UNIT-I: History of data warehousing

Unit Structure

1.1.0 OBJECTIVES

In this chapter you will learn about:

- Basic of technology, web, data representation etc.
- Basics of Data warehouse and its environment.
- Evolution of Data warehouse 2.0
- Business impact of data warehouse.

1.1.1 INTRODUCTION

In the beginning there were simple mechanisms for holding data. There were punched cards. There were paper tapes. There was core memory that was hand beaded. In the beginning storage was very expensive and very limited.

A new day dawned with the introduction and use of magnetic tape. With magnetic tape, it was possible to hold very large volumes of data cheaply. With magnetic tape, there were no major restrictions on the format of the record of data. With magnetic tape, data could be written and rewritten. Magnetic tape represented a great leap forward from early methods of storage. But magnetic tape did not represent a perfect world. With magnetic tape, data could be accessed

only sequentially. It was often said that to access 1% of the data, 100% of the data had to be physically accessed and read. In addition, magnetic tape was not the most stable medium on which to write data. The oxide could fall off or be scratched off of a tape, rendering the tape useless. Disk storage represented another leap forward for data storage. With disk storage, data could be accessed directly. Data could be written and rewritten. And data could be accessed en masse. There were all sorts of virtues that came with disk storage.

1.1.2 Database management system

Soon disk storage was accompanied by software called a "DBMS" or "data base management system." DBMS software existed for the purpose of managing storage on the disk itself. Disk storage managed such activities as

■ identifying the proper location of data;
■ resolving conflicts when two or more units of data were mapped to the same physical location;
■ allowing data to be deleted;
■ spanning a physical location when a record of data would not fit in a limited physical space and so forth.

Among all the benefits of disk storage, by far and away the greatest benefit was the ability to locate data quickly. And it was the DBMS that accomplished this very important task.

1.1.3 Personal computers and 4GL technology

To placate the end user's cry for accessing data, two technologies emerged—personal computer technology and 4GL technology. Personal computer technology allowed anyone to bring his/her own computer into the corporation and to do his/her own processing at will. Personal computer software such as spreadsheet software appeared. In addition, the owner of the personal computer could store his/her own data on the computer. There was no longer a need for a centralized IT department. The attitude was if the end users are so angry about us not letting them have their own data, just give them the data. At about the same time, along came a technology called "4GL" fourth-generation technology. The idea behind 4GL technology was to make programming and system development so straightforward that anyone could do it. As a result, the end user was freed from the shackles of having to depend on the IT department to feed him/her data from the corporation. Between the personal computer and 4GL technology, the notion was to emancipate the end user so that the end user could take his/her own destiny into his/her own hands. The theory was that freeing the end user to access his/her own data was what was needed to satisfy the hunger of the end user for data. And personal computers and 4GL technology soon found their way into the corporation.
But something unexpected happened along the way. While the end users were now free to access data, they discovered that there was a lot more to making good decisions than merely accessing data.

The end users found that, even after data had been accessed,

■ if the data was not accurate, it was worse than nothing, because incorrect data can be very misleading;

■ incomplete data is not very useful; data that is not timely is less than desirable;

■ when there are multiple versions of the same data, relying on the wrong value of data can result in bad decisions;
■ data without documentation is of questionable value.

It was only after the end users got access to data that they discovered all the underlying problems with the data.

1.1.4 Spider web environment

The result was a big mess. This mess is sometimes affectionately called the "spider's web" environment. It is called the spider's web environment because there are many lines going to so many places that they are reminiscent of a spider's web.

Figure 1.1.1 illustrates the evolution of the spider's web environment in the typical corporate IT environment.

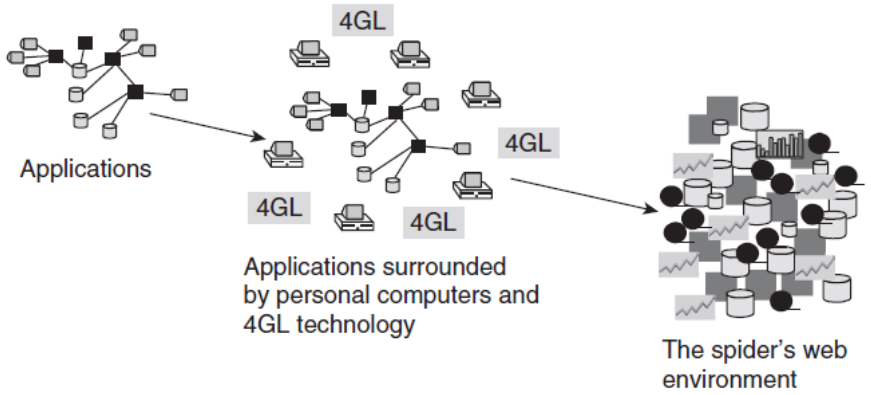


FIGURE 1.1.1 The early progression led to the spider's web environment.

The spider's web environment grew to be unimaginably complex in many corporate environments. As testimony to its complexity, consider the real diagram of a corporation's spider's web of systems shown in Figure 1.1.2.

One looks at a spider's web with awe. Consider the poor people who have to cope with such an environment and try to use it for making good corporate decisions. It is a wonder that anyone could get anything done, much less make good and timely decisions.

The truth is that the spider's web environment for corporations was a dead end insofar as architecture was concerned. There was no future in trying to make the spider's web environment work.

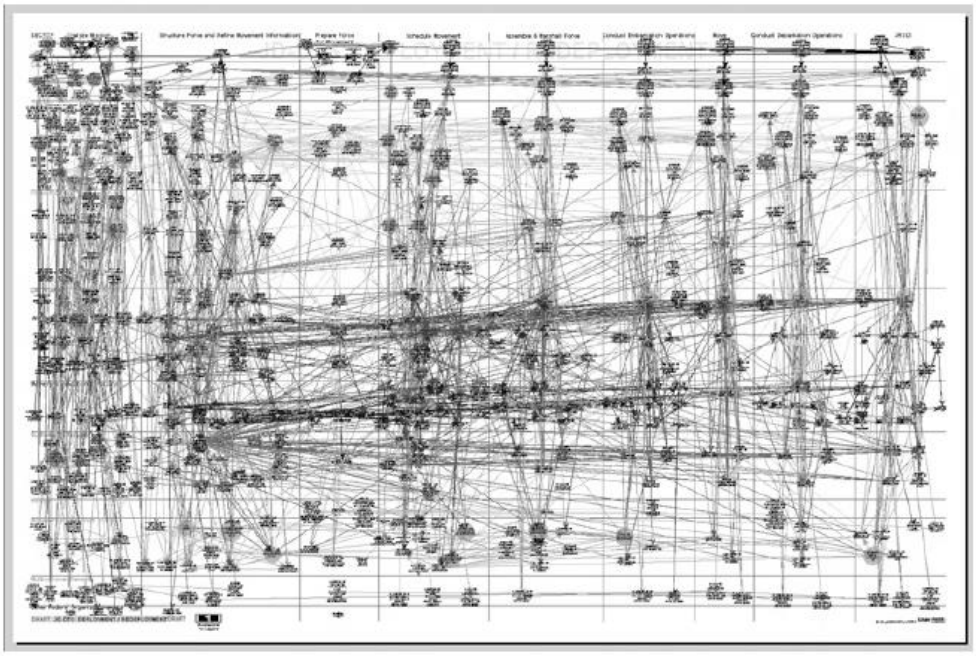


FIGURE 1.1.2 A real spider's web environment.

The frustration of the end user, the IT professional, and management resulted in a movement to different information architecture. That information systems architecture was one that centered on a data warehouse.

1.1.5 Evolution from business perspective

The progression that has been described has been depicted from the standpoint of technology. But there is a different perspective the perspective of the business. From the perspective of the business person, the progression of computers began with simple automation of repetitive activities. The computer could handle more data at a greater rate of speed with more accuracy than any human was capable of. Activities such as the generation of payroll, the creation of invoices, the payments being made, and so forth are all typical activities of the first entry of the computer into corporate life.

Soon it was discovered that computers could also keep track of large amounts of data. Thus were "master files" born. Master files held inventory, accounts payable, accounts receivable, shipping lists, and so forth. Soon there were online data bases, and with online data bases the computer started to make its way into the core of the business. With online data bases airline clerks were emancipated. With online processing, bank tellers could do a whole new range of functions. With online processing insurance claims processing was faster than ever.

It is in online processing that the computer was woven into the fabric of the corporation. Stated differently, once online processing began to be used by the business person, if the online system went down, the entire business suffered, and suffered immediately. Bank tellers could not do their job. ATMs went down. Airline reservations went into a manual mode of operation, and so forth.

Today, there is yet another incursion by the computer into the fabric of the business, and that incursion is into the managerial, strategic decision making aspects of the business. Today, corporate decisions are shaped by the data flowing through the veins and arteries of the corporation. So the progression that is being described is hardly a techno centric process. There is an accompanying set of business incursions and implications, as well.

1.1.6 Data warehouse environment

Figure 1.1.3 shows the transition of the corporation from the spider's web environment to the data warehouse environment. The data warehouse represented a major change in thinking for the IT professional. Prior to the advent of the data warehouse, it was thought that a data base should be something that served all purposes for data. But with the data warehouse it became apparent that there were many different kinds of data bases.

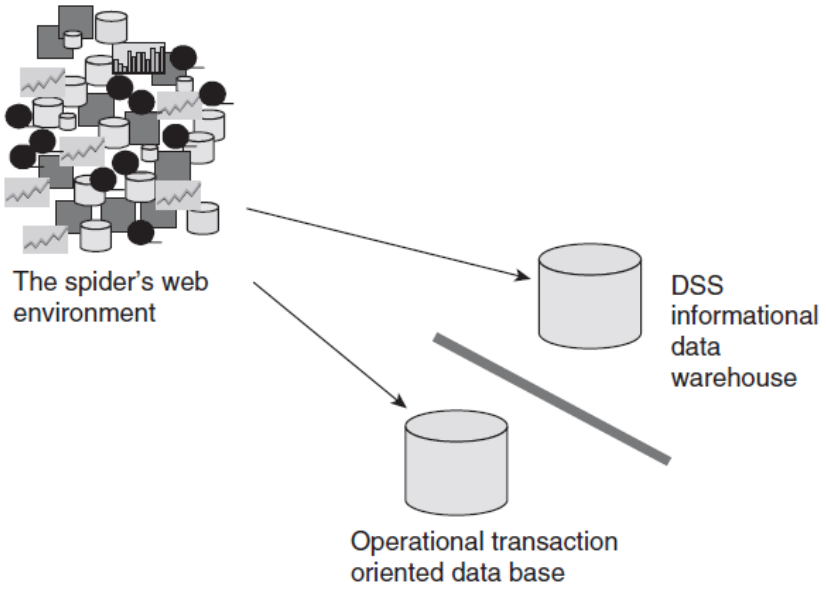


FIGURE 1.1.3 A fundamental division of data into different types of data bases was recognized.

1.1.7 What is data warehouse?

The data warehouse is a basis for informational processing. It is defined as being

- subject oriented;
- integrated;
- nonvolatile;
- time variant;
- a collection of data in support of management's decision.

This definition of a data warehouse has been accepted from the beginning. A data warehouse contains integrated granular historical data. If there is any secret to a data warehouse it is that it contains data that is *both* integrated and granular. The integration of the data allows a corporation to have a true enterprise-wide view of the data. Instead of looking at data parochially, the data analyst can look at it collectively, as if it had come from a single well-defined source, which most data warehouse data assuredly does not. So the ability to use data

warehouse data to look across the corporation is the first major advantage of the data warehouse. Additionally, the granularity—the fine level of detail—allows the data to be very flexible. Because the data is granular, it can be examined in one way by one group of people and in another way by another group of people. Granular data means that there is still only one set of data—one single version of the truth. Finance can look at the data one way, marketing can look at the same data in another way, and accounting can look at the same data in yet another way. If it turns out that there is a difference of opinion, there is a single version of truth that can be returned to resolve the difference. Another major advantage of a data warehouse is that it is a historical store of data. A data warehouse is a good place to store several years' worth of data. It is for these reasons and more that the concept of a data warehouse has gone from a theory derided by the data base theoreticians of the day to conventional wisdom in the corporation today. But for all the advantages of a data warehouse, it does not come without some degree of pain.

1.1.8 Integrating data

The first (and most pressing) pain felt by the corporation is that of the need to integrate data. If you are going to build a data warehouse, you *must* integrate data. The problem is that many corporations have legacy systems that are for all intents and purposes intractable. People are really reluctant to make any changes in their old legacy systems, but building a data warehouse requires exactly that. So the first obstacle to the building of a data warehouse is that it requires that you get your hands dirty by going back to the old legacy environment, figuring out what data you have, and then figuring out how to turn that application-oriented data into corporate data. This transition is *never* easy, and in some cases it is almost impossible. But the value of integrated data is worth the pain of dealing with non-integrated, application-oriented data.

1.1.9 Volumes of data

The second pain encountered with data warehouses is dealing with the volumes of data that are generated by data warehousing. Most IT professionals have never had to cope with the volumes of data that accompany a data warehouse. In the application system environment, it is good practice to jettison older data as soon as possible. Old data is not desirable in an operational application environment because it slows the system down. Old data clogs the arteries. Therefore any good systems programmer will tell you that to make the system efficient, old data must be dumped. But there is great value in old data. For many analyses, old data is extremely valuable and sometimes indispensible. Therefore, having a convenient place, such as a data warehouse, in which to store old data is quite useful.

1.1.10 Different development approach

A third aspect of data warehousing that does not come easily is the way data warehouses are constructed. Developers around the world are used to gathering requirements and then building a system. This time-honored approach is drummed into the heads of developers as they build operational systems. But a data warehouse is built quite differently. It is built iteratively, a step at a time. First one part is built, and then another part is built, and so forth. In almost every case, it is a prescription for disaster to try to build a data warehouse all at once, in a "big bang" approach. There are several reasons data warehouses are not built in a big bang approach. The

first reason is that data warehouse projects tend to be large. There is an old saying: "How do you eat an elephant? If you try to eat the elephant all at once, you choke. Instead the way to eat an elephant is a bite at a time." This logic is never more true than when it comes to building a data warehouse. There is another good reason for building a data warehouse one bite at a time. That reason is that the requirements for a data warehouse are often not known when it is first built. And the reason for this is that the end users of the data warehouse do not know exactly what they want. The end users operate in a mode of discovery. They have the attitude. "When I see what the possibilities are, then I will be able to tell you what I really want." It is the act of building the first iteration of the data warehouse that opens up the mind of the end user to what the possibilities really are. It is only after seeing the data warehouse that the user requirements for it become clear. The problem is that the classical systems developer has never built such a system in such a manner before. The biggest failures in the building of a data warehouse occur when developers treat it as if it
were just another operational application system to be developed.

1.1.11 Evolution to DW 2.0 environment

This chapter has described an evolution from very early systems to the DW 2.0 environment. From the standpoint of evolution of architecture it is interesting to look backward and to examine the forces that shaped the evolution. In fact there have been many forces that have shaped the evolution of information architecture to its highest point—DW 2.0.
Some of the forces of evolution have been:

■ The demand for more and different uses of technology: When one compares the very first systems to those of DW 2.0 one can see that there has been a remarkable upgrade of systems and their ability to communicate information to the end user. It seems almost inconceivable that not so long ago output from computer systems was in the form of holes punched in cards. And end user output was buried as a speck of information in a hexadecimal dump. The truth is that the computer was not very useful to the end user as long as output was in the very crude form in which it originally appeared.

■ Online processing: As long as access to data was restricted to very short amounts of time, there was only so much the business person could do with the computer. But the instant that online processing became a possibility, the business opened up to the possibilities of the interactive use of information intertwined in the day-to-day life of the business. With online processing, reservations systems, bank teller processing, ATM processing, online inventory management, and a whole host of other important uses of the computer became a reality.

■ The hunger for integrated, corporate data: As long as there were many applications, the thirst of the office community was slaked. But after a while it was recognized that something important was missing. What was missing was corporate information. Corporate information could not be obtained by adding together many tiny little applications. Instead data had to be recast into the integrated corporate understanding of information. But once corporate data became a reality, whole new vistas of processing opened up.

■ The need to include unstructured, textual data in the mix: For many years decisions were made exclusively on the basis of structured transaction data. While structured transaction data is certainly important, there are other vistas of information in the corporate environment. There is a wealth of information tied up in textual, unstructured format. Unfortunately unlocking the

textual information is not easy. Fortunately, textual ETL (extract/transform/load) emerged and gave organizations the key to unlocking text as a basis for making decisions.

■ Capacity: If the world of technology had stopped making innovations, a sophisticated world such as DW 2.0 simply would not have been possible. But the capacity of technology, the speed with which technology works, and the ability to interrelate different forms of technology all conspire to create a technological atmosphere in which capacity is an infrequently encountered constraint. Imagine a world in which storage was held entirely on magnetic tape (as the world was not so long ago.) Most of the types of processing that are taken for granted today simply would not have been possible.

■ Economics: In addition to the growth of capacity, the economics of technology have been very favorable to the consumer. If the consumer had to pay the prices for technology that were used a decade ago, the data warehouses of DW 2.0 would simply be out of orbit from a financial perspective. Thanks to Moore's law, the unit cost of technology has been shrinking for many years now. The result is affordability at the consumer level.

These then are some of the evolutionary factors that have shaped the world of technology for the past several decades and have fostered the architectural evolution, the epitome of which is DW 2.0.

1.1.12 Business impact of the data warehouse

The impact of the data warehouse on business has been considerable. Some of the areas of business that are directly impacted by the advent of the data warehouse are:

■ The frequent flyer programs from the airline environment: The single most valuable piece of technology that frequent flyer programs have is their central data warehouse.

■ Credit card fraud analysis: Spending profiles are created for each customer based on his/her past activities. These profiles are created from a data warehouse. When a customer attempts to make a purchase that is out of profile, the credit card company checks to see if a fraudulent use of the credit card is occurring.

■ Inventory management: Data warehouses keep detailed track of inventory, noting trends and opportunities. By understanding at a detailed level the consumption patterns for the goods managed by an organization, a company can keep track of both oversupply and undersupply.

■ Customer profiles: Organizations wishing to "get to know their customer better" keep track of spending and attention patterns as exhibited by their customers and prospects. This detailed information is stored in a data warehouse.

And there are many other ways in which the data warehouse has impacted business. In short the data warehouse becomes the corporate memory. Without the data warehouse there is—at best—a short corporate memory. But with a data warehouse there is a long and detailed corporate memory. And that corporate memory can be exploited in many ways.

1.1.13 Components of data warehouse environment

There are various components of a data warehouse environment. In the beginning those components were not widely recognized.

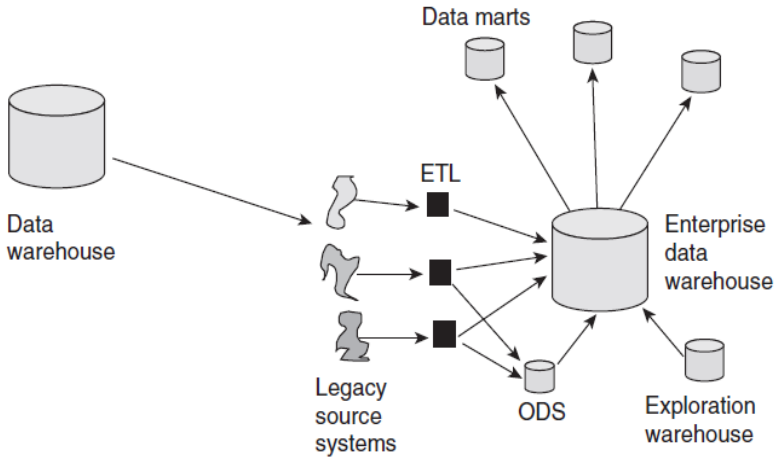


FIGURE 1.1.4 Soon the data warehouse evolved into a full-blown architecture sometimes called the corporate information factory.

But soon the basic components of the data warehouse environment became known. Figure 1.1.4 illustrates the progression from an early stand-alone data warehouse to full-fledged data warehouse architecture. The full-fledged architecture shown in Figure 1.1.4 includes some notable components, which are discussed in the following sections.

1.1.14 Evolution of data warehouse from the business perspective

In the earliest days of computing end users got output from computers in a very crude manner. In those days end users read holes punched in cards and read hexadecimal dumps in which one tiny speck of information hid in a thousand pages of cryptic codes. Soon reports became the norm because the earliest forms of interface to the end users were indeed very crude.

Soon end users became more sophisticated. The more power the end user got, the more power the end user could imagine. After reports came, online information was available instantaneously (more or less). And after online transaction processing, end users wanted integrated corporate data, by which large amounts of data were integrated into a cohesive whole. And then end users wanted historical data.

Throughout this progression came a simultaneous progression in architecture and technology. It is through first-generation data warehouses that the end user had the ultimate in analytical capabilities.

Stated differently, without first-generation data warehouses, the end user was left with only a fraction of the need for information being satisfied. The end user's hunger for corporate information was the single most important driving force behind the evolution to first generation data warehousing.

1.1.15 Other notions about data warehouse

But there are forces at work in the marketplace that alter the notion of what a data warehouse is. Some vendors have recognized that a data warehouse is a very attractive thing, so they have "mutated" the concept of a data warehouse to fit their corporate needs, even though the data warehouse was never intended to do what the vendors advertised.

Some of the ways that vendors and consultants have mutated the notion of a data warehouse are shown in Figure 1.1.5. Figure 1.1.5 shows that there are now a variety of mutant data warehouses, in particular,

■ the "active" data warehouse;
■ the "federated" data warehouse;

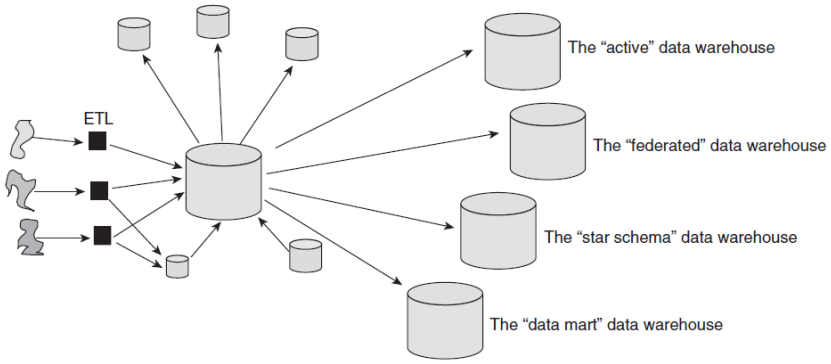


FIGURE 1.1.5 Soon variations of the data warehouse began to emerge.

■ the "star schema" data warehouse (with conformed dimensions);
■ the "data mart" data warehouse.

While each of these mutant data warehouses has some similarity to what a data warehouse really is, there are some major differences between a data warehouse and these variants. And with each mutant variation come some major drawbacks.

1.1.16 Federated data warehouse

In the federated data warehouse approach, there is no data warehouse at all, usually because the corporation fears the work required to integrate data. The idea is to magically create a data warehouse by gluing together older legacy data bases in such a way that those data bases can be accessed simultaneously. The federated approach is very appealing because it appears to give the corporation the option of avoiding the integration of data. The integration of old legacy systems is a big and complex task wherever and whenever it is done. There simply does no way around have to do integration of older data unless you use the federated approach.
But, unfortunately, the federated approach is more of an illusion than a solution. There are *many* fundamental problems with the federated approach, including but not limited to:

■ Poor to pathetic performance: There are many reasons federating data can lead to poor performance. What if a data base that needs to be joined in the federation is down or is currently being reorganized? What if the data base needed for federation is participating in OLTP? Are

there enough machine cycles to service both the OLTP needs and the federated needs of the data base? What if the same query is executed twice or the same data from different queries is needed? It has to be accessed and federated each time it is needed, and this can be a wasteful approach to the usage of resources. This is just the short list of why performance is a problem in the federated environment.

■ Lack of data integration: There is no integration of data with the federated approach. If the data that is being federated has already been integrated, fine; but that is seldom the case. Federation knows little to nothing about the integration of data. If one file has dollars in U.S. dollars, another file has dollars in Canadian dollars, and a third file has dollars in Australian dollars, the dollars are all added up during federation. The fundamental aspects of data integration are a real and serious issue with the federated approach.

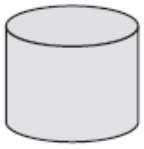
■ Complex technical mechanics: However federation is done; it requires a complex technical mechanism. For example, federation needs to be done across the DBMS technology of multiple vendors. Given that different vendors do not like to cooperate with each other, it is no surprise that the technology that sits on top of them and merges them together is questionable technology.

■ Limited historical data: The only history available to the federated data base is that which already exists in the federated data warehouse data bases. In almost every case, these data bases jettison history as fast as they can, in the quest for better performance. Therefore, usually only a small amount of historical data ever finds its way into the federated data warehouse.

■ Nonreplicable data queries: There is no repeatability of queries in the federated environment. Suppose a query is submitted at 10:00 AM against data base ABC and returns a value of $ 156.09. Then, at 10:30 AM a customer comes in and adds a payroll deposit to his account, changing the value of the account to $ 2971.98. At 10:45 AM the federated query is repeated. A different value is returned for the same query in less than an hour.

■ Inherited data granularity: Federated data warehouse users are stuck with whatever granularity is found in the applications that support the federated query. As long as the granularity that already exists in the data base that will support federation is what users want, there is no problem. But as soon as a user wants a different level of data granularity either higher or lower a serious fundamental problem arises.

Figure 1.1.6 summarizes the fundamental problems with the federated approach to data warehousing.



Problems
– poor to pathetic performance
– no integration of data
– complex technical implementation
– no history
– no repeatability of queries
– improper granularity of data

FIGURE 1.1.6 The federated data warehouse.

1.1.17 Star schema

The star schema approach to data warehousing requires the creation of fact tables and dimension tables. With the star schema approach, many of the benefits of a real data warehouse can be achieved. There are, nevertheless, some fundamental problems with the star schema approach to data warehousing, including:

■ Brittleness: Data warehouses that consist of a collection of star schemas tend to be " brittle. " As long as requirements stay exactly the same over time, there is no problem. But as soon as the requirements change, as they all do over time, either massive changes to the existing star schema must be made or the star schema must be thrown away and replaced with a new one. The truth is that star schemas are designed for a given set of requirements and only that given set of requirements.

■ Limited extensibility: Similarly, star schemas are not easy to extend. They are pretty much limited by the requirements upon which they were originally based. One audience: Star schemas are optimized for the use of one audience. Usually one set of people fi nd a given star schema optimal, while everyone else finds it less than optimal. The essential mission of a data warehouse is to satisfy a large and diverse audience.

A data warehouse is suboptimal if there are people who are served in a less than satisfactory manner, and creating a single star schema to serve all audiences does just that.

■ Star schema proliferation: Because a single star schema does not optimally satisfy the needs of a large community of users, it is common practice to create multiple star schemas. When multiple star schemas are created, inevitably there is a different level of granularity for each one and data integrity comes into question. This proliferation of star schemas makes looking for an enterprise view of data almost impossible.

■ Devolution: To solve the problem of multiple granularities across different stars, the data in each star schema must be brought to the lowest level of granularity. This (1) defeats the theory of doing star schemas in the first place and (2) produces a classical relational design, something the star schema designer finds abhorrent.

Figure 1.1.7 shows the challenges of using a star schema for a data warehouse.

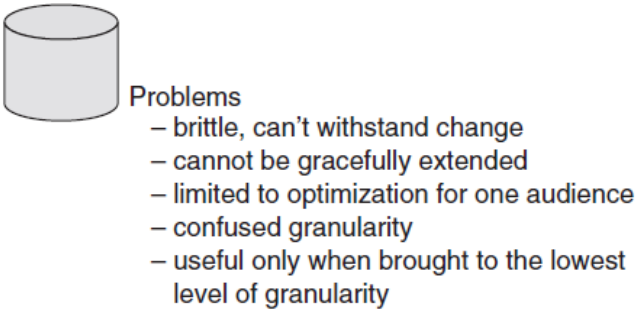


FIGURE 1.1.7 The star schema data warehouse.

1.1.18 Data mart

Many vendors of online application processing (OLAP) technology are enamored of the data mart as a data warehouse approach. This gives the vendor the opportunity to sell his/her products first, without having to go through the building of a real data warehouse. The typical OLAP vendor's sales pitch goes" Build a data mart first, and turn it into a data warehouse later. " Unfortunately there are many problems with building a bunch of data marts and calling them a data warehouse. Some of the problems are:

■ No reconcilability of data: When management asks the question " What were last month's revenues? " accounting gives one answer, marketing gives another, and finance produces yet another number. Trying to reconcile why accounting, marketing, and sales do not agree is very difficult to do.

■ Extract proliferation: When data marts are first built, the number of extracts against the legacy environment is acceptable, or at least reasonable. But as more data marts are added, more legacy data extracts have to be added. At some point the burden of extracting legacy data becomes unbearable.

■ Change propagation: When there are multiple data marts and changes have to be made, those changes echo through all of the data marts. If a change has to be made in the finance data mart, it probably has to be made again in the sales data mart, then yet again in the marketing data mart, and so forth. When there are many data marts, changes have to be made in multiple places. Furthermore, those changes have to be made in the same way. It will not do for the finance data mart to make the change one way and the sales data mart to make the same change in another way. The chance of error quickly grows at an ever-increasing rate. The management, time, money, and discipline required to ensure that the propagating changes are accurately and thoroughly completed are beyond many organizations.

■ Non extensibility: When the time comes to build a new data mart, unfortunately most must be built from scratch. There is no realistic way for the new data mart to leverage much or even any of the work done by previously built data marts. All of these factors add up to the fact that with the data mart as a data warehouse approach, the organization is in for a maintenance nightmare.

