three tier solution would flow from Data Warehouse -> Data Mart -> Client Workstation. Typically, one witnesses the movement of data warehouse data to local data marts as data warehouse acceptance unfolds.

One cannot hear about data warehousing without discussions about OLAP (on-line analytical processing). This is the equivalent to OLTP (on-line transaction processing) of the application world. The difference is as complete as Bill Inmon describes. OLTP data is highly denormalized, since it is inquired upon and updated often. OLAP however requires normalization, that is, optimized for slicing and dicing. Since the data itself does not change, OLAP data structures can be heavily keyed and often included aggregate or summary data within the detail records. It would not be uncommon for summary data or metadata (data about the data - the yellow pages for your warehouse) to occupy more disc space than the detailed information itself. Whereas OLAP is not new to Relational Database Technology (RDBMS), the very first could not handle OLTP. Now OLAP & OLTP can co-exist up to 100GB of information without special consideration. It should be understood that there is generally a much higher I/O wait doing OLAP than a CPU wait. Most technologies use read-ahead solutions to minimize the frequency and duration of these waits.

### **What is Data Mining?**

This term has become as readily accepted as the warehousing technologies that have made it possible. Mario Schkolnick of IBM Research coined the phrase within the following definition:

“Data mining can be thought of as the efficient discovery of previously unknown facts (nuggets) found in very large databases”

The concept is one of discovery vs. verification. The availability of the data warehouse and OLAP allows associations to be sought such that the presence of one set of items implies other items. This is a fundamentally different approach to having something pointed out to you (like your company has lost a good customer) and reports are reactively executed to verify this after the fact.

### **What are the Benefits of Data Warehousing?**

Other than the aforementioned superior access and quality (accuracy & consistency) of information, a successful project will also actually lead to reduced costs / increased productivity in MIS. The operational throughput will increase on the production systems while data warehouse subscribers will be less frequently lined up with requests for MIS reports. It is also an ideal way to leverage a client / server environment since it is actually a fresh start from a tools and technology perspective. The infrastructure that you erect for this environment should be designed to serve your company's needs going forward, whilst building a competitive advantage at the same time by providing functionality not possible in aging systems.

### **The Challenges of Data Warehousing**

For a successful project to be undertaken, it needs sponsorship and buy-in from the highest level of management that you can access. The scope of these projects can be so vast that it is not uncommon to find many shelved in a very delayed and over-budget condition. One of the first steps is to set realistic goals and expectations, and to conduct a cost justification analysis around them. In addition to configuring software and hardware requirements, a most often underestimated component are staffing and training considerations. It is very possible (in fact, desirable) to take key individuals within an organization and move them from their current roles onto a data warehouse steering committee. Also, anticipate disc requirements to grow over time and budget accordingly. It will make sense to multiply the number of data marts in the environment, possibly one per functional department (i.e. Financial server, Marketing server). This is a golden opportunity to evaluate and correct network traffic and future needs.

### **Truths about Data Warehousing**

A data warehouse is a dynamic, growing environment. Accordingly, they *become high-maintenance* systems. It is recommended that someone with RDBMS experience be charged with the responsibility to administer the condition of the data structures. Disc fragmentation can be a problem in particular, if not monitored carefully. It will also be essential to watch for changing network traffic patterns. It is probably the time to start thinking about Database / Network Administrator position(s) for your company if they do not presently exist.

Do not be surprised to find problems in the feeder systems. You may find inconsistencies that have existed untouched in your production environment that have been unchecked for years. It is your job and responsibility to find and address these problems. The worst thing that could be done is to replicate process flaws into your next generation solution. Data scrubbing (or data cleansing) is the term given to the process whereby data is conformed or mapped into a single form, thereby ensuring consistency. This step is particularly necessary when receiving data from multiple applications residing on multiple systems. Data structures change over time (for example, in software releases), and it will be important to address these as you “back-fill” your warehouse with historical information. Now is the ideal opportunity to avoid replication of garbage data collected within the operational system.

Remember too that as data volumes increase, the need for aggregation will increase. Summary data and detail data will co-exist, and potentially the summary data can consume more disc space than the data itself. There is considerable overhead in the creation of aggregate information that should be anticipated and planned for.

A system **must add value** to the business system (i.e. it must be relevant, accurate and readily available). Ensure that there is a *business case* and **not** just a *technology case* before you embark on this long journey.