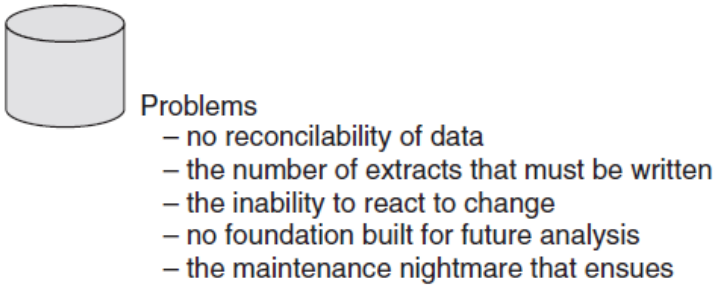


FIGURE 1.1.8 The data mart data warehouse

Figure 1.1.8 depicts the problems with the data mart as a data warehouse approach. It is an interesting note that it is not possible to build a data mart and then grow it into a data warehouse. The DNA of a data mart is fundamentally different from the DNA of a data warehouse. The theory of building a data mart and having it grow up to be a data warehouse is akin to asserting that if you plant some tumbleweeds, when they grow up they will be oak trees. The DNA of tumbleweed and the DNA of an oak tree are different. To get an oak tree, one must plant acorns. When planted, tumbleweed seeds yield tumbleweeds. The fact that in the springtime, when they first start to grow, oak trees and tumbleweeds both appears as small green shoots is merely a coincidence. Likewise, while there are some similarities between a data mart and a data warehouse, thinking that one is the other, or that one can mutate into the other, is a mistake.

1.1.19 LET US SUM UP

Data warehousing has come a long way since the frustrating days when user data was limited to operational application data that was accessible only through an IT department intermediary. Data warehousing has evolved to meet the needs of end users who need integrated, historical, granular, flexible, accurate information. The first-generation data warehouse evolved to include disciplined data ETL from legacy applications in a granular, historical, integrated data warehouse. With the growing popularity of data warehousing came numerous changes—volumes of data, a spiral development approach, heuristic processing, and more. As the evolution of data warehousing continued, some mutant forms emerged:

■ Active data warehousing
■ Federated data warehousing
■ Star schema data warehouses
■ Data mart data warehouses

While each of these mutant forms of data warehousing has some advantages, they also have introduced a host of new and significant disadvantages. The time for the next generation of data warehousing has come.

1.1.20 List of References

- DW2.0 The architecture for Next Generation of Datawarehousing W.H. Inmon, Derek Strauss, Genia Neushloss, ELSEVIER.
- Building the data warehouse, W.H.Inmon, third Edition, Wiley.
- Data warehousing, S. Mohanty, TMH .
- The Data Warehouse Lifecycle toolkit", Ralph Kimball ,John Wiley.

1.1.21 Unit End Exercises

1) Write short note on Spider Web Environment.
2) Outline the factors which led to the evolution of DW 2.0
3) What are impacts of the data warehouse on business?
4) Write a short note on star schema.

UNIT-I: Introduction

Unit Structure

1.2.0 Objectives
1.2.1 Introduction
1.2.2 Lifecycle of data warehouse
1.2.3 reasons for different sectors
1.2.4 metadata,
1.2.5 Access of data
1.2.6 structures data/ unstructured data
1.2.7 Textual analysis
1.2.8 blather

1.2.0 OBJECTIVES

In this chapter you will learn about:

- Lifecycle of data warehouse.
- What is metadata?
- Types of data-structured and unstructured.
- Application of DW.

1.2.1 INTRODUCTION

To resolve all of the choices in data warehouse architecture and clear up all of the confusion, there is DW 2.0. DW 2.0 is the definition of data warehouse architecture for the next generation of data warehousing. To understand how DW 2.0 came about, consider the following shaping factors:

■ In first-generation data warehouses, there was an emphasis on getting the data warehouse built and on adding business value. In the days of first-generation data warehouses, deriving value meant taking predominantly numeric-based, transaction data and integrating that data. Today, deriving maximum value from corporate data means taking ALL corporate data and deriving value from it. This means including textual, unstructured data as well as numeric, transaction data.

■ In first-generation data warehouses, there was not a great deal of concern given to the medium on which data was stored or the volume of data. But time has shown that the medium on which data is stored and the volume of data are, indeed, very large issues.

■ In first-generation data warehouses, it was recognized that integrating data was an issue. In today's world it is recognized that integrating old data is an even larger issue than what it was once thought to be.

■ In first-generation data warehouses, cost was almost a nonissue. In today's world, the cost of data warehousing is a primary concern.

■ In first-generation data warehousing, metadata was neglected. In today's world metadata and master data management are large burning issues. In the early days of first-generation data

warehouses, data warehouses were thought of as a novelty. In today's world, data warehouses are thought to be the foundation on which the competitive use of information is based. Data warehouses have become essential.

■ In the early days of data warehousing, the emphasis was on merely constructing the data warehouse. In today's world, it is recognized that the data warehouse needs to be malleable over time so that it can keep up with changing business requirements.

■ In the early days of data warehousing, it was recognized that the data warehouse might be useful for statistical analysis. Today it is recognized that the most effective use of the data warehouse for statistical analysis is in a related data warehouse structure called the exploration warehouse.

We are indeed much smarter today about data warehousing after several decades of experience building and using these structures.

1.2.2 Lifecycle of data warehouse

In first-generation data warehousing it was thought that data need only be placed on a form of disk storage when a data warehouse was created. But that is only the beginning—data merely starts a life cycle as it enters a data warehouse. It is naïve of developers of first generation data warehouses to think otherwise.

In recognition of the life cycle of data within the data warehouse, the DW 2.0 data warehouse includes four life-cycle " sectors " of data. The first sector is the Interactive Sector. Data enters the Interactive Sector rapidly. As data settles, it is integrated and then is passed into the Integrated Sector. It is in the Integrated Sector that—not surprisingly— integrated data is found. Data remains in the Integrated Sector until its probability of access declines. The falling off of the probability of data access usually comes with age. Typically, after 3 or 4 years the probability of data access in the Integrated Sector drops significantly. From the Integrated Sector the data can then move on to one of two sectors. One sector that the data can go to is the Near Line Sector. In many ways, the Near Line Sector is like an extension of the Integrated Sector. The Near Line Sector is optional. Data does not have to be placed there. But where there is an extraordinarily large amount of data and where the probability of access of the data differs significantly, then the Near Line Sector can be used. The last sector is the Archival Sector. Data residing in the Archival Sector has a very low probability of access. Data can enter the Archival Sector from either the Near Line Sector or the Integrated Sector. Data in the Archival Sector is typically 5 to 10 years old or even older. What does the life cycle of data look like as it enters the DW 2.0 data warehouse? Figure 1.2.1 illustrates the DW 2.0 data life cycle.

Data enters the DW 2.0 environment either through ETL from another application or from direct applications that are housed in the Interactive Sector. The Interactive Sector is the place where online update of data occurs and where high-performance response time is found. The data that enters the Interactive Sector is very fresh, perhaps only a few seconds old. The other kind of data that is found in the Interactive Sector is data that is transacted as part of a co resident application. In this case, the data is only milliseconds old.

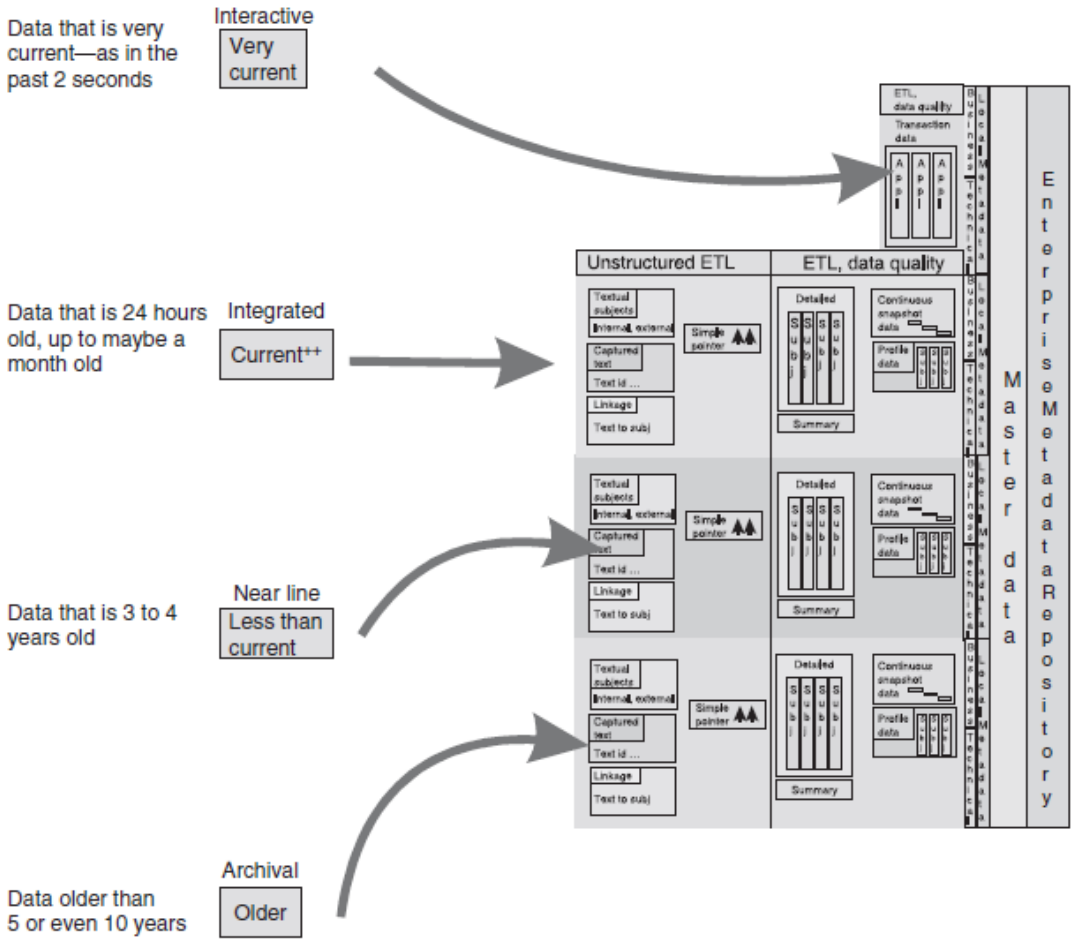


FIGURE 1.2.1 There is a life cycle of data within the DW 2.0 environment. And there is a sector of data that corresponds to the different ages of data in the data warehouse.

As an example of the data found in the interactive sector, consider an ATM transaction. In an ATM transaction, the data is captured upon the completion of an ATM activity. The data may be as fresh as less than- a-second old when it is entered into the Interactive Sector. The data may enter the Interactive Sector in one of two ways. There may be an application outside of the DW 2.0 data warehouse by which data is captured as a by-product of a transaction. In this case, the application executes the transaction and then sends the data through ETL into the Interactive Sector of the data warehouse.

The other way the data can enter the Interactive Sector is when the application is also part of a DW 2.0 data warehouse. In this case, the application executes the transaction and enters the data immediately into the Interactive Sector.

The point is that in the Interactive Sector the application may exist outside of the sector or the application may actually exist inside the Interactive Sector. Transaction data is inevitably application oriented, regardless of its origins. Data in the Interactive Sector arrives in an application state. At some point in time, it is desirable to integrate transaction, application data. That point in time may be seconds after the data has arrived in the Interactive Sector or the integration may be done days or weeks later. Regardless, at some point in time, it is

desirable to integrate application data. That point in time is when the data passes into the Integrated Sector.

Data enters the Integrated Sector through ETL. By passing into the Integrated Sector through ETL, the data casts off its application state and acquires the status of corporate data. The transformation code of ETL accomplishes this task. Once in the integrated state, the data is accumulated with other similar data. A fairly large collection of data resides in the Integrated Sector, and the data remains in the integrated state as long as its probability of access is high. In many organizations, this means that the data stays in the Integrated Sector for from 3 to 5 years, depending on the business of the organization and the decision support processing that it does.

In some cases, there will be a very large amount of data in the Integrated Sector coupled with very vigorous access of the data. In this case it may be advisable to use near-line storage as a cache for the integrated data. The organization can make a very large amount of data electronically available by using the Near Line Sector. And the cost of integrated data storage with near-line storage makes the cost of the environment very acceptable. Data is placed in near-line storage when the probability of access of the data drops significantly. Data whose probability of access is high should not be placed there. It is assumed that any data placed in near-line storage has had its probability of access verified by the analyst controlling the storage location of corporate data.

The last DW 2.0 sector of data is the Archival Sector. The Archival Sector holds data that has been collected electronically and may see some usage in the future. The Archival Sector is fed by the Near Line Sector or the Integrated Sector. The probability of access of data in the Archival Sector is low. In some cases, data may be stored in the Archival Sector for legislated purposes, even where it is thought the probability of access is zero.

1.2.3 Reasons for different sectors

There are many reasons for the different data sectors in the DW 2.0 environment. At the heart of the different sectors is the reality that as data passes from one sector to another the basic operating characteristics of the data change. Figure 1.2.2 illustrates some of the fundamental operating differences between the various sectors. Figure 1.2.2 shows that the probability and the pattern of access of data from sector to sector differ considerably. Data in the Interactive Sector is accessed very frequently and is accessed randomly. Data in the Integrated Sector is also accessed frequently, usually sequentially and in bursts.

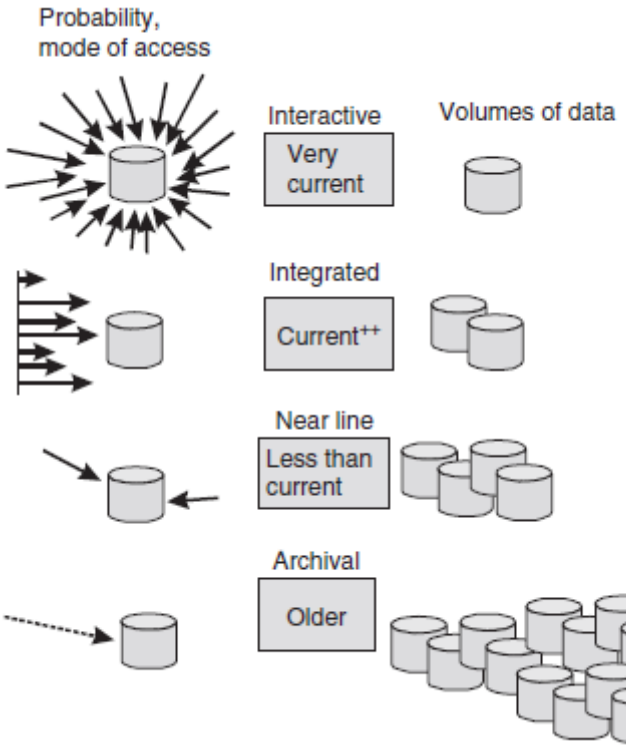


FIGURE 1.2.2 As data goes through its life cycle within DW 2.0, its probability of access and its volume change dramatically

Data in the Near Line Sector is accessed relatively infrequently, but when it is accessed, it is accessed randomly. Data in the Archival Sector is accessed very infrequently, and when it is accessed, it is accessed sequentially or, occasionally, randomly. In addition to the various patterns of access, data in different sectors are characterized by distinctly different volumes. There is relatively little data in the Interactive Sector. There is more data in the Integrated Sector. If an organization has near-line data at all, there is usually a considerable amount of data in the Near Line Sector. And Archival Sector data can mount up considerably. Even though there will be relatively little archival data in the early years of collecting it, as time passes, there is every likelihood that a large amount of data will collect in the Archival Sector.

There is a conundrum here. In the classical data warehouse, all data was just placed on disk storage, as if all data had an equal probability of access. But as time passes and the volume of data that is collected mounts, the probability of access of data drops, giving rise to a certain Irony: The more data that is placed on disk storage, the less the data is used.

This is expensive, and poor performance is the result. In fact, it is *very* expensive. Poor performance and high cost are not the only reasons first-generation data warehousing is less than optimal. There are other good reasons for dividing data up into different life-cycle sectors. One reason is that different sectors are optimally served by different technologies.

1.2.4 Metadata

As a case in point, consider metadata. Metadata is the auxiliary descriptive data that exists to tell the user and the analyst where data is in the DW 2.0 environment. Figure 1.2.3 illustrates the very different treatment of metadata in the Interactive and Archival Sectors of the DW 2.0 architecture.

Figure 1.2.3 shows that it is common industry practice to store metadata separately from the actual data itself. Metadata is stored in directories, indexes, repositories, and a hundred other places. In each case, the metadata is physically separate from the data that is being described by the metadata.

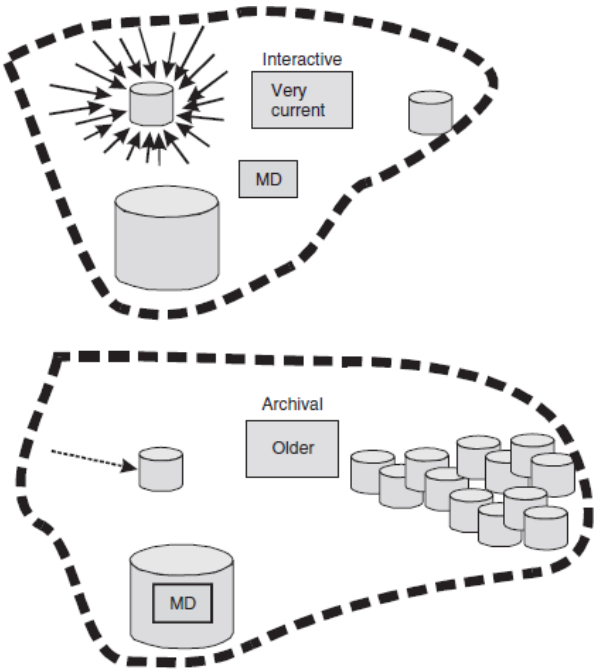


FIGURE 1.2.3 With interactive data, metadata is stored separately; with archival data, metadata is stored directly with the data.

In contrast, metadata is stored physically with the data that is being described in the Archival Sector. The reason for including metadata in the physical embodiment in the archival environment is that archival data is like a time vault containing data that might not be used for 20 or 30 years. Who knows when in the future archival data will be used or for what purpose it will be used? Therefore, metadata needs to be stored with the actual data, so that when the archival data is examined it will be clear what it is. To understand this point better,
consider the usage of archival data that is 30 years old. Someone accesses the archival data and then is horrified to find that there is no clue no Rosetta Stone indicating what the data means. Over the 30-year time frame, the metadata has been separated from the actual data and now no one can find the metadata. The result is a cache of archival data that no one can interpret.

However, if the metadata had been stored physically with the actual data itself, then when the archivist opened up the data 30 years later, the meaning, format, and structure of the actual data would be immediately apparent. From the standpoint of the end user, there is a relationship between end user satisfaction and metadata. That relationship is that with metadata, the end user can determine if data and analysis already exist somewhere in the corporation. If there is no metadata, it is difficult for the business person to determine if data or analysis already is

being done. For this reason, metadata becomes one of the ways in which the business person becomes enamored of the DW 2.0 environment.

1.2.5 Access of data

Access of data is another fundamental difference between the various DW 2.0 data sectors. Consider the basic issue of the access of data depicted in Figure 1.2.4.

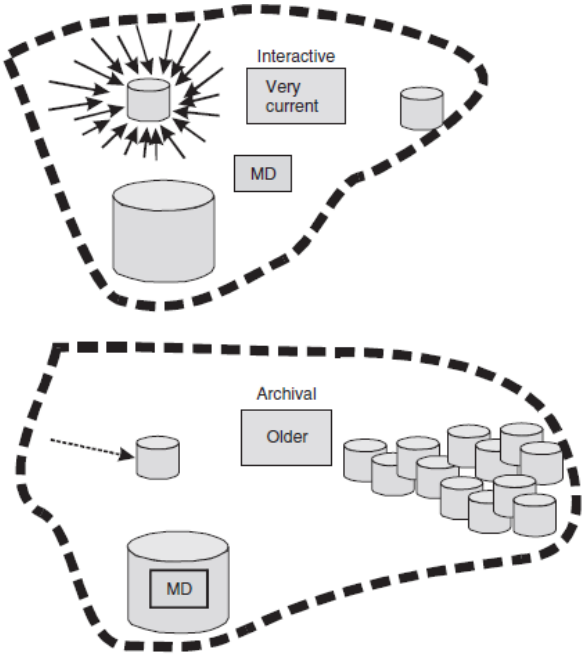


FIGURE 1.2.4 Another major difference is in the mode of access of data.

This figure highlights the very fundamental differences in the pattern and frequency of data access in the Interactive Sector and the Archival Sector. Interactive data is accessed randomly and frequently. One second, one transaction comes in requiring a look at one unit of data. In the next second, another transaction comes in and requires access to a completely different unit of data. And the processing behind both of these accesses to data requires that the data be found quickly, almost instantaneously. Now consider the access of archival data. In the case of archival data, there is very infrequent access of data. When archival data is accessed, it is accessed sequentially, in whole blocks of records.

Furthermore, the response time for the access of archival data is relaxed. It is seen then that the pattern of access of data is very different for each sector of data in the DW 2.0 architecture. The technology for the sectors varies as well. Therefore, it is true that no single technology no "one size fits all" technology is optimal for the data found in a modern data warehouse. The old notion that you just put data on disk storage and everything takes care of itself simply is not true. While the life cycle of data is a very important aspect of DW 2.0, it
is not the only departure from the first-generation data warehouse. Another major departure is the inclusion of both structured and unstructured data in the DW 2.0 environment.

1.2.6 Structures data/ unstructured data

There are two basic types of data structured and unstructured. Structured data is data that comes repetitively in the same format and layout, usually through the issuance of transactions. Typical examples of structured data include data generated by bank transactions, airline reservation transactions, insurance transactions, manufacturing transactions, retail transactions, and so forth. Structured data is conveniently stored in data base records, where there are attributes, keys, indexes, tables, and so forth. In fact the entire world of structured
data is well served by standard data base technology. The other world is that of unstructured data. Unstructured data exists in two basic forms textual and non textual. Textual unstructured data occurs in many places—emails, telephone conversations, PowerPoint presentations, and more. Non textual unstructured data occurs as graphics and images, including but not limited to photographs, X-rays, MRIs, diagrams, and illustrations.

While current technology is not yet able to handle non textual data with much elegance, textual unstructured data is another story. Textual unstructured data can be captured and manipulated. Standard data base technology has a hard time with textual data because textual
data is not repetitive like structured data. However, this does not mean that there is not great value in textual data. Indeed, there is great value in it. It is just that the textual data is not easily handled and manipulated by standard data base technology.

DW 2.0 calls for unstructured textual data to be placed in a data warehouse and to be integrated along with structured data. Great opportunity for the innovative use of information arises by so doing.

Figure 1.2.5 shows the fundamental division between structured data and unstructured data in the DW 2.0 architecture and highlights the need to incorporate and integrate both kinds of data. There are many challenges in the bringing together of structured and unstructured data in the DW 2.0 environment. One challenge is the integration of structured and unstructured data. Some unstructured data cannot be integrated with structured data, but other unstructured
data can. In fact unstructured data comes in two “flavors” unstructured data and semi structured data. Unstructured data is that which is just text written in a free-form manner. A book, a manual, or a training course often has large amounts of unstructured text. Semi structured data is data that is textual, but in which there is repeatability of the format of the text. As an example, a recipe book is a single document. But within the recipe book are many recipes. Each recipe has its own ingredients and its own instructions. Each recipe is a form of semi structured data.

The integration of textual data into the DW 2.0 environment requires that both unstructured data and semi structured data be used as input.

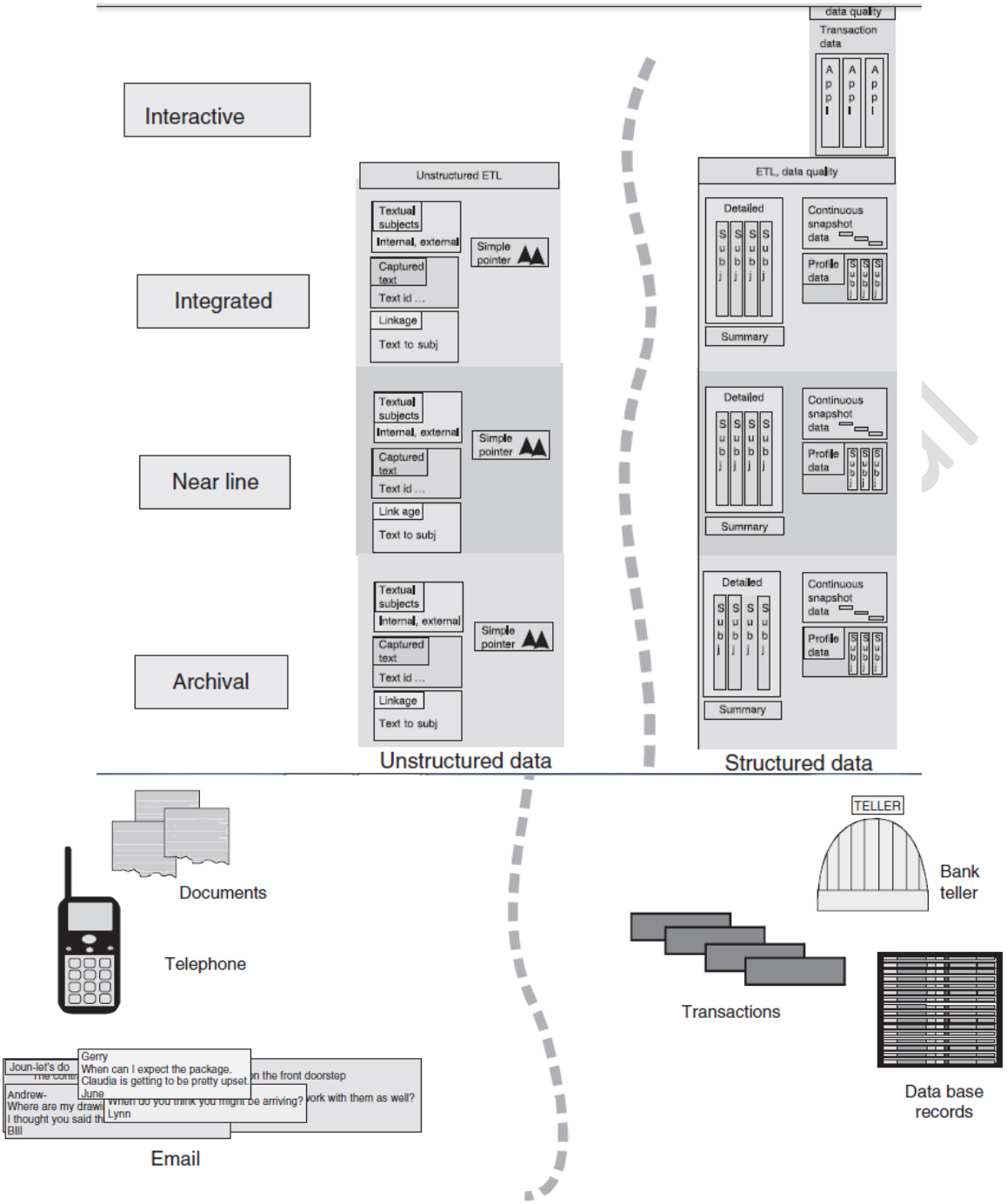


FIGURE 1.2.5 There are two distinct worlds that are very different in many ways.

1.2.7 Textual analysis

The integration of unstructured and structured data in the DW 2.0 data warehouse enables a different kind of analysis than has ever been done before analytical processing against textual data. Analytical processing can be done against unstructured data or against a combination of structured and unstructured data, as represented in Figure 1.2.6.

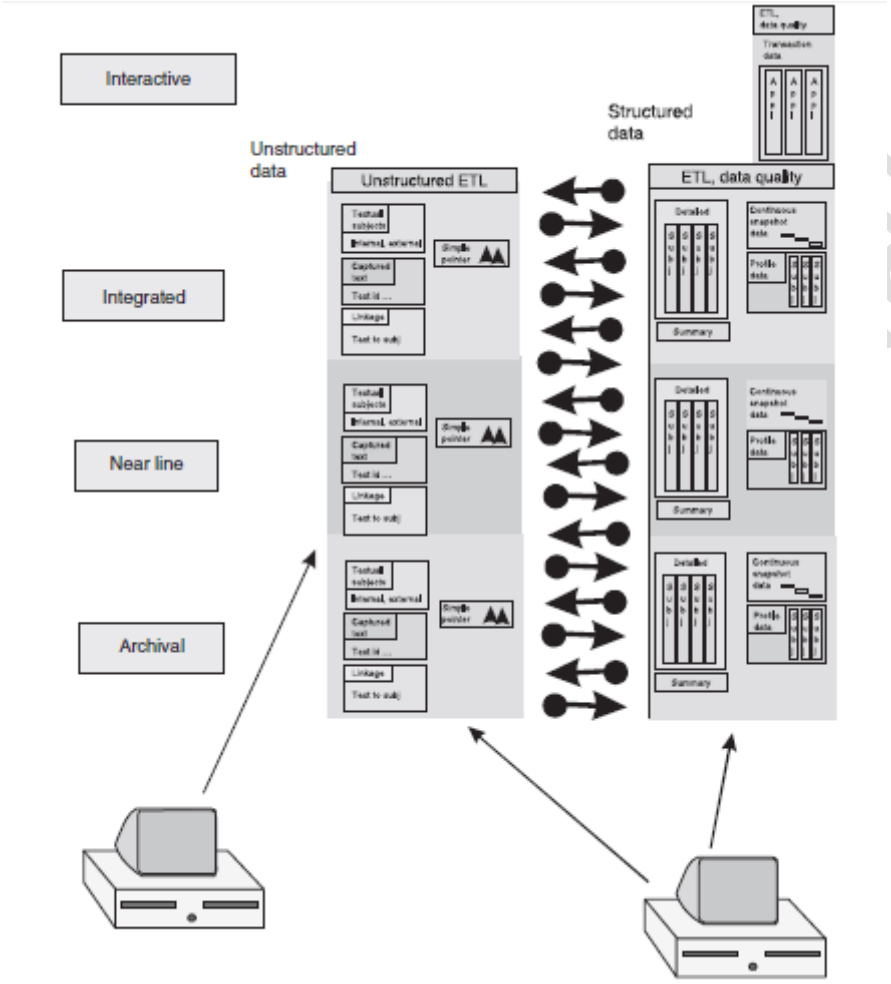


FIGURE 1.2.6 When unstructured data is included in a data warehouse: (a) unstructured data can be accessed and analyzed by itself, (b) unstructured data and structured data can be analyzed together, and (c) some unstructured data can be closely linked with structured data.