### **Data Warehousing: Do's and Don'ts**

#### **Do's:**

- Find a business sponsor at the highest level
- Involve the key business users (no-one knows the requirements and data better than the everyday user)
- Require proof-of-concept using your own company's data (and volume test)
- Select a vendor that you can trust that will share the risk
- Grow the scope of the warehouse - it should be an iterative process. Start with a pilot application that can be an area of particular need. This will increase the potential for success and acceptance.

#### **Don'ts:**

- Over-sell the utility or promise of this solution
- Expect to deliver the entire solution overnight
- Get distracted by industry hype or vendor promises
- Ignore the business requirements
- Sell technology first
- Refuse help from the users who know the business best

### **Building a Pilot and Selecting the Tools: a Case Study**

The remainder of this paper will focus on the tool selection process by examining a small scale data mart project that you can get of the ground fairly quickly and successfully in your company.

Let's pick a functional area that everyone can relate to. Every business must have customers and sales orders. In a traditional OLTP environment there is a customer master data set and sales order headers

& sales order lines. I will also assume that this data resides on a single HP3000 running the business production process (these assumption are merely to establish a starting point). Problems arise when intensive analysis is done on these sets "in the heat of battle" during the day - in fact some larger order entry systems are 24 by 7 anyway. We must get this information onto a second platform in order to run unencumbered by the production system.

The ideal vehicle for such a starter project would be an NT Server running Microsoft Back Office products. The initial cost of entry is low, assuming you do not already have a NT Server presently that you could rent space on (an underutilized mail server perhaps?). In our example, let us put a 5 client license pack for SQL Server 6.5 and NT Server 4.0 on a 200MHz Pentium PC with 64MB of memory. It is my observation that memory is more important than clock cycles for operational efficiencies. A 2GB disc drive will suffice, although the more disc spindles you have, the better performance will be (namely so that log files, indices and data can be separated). The other important consideration is network connectivity. If you can segregate network traffic for this application (through an Ethernet switch) and connect at 100 Mbps, you will be off to a great start.

First, a blueprint must be made up to identify what data should come off the host and how often. Let's say we need sales order headers and lines and an updated customer file daily. Bring only the data that you will need for analytical purposes. Do not, for example, bring comment lines.

A batch file can then be written on the HP3000 that uses a report writer or creates a drop of Image/SQL data tables into basically an ASCII format. Image/SQL shall be used in the example since it is a product that is bundled with the operating system on the HP3000. Once the tables have been dropped, we will need a mechanism for getting the data files over to the NT Server. Again, since FTP is now bundled with MPE, we can use it in a job to export the data files to NT. These files can be "pulled" from the HP3000 by NT or "pushed" by the HP3000. Since we are creating a nightly export job on the 3000 anyway, let's push the files.

At this point, the HP3000 part is complete. A time estimate to do an entire refresh of the customer master and a net change on sales order headers and lines should be measured in minutes and could be appended to your operational system backup job (this estimate obviously varies based on the size of the HP3000, the number of customers that your business has, and how many sales order transactions are processed daily).

It is now time to populate the data mart. Since we are using Microsoft's SQL/Server product, it would make sense to use the BCP utility (bulk copy) to get the data into the data structures. Of course scripts will have to be written prior to this to physically create the data structures. Indexes and columns can be added easily once you are in an RDBMS environment. Thought must be given in advance to assign estimates to disc space requirements, however, since it sometimes requires a rebuild in order to expand disc space allocated. In this example, it would be recommended to built a table that comprised of both sales order headers and lines, perhaps with a unique key on the concatenation of sales order | line number. Performance will be significantly improved as a result of this denormalization effort.

Now you can start developing applications that tie directly to the SQL/Server data source, instead of perhaps using ODBC to go through Image/SQL to get to the production system. Additionally, you will have much better performance as a result of not competing with the production system, and that you are on a dedicated system using a native environment (Microsoft's 32 bit Back Office). On the client side, let's use a 32 bit operating system (Windows 95 or NT Workstation) with the GUI development being done in Visual Basic 5.0 using 3<sup>rd</sup> party ActiveX add-ins.

Even without prior VB knowledge, small applications can be put together in weeks rather than months. The included code sampling and screen shots are demonstrations of an application that was put together using such an approach.