1.2.8 Blather

The challenges of incorporating unstructured data with structured data in a DW 2.0 data warehouse are many. One of the challenges is that of screening unstructured data. For a variety of reasons it is true that unstructured data contains what can be called "blather." Blather is data that has no meaning to the business of the corporation. A typical example of blather occurs in email. Consider the email message, "Honey, what's for dinner tonight?"

The world of email is full of personal content like this. Personal email has no relevance or bearing on the business of the corporation. If one is storing email, one normally does not need

to see or keep blather. It just gets in the way. Email is not the only unstructured data that needs to be screened for blather. All unstructured data needs to be screened for blather. Stated differently, if the corporation does not screen for blather, then the unstructured data that gets loaded into the DW 2.0 environment may be largely irrelevant and bloated, neither of which is conducive to good analysis. Therefore, screening is an important process in the collection and management of unstructured data.

Figure 1.2.7 represents the screening of unstructured data before it enters the DW 2.0 environment. Screening unstructured data for blather is just the first of many steps that need to be done to prepare unstructured data for the DW 2.0 environment. Another major activity in the preparation of unstructured data for the DW 2.0 environment is that of creating a general (or "normalized") textual foundation of the unstructured data. If unstructured data is going to be useful analytically, it must be transformed into data that can be analyzed both generally and specifically.

To understand the need for both a general and a specific foundation for unstructured text, consider this. In almost every organizational case, unstructured text is generated by multiple sources. Whatever unstructured data is being considered, whether it be documents, emails, medical records, etc., it is a good bet that the unstructured data was not written by the same person.

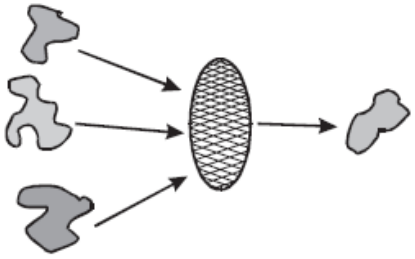


FIGURE 1.2.7 Unstructured data needs to be screened.

1.2.9 Issue of terminology

Because text is written by many different people, the different terminologies used by different people must be taken into consideration. Due to differences in background, age, ethnicity, social class, education, country of origin, native language, and many other factors, people have many different ways of expressing the same thing. If these different ways of expressing the same things are not "rationalized" (or "normalized"), then it will be impossible to do a meaningful analysis on the textual data.

Therefore, if textual data is to be used for textual analytics, it must first be subjected to a normalization process. There is another reason text needs to be normalized before textual analysis can occur. That reason is that when an author writes text the author may have no idea that the text will be read and used as input into a data warehouse. Therefore there is no consideration by the author of any activity of terminology resolution. The analyst doing preparation of text for entry into a data base has the challenge of reading the text and describing to the system that will ingest the text how the text is to be interpreted. The process of describing

text to a machine is not easy and must be done as part of the terminology rationalization process.

The normalization process for text requires that the text be cast in two forms a specific form and a general form. The specific form is usually what the person has said or written. The general form is the normalized value of the data represented by what the person said or wrote.

As an example of the general and the specific forms of data, suppose that a person has written, "The tarsus has been subjected to stress and has disarticulated." This expression would be the specific original text. Now suppose someone were doing a search on broken bones. Looking up bones would not turn up tarsus, and looking up broken would not turn up disarticulated. But if the text had been preprocessed into specific and general terms, the term tarsus could have been recognized as a bone and the term disarticulated could have been recognized as an expression for broken. The creation of the general from the specific could have created two expressions tarsus/ bone and disarticulated/broken. After the connection between the specific and the general has been made, and after both the specific and the general forms have been placed in the DW 2.0 environment, an analyst looking for broken bones in the unstructured data will immediately find a disarticulated tarsus.

So a second major step in preparing unstructured data for the DW 2.0 environment is to read the specific data and to superimpose the general data over the specific to make the data fit for analysis. Anything short of that is just a waste of time and opportunity.

1.2.10 Specific text

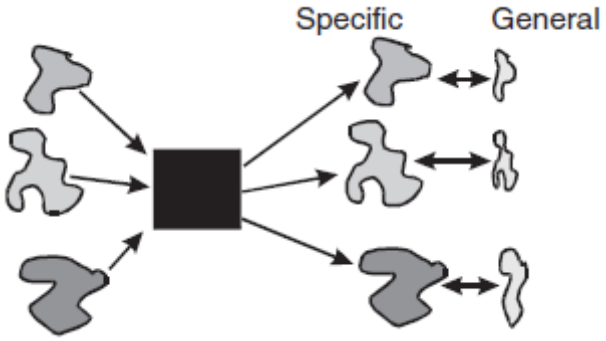


FIGURE 1.2.8 A transformation needs Specific General to be made from raw text to specific/ general text.

Figure 1.2.8 illustrates the need for reading specific data and creating a general interpretation of the general textual data before the data is passed into a DW 2.0 environment. The recognition of the life cycle of data in the data warehouse and the inclusion of unstructured data in the data warehouse are important advances over first-generation data warehouses. But they are not the only improvements that DW 2.0 has to offer next-generation data warehouse users. From the business perspective it is mandatory that terminology be rationalized. If the text is not normalized, it is a futile effort to try to allow the business person to do analytical processing against textual data.

1.2.11 Metadata – a major component

The DW 2.0 approach recognizes that metadata is a major and critically important part of the data warehouse infrastructure. For a variety of reasons, metadata was never recognized or included as a major part of the first-generation data warehouse. With the advent of the next generation of data warehousing, metadata will not be forgotten again.

Metadata is the information that sits atop the data and describes what is in the actual data. Metadata is important in any environment, but is especially important in the DW 2.0 environment. So why is metadata important to DW 2.0 when it was optional or even forgotten in first-generation data warehouses? There are some very good, very compelling reasons for the importance of metadata in the DW 2.0 architecture:

■ Size and diversity: Today's data warehouses are bigger and more diverse than previous data warehouses. While it once may have been possible to keep track of what data was in a data warehouse informally, due to the volume and diversity of data warehouse data today, it is not possible to keep track of the contents of a data warehouse.

■ More diverse users: There are more, and more-diverse, users for today's data warehouses. There once were only a few users who formed a close-knit community. Today there are many users with diverse backgrounds. Keeping these users informed about what is in a data warehouse is a full-time job unto itself.

■ Broad metadata scope: Metadata lies at the heart of successful DSS (Decision Support Systems) processing. To do optimal DSS processing, the end user analyst has to know many things about the data that is available for analysis. The end user analyst needs to know where the data is from, what the data means, what calculations were made, what source data was included, what source data was excluded, when the data became available, and more. All of this metadata information is important to the end user analyst.

■ Administrative requirements: Administration of the data warehouse environment becomes more complex as the data warehouse ages and grows. The better the metadata, the better and easier the administration of the warehouse environment.

These, then, are just a few of the reasons for the inclusion of metadata in DW 2.0. Figure 1.2.9 shows the metadata layer in the DW 2.0 architecture. In a way, the metadata component of the DW 2.0 architecture is simple—at least in concept. The metadata that resides in the DW 2.0 environment merely serves to describe the actual DW 2.0 data. Figure 1.2.10 illustrates the DW 2.0 metadata perspective. It is worth noting that DW 2.0 metadata needs to describe *both* structured and unstructured data. It is true that the classical use of metadata has been for the description of structured data. But with the inclusion of unstructured data, metadata is revealed as useful for describing the unstructured data as well. Indeed, part of the integration of structured and unstructured data is the integration of metadata describing both of these components of the DW 2.0 data environment.

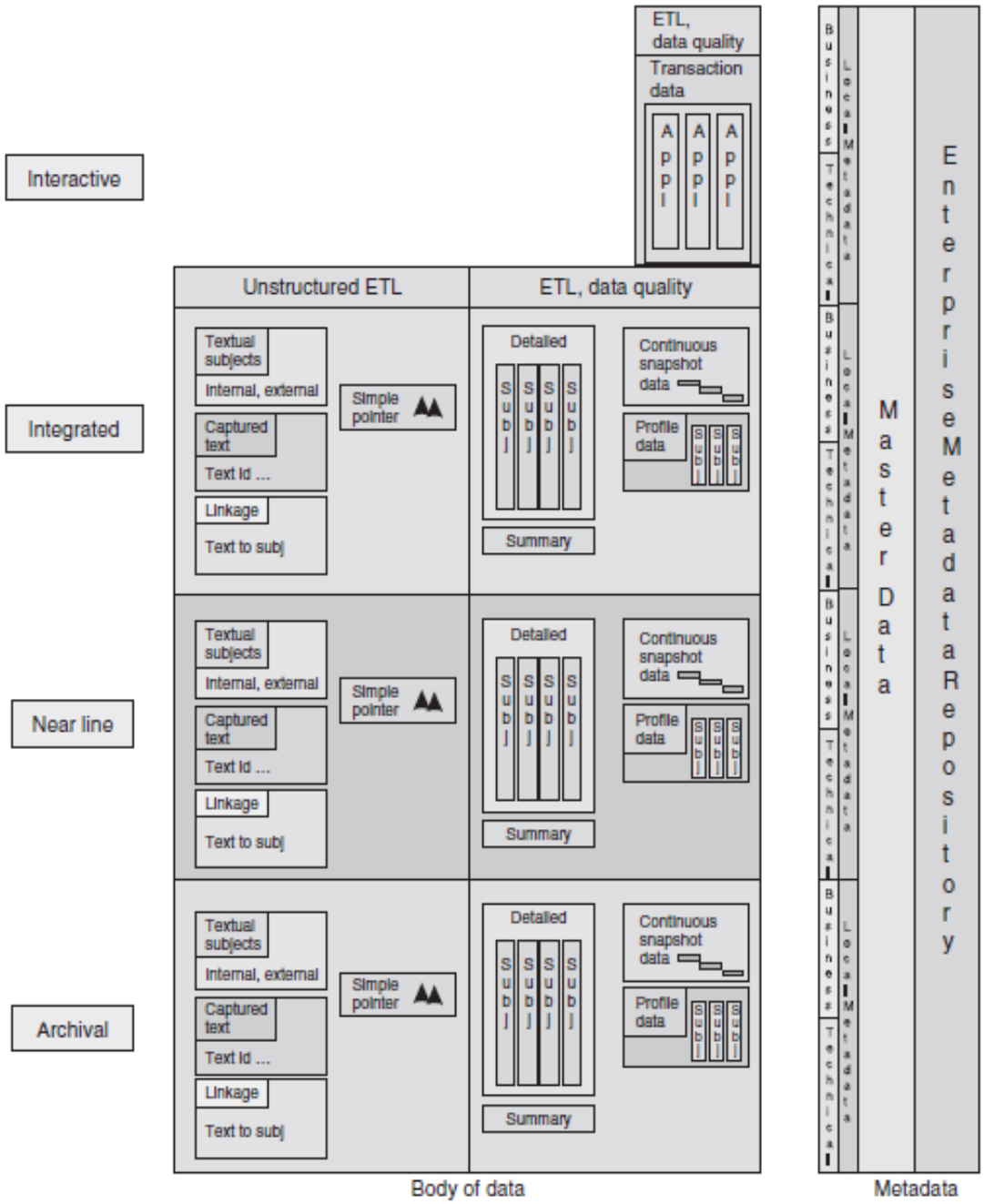


FIGURE 1.2.9 Another major component of DW 2.0 is metadata

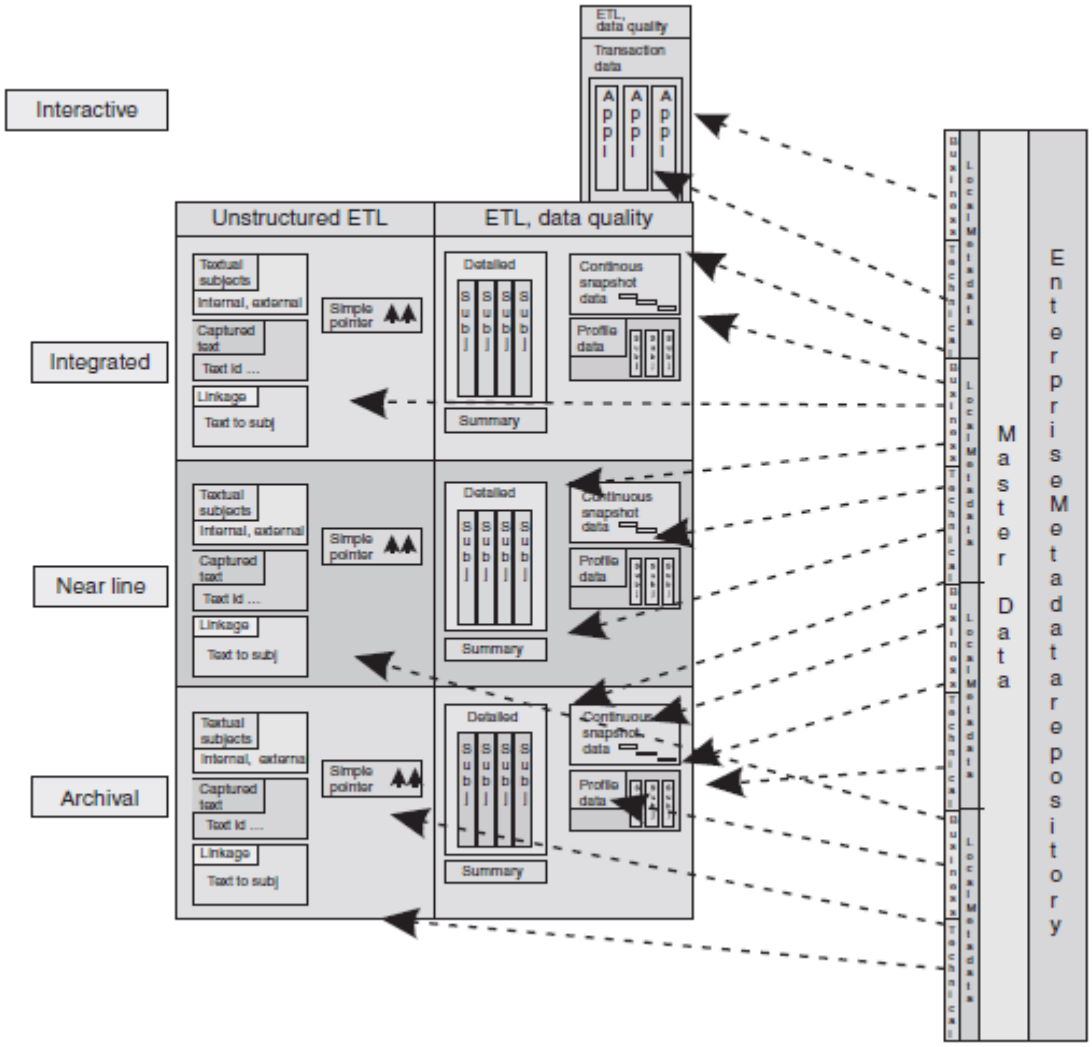


FIGURE 1.2.10 A simple way of thinking about metadata is that metadata describes one or more aspects of the body of data that lies in DW 2.0.

1.2.12 Local metadata

Go to any conference and listen to the speakers and the hue and cry for metadata will be unavoidable. The fact of the matter is that there is plenty of metadata around today. There is metadata in the ETL tools, metadata in DBMS directories, and metadata in BI (Business Intelligence) tools . . . metadata everywhere! But that metadata is all specific to the tools that are being used. It can be called "local metadata." What is missing is *enterprise-wide* metadata. DW 2.0 calls for both layers of metadata—local and enterprise-wide. Figure 1.2.11 illustrates the resulting DW 2.0 hierarchy, relationships, and subtypes. Figure 1.2.11 shows that the DW 2.0 architecture includes several types of metadata and an overall metadata structure. There is master metadata or "reference data. " There is enterprise-wide metadata. There is local metadata, such as that found in a given tool. Then, within the local metadata, there are both business and technical metadata. Business metadata is metadata that is written in the language of the business person and is appropriate to the business of the corporation.

Technical metadata (the kind most people are familiar with) is useful to and relevant to the corporation's technicians. It is fair to say that today most metadata in the corporation is metadata relating to technology, not business.

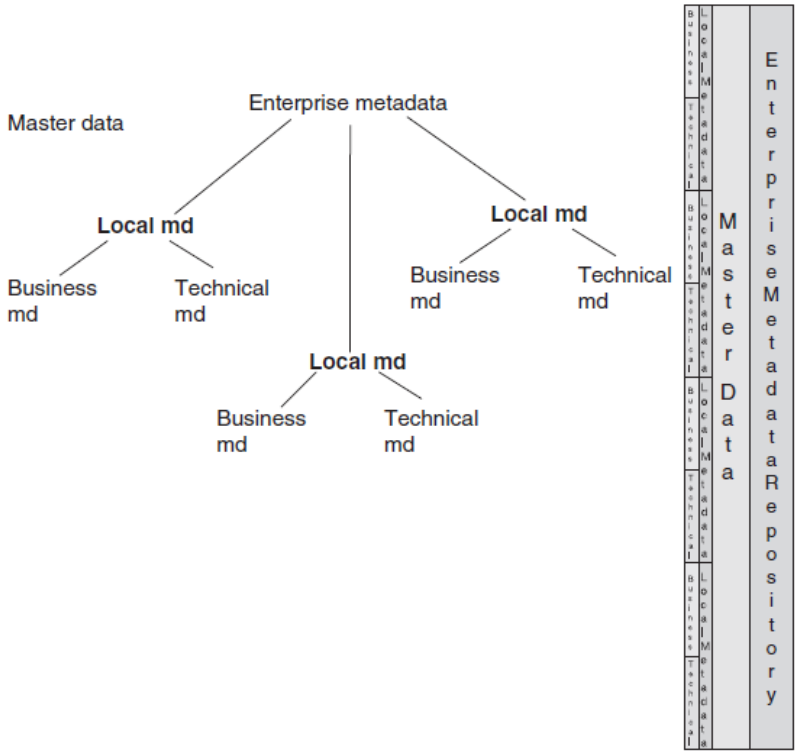


FIGURE 1.2.11 The structure of the metadata environment.

DW 2.0 offers some major significant improvements over first generation data warehouses. DW 2.0 offers the recognition that there is a life cycle of data once it enters the data warehouse, the notion that structured and unstructured data belong in the data warehouse and the idea that metadata is an essential and normal part of the data warehouse.

There is yet another major change from the first-generation data warehouse environment to the DW 2.0 environment. That improvement is the notion that the technical foundation of the data warehouse should not be set in technology that cannot be easily changed.

1.2.13 Changing business requirements

It is said that the only two constants are death and taxes. There should be a third—business change. Business change happens to everyone, albeit at different rates and in different manners. Figure 1.2.12 depicts business requirements changing over time. While business change is a predictable constant reality, changing the technological infrastructure upon which the business depends is a different matter altogether. For a variety of reasons, the technology
that underpins most businesses is rooted in cement. Making changes to the technological infrastructure is very, very difficult to do. There is a disparity, then, between business that changes all the time and a technological infrastructure that does not change. In Figure 1.2.13, a data warehouse has been built to meet requirements depicted as an orange box. When the business requirements and conditions change, they need to look like a blue circle. The technical infrastructure, depicted by a data can, is originally engineered to support the initial orange-box

business requirements. This technical infrastructure remains static and unchanged after the changes are made in the business. An organization's data warehouse is a significant, often mission-critical, component of its technology infrastructure. The problem is that the data warehouse cannot change easily and is designed to meet the business requirements of an earlier time. Little by little the technology infrastructure changes, and one day the technology catches up to where it should be. But by this time the business has changed again. Figure 1.2.14 depicts this never-ending disparity. This, then, is a dilemma for the data warehouse architect. DW 2.0 addresses this dilemma by mandating that the technology be placed on a foundation of dynamism, which can change over time with ease.

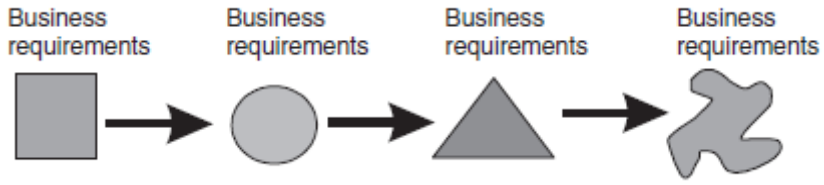


FIGURE 1.2.12 Over time business requirements change.

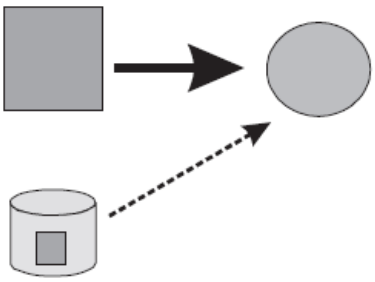


FIGURE 1.2.13 The business requirements have changed but the data warehouse has not.

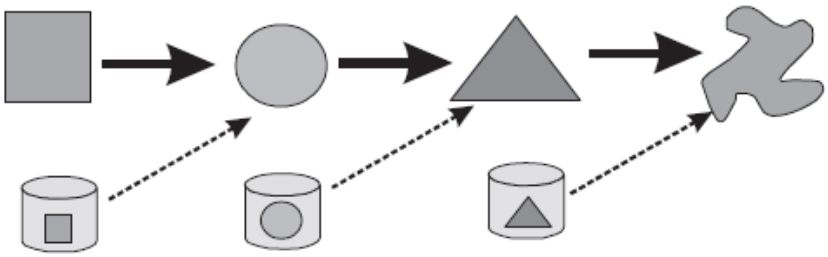


FIGURE 1.2.14 Unless the data warehouse can be placed on a foundation of dynamic technology, it is constantly reflecting yesterday's business requirements.

1.2.14 Flow of data within DW 2.0

The simple diagram that has been used to represent DW 2.0 shows the components and their approximate relationships to each other. What is not shown is the general flow of data throughout the DW 2.0 environment. While it is possible for almost any unit of data to fl ow almost anywhere, in general there is a predictable flow of data through the DW 2.0 for the vast majority of data. For structured processing, data enters the system either directly through an application that is a part of the interactive environment or from an application external to the DW 2.0 environment. Data that originates in an application external to DW 2.0 is processed through an ETL interface and then flows into the Interactive Sector. Once structured data is in the Interactive Sector, it then flows into the Integrated Sector, where the data is integrated and transformed into corporate data. Data subsequently flows from the Integrated Sector either to the Near Line Sector or to the Archival Sector. Near Line Sector data also eventually flows into the Archival Sector. Unstructured data follows a similar path. Unstructured data begins life as a document or some other form of textual data. The textual data undergoes an ETL process for unstructured data. Then, the unstructured data enters the Integrated Sector of the DW 2.0 environment.

It is questionable if the Near Line Sector applies to unstructured data. However, if there is a need for near-line data in the unstructured environment, the unstructured data can flow into that environment. In any case, data flows into the archival portion of unstructured data either from the Integrated Sector or from the Near Line Sector if it has been used. The flow of both structured and unstructured data through the components of the DW 2.0 environment is depicted in Figure 1.2.15.

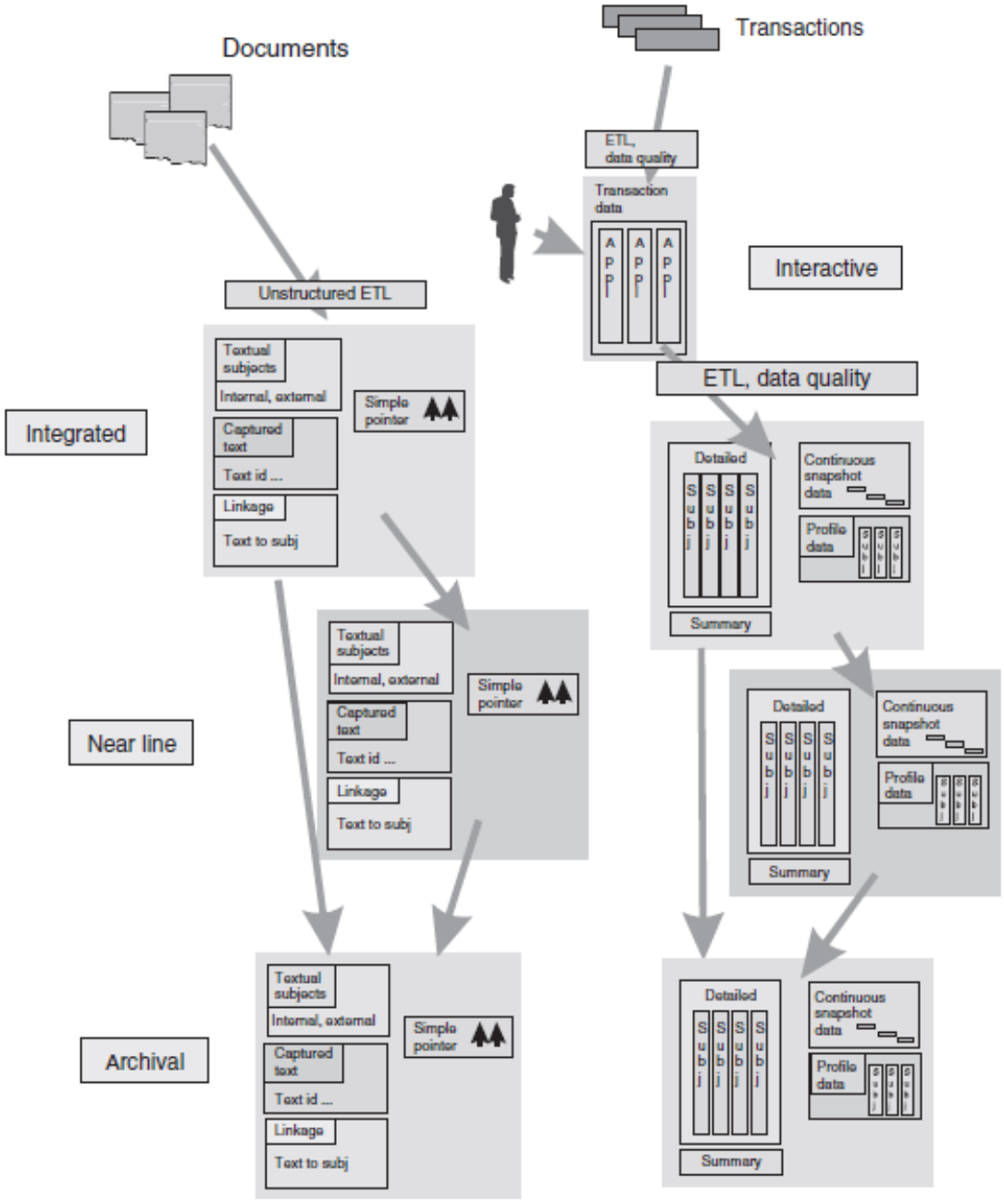


FIGURE 1.2.15 The general flow of data throughout the DW 2.0 environment.

1.2.15 Volumes

Another interesting way to view structured and unstructured data in the DW 2.0 environment is from the perspective of where the volumes of data reside, as shown in Figure 1.2.16. There is typically relatively little interactive data in the structured portion of the DW 2.0 environment. The volume of structured integrated data increases substantially. When the Near Line Sector is used, the volume of structured data it must support increases yet again. And then, as the Archival Sector reaches maturity, the volume of archived structural data increases very dramatically.

In contrast, the volume of data in all of the life-cycle sectors of the unstructured environment is invariably higher and increases at a faster rate than its structured environment counterpart.

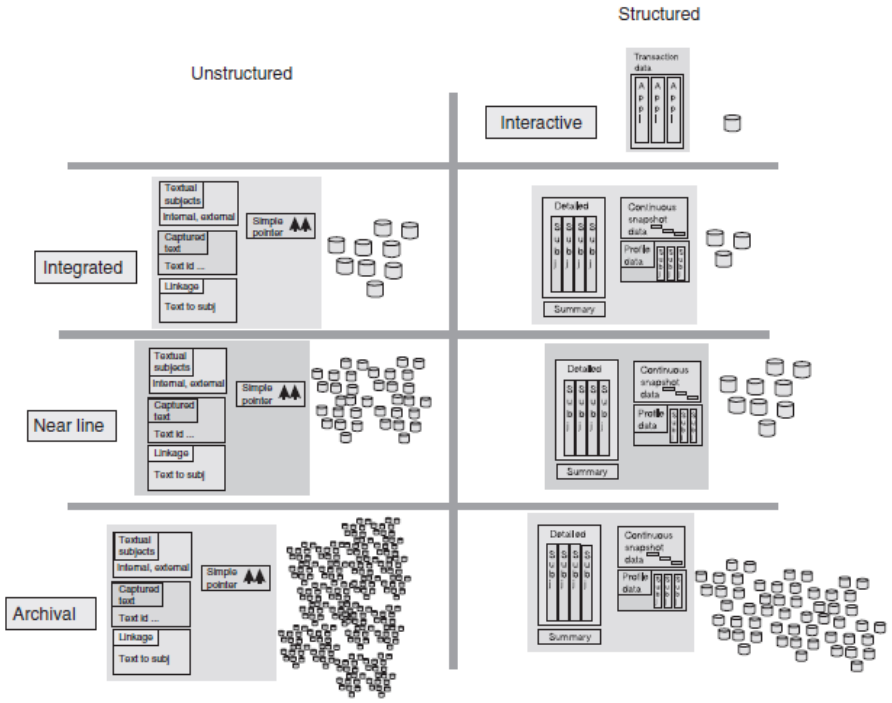


FIGURE 1.2.16 The expected volumes of data at the different sectors in the DW 2.0 environment.

It is estimated that there is four to five times as much unstructured data in the typical corporation as there is structured data. When blather and other non useful text are factored out, there probably remains two to three times as much data in the unstructured environment as there is in the structured environment. These proportional data volume relationships are also illustrated in Figure 1.2.16

1.2.16 Useful applications

DW 2.0 offers the possibility of new applications that have never before been possible. DW 2.0's support for both structured and unstructured data in the data warehouse environment gives rise to some of the most interesting applications from the blending of both kinds of data. Figure1. 2.17 shows that when unstructured communications, such as email, co reside in a data warehouse with demographics derived from structured data, it is possible for the first time to create a true 360 ° view of a customer.

Another useful combination of the unstructured data and structured data environment occurs when unstructured doctor's notes meet laboratory test results. A complete, composite, electronic patient record becomes possible, practical, and easy to achieve, as represented in Figure 1.2.18.

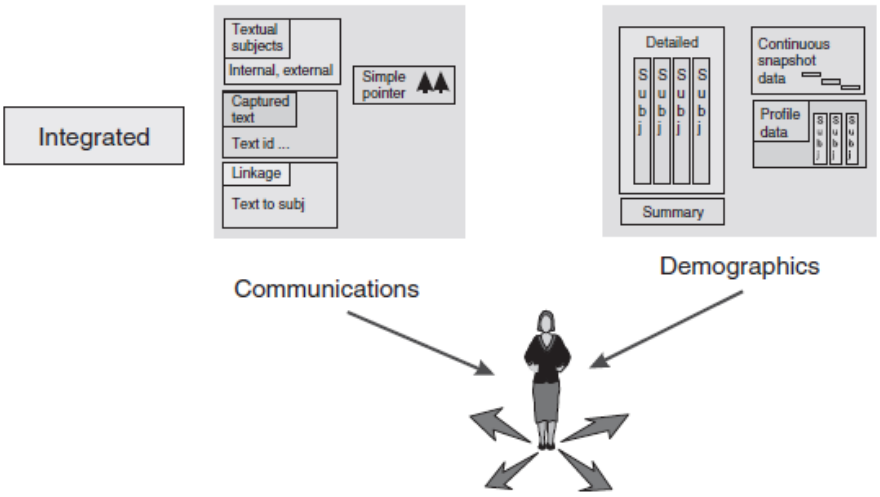


FIGURE 1.2.17 A true 360° view of the customer.

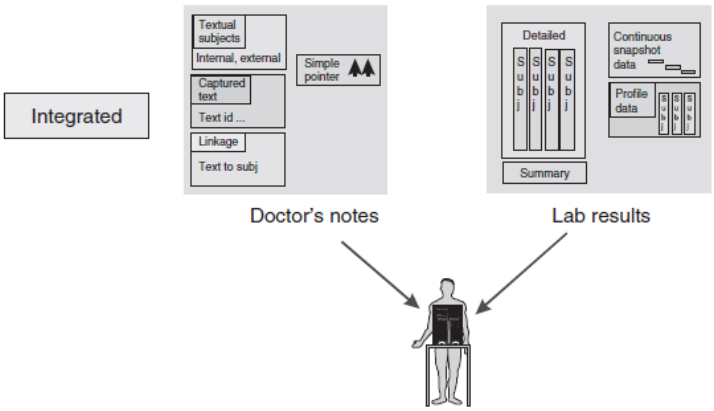


FIGURE 1.2.18 Creating the personal medical health record.

1.2.17 DW 2.0 and referential integrity

Referential integrity has been around for a long time. Referential integrity suggests that data residing in a data base must also be governed by a set of logical rules. For example, if the medical procedure is childbirth, the patient's sex must equal female. Or, if a purchase is made, there must be a product or service that has been purchased. The DW 2.0 approach extends the notion of referential integrity. In DW 2.0 there is inter-sector referential integrity and there is intra-sector referential integrity. Figure 1.2.19 shows the two different types of referential integrity. In Figure 1.2.19, inter-sector referential integrity refers to preservation of the integrity of data as it passes from one sector to another. Intra-sector referential integrity refers to preservation of the integrity of data within a sector.

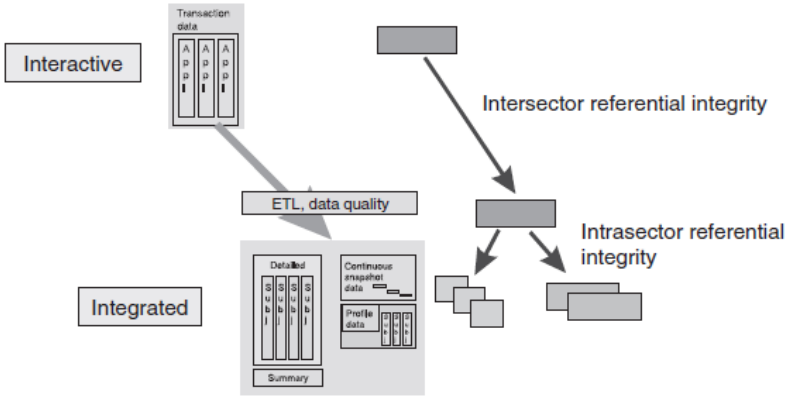


FIGURE 1.2.19 There are two forms of referential integrity within the DW 2.0 environment inter-sector and intra-sector.

1.2.18 Reporting in DW 2.0

Reporting can occur almost anywhere within the DW 2.0 environment. There is no one place where reports are run. Instead, there are different kinds of reports that are run in different places. Figure 1.2.20 shows that some reports are run operationally out of the Interactive Sector. Others are run out of the Integrated Sector. Some reports are run using a combination of structured and unstructured data from the Interactive and the Integrated Sectors of the DW 2.0 environment. And yet other reports are run from the unstructured portion of the DW 2.0 environment.

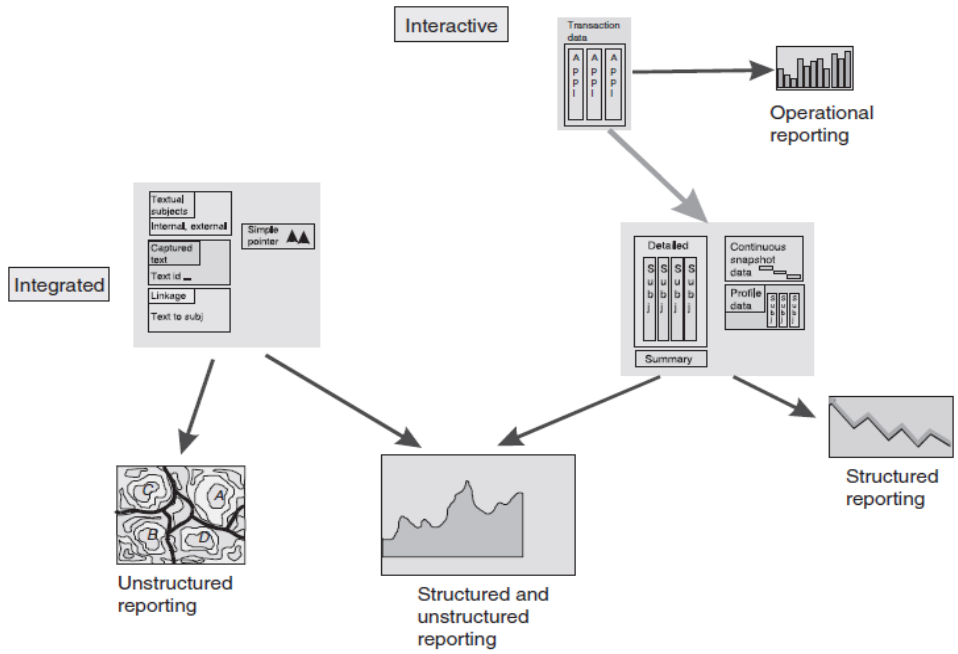


FIGURE 1.2.20 There are different kinds of reporting that occur throughout the DW 2.0 environment.

1.2.19 LET US SUM UP

DW 2.0 is the next generation of architecture for the data warehouse environment. There are many differences between DW 2.0 and first-generation data warehousing. The four largest differences are

■ recognition of the life cycle of data as it passes into and then resides in a data warehouse;
■ inclusion of unstructured data in the data warehouse;
■ inclusion of metadata in the DW 2.0 environment;
■ placement of DW 2.0 on a technology foundation that can change over time.

DW 2.0 has four major data life-cycle sectors:

■ Interactive Sector, where data warehousing is done in an update mode at transaction response-time rates
■ Integrated Sector, where data is integrated and analytical processing can be done
■ Near Line Sector, which serves as a caching area for data in the Integrated Sector
■ Archival Sector, where data that may still be needed goes after a significant decline in the probability that it will be accessed DW 2.0 includes unstructured data as well as structured data.

DW 2.0 includes unstructured text that is put into a data warehouse must first go through an integration process. The integration process is necessary to prepare the unstructured text for textual analytics. If integration is not done for unstructured text, textual analytics cannot be done effectively.

1.2.20 List of References

- DW2.0 The architecture for Next Generation of Datawarehousing W.H. Inmon, Derek Strauss, Genia Neushloss, ELSEVIER.
- Building the data warehouse, W.H.Inmon, third Edition, Wiley.
- Datawarehousing, S. Mohanty, TMH .
- The Data Warehouse Lifecycle toolkit", Ralph Kimball ,John Wiley.

1.2.21 Unit End Exercises

1) Explain in brief the Life Cycle of data in DW 2.0.
2) List the reasons which make Metadata very important in DW 2.0 architecture.
3) What are the steps involved in processing of unstructured data?
4) What is metadata? Give the reasons for the importance of metadata in the DW2.0 architecture.

UNIT-I: DW Components

Unit Structure

1.3.0 OBJECTIVES

In this chapter you will learn about:

- How data is entering in interactive sector and ETL?
- Data is integrated in integrated sector.
- Near Line Sector contains data that is a mirror image of the data found in the Integrated Sector
- Archival Sector contains packages of data that are self-contained.

1.3.1 INTRODUCTION

DW 2.0 is made up of four different sectors: the Interactive Sector, the Integrated Sector, the Near Line Sector, and the Archival Sector. Depending on the size and age of the data warehouse, different sectors will either be in use or not be in use. For example, in the early days of a data warehouse, it is unlikely that any archival data will be present. For a small data warehouse, there may never be any near-line storage. The exact implementation of the DW 2.0 data warehouse varies considerably among different corporations. Each of the different sectors has its own set of considerations and characteristics. Indeed, within the same sector, the considerations for structured data and unstructured data vary considerably. From the business perspective, there are usually different types of users that access and analyze data from each sector.

The office community makes widespread usage of the Interactive Sector. Day-to-day office activities are transacted here. The Integrated Sector indirectly supports all levels of management from the junior level to the president of the company. The Near Line Sector is used by the analytical community, and the Archival Sector is infrequently used, or is used by an actuarial or engineering community.

There are then different communities of users who access and use the different sectors of the DW 2.0 environment.

1.3.2 Interactive sector

The Interactive Sector is the place where data enters the DW 2.0 environment. Data enters either through ETL, where the application is external to DW 2.0, or as part of a transaction from an application implemented inside of the Interactive Sector. Figure 3.1 shows the various places from which data enters the Interactive Sector.

There may be multiple applications within the Interactive Sector, as depicted in Figure 1.3.1. The applications may or may not have integrated data. The applications in the Interactive Sector can do updates and can have high-performance transaction processing in which sub second response time is the norm.

The workload of transactions passing through the Interactive Sector is illustrated by Figure 1.3.2. In Figure 1.3.2 it is seen that there are many small transactions passing through the interactive workload. This is the equivalent of having a road on which only Porsches and Ferraris are allowed to run. Because there are no slow-running vehicles on the road, the average speed is very fast.

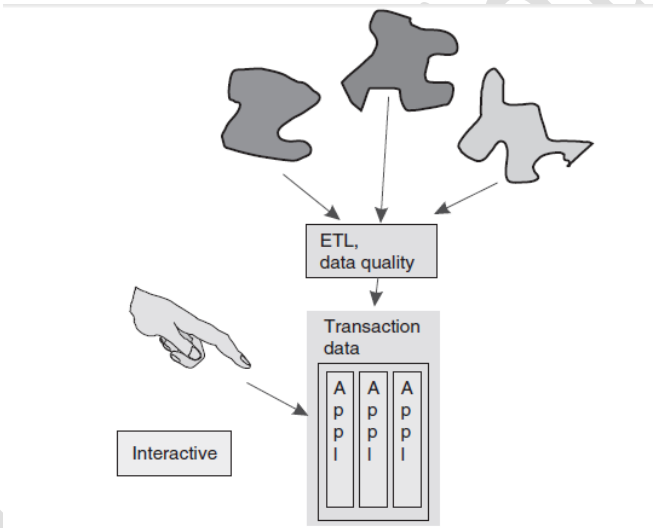


FIGURE 1.3.1 Where input to the interactive layer comes from

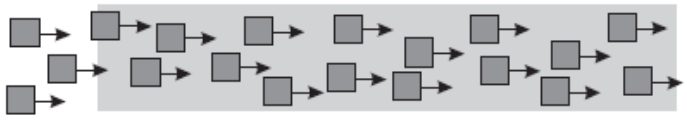


FIGURE 1.3.2 What the workload going on inside the Interactive Sector looks like.

There is consistently good response time for the activities that are running through the system. Another feature of the Interactive Sector is the volume of data that is managed by the technology of the sector. There is only a modest amount of data that is found in the Interactive Sector, as illustrated in Figure 1.3.3.

The range of data is anywhere from a few gigabytes to up to a terabyte of data running in the Interactive Sector. Relative to other parts of the DW 2.0 environment, the volumes of interactive data that are found here are small. The interactive data almost always resides on disk storage. Because interactive data is stored on disk, and because the interactive workload is consistently made up of small, fast transactions, the response times that are achieved are very fast. Figure 1.3.4 represents the good response time that is the norm for the Interactive Sector.

In addition to having fast performance, the transactions that are run through the Interactive Sector are able to do updates. Data can be added, deleted, or modified, as shown in Figure 1.3.5 for the Interactive Sector.

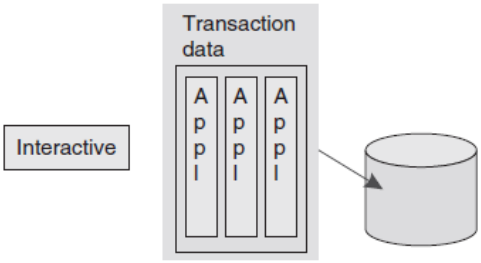


FIGURE 1.3.3 There is a modest amount of data being managed by the interactive environment

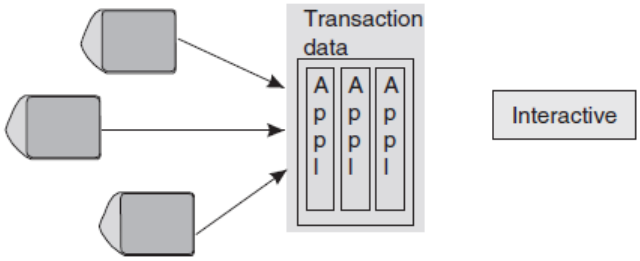


FIGURE 1.3.4 Access to the interactive environment is measured in seconds.

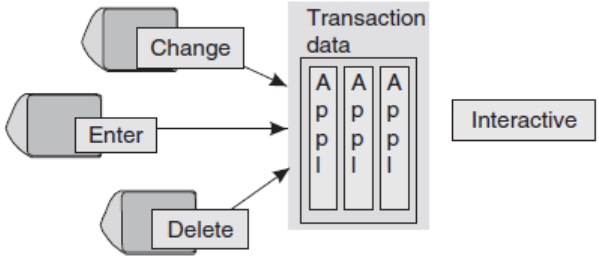


FIGURE 1.3.5 Transactions can update data in the Interactive Sector

One feature of the interactive environment is that because data can be updated, the value that is given to a query is relative to the moment in time the query is made. In other words, if a query is executed at one moment in time and the same query is performed at a later point in time the two queries will produce different answers.

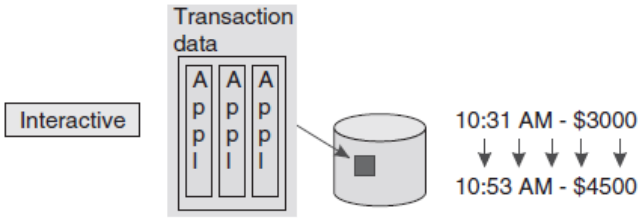


FIGURE 1.3.6 Because data can be updated in the interactive environment, any value that has been accessed is accurate only as of the moment of access

Figure 3.6 shows a query done at 10:31 AM yielding a value of $3000. The same query done at 10:53 AM returns a result of $4500. In between the queries, someone had added money to the account. If data enters the Interactive Sector from an exterior application, the data flows through an ETL layer. In other words, if an application exists outside of the Interactive Sector and if it is desired to have data from that application enter the Interactive Sector, the data must be processed by an ETL tool.

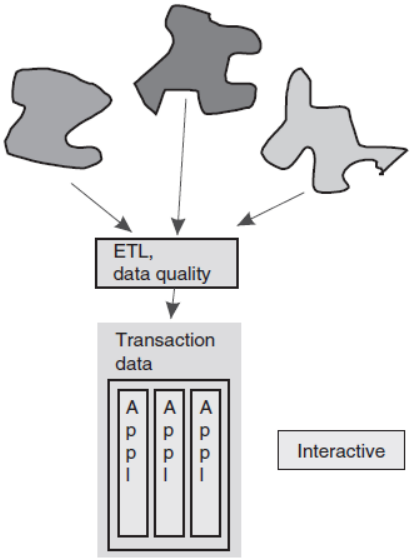


FIGURE 1.3.7 When data enters the Interactive Sector, the data may or may not be transformed.

The data normally enters via passage through an ETL layer Figure 1.3.7 illustrates this circumstance. It is possible to put data into the Interactive Sector without having to integrate the data separately using an ETL tool. In this case the data is integrated as it passes into the Integrated Sector. Data inside the Interactive Sector may or may not have referential constraints placed on it. The choice to use or not to use referential integrity depends entirely on the application that is being run.

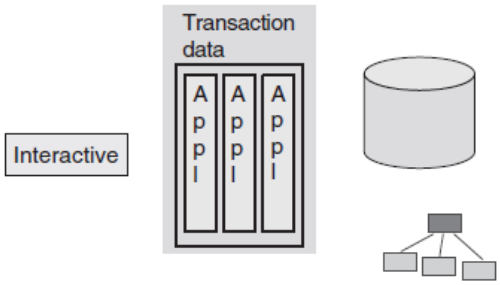


FIGURE 1.3.8 Referential integrity can be enforced at the intra-sector level in the Interactive Sector

Figure 1.3.8 suggests that the Interactive Sector may or may not include optional referential integrity. The pattern of access for data found in the Interactive Sector is that access is fast in sub seconds. When interactive data is accessed, there is a desire to access only a few records of data per access. And the pattern of access is random. One interactive data access will go to one location and another access will go to an entirely different and random location. Because of this pattern of access of data inside the Interactive Sector, disk storage is ideal. Figure 1.3.9 depicts the general pattern of rapid, random data access in the Interactive Sector

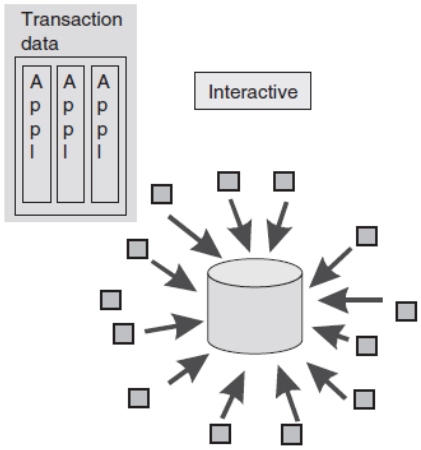


FIGURE 1.3.9 The pattern of access: (a) fast, (b) random, and (c) for small amounts of data.

Only small amounts of historical data are found inside the Interactive Sector. Typical interactive historical data is a day old or even just an hour old. It is really unusual to find data as old as a month in the Interactive Sector. Data passes into the Integrated Sector before it gets to be old.

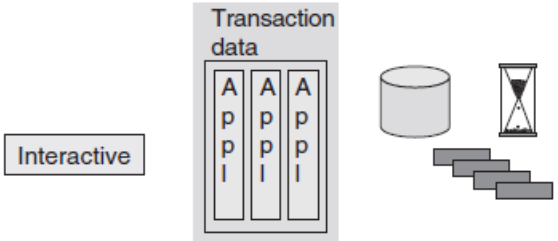


FIGURE 1.3.10 There are only small amounts of historical data to be found in the interactive environment.

Figure 1.3.10 captures the notion that not much historical data is maintained in the Interactive Sector. There is a very uneven level of granularity of data found in the Interactive Sector. Some applications will have a fine degree of granularity, while other applications will aggregate data. There just isn't any consistency of design when it comes to the granularity of application data, as shown in Figure 1.3.11.

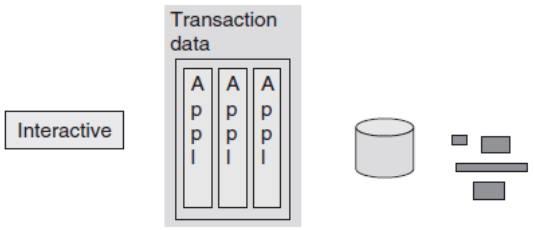


FIGURE 1.3.11 The granularity of the data inside the Interactive Sector varies considerably from application to application.

Data passes from the Interactive Sector to the Integrated Sector. If data has come from outside of the Interactive Sector, it passes to the Integrated Sector. If data comes from the execution of an application residing in the Interactive Sector, then the data is gathered as a by-product of the execution and is passed on to the Integrated Sector. Figure 1.3.12 shows that all data that passes into or through the Interactive Sector eventually moves into the Integrated Sector.

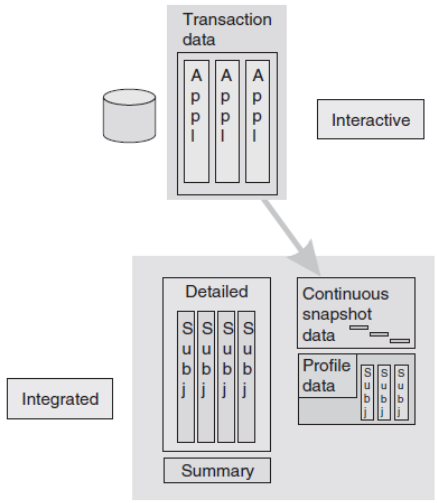


FIGURE 1.3.12 Where data goes after it leaves the Interactive Sector.

1.3.3 Integrated sector

The Integrated Sector is the place where application and transaction data eventually meet and become corporate data. As an example of the differences between application data and corporate data, consider three applications, A, B, and C, that collect revenue information. Application A collects revenue information in euros. Application B collects data in U.S. dollars. And application C collects revenue information in pesos. As this application data passes into the Integrated Sector, the data is all converted into a common currency. And there are *many* conversions that have to be done to transform operational application and interactive transaction data into corporate data.

For example, suppose application A has a month-end closing date of the 31st, application B has a closing date of the calendar end of the month, and application C has a corporate calendar closing date. It is for this reason that the closing reports from the applications will not reconcile to the corporate closing reports. They may have different actual closing dates. These inconsistencies are resolved when all three closing dates are converted to a common closing date as data is passed into the corporate Integrated Sector.

Another type of reconciliation is that of data key reconciliation. Application A has one key structure, application B has another key structure, and application C has yet another key structure. When data from these applications pass into the Integrated Sector, the keys are converted into a common key structure that serves the entire corporation. Here is another example. Application A has a date format of YYMMDD, application B has a date format of MMDDYY, and application C has a date format of YYYYMMDD. When dates are passed into the Integrated Sector, they are all converted into a single common format.

These are just a few examples of the many types of conversion that occur as data passes from the Interactive to the Integrated Sector. The Integrated Sector contains different kinds of structures. The following types of data structures are found in the Integrated Sector:

■ Subject-oriented detailed data—This is where data is organized into major subject areas and where detail is kept. As an example of the way that detail is organized into a subject area, suppose that the particulars of a sales transaction come to the Integrated Sector. From the sale data, the item sold will go to the products sold subject area, the person to whom the item was sold may have some purchase preference data that needs to go in the customer subject area, and the revenue amount will go to the revenue subject area.

■ Small amounts of summary data—The summary data found in the Integrated Sector is summary data that is used in many places and summary data that doesn't change. For example, for a publicly traded corporation, a quarterly statement of revenue, expenses, profitability, and other information goes into a public place (i.e., the Integrated Sector) where anyone that needs this information can have access to it.

■ Continuous time span data—For some data that changes slowly, it is useful to place it in a continuous time span structure. For example, a continuous time span structure can be used to track the name and address of a customer. The customer doesn't move often and the customer's name changes infrequently only when the customer's marital status changes. Therefore, it is possible and reasonable to keep a continuous record of the information about a customer in the Integrated Sector.

■ Profile data—Profile data is data collected from a wide variety of sources about a single subject. A customer record is a simple example of a profile record. A profile record for a customer is kept to track such things as the customer's demographics, the last items the customer bought, the times of day when the customer is active, the locations the customer is active in, and so forth. Looking at the profile record provides a lot of information about the customer in a thumbnail sketch. The data in the Integrated Sector is collected after the interactive data is passed through a layer of ETL processing. It is at the moment of ETL processing that data quality processing is also done. Some simple examples of data quality processing that are done here are domain checking and range checking. An example of domain

checking is ensuring gender codes such as M (male), F (female), or U (unknown). The logic of range checking might read, if age is greater than 150, flag it as likely an error.
Once the data is collected, integrated, and passed through a data quality editor, it is then entered into the Integrated Sector, as depicted in Figure 1.3.13.

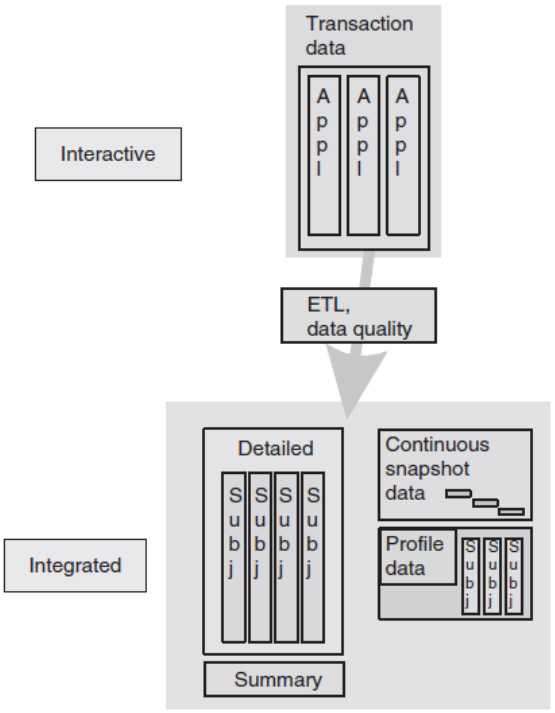


FIGURE 1.3.13 How data enters the Integrated Sector

The workload passing through the Integrated Sector is very mixed, as illustrated by Figure 1.3.14. The workload shown in Figure 1.3.14 is like a highway on which there is Porsches, Ferraris, and lots of semitrailers all sharing the same roadway. The speed of a vehicle on this highway depends on the vehicle in front of it. It doesn't matter that a Porsche can go 185 mph if it is stuck behind a semi going 25 mph. Contrast this workload with the workload of the Interactive Sector and it becomes clear that there is a world of difference between the two highways.

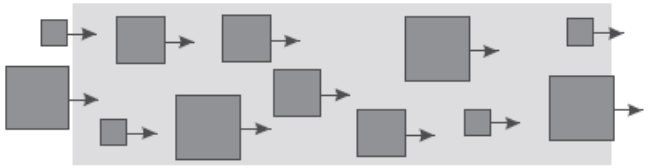


FIGURE 1.3.14 What the workload going on inside the Integrated Sector looks like.

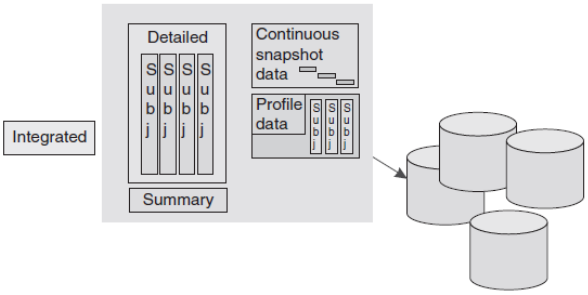


FIGURE 1.3.15 There is a fairly significant amount of data being managed by the integrated environment.