Here is a sample job that will transfer sales order headers and lines to flat files on the NT Server:

```
!JOB TRANEXP, MANAGER.SYS,PUB
```

```

!SETVAR YY “!HPYEAR”
!IF LEN(“!HPMONTH”) < 2 THEN
!  SETVAR MM “0” + “!HPMONTH”
!ELSE
!  SETVAR MM “!HPMONTH”
!ENDIF
!IF LEN(“!HPDATE”) < 2 THEN
!  SETVAR DD “0” + “!HPDATE”
!ELSE
!  SETVAR DD “!HPDATE”
!ENDIF
!SETVAR MYDATE “!YY” + “!MM” + “!DD”
!COMMENT SCRIPT FILES & EXTRACT FILES
!PURGE UNLSHIPL
!CONTINUE
!PURGE UNLSHIPH
!CONTINUE
!PURGE EXPSHIPL
!CONTINUE
!PURGE EXPSHIPH
!CONTINUE
!ECHO CONNECT TO ‘IMAGEDBE’;>ISQLCMDS
!ECHO START UNLSHIPL;>>ISQLCMDS
!ECHO START UNLSHIPH;>>ISQLCMDS
!ECHO EXIT;>>ISQLCMDS
!ECHO YES;>>ISQLCMDS
!ECHO UNLOAD TO EXTERNAL > UNSHIPL
!ECHO EXPSHIPL FROM “SELECT * FROM SALESDB.SHIP_LINE >> UNSHIPL
!ECHO WHERE SL_REQ_DATE = !MYDATE” >> UNSHIPL
!ECHO DSHIPL 10 18 2 4 8 10 2 26 24 12 2 4 2 14 >> UNSHIPL
!ECHO 13 6 13 6 13 6 13 6 13 6 13 6 13 6 13 6 13 6 >> UNSHIPL
!COMMENT 13 6 DEPICTS AN R2 DATA TYPE & 33 12 IS AN R4
!COMMENT OTHER TYPES ARE X10, X18, X2 ETC.
!ECHO 33 12 33 12 33 12 33 12 3 12 >> UNSHIPL
!ECHO 33 12 33 12 33 12 33 12 3 12 >> UNSHIPL
!ECHO 33 12 33 12; >> UNSHIPL
!SAVE UNSHIPL
!ECHO UNLOAD TO EXTERNAL > UNSHIPH
!ECHO ESHIPH FROM “SELECT * FROM SALESDB.SHIP_HEADER >> UNSHIPH
!ECHO WHERE SH_ORDER_DATE = !MYDATE” >> UNSHIPH
!ECHO DSHIPH 6 8 30 30 12 30 30 30 30 10 12 12 >> UNSHIPH
!ECHO 2 2 4 2 4 20 60 60 60 >> UNSHIPH
!ECHO 33 12 33 12 33 12; >> UNSHIPH
!SAVE UNSHIPH
!RUN ISQL.PUB.SYS < ISQLCMDS
!COMMENT INVOLVE FTP TO NT SERVER
!FTP.ARPA.SYS
OPEN NTSERVER
USER ANONYMOUS
TRANJOB@HP3000.COM
PUT EXPSHIPL
PUT EXPSHIPH
QUIT
!EOJ

```

A typical script that would bulk copy these files into MS-SQL/Server might look like this:

```
C:\MSSQL> bcp datawhse..tblSolNew in c:\expsol. /fsol.fmt /SNTSERVER /Usa /P
```

Starting copy...

1000 rows sent to SQL Server. 1000 total

1236 rows copied.

Network packet size (bytes): 4096

Clock Time (ms.): total = 2223 Avg = 1 (556.01 rows per sec.)

```
C:\MSSQL> bcp datawhse..tblSohNew in c:\expsoh. /fsoh.fmt /SNTSERVER /Usa /P
```

Starting copy...

1000 rows sent to SQL Server. 1000 total

1040 rows copied.

Network packet size (bytes): 4096

Clock Time (ms.): total = 1462 Avg = 1 (711.35 rows per sec.)

The following 5 screen captures demonstrate how a Visual Basic screen can show such vital business indicators as who are my top 10 customers, year-to-date and from a profitability standpoint, what products are my largest customer buying during this time frame.



**Conclusions:**

Data warehousing solutions can be vast, cost millions and take years to assemble. It is the hope that this paper has shown that pilot projects can come out of the gate quickly and effectively. Typical costs (excluding hardware) should be from \$1000 to \$1500 per seat. The cost of implementing a data warehousing solution is small compared with the advantages that can be gained by deploying a decision support system correctly. Generally speaking, years have been spent capturing valuable data everyday in your company's "legacy" environment. It is now time to start turning that data into meaningful information. Once a knowledge worker has a tool and this ability within their reach, the benefits to the organization can be immense.