There is a good reason for the mixed workload of the Integrated Sector. Some people have to access a lot of data and other people want only a small amount of data. It is as simple as that. But all of them want it from the Integrated Sector, so there is a very mixed workload. The Integrated Sector usually involves a significant amount of data, as depicted by Figure 1.3.15. There are some good reasons for the large amount of data found in the Integrated Sector, including:

■ The data is granular: There are a lot of atomic units of data to be collected and managed.
■ The data is historical: There is often from 3 to 5 years worth of data.
■ The data comes from a wide variety of sources.

Combine these three factors and the result is a lot of data in the Integrated Sector. Corresponding to the mixed workload is the response time that is expected in the Integrated Sector, where the response time varies anywhere from 10 seconds to much longer. The reason for the variance is the mixed workload. When large data retrieval jobs are being executed against the Integrated Sector, small data queries against the sector are likely to be held up and delayed. On the other hand, users who access the Integrated Sector when no one else is using it can expect good response time.

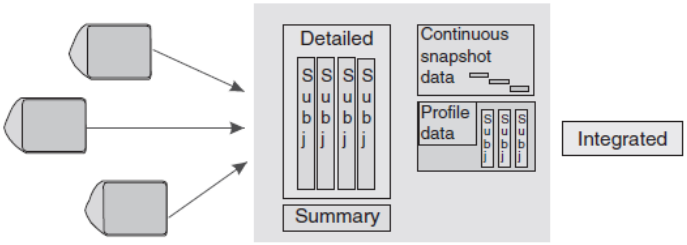


FIGURE 1.3.16 Access to the integrated environment is measured anywhere from 10 seconds to 2 hours or more.

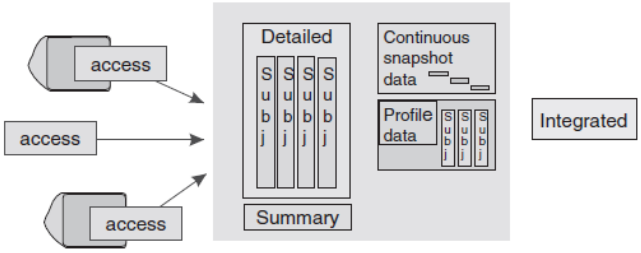


FIGURE 1.3.17 Transactions can only access data in the Integrated Sector

One way that people try to regulate response time is not to let the semis on the highway during rush hours let only the Porsches and Ferraris on the road during the hours of 8:00 AM to 3:30 PM. While this works well, it penalizes the semitrailer drivers who have a legitimate need to get on the highway. In other words, the response time for small jobs that need to use Integrated Sector data can be improved during peak usage hours by confining large queries against the sector to off hours, a less than desirable situation for large query users.

So there are ways around the mixed workload found on the Integrated Sector highway. Figure 3.16 depicts the response time expectations that accompany the Integrated Sector. The transactions that access data in the Integrated Sector can only read data. Unlike the Interactive Sector, where data can be added, deleted, or altered, the data in the Integrated Sector can only be accessed, not updated.

Figure 1.3.17 illustrates data access from the Integrated Sector. Because data cannot be added, deleted, or changed in the Integrated Sector does not mean that changes to data cannot be accommodated. Instead changes to data are accomplished in a different manner. Changes to data in the Integrated Sector are done by creating a new record whenever a change is made to the data. When a bank account changes, a new record is created. When a person's address changes, a new record is created. When a person's insurance coverage changes, a new record is created, and so forth. By creating a new record every time a change is made, a historical track record of the data's evolution is maintained. In addition, once a value is correctly placed in the Integrated Sector, it is never altered. The record may be sent to near-line storage or to archival storage, but after the record is correctly created, it is never changed. This means of dealing with change is very different from that found in the Interactive Sector, where changes to the same records are made all the time.

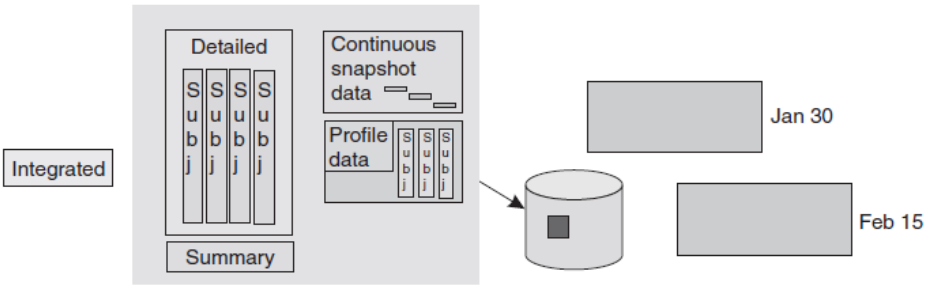


FIGURE 1.3.18 When data values change in the Integrated Sector, a record of the change is created and entered. Thus a historical record of change is kept.

Figure 1.3.18 illustrates how changes are recorded in the Integrated Sector. There is an interesting effect of tracking historical changes to data as it is done in the Integrated Sector. Namely, once a question is asked, it will always get the same answer, even if it is asked at a later point in time. To illustrate this constancy of data, suppose that at 10:31 AM a banker wants to know what the bank's year-to-date revenues are as of 10:31 AM this morning. The answer retrieved from the Integrated Sector comes back as $3000. Now, for whatever reason, the banker asks the same question at 10:53 AM. Although the bank's revenues have changed since 10:31 in the bank's operational and interactive applications, the answer to the banker's query against the Integrated Sector of the data warehouse has not. The banker will get back the same answer $3000.

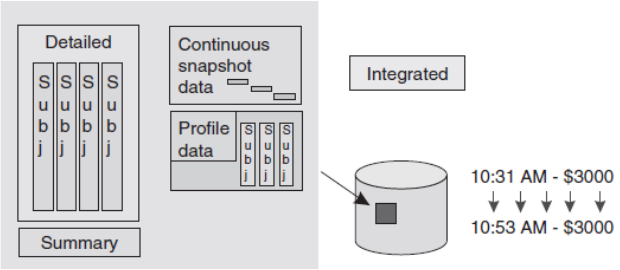


FIGURE 1.3.19 Because data cannot be updated in the integrated environment, any value that has been accessed is accurate at any time.

Figure 3.19 illustrates the constancy of data that is characteristic of processing in the Integrated Sector.

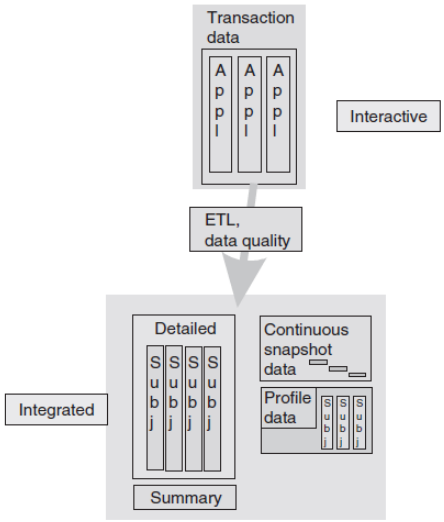


FIGURE 1.3.20 When data enters the Integrated Sector, it is always integrated after passing through ETL processing.

Figure 1.3.20 emphasizes that there are no exceptions to the integration of data as it enters the Integrated Sector. Data entry to the Integrated Sector is necessarily a one-way street and a highly controlled route.

There are two forms of referential integrity that are relevant to and are found in the Integrated Sector. The first type is inter-sector referential integrity. Inter-sector referential integrity refers to the integrity of data as it passes across sectors. In other words, if data passes from the Interactive Sector to the Integrated Sector, the data will recognizably have its source and its target accounted for and no data will be missing. There can be no entry of data in the Integrated Sector that does not have a corresponding entry in the Interactive Sector and vice versa there can be no entry of data in the Interactive Sector that does not have a corresponding entry in the Integrated Sector.

However, just because there are corresponding entries in each sector does not necessarily mean that the values of the entries are the same. One entry may be in euros and the other entry may be in U.S. dollars.

Just because two data elements do not have the same value does not mean that they are not corresponding entries. The second kind of referential integrity that can be found in the Integrated Sector is intra-sector referential integrity. Intra-sector referential integrity means that there may be a relationship between data elements within the same sector. Both types of referential integrity are possible in the Integrated Sector, as depicted by Figure 1.3.21. In comparison to the Interactive Sector, the pattern of access for data in the Integrated Sector is one in which fewer calls for data are made, but the calls typically request more data. Figure 1.3.22 illustrates the general pattern of access that is expected for the Integrated Sector.

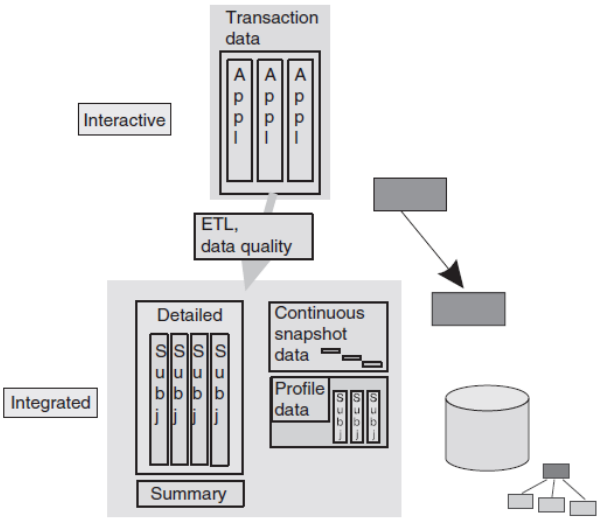


FIGURE 1.3.21 Referential integrity can be enforced at the inter-sector level or the intra-sector level.

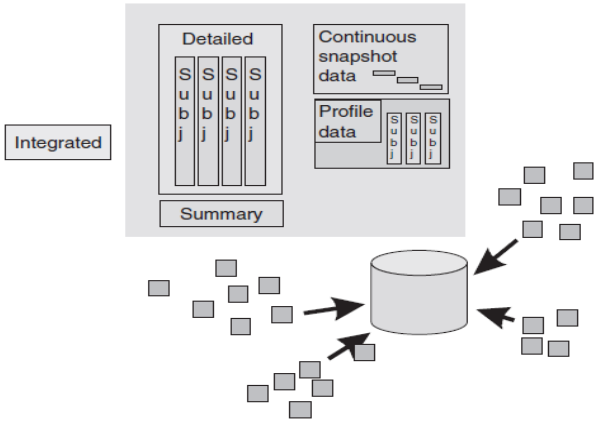


FIGURE 1.3.22 The pattern of access: (a) relaxed speed, (b) sequentially random, (c) for large amounts of data.

This pattern of data access is expected with a mixed workload of small-to-large data retrieval requests. Another difference between the Interactive Sector and the Integrated Sector has to do with the amount of historical data that is found in the different environments. In the integrated environment there is a significant amount of historical data. It is normal to find from 3 to 5 years' worth of data in the Integrated Sector. In contrast, it is rare to find more than 30 days' worth of data in the Interactive Sector.

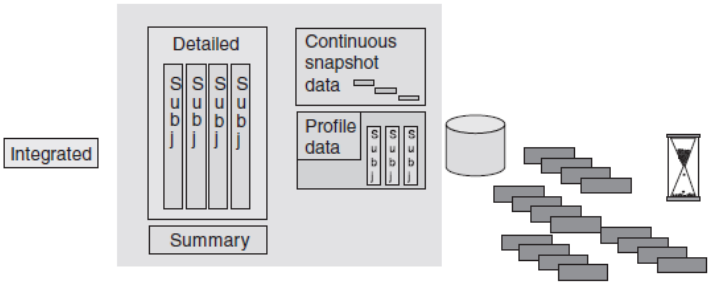


FIGURE 1.3.23 There are fairly large amounts of historical data to be found in the integrated environment.

Figure 1.3.23 illustrates the large amount of historical data that is found in the Integrated Sector. Data granularity is another major difference between the Interactive Sector and the Integrated Sector. In the Interactive Sector, there is mixed granularity. Some data is granular and some data is not. The Integrated Sector, on the other hand, contains the lowest level of granularity of data found in the corporation. It is mandatory that the data in the Integrated Sector be at the very finest and atomic that there is. The Integrated Sector will support many forms of DSS (Decision Support Systems) processing, and each of the DSS processes will have its own requirements for data. Therefore, the lower the level of granularity at the integrated level, the more forms of DSS processing are supported. Stated differently, when the level of granularity of data is raised in the Integrated Sector, fewer forms of subsequent DSS processing can be supported.

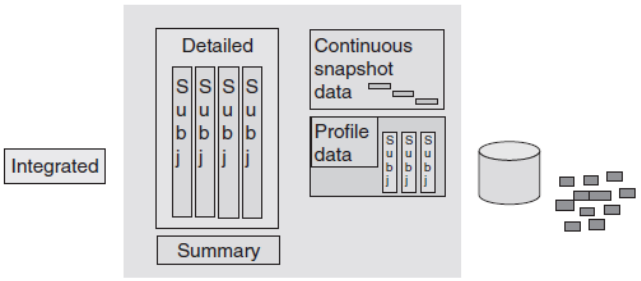


FIGURE 1.3.24 The granularity of the data inside the Integrated Sector is as low as it gets anywhere in the information systems environment

Figure 1.3.24 depicts the need for low levels of granularity at the Integrated Sector. Once data leaves the Integrated Sector, it can go either to the Near Line Sector or to the Archival Sector (Figure 1.3.25). Data goes to the Near Line Sector when there is a lot of data and there is a need for caching of data. Data goes to the archival environment when the probability of access of the data drops significantly. Usually, but not always, this means that as data ages it is sent to the archival environment.

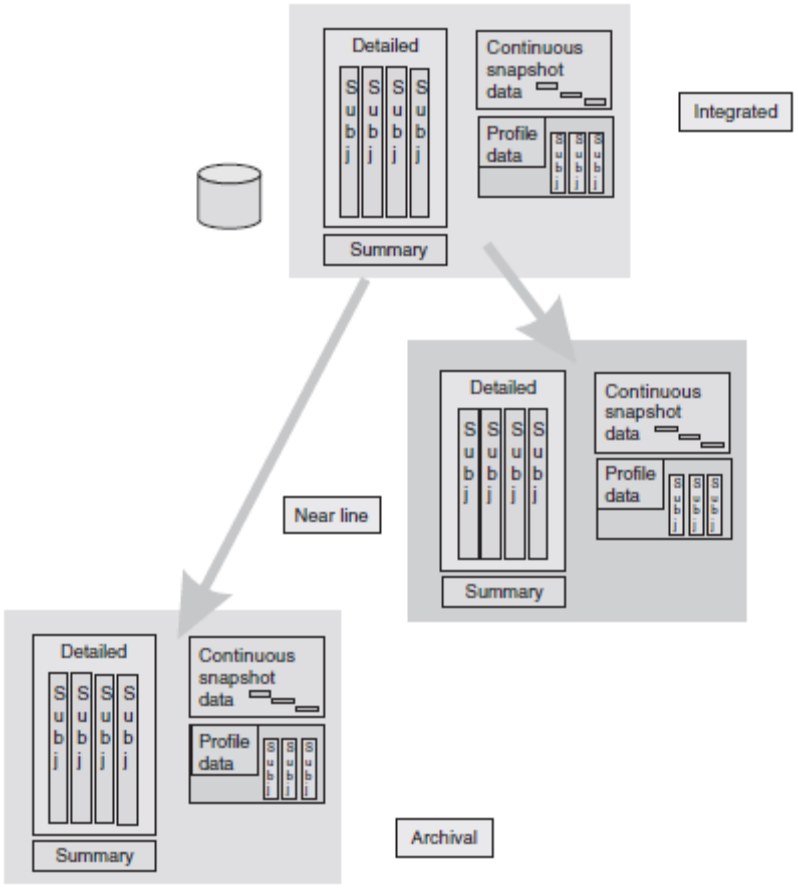


FIGURE 3.25 Where data goes after it leaves the Integrated Sector

1.3.4 Near Line sector

The Near Line Sector is a form of caching for the Integrated Sector. The Near Line Sector may or may not be used for caching—it depends entirely upon the nature of the data warehouse. When a data warehouse's integrated Sector is very large, easing the burden on the Integrated Sector by inclusion of a Near Line Sector makes sense. When there is not all that much data in the Integrated Sector, then using the Near Line Sector probably is not necessary.

There are two reasons for using the Near Line Sector—cost and performance. Near-line storage costs significantly less than disk storage. So if the hardware costs of disk storage for the Integrated Sector are prohibitive, then much of the data in the Integrated Sector can be downloaded into the Near Line Sector for tremendous cost savings.

Performance is enhanced by downloading data with a low probability of access to the Near Line Sector. Because only data with a low probability of access is sent to the Near Line Sector, the data remaining in disk storage in the Integrated Sector is freed from the overhead of "bumping into" large amounts of data that is not going to be used. Near-line storage is sequential data storage on cartridges that are managed robotically. Near-line storage is used for storing large quantities of data very inexpensively. Data is still electronically available when it is maintained on near-line storage, but the cost of storage is significantly reduced compared to the cost of storing all data in the Integrated Sector on disk storage.

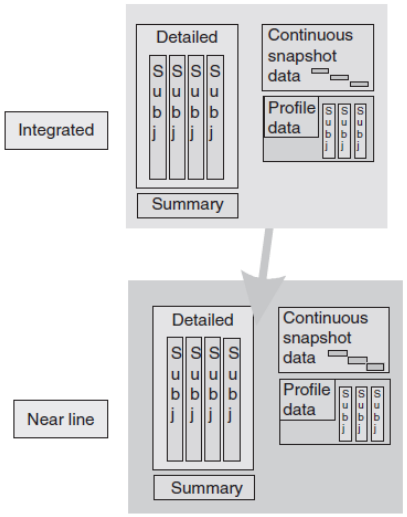


FIGURE 1.3.26 The source of near-line data is the Integrated Sector.

Figure 1.3.26 depicts Near Line Sector data arriving from the Integrated Sector. Once data has been placed in near-line storage, it is subjected to a workload like any other environment. The typical workload for near-line storage does not need much data access activity. The reason there is a diminished need for access to the near-line data is that only data with a low probability of access has been placed in near-line storage. In fact, if there is frequent access of any particular type of near-line data, then that data needs to be moved back up into the Integrated Sector. On the relatively rare occasion when there is a need for access to near-line data, the access is either for a string of records or for a single record. Figure 1.3.27 depicts the typical workload associated with near line storage.

FIGURE 1.3.27 What the workload going on inside the Near Line Sector looks like.

One of the burning issues is how data moves from the Integrated Sector to the Near Line Sector. There are basically two ways that data can be managed when it comes to its placement manually or in an automated manner.

When data is managed manually, there is an administrator who monitors the usage of the data in the Integrated Sector or receives a request for movement of data. The administrator then moves the data either to or from the Integrated Sector to the Near Line Sector.

The other alternative for the management of the movement of the data is through a CMSM a cross-media storage manager. The CMSM sits between the Integrated Sector and the Near Line Sector. The CMSM automatically manages the placement of the data from one environment to the next.

The CMSM can operate in a mode of transparency. A transparent CMSM examines requests coming into the system and sees if any of the requests are looking for data that is managed in near-line storage. When a request for data that is being managed in near-line storage arrives, the CMSM queues the transaction, goes and finds the data in near-line storage that has been requested, and loads the data onto disk storage. Then the transaction is de-queued and executed. Upon going into execution, the transaction finds all the data it needs residing in disk storage where the CMSM put it. There are other ways that the CMSM can operate. What has just been described is only one of several modes of CMSM operation.

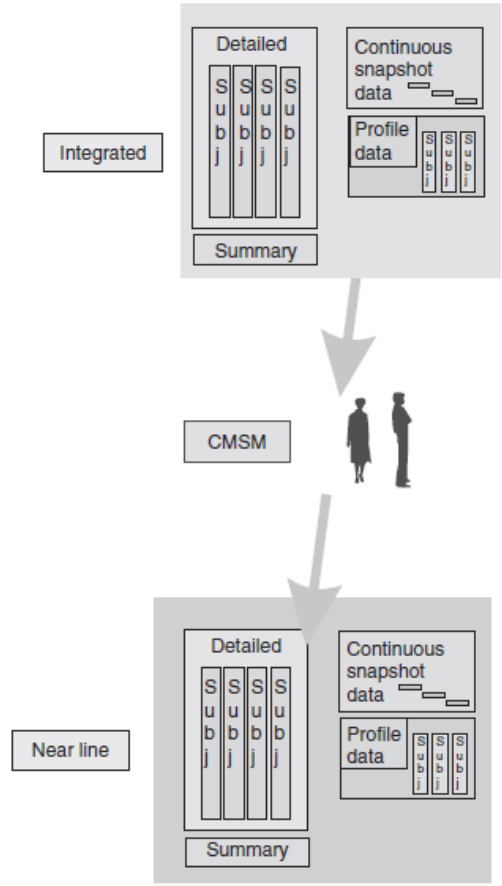


FIGURE 1.3.28 There are essentially two ways to manage the flow of data between the Integrated Sector and the Near Line Sector.

Figure 1.3.28 shows the interface between the Integrated Sector and the Near Line Sector. As a rule, the data that resides in the Near Line Sector is a mirror image of the structure and format of the data that resides in the Integrated Sector. The design of the data, the DBMS, and the release of the DBMS are all identical to their Integrated Sector counterparts. There is a good reason for this extreme similarity of data from the integrated and near-line environments. That reason is that data needs to be effectively interchangeable between the two environments. It is certainly obvious that data moves from the integrated environment to the near-line environment. What may not be so obvious is that data also moves from the near-line environment back to the integrated environment. Data moves from the near-line environment to the integrated environment whenever the probability of the data being accessed arises. Therefore, it becomes easy to move data from the near-line environment to the integrated environment when the data is identical in format, structure, and technology. Let any one of

these factors slip and it becomes much harder to move near-line data back into the integrated environment.

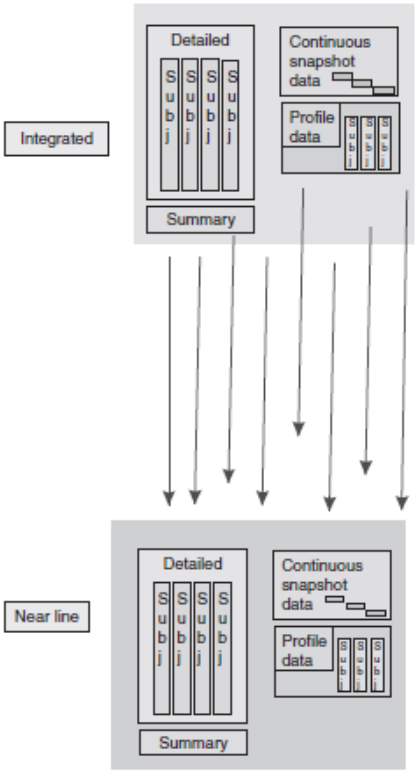


FIGURE 1.3.29 Structurally, the near-line environment is a carbon copy of the integrated environment.

Figure 1.3.29 shows that the data is identical in structure, format, and technology in the integrated and near-line environments. One of the main benefits of the near-line environment is that it can manage *huge* amounts of data, far more than can be managed gracefully in the interactive or integrated environments. It is possible to manage hundreds of terabytes of data in the near-line environment.

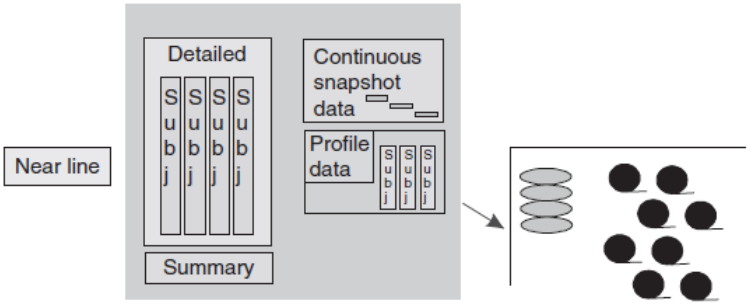


FIGURE 1.3.30 There can be a very large amount of data being managed by the near-line environment.

Figure 1.3.30 depicts the ability of the near-line environment to manage huge amounts of data. Upon leaving the Near Line Sector, data normally moves into the Archival Sector. Note that

the Archival Sector may be fed data directly from the Integrated Sector without passing through the Near Line Sector. However, if the data has been moved into the Near Line Sector, then it is normally moved from there to the Archival Sector. The movement of data to the Archival Sector is made when the probability of accessing the data drops significantly.

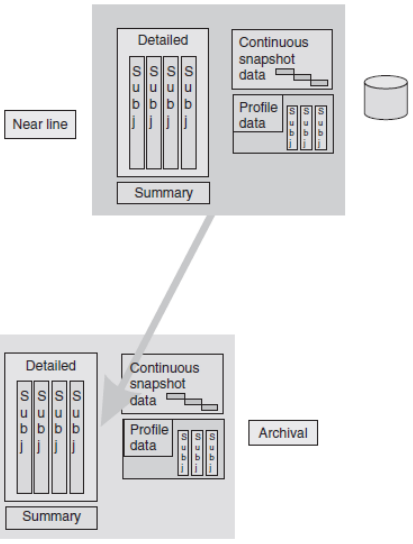


FIGURE 1.3.31 Where data goes after it leaves the Near Line Sector

Figure 1.3.31 illustrates the movement of data from the Near Line Sector to the Archival Sector.

1.3.5 Archival sector

The Archival Sector is the place where data goes when the probability of it being accessed becomes very small. In some cases, data is placed in the Archival Sector not due to any probability of access, but because of legislative reasons. In some cases the government mandates the storage of data for long periods of time.

Putting data in archives used to be a one-way street. Data that went into archives became inaccessible. Often when a person went to open a magnetic tape that had been assigned to archival processing, oxide dust fell out of the plastic container the moment it was opened. This of course meant that the magnetic tape had decomposed, which in turn meant that the tape was now useless because it could not be read.

Today when data is placed in archival storage, the data must be readable at a later point in time otherwise the archival environment is a waste of time and money. In many ways building an archival environment is like building a time capsule. People put all sorts of things in a time capsule, not knowing when the capsule will be opened or by whom. The same analogy is true for the data stored in the archival environment. When an archival environment is populated with data, the future needs of the data's users are unknown and the moment in time when the data will be needed is also unknown.

Therefore, the archival environment needs to organize data so that it is in "time capsules" that are completely self-contained.

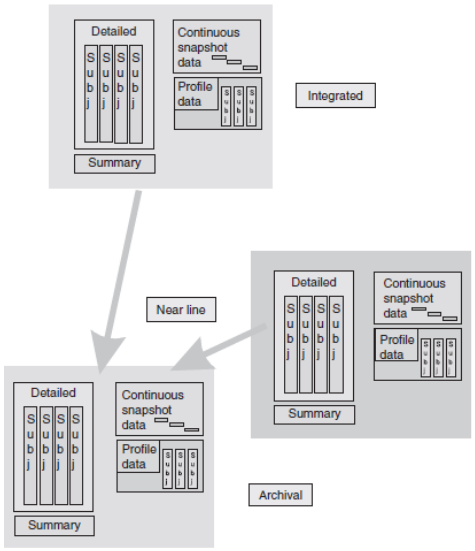


FIGURE 1.3.32 Archival data can come from either the Integrated Sector or the Near Line Sector.

Figure 1.3.32 recaps how data enters the archival environment from either the integrated environment or the near-line environment. The workload associated with the archival environment is very unusual. For long periods of time months or even years there is often no access of archival data whatsoever. Then one day there is a need for the data—either a few records or a whole long sequential string of data.

The biggest problem with archival data is usually finding the required data. Whereas in other environments the emphasis is on sub second response time, in the case of archival data the issue is that of even being able to find whatever data you are looking for. Usually there is so much archival data, and the basis for searching the data is so obscure, that finding data in the archival environment is like looking for the proverbial needle in a haystack.

FIGURE 1.3.33 What the workload going inside the Archival Sector looks like.

Figure 1.3.33 depicts the workload associated with the archival environment. The volumes of data found in the Archival Sector can be tremendous. Over time, it is expected that there will be more data in the Archival Sector than anywhere else. In the early days of a data warehouse's life cycle there usually is very little data in its archives. But over time, as the data warehouse matures, its archives can accumulate and grow to contain utterly huge amounts of data.

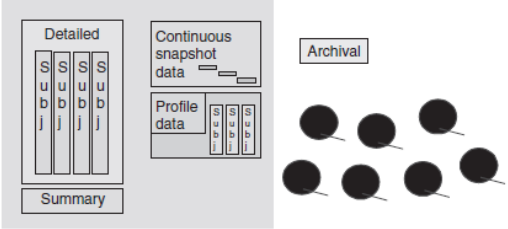


FIGURE 1.3.34 There can be a huge amount of data being managed by the archival environment.

Figure 1.3.34 shows the large amount of data that can be found in the Archival Sector. Access to the archival environment in terms of response time is measured in units of time not found elsewhere in the DW 2.0 architecture. It is normal to expect for it to take days or even weeks to find data in the archival environment. This of course depends on the volume of data in the archival environment, how well the data is indexed, the data search criteria, and the technology used to store the archival data. In some cases a search may be completed rather
quickly but general expectations for retrieving data from the archival environment had better not be set too high.

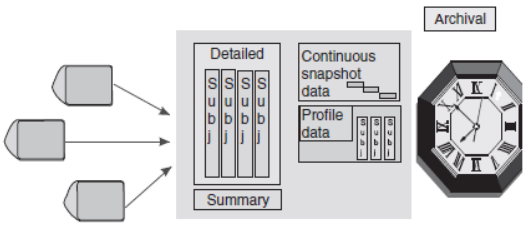


FIGURE 1.3.35 Access to the archival environment is measured in hours or even days.

Figure 1.3.35 illustrates the search time expectations for an archival environment. On occasion, upon completing a search, data can be moved from the archival environment to the Integrated Sector. This kind of archival data restoration is indicated when there is reason to suspect that the data will be needed for lots of analysis and access. Going to the archival environment is a rather painful experience in most environments. By moving archival data that is likely to be used frequently back to the Integrated Sector, the often-painful experience of having to go back to the archival environment to retrieve it again can be mitigated.

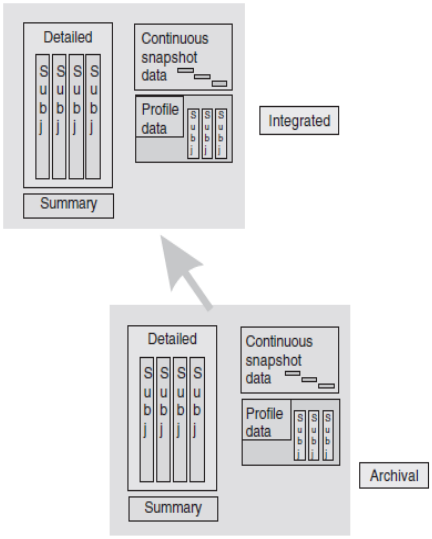


FIGURE 1.3.36 On occasion data can be moved from the archival environment to the integrated environment.

Figure 1.3.36 shows that data can be restored to the Integrated Sector from the Archival Sector. One of the most useful things that can be done with archival data is the creation of passive indexes. Usually the archival environment just sits there. Occasionally someone loads more data into the archival environment. But for the most part, the archival environment is essentially void of activity.

Then along comes a day when someone wants something from the archival environment. Suddenly there is a big rush to push a lot of data through the system. In the archival environment, there are typically long periods of inactivity followed by a few short periods of frantic voluminous activity. And when it comes to finding data in the archival environment, the search is always questionable. These are the conditions that commend the creation and use of passive indexes.

Under normal circumstances, indexes are created for the purpose of rapid access into a data base using known requirements. But in the case of archival data, there are few predictable paths of access. Therefore, while the data in the Archival Sector is just sitting there, creating indexes based on probable paths of access is a good use of time. Once several passive indexes are created, when it comes time to search the Archival Sector, with any luck the data that is being sought can be easily and quickly found. Only in the case in which data is being sought where a passive index has not been built will it be necessary to do a full sequential search of the data.

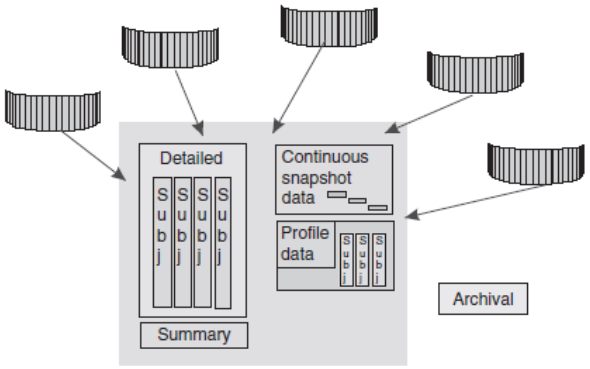


FIGURE 1.3.37 Archival data sits there and is not often used. It is a good idea to use this idle time to create what is termed a "passive index."

Figure 1.3.37 shows the creation of passive indexes on likely paths to and through the Archival Sector. When data is sent to the Archival Sector, it may or may not be appropriate to preserve the structure that the data had in the integrated or near-line environments. There are advantages and disadvantages to both preserving the structure of the data and not preserving the structure of the data. One advantage of preserving the structure of the data as it passes into the Archival Sector is that it is an easy thing to do.

The data is simply read in one format and written out in the same format. That is about as simple as it gets. But there are some reasons this approach may not be optimal. One reason is that once the data becomes archived, it may not be used the same way it was in the integrated environment. The format and structure of data in the Integrated Sector may not be appropriate at all for data in the Archival Sector. In addition, data in the Integrated Sector is usually compatible with a particular release of the software that uses it. By the time archival data is retrieved from the archival environment, it is likely that the release of the software that will use it is long gone. Therefore, it may not make sense to structure the data in the Archival Sector the same way the data was structured in the Integrated Sector.

The data can and often should be broken into finer pieces as it is placed in the Archival Sector. Then, when it is searched or when it is accessed it is simpler to query. Additionally, there is no reason the data cannot be placed in the archival environment in two formats in its original structure from the Integrated Sector and in a simpler, more universal format.

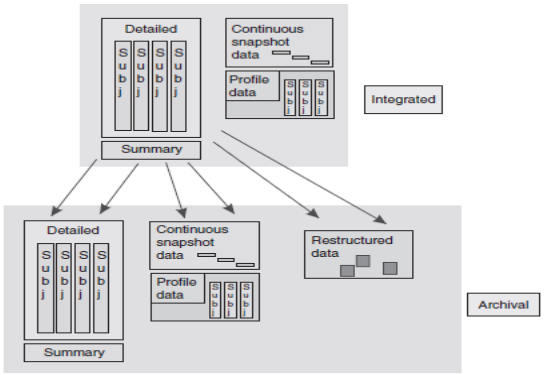


FIGURE 1.3.38 When data is sent to the archive environment, the structure of the data may be preserved, the data may be restructured, or both.

Figure 3.38 illustrates the dual structuring of data in the Archival Sector. Data placed in the Archival Sector needs to be as free as possible from software release constraints and conventions.

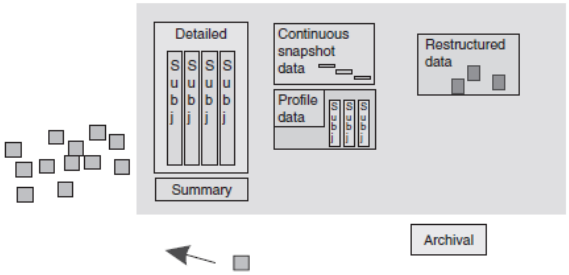


FIGURE 1.3.39 The pattern of access for the archival environment is: (a) very infrequent access; (b) when access occurs, most of it is for lots of sequential data; (c) a few accesses are made for specific records.

There is a predictable pattern of access for data found in the Archival Sector, as depicted by Figure 1.3.39. Archival Sector data is accessed very infrequently, and when it is accessed, it is usually the case that entire sets of archival data must be accessed. Only rarely will a single record in the archival environment need to be retrieved.

One of the interesting aspects of accessing data in the archival environment is that oftentimes the data will need to be located based on obscure fields or data values. Occasionally there is a need to access archival data by one "standard" key and identifier values. But, often there needs to be an access and qualification of data based on very non-normal types of data.

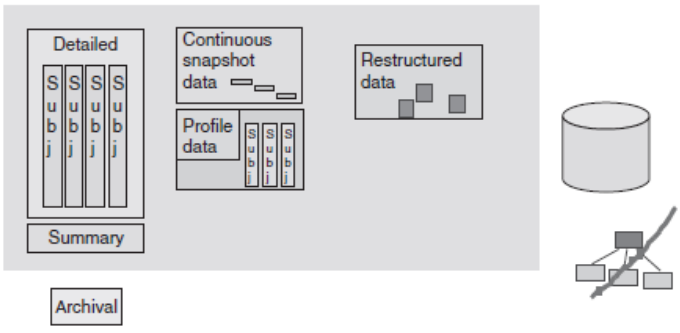


FIGURE 1.3.40 Normally referential integrity is not enforced at the Archival Sector.

Due to the volume of archival data and the fact that archival data is retained for long periods of time, referential integrity constraints are not a part of the Archival Sector, as illustrated in Figure 1.3.40.

The Archival Sector is often searched when the goal is to find *any* relevant data and move it to the Integrated Sector or an exploration facility. But there are occasions when it makes sense to search the Archival Sector by itself. In other words, the Archival Sector can be used as a basis

for decision making. There are, however, some major drawbacks to this approach, including but not limited to the following:

■ There is really a lot of data in the Archival Sector.
■ The data in the Archival Sector needs to be searched sequentially.
■ There are no indexes that are useful for the searches that need to be done.

Further, the data query and analysis technology that is available for the Archival Sector is limited, compared to what is available in other arenas. However, on occasion the Archival Sector can be accessed and analyzed in a stand-alone fashion. Figure 1.3.41 shows this capability.

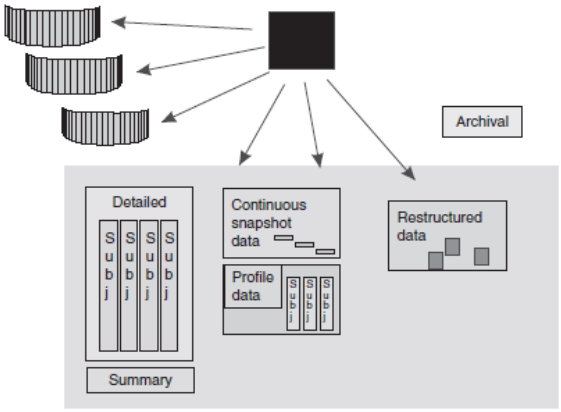


FIGURE 1.3.41 It is always possible to search sequentially and process the archival environment by itself.

1.3.6 LET US SUM UP

The place where data enters the DW 2.0 environment under normal circumstances is the Interactive Sector. Data can enter through ETL or directly. The Interactive Sector is application oriented, the sector in which update to data can be done, and the environment that supports 2- to 3-second response time. The workload passing through the Interactive Sector is small and fast. No large transactions are allowed to pass through the Interactive Sector.

The pattern of data access in the Interactive Sector is random, fast, and small, in terms of the amount of data accessed. There is limited historical data in the Interactive Sector. Integrated Sector data is integrated before the data is allowed to enter the environment. The integration is normally done by an ETL tool. The integration represents a change in the state of the data. While data from the Interactive Sector is application-oriented data, that in the Integrated Sector is corporate data.

The workload of data passing into and out of the integrated environment is mixed, including both small and large transactions. The response time in the Integrated Sector is also mixed, varying from seconds to hours. There is usually a large volume of data in the Integrated Sector. The Integrated Sector usually contains 3 to 5 years' worth of data. No data updating occurs in the Integrated Sector. Snapshots of data are taken and are inserted into the data base when a change of data is needed. A historical track record of the data is created. The pattern of access

is infrequent, and large volumes of data are accessed, often en masse. When data leaves the Integrated Sector, the data goes either to the Near Line Sector or to the Archival Sector.

The Near Line Sector acts like a cache for the Integrated Sector. The Near Line Sector operates on non-disk-storage-based technology. The Near Line Sector contains data that is a mirror image of the data found in the Integrated Sector. The Near Line Sector is linked to the Integrated Sector either manually or by means of a CMSM—a cross-media storage manager.

The workload on the Near Line Sector is infrequent loading of data. But when data is loaded, it is loaded en masse. Data is placed in the Near Line Sector when the probability of access drops. The Archival Sector is where data goes when the probability of access diminishes significantly. The Archival Sector contains packages of data that are self-contained. These self-contained packages are like time capsules, to be opened up at a later and unspecified moment in time. The creation of passive indexes is a good idea for archival data. There normally is a lot of data in the archival environment. It is conceivable that there could be hundreds of years of data in the archival environment. To be useful, archival data must be free from software release and product considerations, because it is unlikely that the same releases and products will be available when the data needs to be used.

Unstructured data must first be integrated before it can be useful for textual analytics. Unstructured data passes through an ETL layer before entering the unstructured DW 2.0 environment. There are usually large volumes of data found in the unstructured environment. There may not be an Archival Sector for unstructured data. There probably is no Near Line Sector for unstructured data.

1.3.7 List of References

- DW2.0 The architecture for Next Generation of Datawarehousing W.H. Inmon, Derek Strauss, Genia Neushloss, ELSEVIER.
- Building the data warehouse, W.H.Inmon, third Edition, Wiley.
- Datawarehousing, S. Mohanty, TMH .
- The Data Warehouse Lifecycle toolkit", Ralph Kimball ,John Wiley.

1.3.8 Unit End Exercises

1) List and explain briefly the various data structures found in integrated Sector.
2) Explain unstructured processing of Integrated Sector of DW 2.0
3) Write a short note on interactive sector.

UNIT-II: Metadata in DW

Unit Structure

2.1.0 OBJECTIVES

In this chapter you will learn about:

- What is metadata?
- Active or passive repository.
- Taxonomy-Internal and External.
- Metadata in archival sector.

2.1.1 INTRODUCTION

One of the essential ingredients of the DW 2.0 architecture is metadata. Unlike first-generation data warehouses where metadata either was not present or was an afterthought, metadata is one of the cornerstones of the DW 2.0 data warehouse. There are many reasons metadata is so important. Metadata is important to the developer, who must align his/her efforts with work that has been done previously. It is important to the maintenance technician, who must deal with day-to-day issues of keeping the data warehouse in order. It is perhaps most important to the end user, who needs to find out what the possibilities are for new analysis.

The best way to understand the value of metadata in DW 2.0 is to see it as acting like a card catalog in a large public library. How is information found in the public library? Do people walk in and look for a book by passing from one row of books to the next? Of course people can do that. But it is a colossal waste of time. A much more rational way to locate what you are looking for in a library is to go directly to the card catalog. Compared to manually searching through every book in the library, searching a card catalog is immensely faster.

Once the book that is desired is located in the catalog, the reader can walk directly to where the book is stored. In doing so, vast amounts of time are saved in the location of information. Metadata in DW 2.0 plays essentially the same role as the card catalog in the library. Metadata allows the analyst to look across the organization and to see what analysis has already been done.

2.1.2 Reusability of data analysis

Consider the end user. End users sit on the sidelines and feel the need for information. This need for information may come from a management directive, from a corporate mandate, or simply from the end user's own curiosity. However it comes, the end user ponders how to approach the analysis. Metadata is the logical place to which to turn. Metadata enables the analyst to determine what data is available. Once the analyst has identified the most likely place to start, the analyst can then proceed to access the data. Without metadata, the analyst has a really hard time identifying the possible sources of data. The analyst may be familiar with some sources of data, but it is questionable whether he/she is aware of all of the possibilities. In this case, the existence of metadata may save huge amounts of unnecessary work.

Similarly, the end user can use metadata to determine if an analysis has already been done. Answering a question may be as simple as looking at what someone else has done. Without metadata, the end user analyst will never know what has already been done. For these reasons, then (and plenty more!), metadata is a very important component of the DW 2.0 architecture.

2.1.3 Metadata

Metadata has a special role and implementation in DW 2.0. Separate metadata is required for each sector of DW 2.0. There is metadata for the Interactive Sector, metadata for the Integrated Sector, metadata for the Near Line Sector and metadata for the Archival Sector.
The metadata for the Archival Sector is different from the other metadata, because Archival Sector metadata is placed directly in the archival data. The reason for this is to ensure that over time the metadata is not separated or lost from the base data that it describes.

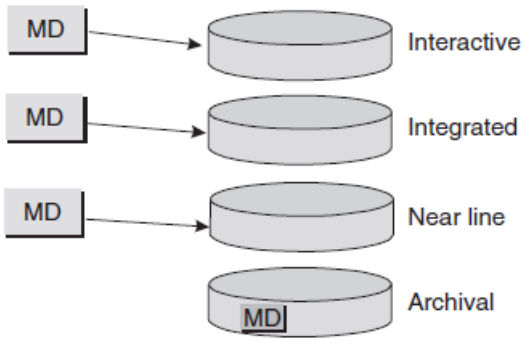


FIGURE 2.1.1 Metadata and the DW 2.0 environment.

Figure 2.1.1 depicts the general location of metadata in the DW 2.0 architecture. There is a general architecture for metadata in DW 2.0. There are actually two parallel metadata structures—one for the unstructured environment and one for the structured environment.

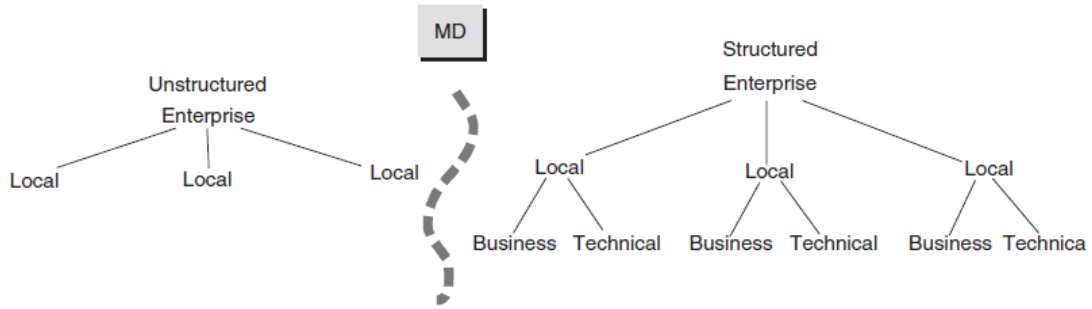


FIGURE 2.1.2 The general structure of metadata.

Figure 2.1.2 shows the high level of structure of DW 2.0 metadata. For unstructured data, there are really two types of metadata—enterprise and local. The enterprise metadata is also referred to as general metadata, and the local metadata is referred to as specific metadata. For structured metadata, there are three levels—enterprise, local, and business or technical. There is an important relationship between these different kinds of metadata. The best place to start to explain these relationships is the local level. Local metadata is a good place to start, because most people have the most familiarity with that type of metadata. Local metadata exists in many places and in many forms. It exists inside ETL processes. It exists inside a DBMS directory. It exists inside a business intelligence universe. Local metadata is metadata that exists inside a tool that is useful for describing the metadata immediate to the tool. For example, ETL metadata is about data sources and targets and the transformations that take place as data is passed from source to target. DBMS directory metadata is typically about tables, attributes, and indexes. Business intelligence (BI) universe metadata is about data used in analytical processing. There are many more forms of local metadata other than these common sources.

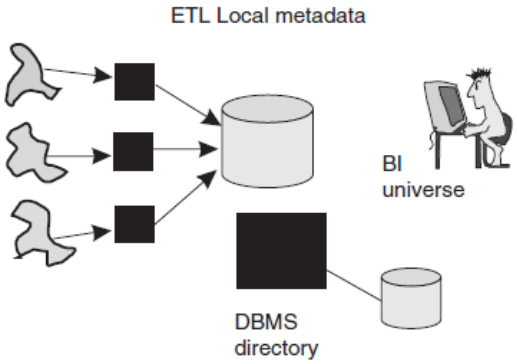
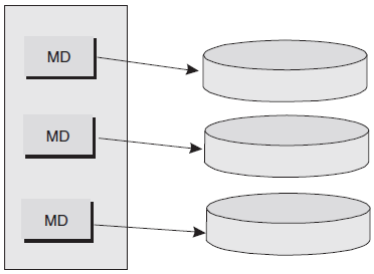


FIGURE 2.1.3 Local metadata already exists in a lot of places.

Figure 2.1.3 depicts examples of local metadata. Local metadata is stored in a tool or technology that is central to the usage of the local metadata. Enterprise metadata, on the other hand, is stored in a locale that is central to all of the tools and all of the processes that exist within the DW 2.0 environment.



■ **FIGURE 2.1.4** A repository is a place where metadata is stored.

Figure 2.1.4 show that enterprise metadata is stored for each sector of the DW 2.0 environment in a repository. In this figure it is seen that sitting above each sector of DW 2.0 is a collection of enterprise metadata, and that all of the enterprise metadata taken together forms a repository. Actually, all of the sectors except the Archival Sector have their metadata stored in a repository.

2.1.4 Active/ passive repository

There are two basic types of metadata repositories—active and passive. An active repository is one in which the metadata interacts in an ongoing manner with the development and query activities of the system. A passive repository is one in which the metadata does not interact in any direct manner with the development and/or the query activities of the end user. Figure 2.1.5 depicts a passive metadata repository.

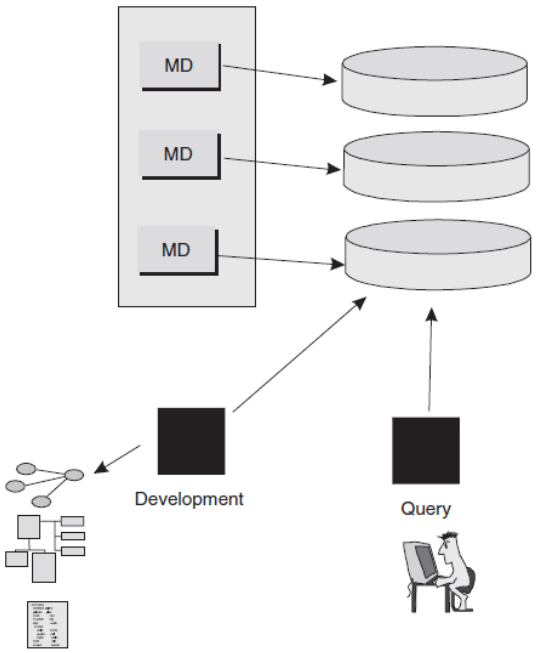


FIGURE 2.1.5 A passive repository

The passive repository is the least desirable option, because end-user and developer activity is independent of the metadata repository. Any work that becomes optional quickly becomes work that is not done at all, because most organizations are compelled to minimize the amount of work, reduce costs, and meet deadlines. The passive metadata repository quickly assumes

the same position as documentation. In the time-honored fashion of developers, documentation simply does not get done. If by some miracle the passive metadata repository does get built, it is soon out of date as changes to the system are not reflected in the passive repository.

The other style of metadata repository is the active repository. The active repository is the place where enterprise metadata is kept when the metadata is actively used during the development process and for data-querying activities. Figure 2.1.6 shows an active metadata repository.

This figure illustrates that the development process and the query process are intimately intertwined with the metadata repository. When a change needs to be made to the data model, when the most current description of the data is needed, or when the source of data is needed, these activities and more rely on the active repository as their foundation.

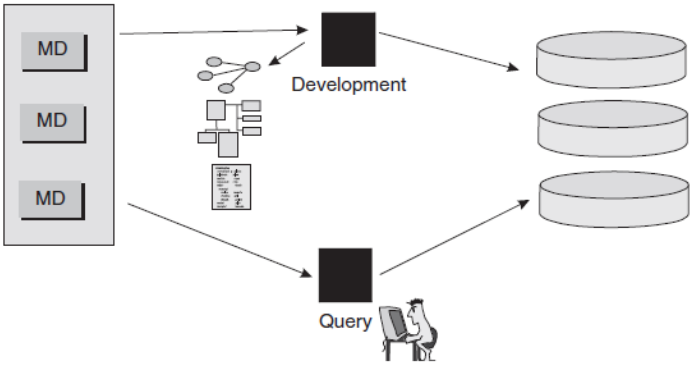


FIGURE 2.1.6 An active repository.

2.1.5 Enterprise metadata

The enterprise active metadata repository is used when a query needs to be made against a table, when the relationship between the local and the global needs to be examined, or when the analyst just wishes to see what the data possibilities are.

Enterprise metadata has many different relationships with local metadata. One such relationship is semantics. In a semantics relationship, the enterprise describes a global term for the corporation. Then the local usage of that term is described, along with a pointer to the local system where the term is found. For example, suppose there is an enterprise term " revenue " and three local systems that refer to that term. In one local system the term is the same— " revenue. " In another local system, the term is " money, " and in another the term is " funds. " These three different words mean the same thing in the example enterprise. Enterprise metadata is an excellent way to ensure that the terms are consistently recognized and interpreted as the synonyms they are in the enterprise's dialect. Metadata provides a way to unite vocabularies at the enterprise level and the local level. This capability is illustrated in Figure 2.1.7.

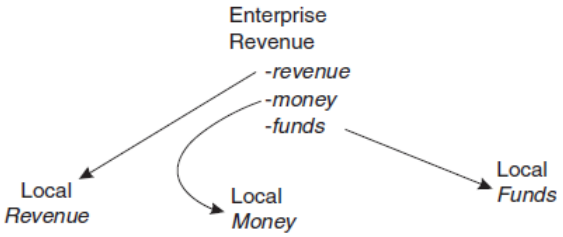


FIGURE 2.1.7 An example of the relationship between enterprise metadata and local metadata.

But semantic relationships are hardly the only type of enterprise/ local relationship that can be found in the metadata of the DW 2.0 environment. Another key type of data relationship commonly found in organizations is that of enterprise subject area definitions.

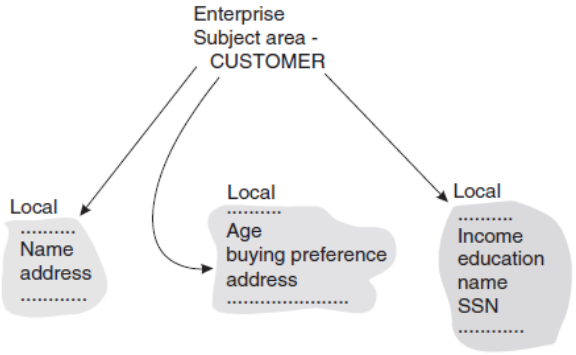


FIGURE 2.1.8 An example of the specification of a major subject area at the enterprise level.

Figure 2.1.8 shows one such definition. This figure shows a major subject area, Customer, defined at the enterprise level. The different places at the local level where information about Customer is found are shown. One local system has named and addresses information about a Customer. Another local system has age and buying preference information about a Customer. Yet another local system maintains Customer income, education, and Social
Security number. The enterprise metadata level can be used to identify where the local systems have supporting data for major business subjects.

2.1.6 Metadata and the system record

Metadata can also be used to define the system of record for the enterprise's data subjects and their data attributes. In the system of record, the definitive source for a unit of data is declared.

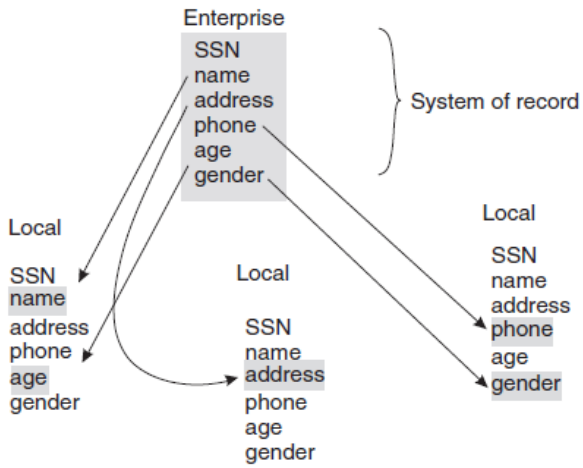


FIGURE 2.1.9 The system of record is identified in the enterprise level of metadata.

Figure 2.1.9 shows that the system of record for each element of data may be identified. It is not unusual for there to be more than one system of record for various data attributes of a major subject of the enterprise. It is noteworthy that in the three preceding examples, there is some overlap between the definitions of data and relations defined between the local level of metadata and the enterprise level of metadata. However, there are some dissimilarities as well. The differences between the relationships that have been described are subtle. This level of subtlety can be represented by enterprise metadata and its relationship with local metadata in the DW 2.0 environment. But there are other metadata relationships that are also important in the DW 2.0 environment. For example, there are two distinct types of metadata in the local level of metadata—business and technical. Business metadata is information about the data that is useful to and found in the jargon of the business person. Technical metadata is information about the data that is useful to and found in the jargon of the technician. An example of business metadata might be the definition of "revenue" as "money or instruments paid for services or products." An example of technical metadata might be the attribute definition "REV-DESIGNATED PIC 9999.99" contained in a table named "ABC. "

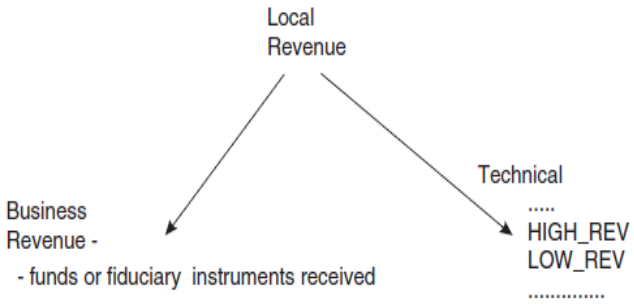


FIGURE 2.1.10 An example of local business metadata and local technical metadata.

Figure 2.1.10 shows that the local level of metadata has its own subdivisions for business and technical metadata. Unstructured data in DW 2.0 has its own metadata. The metadata for the unstructured environment is very different from the metadata for the structured environment. Taxonomies are an example of metadata for the unstructured environment.

2.1.7 Taxonomy

Simply stated, taxonomy is a detailed subdivision of a larger subject. Taxonomy has a detailed breakdown of the components of a given subject. Glossaries and ontologies are related to taxonomies.

Two basic types of taxonomies are found in the unstructured DW 2.0 environment—internal and external. Internal taxonomies are constructed using only the words and phrases found in the text itself. For example, suppose the text in question is a series of contracts. The taxonomies for a group of contracts might well contain major themes, such as contract, terms, length of agreement, and payments. The internal taxonomy is a declaration of the major subject areas found within the text of unstructured data itself. Sometimes internal taxonomies are called " themes " of a text. The other type of taxonomy found in the unstructured DW 2.0 environment is external. An external taxonomy can come from anywhere. An external taxonomy is something that is developed entirely from the "real world." Examples of some external taxonomy include

- Sarbanes Oxley;
- Basel II;
- import/export regulations;
- major league baseball;
- Dewey decimal numbering scheme;
- recipes from Emeril.

There may or may not be a relationship between an external taxonomy and a body of unstructured data. For example, suppose the external taxonomy major league baseball is compared to some unstructured contract text. Unless the contracts are for baseball players, there most likely is little or no overlap between the contracts and major league baseball. In contrast, suppose the body of text is a collection of emails authorizing corporate expenditures and the external taxonomy is Sarbanes Oxley. In this case there would indeed be major overlap between the taxonomy and the body of unstructured data.

2.1.8 Internal and external taxonomy

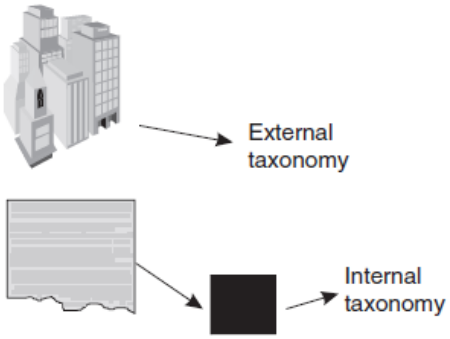


FIGURE 2.1.11 Where different taxonomies come from.

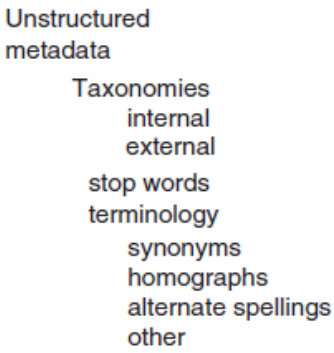


FIGURE 2.1.12 Some forms of unstructured metadata.

Figure 2.1.11 shows that there are two different kinds of taxonomies. There are many different forms of unstructured metadata. Figure 2.1.12 illustrates just a sampling of the different kinds of unstructured metadata. Some other types of metadata that are common to the unstructured environment include:

■ Stop words—words that are used in speech but which are not central to the meaning of the text. Typical stop words include a, and, the, was, that, which, where, to.
■ Synonyms—words that mean the same thing but have different spellings, the *fur* and the *coat* of a cat, for example.
■ Homographs—words that are spelled the same but have different meanings, for example the *bow* of a ship versus a *bow* and arrow.
■ Alternate spellings—two or more acceptable ways of spelling the same word, for example, *color* and *color.*

2.1.9 Metadata in archival sector

There is one anomaly in the DW 2.0 environment when it comes to metadata, and that is the metadata that is associated with the Archival Sector. The metadata that is associated with archival processing is stored with the archival data itself. The reason for storing the archival metadata with the archival data is that it is assumed that the metadata will be lost over time if it is collocated with its associated archival content data. Of course, it is possible to store a separate set of metadata with the archival environment, but the first place where historical metadata is most often sought and likely to be most useful is with the archival data itself. Figure 2.1.13 suggests that archival data should contain its own metadata.

FIGURE 2.1.13 Metadata in the Archival Sector.

2.1.10 Maintaining metadata

One of the biggest challenges associated with metadata is not the initial creation of the metadata environment, but rather the ongoing maintenance of the metadata environment. Change is a constant and it affects metadata just as much as it does everything else.

Figure 2.1.14 shows that change occurs and that when the environment is an active metadata environment it is easier to keep up with change than in a passive environment. It is much too easy to just ignore change with a passive metadata environment. Change occurs and the associated alterations to passive metadata tend to be put off. Then one day, someone wakes up and sees that normal, routine changes have not been reflected in the metadata environment for a long time, resulting in a passive metadata that is so far out of sync with the present that it is virtually useless, often coincidentally when it is most needed. With an active metadata repository, changes need to be reflected regularly in the metadata to keep up with the normal evolution and maintenance of existing systems. Whenever the systems are changed, the metadata must be changed as well.

Using metadata is just as important as the storing and periodic update of metadata. Although there are many ways to use metadata, perhaps the most effective way is at the point of end user interface—the point at which the end user is doing interactive processing.

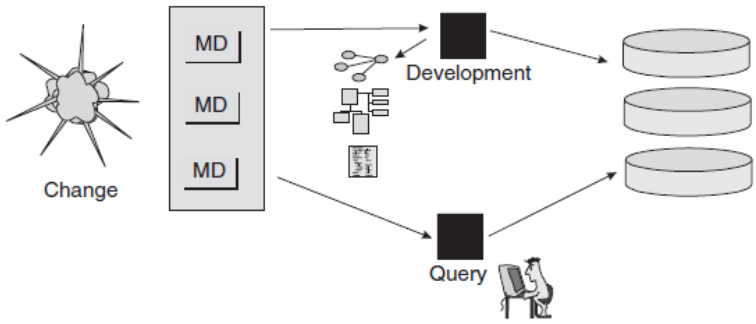


FIGURE 2.1.14 Where there is an active repository it is relatively easy to keep up with change.

2.1.11 Using metadata – an example

For an example of the usage of DW 2.0 metadata, consider the following scenario. An airline agent—P. Bruton—pulls up a screen with the flight details for a client—Bill Inmon.

FIGURE 2.1.15 An end user initiates a screen or report.

Figure 2.1.15 shows the screen that she has pulled up. P. Bruton is not familiar with the reservations made by Tiki Airlines. She sees a data field on the screen that interests her. She highlights the CONNECTIONS field on the Tiki Airlines screen, as shown in Figure 2.1.16.

FIGURE 2.1.16 The user selects a unit of data that is of interest.

Next, P. Bruton hits a Function key, and the popup menu shown in Figure 2.1.17 appears. The menu asks her to identify what aspect of CONNECTIONS she wants to see. The options on the menu are " AKA " (also known as), " definition, " " formula, " " where used, " and " owned by. " P. Bruton selects the definitions option. Figure 2.1.17 shows the menu of types of metadata that are available.

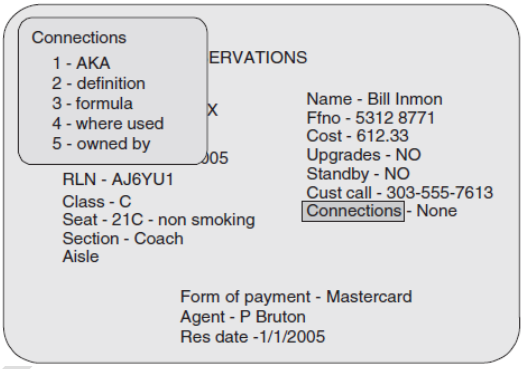


FIGURE 2.1.17 The system describes the possibilities of metadata that can be displayed.

Now the system goes and searches its metadata information for a definition of CONNECTIONS. Upon finding the definition, the system displays a message showing the definition of CONNECTIONS. Figure 2.1.18 shows the display of a metadata definition in an interactive mode.

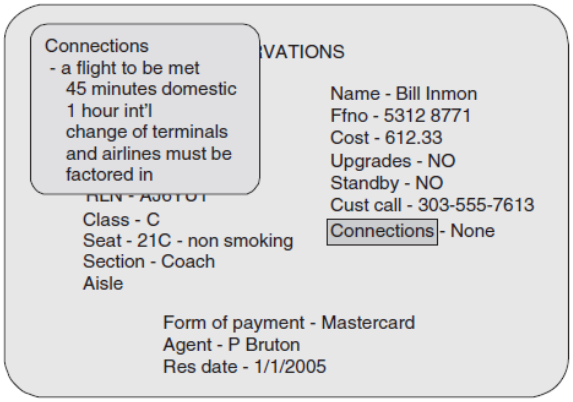


FIGURE 2.1.18 The desired metadata is displayed.

Note that the accessing of metadata never entailed leaving the original screen. The display of metadata was done on top of an analytical process. The metadata was displayed interactively and became part of the analytical process.

2.1.12 End user perspective

There are many purposes served by metadata in DW 2.0. Metadata serves to interconnect the different sectors of data. It serves as documentation for the environment. It serves as a road map for adding to the DW 2.0 environment. But the most important role metadata plays is as a guide to the contents and relationships of data found in DW 2.0. It is the end user who needs a guide to the data and relationships that are found in DW 2.0. When the end user has a guide to what data already exists in the DW 2.0 environment, there is the possibility of the reuse of data. Stated differently, when there is no metadata in DW 2.0, the end user always has to invent every new analysis from scratch.

The end user is incapable of seeing what analytical work has been done before. Therefore everything must be new. The problem with every analysis being new is that in many cases the wheel gets to be reinvented over and over. But with metadata there is no need to reinvent the wheel. One analyst can build on the work of another analyst. From the standpoint of the business user, there is another important role for metadata and that role is in showing the heritage of data. In many cases in which an analyst considers using a unit of data as part of an analysis, the business user needs to know where the data came from and how it was calculated. It is metadata that provides this very important function in DW 2.0.

There is yet another important role played by metadata in the eyes of the business user. On occasion there is a need for compliance of data. There is Sarbanes Oxley and there is Basel II. Metadata provides the key to the audit trail that is so vital to compliance in the analytical environment. There are then some very pragmatic reasons metadata plays such an important role in the eyes of the business user in DW 2.0.

2.1.13 LET US SUM UP

Metadata is the key to reusability of data and analysis. Metadata enables the analyst to determine what has already been built. Without metadata, the analyst has a hard time finding out what infrastructure is already in place. There are four levels of metadata:

■ Enterprise
■ Local
■ Business
■ Technical

There is metadata for both the structured and the unstructured DW 2.0 environments. A metadata repository can be either active or passive. As a rule, an active metadata repository is much more desirable and useful than a passive metadata repository. When development and analysis are used interactively with a metadata repository, the repository is considered to be active. The system of record of data warehouse data is ideally defined in a metadata repository.

Unstructured metadata consists of taxonomies, glossaries, and ontologies. The forms of metadata may be internal or external. Archival metadata is stored directly in the archival environment. By collocating the metadata with the physical storage of archival data it describes, a time capsule of data can be created.

2.1.14 List of References

- DW2.0 The architecture for Next Generation of Datawarehousing W.H. Inmon, Derek Strauss, Genia Neushloss, ELSEVIER.
- Building the data warehouse, W.H.Inmon, third Edition, Wiley.
- Datawarehousing, S. Mohanty, TMH .
- The Data Warehouse Lifecycle toolkit", Ralph Kimball ,John Wiley.

2.1.15 Unit End Exercises

1) Describe internal and external taxonomy in the unstructured DW 2.0 with the help of an example.
2) Differentiate between Active and Passive Repository.
3) Explain general architecture for metadata in DW 2.0

UNIT-II: Methodology and Approach for DW

Unit Structure

2.2.0 OBJECTIVES

In this chapter you will learn about:

- Spiral model methodology
- Seven stream approach
- How stream are used for designing DW/BI?

2.2.1 INTRODUCTION

To be successful with DW 2.0, organizations need to adopt a spiral development approach in which numerous small iterations of data warehouse development are accomplished frequently. There should be a period of no longer than 3 months for each iteration of development. The spiral development approach to the building of a data warehouse is a standard practice that has proven its worth around the world as people build data warehouses.
So what is the spiral methodology all about? To describe the spiral methodology, it is necessary first to have a look at its exact opposite, the waterfall development methodology.

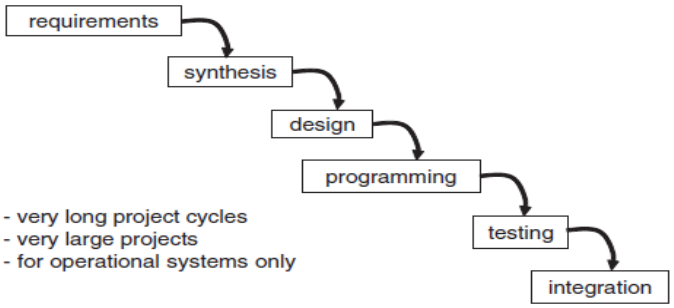


FIGURE 2.2.1 Waterfall development methodology

The diagram in Figure 2.2.1 illustrates the typical waterfall development methodology life cycle. The waterfall methodology has also been called “SDLC,” which is short for “systems development life cycle.” Waterfall methods have historically been very successful in delivering

online transaction processing systems. These operational systems normally have very long project cycles. They are very large projects, and all the requirements are typically documented up front. During the early days of data warehousing, DW 1.0, it was mistakenly thought that waterfall methodology could be applied to data warehouse projects.

2.2.2 Spiral model methodology

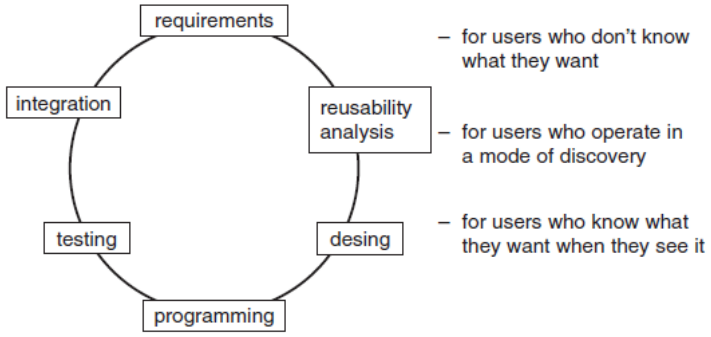


FIGURE 2.2.2 Spiral development methodology

In contrast to the above, the spiral methodology, illustrated in Figure 2.2.2, is ideally suited to users who do not know what they want. Most of the time when a data warehouse project is undertaken, it is impossible to gather all the business requirements up front. This is not due to anyone's fault—it is not because the business cannot make up its mind, and it is not because the information technology group is too out of touch to be able to understand the business's needs.

It is simply the nature of the beast. Business intelligence capability is normally developed in a mode of discovery. The members of the business community will ultimately know what they want when they see it—and as soon as they get what they want, they understand that they want and need something else. The BI requirements goal posts are constantly changing, and it is understandable that this should be the case, because the business is constantly changing. In the development of an operational system, if the business asks for changes to be made to the system after it is deployed, it is seen as a mark of failure. In contrast, change is seen as a very good thing with data warehouse/business intelligence systems.

A change request is a sign of success—it means the business is using the warehouse; it means the warehouse has stimulated thought, which has now resulted in a request for more or different information.

If no requests for change occur, then the data warehousing initiative is a failure. In short, change is bad in a transaction processing system and change is good in a data warehouse environment. Many people confuse the spiral methodology with the iterative approach that has its roots in the object-oriented disciplines. Although there are some similarities, the two methodologies are actually quite different. Some of the hallmarks of the spiral development methodology are as follows:

- Spiral methodology makes extensive use of prototyping.
- Major tasks in spiral methodology can occur in any sequence.
- It is not necessary to wait for one task to finish before starting the next task.

■ A different project management philosophy and mind-set are required.
■ Culture change in both the business and the IT communities is required.
■ Expectations must be managed, because the first iteration of data warehouse development is incomplete and subject to further refinement.
■ A quarterly delivery cadence is typical, requiring strict scope discipline.

During the first generation of data warehousing, one voice consistently called for the use of spiral methodologies. Larissa Moss has written several books and given numerous public classes on using spiral methodology for building data warehouses. Organizations that have taken heed have benefited from her guidance. Organizations that have disregarded spiral methodology continue to struggle. As the era of DW 2.0 begins, it is imperative that the lessons of past mistakes be heeded.

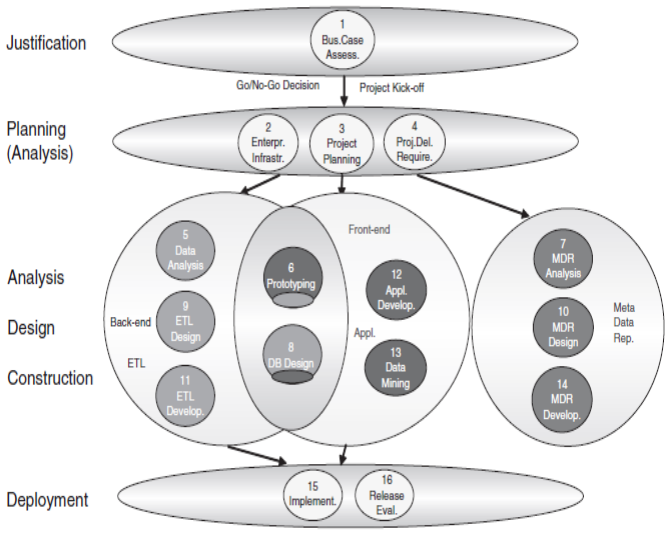


FIGURE 2.2.3 Larissa's three spiral parallel development tracks. Courtesy of Larissa T. Moss, Method Focus, Inc.

Figure 2.2.3 illustrates Larissa Moss's "Business Intelligence Road Map " methodology. At first glance, the major segments of work listed down the left-hand side of Figure 2.2.3 appear to be just like any waterfall methodology. Indeed, many of the same things that people do on waterfall projects are equally applicable in spiral methodology. The diagram highlights, however, that there is a lot of concurrency happening in the analysis, design, and construction work segments. Put this together with the fact that work can begin at any point in the spiral development life cycle (e.g., it is common to start with construction and then work upward) and one can begin to see how vastly different this methodology is from conventional IT development methodologies. Careful consideration must be given to team organization to achieve the concurrent tasks.

The diagram in Figure 2.2.3 highlights three natural groupings of work: back-end work, front-end work, and metadata work. It shows the overlap of back-end and front-end work around prototyping and data base design. The metadata work, although shown off to the right-hand side, is in fact also highly interdependent on the other parallel work. It would probably be more accurate, but less readable, in the diagram to show the metadata work superimposed over the top of the back-end and front-end work. Project management must be able to recognize and manage these interdependencies. It is important to note that although a data warehouse

application iteration can start with construction, the iteration will not be complete unless and until justification, planning, analysis, design, construction, and deployment have all been done. To build an application, it may take several passes through the three spiral work tracks.

Although it is true that for each application all steps need to be done, there is normally no need to revisit "Justification" or even some of the other steps in each pass. Figure 2.2.4 provides further insight into the three parallel spiral development tracks.

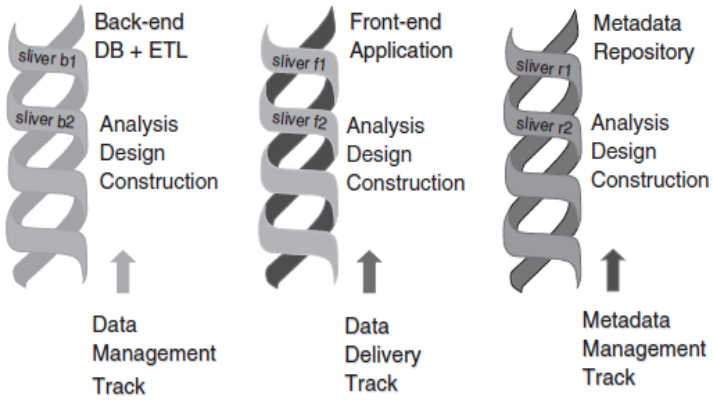


FIGURE 2.2.4 Another illustration of Larissa's three parallel development tracks of spiral methodology. After Larissa T. Moss, Method Focus, Inc.

A spiral methodology produces an application by means of several interim deliverables, or "slivers." Note that each spiral sliver in Figure 2.2.4 addresses a very small, but meaningful, component of the overall application scope. The delivery is small and meaningful, but not completed all at once. The diagram shows two slivers per work track—in reality there may be several slivers per work track, and the back-end, front-end, and metadata repository work tracks seldom have equal numbers of slivers. Successive iterations, delivering successive slivers, ultimately deliver the working application.

The scope of each sliver must be kept extremely small. In addition, the scope must be managed very carefully so as to ensure no scope creep occurs. The goal of spiral methodology is to build an inventory of reusable assets. Larissa shows in Figure 2.2.5 how project constraints have to be reshuffled dramatically, with quality being maximized and scope being minimized.

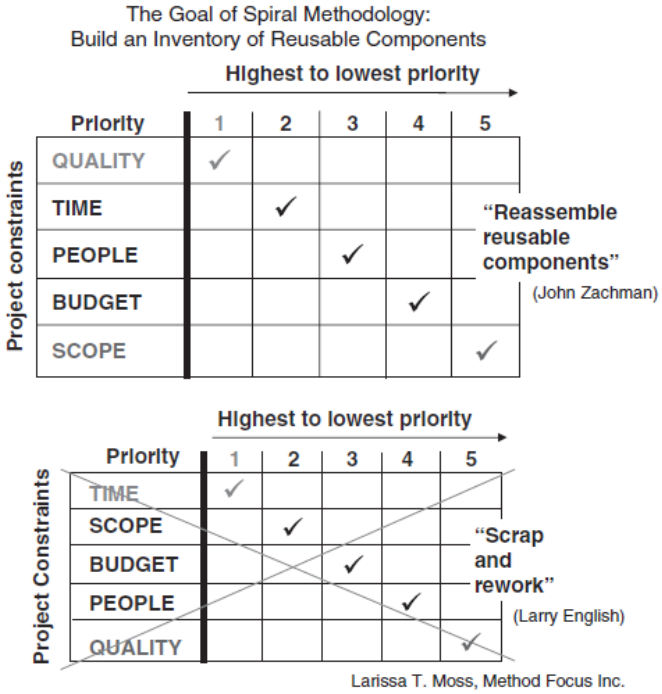


FIGURE 2.2.5 Project constraint reshuffling and component reuse. Courtesy of Larissa T. Moss, Method Focus, Inc.

Using the spiral methodology helps an organization move away from the death spiral of scrap and rework toward a culture based on a return on assets. This new way of thinking about things is designed to focus on reusable components that can be reassembled to meet the business's demand that things be done cheaper, faster, and better. Second-generation data warehousing needs to move away from the conventional return on investment (ROI) approach. The return on investment approach has led many organizations to build successive
"Point solutions" (or data marts), with little or no attention given to the reusability of anything that has been delivered. The ROI imperative says: "get it in as quickly as possible and get some return."

The fact that the initial success cannot be sustained, and the data structures prove to be brittle (not change-enabled), seems to escape the ROI mind-set. In the world of DW 2.0, success is measured by ROA (return on assets). Has the data been reused? Has the metadata been reused? Have the structural business rules been reused? Is the ETL logic reusable and extensible? Are the components of the presentation layer reusable and
flexible? Introducing a spiral methodology is a critical step toward success in second-generation data warehousing. There are several things that an organization can and must do to make spiral development methodology a certain success. A whole new approach to enterprise data needs to be adopted. This approach is discussed in the remainder of this chapter, encapsulated in a tried and tested approach called the "seven streams approach."

2.2.3 Seven streams approach

The basic premise is that business intelligence initiatives should be seen as programs, not projects. Such business intelligence initiatives should be ongoing, becoming more and more honed and sophisticated as business needs change. To achieve and sustain this in a changing business environment requires more than any "data warehouse in a box" or 30- to 90-day "wonder cure" solution can provide. What then are the various factors one needs to take into consideration to achieve sustained success in a business intelligence/data warehousing program? A highly summarized answer to this question is given in the pages that follow, starting with Figure 2.2.6, which illustrates the seven streams approach a proven business intelligence planning and delivery framework.

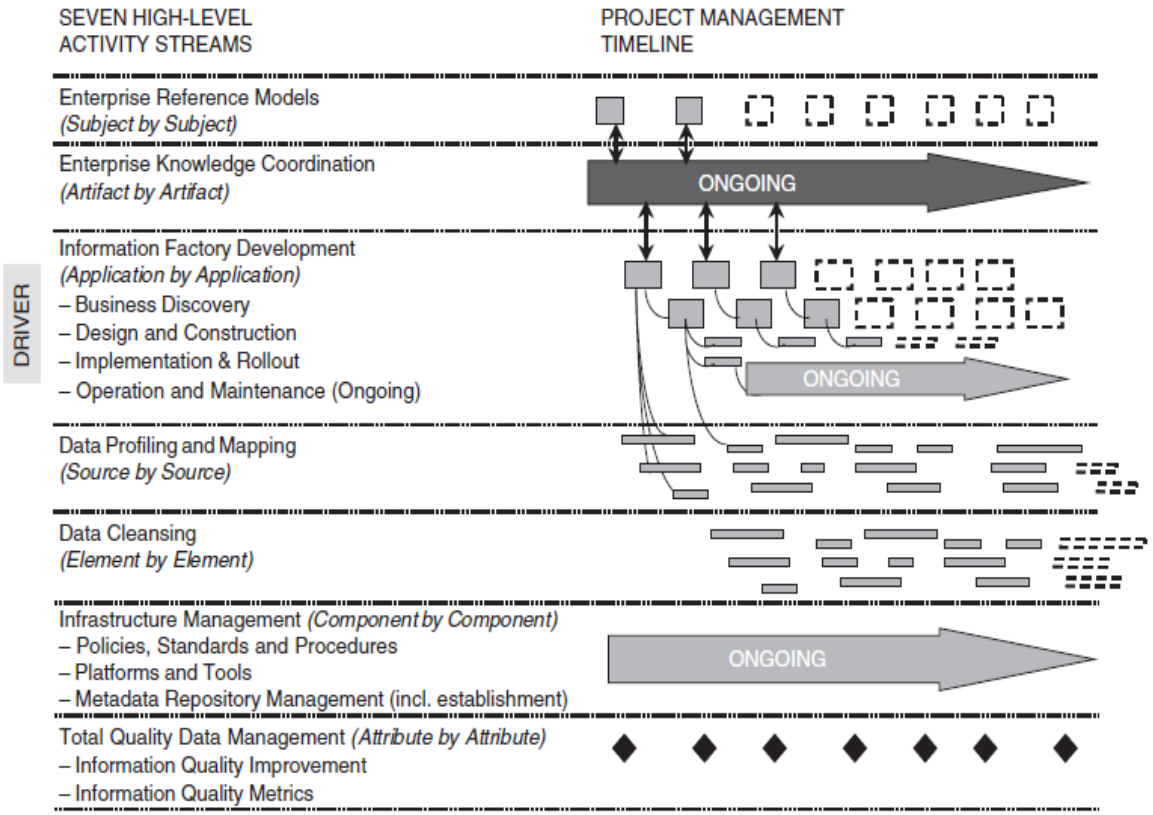


FIGURE 2.2.6 The seven streams approach to DW/BI projects

The key thing to note about the seven streams approach is the fact that each activity stream marches to the beat of a different drum. Each stream is simultaneously initiated, is concurrently driven, and needs to be coordinated and monitored. There is no implied sequence in the way the diagram is organized.

2.2.4 Enterprise reference model

The first activity stream addresses the creation and continued maintenance of a corporate data model (Figure 2.2.7). Of course, this is not done by saying, "stop the world while we build this great gargantuan corporate data model." This is a corporate data model that is built incrementally, subject area by subject area (e.g., customer, product, etc.).

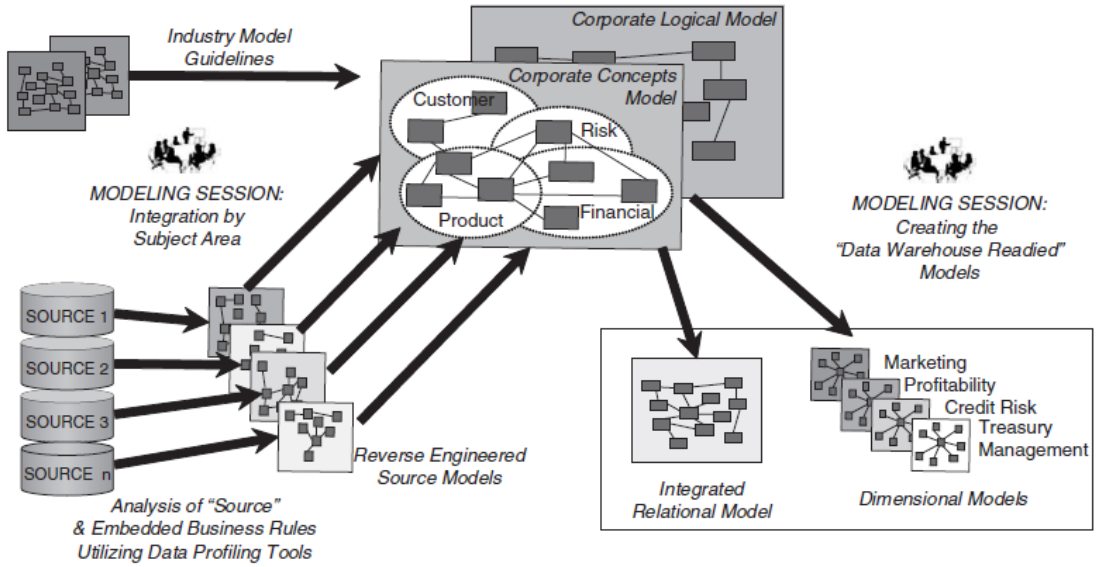


FIGURE 2.2.7 The corporate data model stream.

2.2.5 Enterprise knowledge coordination stream

The next activity stream, knowledge coordination, entails taking the various artifacts that come out of the three data discovery streams (i.e., the corporate data modeling, information factory development, and data profiling streams) and making sense out of those findings, as follows:

■ Corporate data modeling: Corporate data modeling usually entails analyzing the corporation's data top down—identifying the context, concepts, and high-level logical view of the enterprise's data. The corporation's enterprise data is discovered along the path from major business subject (the top) to corporate data entity attribution (the bottom).

■ Information factory development: In the information factory development stream, build activities take place topic by topic or application by application. The business discovery process is driven by the "burning questions" that the business has put forward as its high-priority questions. These are questions for which the business community needs answers so that decisions can be made and actions can be taken that will effectively put money on the company's bottom line. Such questions can be grouped into topics, such as growth, profitability, risk, and so forth. The information required to answer these questions is identified next. Finally the data essential to manufacture the information that answers the burning questions is identified.

■ Data profiling: The data profiling stream entails bottom-up data discovery. The actual detailed data in the company's existing systems is examined and analyzed. Data profiling

identifies and helps resolve redundant data elements; it helps identify the correct data systems of record, as well as systems that should not be considered data warehouse sources; it enables data modelers to map data elements to data entities and fully attribute the company's major data entities. The above three sources of data discovery obviously all need to be tied together and resolved in some way, and that is what happens in the corporate knowledge coordination stream. Artifact by artifact, the output from the three data discovery streams is reconciled. A steady state model is created that provides reusable knowledge about the organization's data and reliable information that can be delivered to stakeholders at the appropriate time. It is best practice for knowledge coordinators to make use of the Zachman framework as a classification schema for the reusable artifacts under their custodianship. Figure 6.8 illustrates how top-down and bottom- up knowledge is coordinated, using the backdrop of the Zachman framework as a "thinking tool."

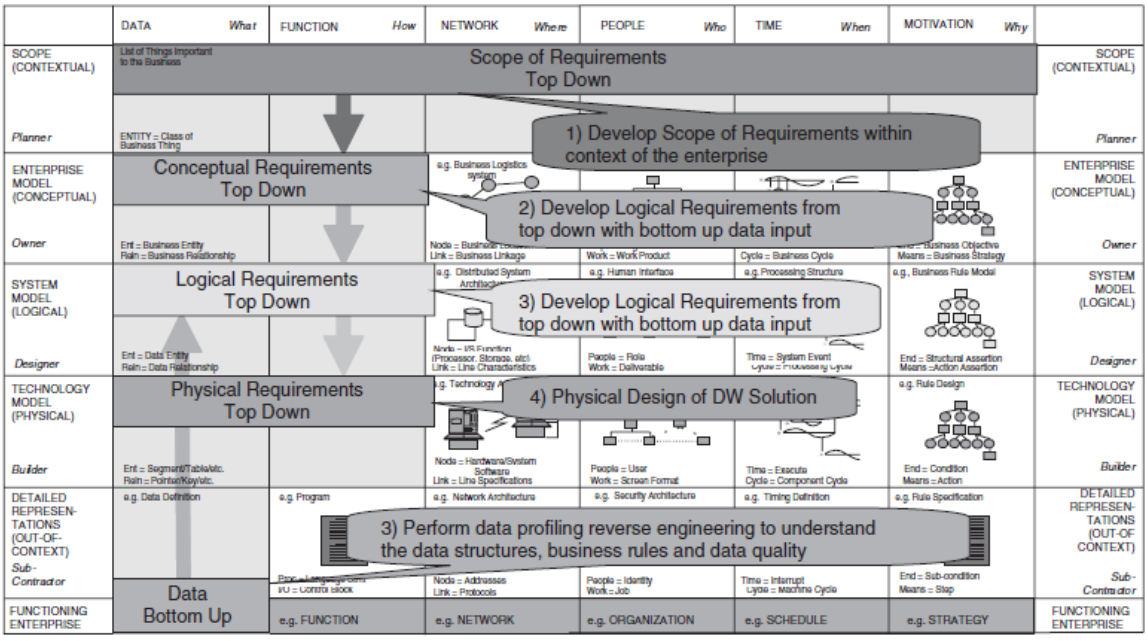


FIGURE 2.2.8 The knowledge coordination stream makes use of the Zachman framework as a classification schema. Published with permission from John A. Zachman,

2.2.6 Information factory development stream

The next stream is the information factory development stream. It is here that building of the information factory takes place. The factory is usually built topic by topic. Each topic contains several burning questions. Of course a topic, such as growth, often spans multiple subject areas, such as customer and product. Topics are often grouped into applications, e.g., an agent scorecard. The information factory development stream is the domain of the spiral methodology. This stream is the "driver stream" in that it sets the priorities for the other six streams.

2.2.7 Data correction stream

The next activity stream is the data correction stream. This stream involves going attribute by attribute through the pertinent source systems and determining what data needs to be corrected, completed, or purged and what data correction rules need to be applied.

2.2.8 Infrastructure stream

The next activity stream is the infrastructure stream. This stream of activities addresses the supporting infrastructure for the scalability of the information factory, including consideration of people, resources, platforms, tools, policies, standards, and procedures, as indicated in Figure 2.2.9. The infrastructure stream is undertaken component by component.

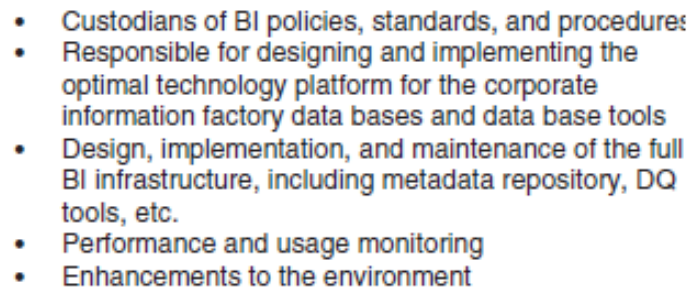
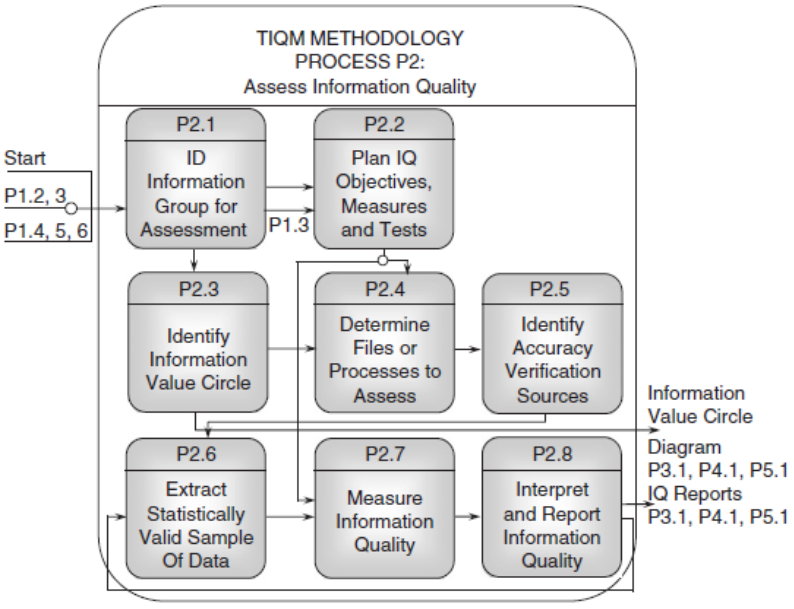


FIGURE 2.2.9 Infrastructure management stream components.

2.2.9 Total information quality management stream

Last but not least, the total information quality management stream concerns data quality monitoring and process improvement, which is achieved process by process. Specific data elements in the environment are examined and their quality is monitored and reported over time. The most comprehensive and rigorous method for addressing total information quality management has been developed by Larry English. His total information quality management methodology (TIQM), formerly called total quality data management (TQdM), consists of several major processes.

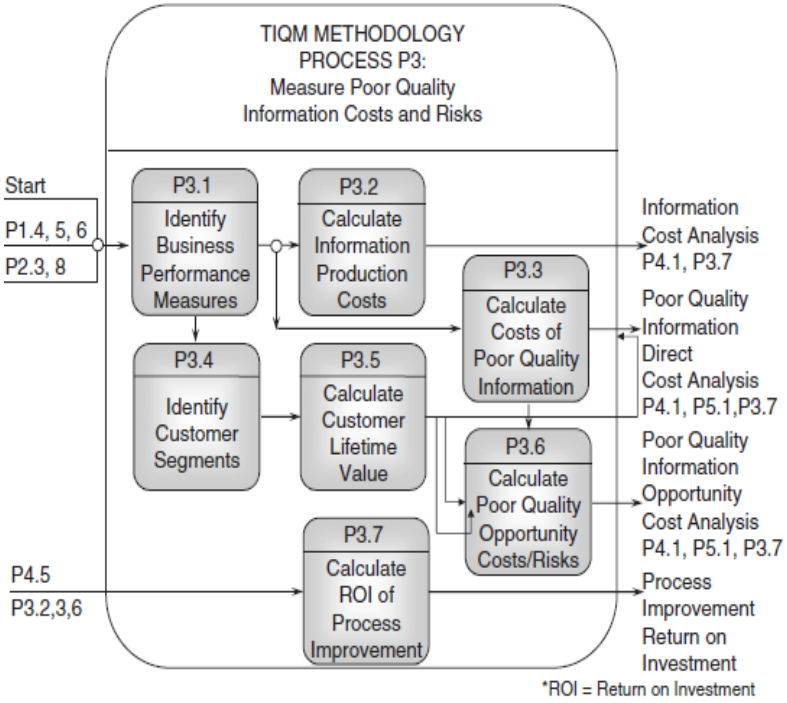
The “assess information quality” process is shown in Figure 2.2.10. Notice that the process is ongoing—see the recursive loop between P2.8 and P2.6 in Figure 2.2.10.



L. English, *Improving Data Warehouse and Business Information Quality*, p. 156. Used with permission.

FIGURE 2.2.10 Information quality assessment in the total information quality management stream. Courtesy of Larry P. English.

In the world of DW 2.0, the organization should measure the quality of information on a regular basis to assure the process stays in control. It is, after all, a truism that "you cannot manage what you cannot measure." After measuring the extent of information quality problems, including accuracy, completeness, and non-duplication, you should measure and calculate the costs of the poor quality information on the downstream processes, including costs to business intelligence processes. This provides the business case for process improvement to identify and eliminate the root causes of defects at the source and through the
information value chain. This process is TIQM Process 3, "measure poor quality information costs and risks," illustrated in Figure 2.2.11.

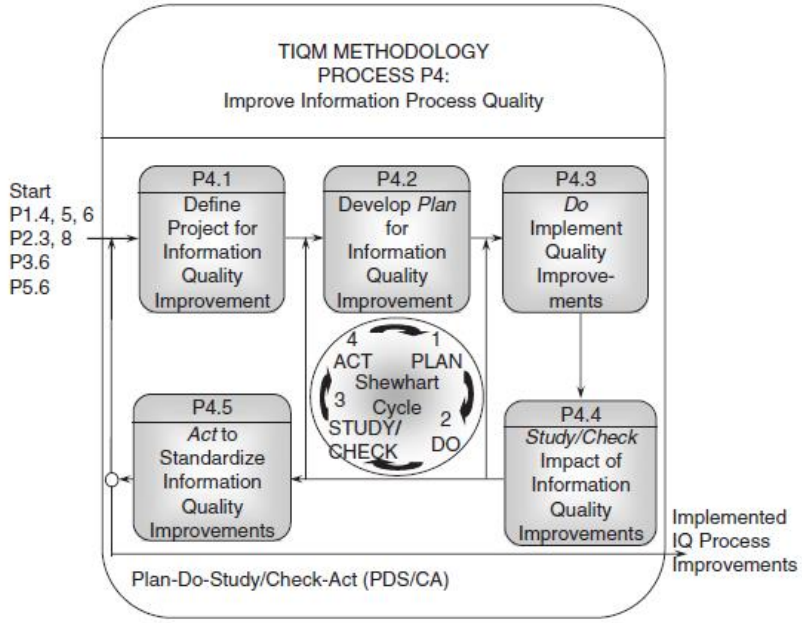


L. English, *Improving Data Warehouse and Business Information Quality*, p. 214.
Used with permission.

FIGURE 2.2.11 Measurement of the costs of poor quality information and the ROI of information process improvements in TIQM. Courtesy of Larry P. English

At a minimum you should measure the direct costs of poor quality information following step P3.3. While the opportunity costs are considered "intangible," they can be dramatic in the form of missed customer revenue and lost customer lifetime value. These are very real costs that come with poor quality information, such as misspelling names, wrong addresses, duplicate customer records, incorrect billing, sending the wrong item on an order.

Understanding the costs of poor quality information enables you to focus on the high pay-off areas for process improvement, always taking a Pareto approach of most important to next-most important. The process "improvement cycle of plan-do-check/study-act" is illustrated in TIQM P4, "improve information process quality," illustrated in Figure 2.2.12.



L. English, *Improving Data Warehouse and Business Information Quality*, p. 290. Used with permission.

FIGURE 2.2.12 The improve information process quality process in TIQM. Courtesy of Larry P. English.

Process 4 is the core competency process in TIQM, required to use the name "quality" in the "information ***quality*** management" label. For this is the process that eliminates the defects which cause business process failure and information scrap and rework. When this process becomes a habit within an organization, it puts that organization on the path to world-class status.

Step P4.1 establishes an initiative for a process improvement based on the organization's project management guidelines.

Step P4.2 first analyzes and identifies the root cause or causes of a broken process causing defective information. This step then defines process improvements that will eliminate the root causes and prevent or significantly reduce the information defects.

Step P4.3 implements the improvements to study and assure they have achieved the improvement goals.

Step P4.4 analyzes the results to assure the improvements worked and documents the lessons learned.

Step P4.5 acts to roll out the improvements to all places where the process is performed, and puts the process in control.

2.2.10 LET US SUM UP

Each of the seven streams in the DW/BI project approach focuses on a different aspect of the corporation's data architecture and is undertaken using a different and correspondingly appropriate work approach:

Stream 1—Enterprise reference modeling is done subject by subject.
Stream 2—Enterprise knowledge coordination is done artifact by artifact.
Stream 3—Information factory development is done topic by topic.
Stream 4—Data profiling and mapping are done source by source.
Stream 5—Data correction is done attribute by attribute.
Stream 6—Infrastructure management is done component by component.
Stream 7—Total information quality management is done process by process to improve and error-proof processes.

Each stream produces deliverables at different rates. The astute DW/BI program manager will recognize what these different rates and rhythms are, will synchronize the work priorities in each of these concurrent streams, and will use this information to define meaningful releases for the organization. The seven streams approach is a framework and tool for designing a DW/BI program that lends itself well to rapid spiral development. The interaction of the seven streams approach and spiral development methodology is graphically depicted in the next few diagrams.

2.2.11 List of References

- DW2.0 The architecture for Next Generation of Datawarehousing W.H. Inmon, Derek Strauss, Genia Neushloss, ELSEVIER.
- Building the data warehouse, W.H.Inmon, third Edition, Wiley.
- Datawarehousing, S. Mohanty, TMH .
- The Data Warehouse Lifecycle toolkit", Ralph Kimball ,John Wiley.

2.2.12 Unit End Exercises

1) Explain key features of Spiral development methodology for DW 2.0.
2) Describe Heuristic Analysis. What is freezing of data and Why is it required?
3) Explain seven streams approach in detail.

UNIT-II: Statistical processing and DW

Unit Structure

2.3.0 OBJECTIVES

In this chapter you will learn about:

- What us statistical analysis?
- Types of transaction
- Exploration processing
- Data mart and backflow of data.
- Perspective of business analyst.

2.3.1 INTRODUCTION

One of the most important functions of any data warehouse is the support of statistical analysis. If an organization has a data warehouse and there is no statistical analysis that is occurring, then a major function of the data warehouse environment is not being exploited. Traditionally, certain industries have made more of the capability of statistical analysis than other industries. While the insurance, manufacturing, and medical research industries have all made extensive use of statistical analysis, there is a place for statistical analysis in practically any industry.

From the end user's perspective, the kind of data that is revealed by statistical analysis is fundamentally different from information determined in other manners. For one example, statistically generated data is almost always used strategically. There are very few cases in which statistically generated data is used tactically. Another reason why statistical analysis is

different from other forms of analysis is that statistical analysis of necessity looks across broad vistas of data. Other forms of analysis look at much smaller sets of data. And a third difference between analysis of statistical data and that of other data is that the vista of information examined by statistics is much longer than other forms of analysis. It is normal when doing statistical analysis to look across 5 years, 10 years, or even more. DW 2.0 supports statistical analysis and processing just as it supports other forms of analysis. Depending on the type and frequency of statistical analysis, DW 2.0 can be used either directly or indirectly.

2.3.2 Two types of transaction

The nature of the statistical analysis transaction lies at the heart of the DW 2.0 support of statistical analysis. Figure 2.3.1 illustrates the two basic transaction types common in data warehousing—a standard query or a statistical analysis.

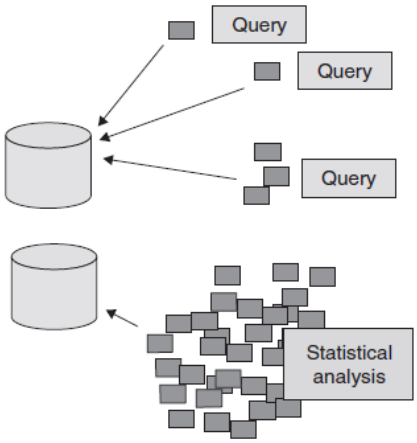


FIGURE 2.3.1 The resources used by statistical analysis far exceed those used by other styles of transactions.

Figure 2.3.1 shows that standard queries typically use only a few resources, because only a few units of data (i.e., records of data) are required. In contrast, statistical analysis typically requires many, many resources to satisfy the statistical query, and many, many records of data are required. Another way of looking at this fundamental difference between query and statistical analyses is to compare the output of a typical query transaction and a typical statistical analysis, as shown in Figure 2.3.2.

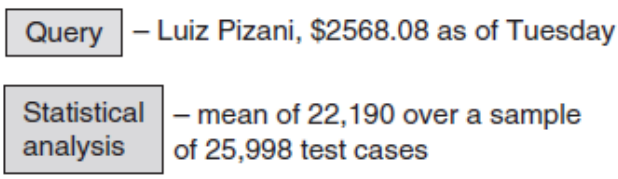


FIGURE 2.3.2 The results of a query are quite different from the results of a statistical analysis.

Figure 2.3.2 represents a query that looks for a small amount of data and reports a small amount of data. In this example, the query has retrieved the record for Luiz Pizani and has found his bank balance. To satisfy the query, only one or two records of data were required for analysis.

In a statistical analysis, however, many records are required. In Figure 2.3.2 is it seen that the results of the query entailed computation of a statistical mean or an average, and to calculate that mean, nearly 26,000 records had to be accessed. Furthermore, the records accessed by the statistical query had to be accessed all at once. The mean value could not be calculated until all the records were available.

2.3.3 Statistical analysis

There are many things that can be done with statistical analysis. One of the simplest statistical analyses that can be done is the creation of a profile of data. A profile of data is a statistical summary of the contents of a body of data. Typical questions answered by a statistical profile of data include: How many records are there? What values are the highest, the lowest? What is the mean, the median, and the mode? Are there values that are out of domain? Are there values that are in domain and appear to be outliers? What is the distribution of data values?
All these questions and more contribute to the creation of a profile of a body of data. The profile of the body of data allows the analyst to look at an overview of the data set to examine the forest, not just the individual trees.

However, there are many other uses for the statistical analysis of data. One such use is the comparison of corporate data to external data. The development of corporate data is the first step in the comparison of corporate data to external data. Then, the external data is captured and put on a similar footing. Then a comparison is made. To illustrate a typical comparison of corporate data to external data, Figure 2.3.3 depicts what might be shown when Coca Cola data is compared to industry-wide beverage sales.

Comparisons of the statistical peaks and valleys may be of particular interest to the beverage business analyst who wants to know, Are Coca Cola sales rising while general industry trends are falling? Are Coca Cola sales falling while industry trends are rising? Is there a general correlative pattern between the two beverage sales curves? The ability to compare and contrast corporate information to external information can lead to truly valuable business insight.

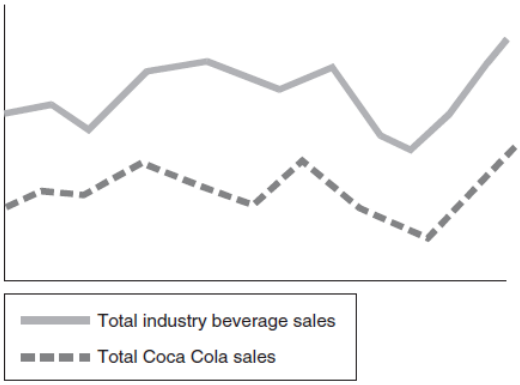


FIGURE 2.3.3 Comparing industry information to corporate information.

2.3.4 Integrity of comparison

The validity of data is one of the key issues in comparative statistical analysis. Conscientious statisticians routinely want to know if they are comparing the same thing—comparing apples with apples, or apples with oranges. Returning to Figure 2.3.3, for example, when industry

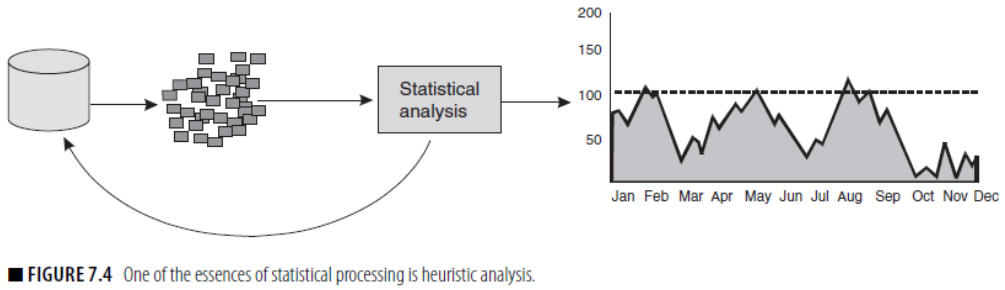
beverage sales are compared to sales of Coca Cola, is the comparison valid? After all, if beverage sales include beer and wine coolers, is it fair and meaningful to compare Coca Cola sales to the sales of these beverages? What if Coca Cola includes such drinks as Minute Maid? Is it fair (or even smart) to compare sales of Minute Maid to sales of Pepsi Cola? The comparison of external data to internal data introduces issues to be resolved before the statistical comparison can be considered a valid one.

There are, however, many important uses for statistical analysis other than the comparison of external to internal data. Another such usage for statistical analysis is to determine trends and patterns in data. The business case for data warehouse support of statistical analysis is a strong one. Even the act of analysis must be considered when pondering statistical analysis. When simple queries are made, the intent is to find data that satisfies an immediate need for information. But, when statistical analysis is done, it typically takes the very different analytical form known as heuristic analysis.

Heuristic analysis is the kind of analysis that is done as a part of discovery. A thing is characterized as heuristic if it involves or serves as an aid to learning, discovery, or problem-solving. In an act of discovery, the analyst does not know what secrets the data holds. The analyst sets out to find or learn what the data contains or means, without knowing
what the data contains or what the analyst is looking for ahead of time.

2.3.5 Heuristic analysis

The heuristic analyst has the attitude, "I don't know what I want, but I will know it when I see it." In heuristic analysis, the next iteration of analysis is determined by the results obtained from the current analysis. In a true heuristic analysis, it is not possible to plan how many iterations of analysis there will be or where the analysis will lead. The heuristic analytical attitude is found everywhere in business. It usually is only the clerks who know exactly what they want.



■ FIGURE 7.4 One of the essences of statistical processing is heuristic analysis.

FIGURE 2.3.4 One of the essences of statistical processing is heuristic analysis.

Figure 2.3.4 depicts the general flow of analysis in a heuristic environment. Statistical processing is associated with heuristic analysis. One of the unusual aspects of heuristic analysis is the need occasionally to "freeze" data. When data is frozen, no new data is ingested into the system. The reason for occasionally freezing data when doing heuristic statistical processing is the need to see if the results obtained by the analysis were caused by a change in algorithms or a change in data. For example, an analyst runs a transaction against a body of data and finds that the data returns an average of 67 units. The analyst then changes the algorithm and reruns the analysis, yielding a new average of 98 units. The question then becomes, Is the change in

results a function of the change in the algorithm or a change in the data? If the second analysis was run against a different set of data, the different results may be a function of operating against different data, rather than the result of changing the algorithm used in calculation.

2.3.6 Freezing data

When fine distinctions like this are important, it is necessary to be able to freeze the data that has been used in a calculation. Freezing the data means that any change in results can definitely be attributed to algorithmic change and nothing else. Figure 2.3.5 depicts data frozen to support heuristic processing and results analysis.

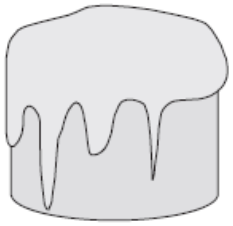


FIGURE 2.3.5 On occasion data needs to be frozen.

2.3.7 Exploration processing

One of the essences of statistical processing is that it is often an exercise in exploration. In many forms of information processing, analysis is done on data whose content, form, and structure are well known. However, in other forms of information processing, analysis is done when the content, form, and structure of the data are not well known. It is this style of processing exploration processing to which statistical analysis is particularly well suited.

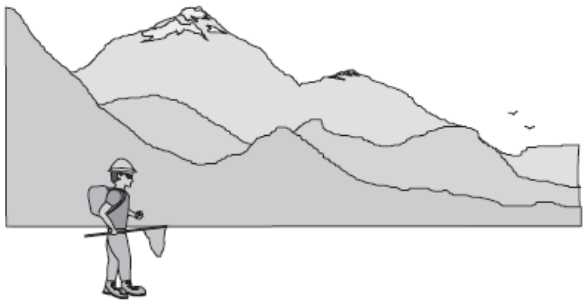


FIGURE 2.3.6 Exploration processing—finding out what the possibilities are.

Figure 2.3.6 depicts exploration processing. The challenge for the DW 2.0 data warehouse environment is how best to support statistical processing. Certainly DW 2.0 holds data that can be very useful in statistical processing. In fact, the DW 2.0 architecture includes the key ingredient for statistical analysis. However, there remain some issues relating to the usage of the data found in the DW 2.0 environment.

2.3.8 Frequency of analysis

The frequency of statistical analysis is germane to the support of statistical processing by DW 2.0. Figure 2.3.7 shows that as the frequency of statistical analysis changes, the supporting DW 2.0 infrastructure also changes.

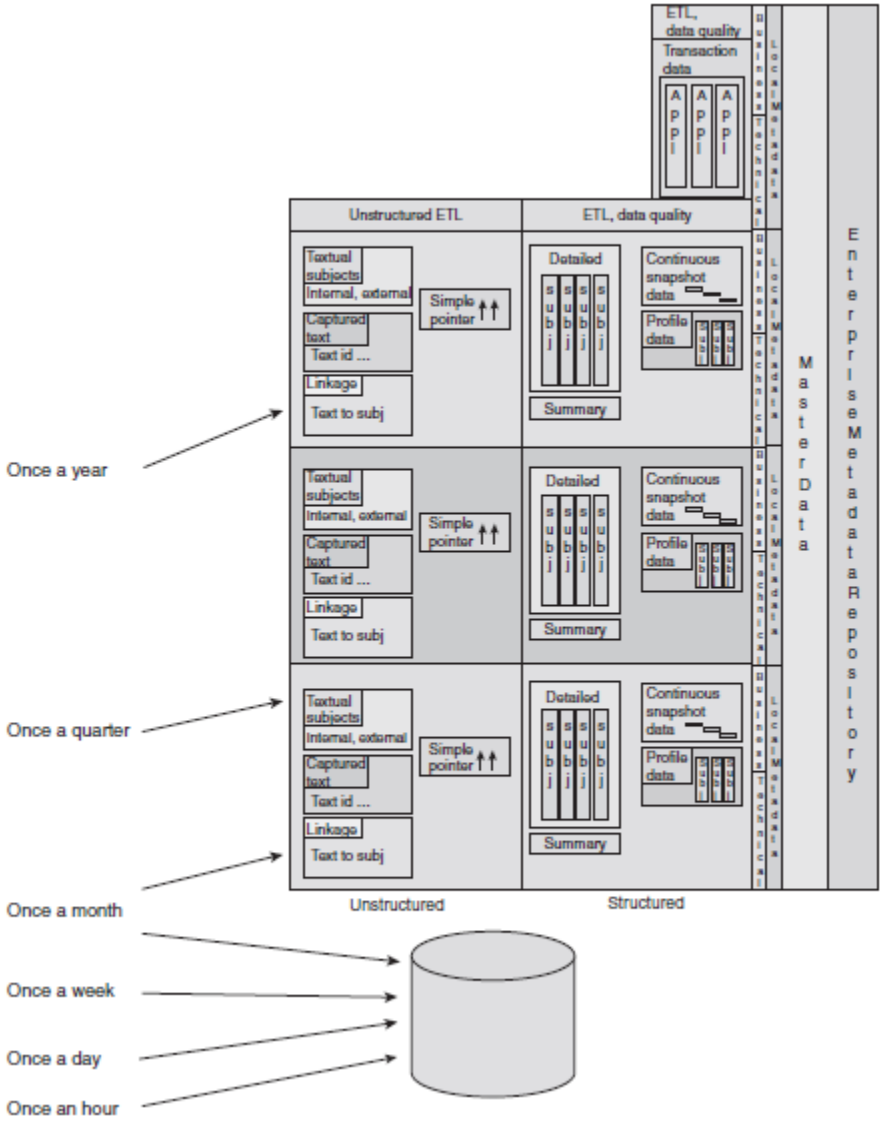


FIGURE 2.3.7 The frequency with which statistical analysis is done determines whether a separate exploration facility is needed.

Figure 2.3.7 suggests that as the frequency of statistical analysis grows, the need for a separate exploration facility increases. If only one statistical analysis is done per year, the basic DW 2.0 infrastructure alone is able to support that level of processing. If statistical analysis is done once a quarter, then the DW 2.0 infrastructure can probably support that level of processing. If statistical analysis is to be done once a month, then maybe the DW 2.0 infrastructure can support that level of processing. But anything more frequent than once a month is unlikely to be able to be supported by the DW 2.0 infrastructure without some addition or enhancement.

And certainly in organizations in which statistical analysis is done once an hour, a separate exploration facility needs to be included in the data warehouse environment.

2.3.9 Exploration facility

The exploration facility is a place where statistical processing can be done with no performance impact on the central DW 2.0 infrastructure. The exploration facility is located in a place that is physically apart from the DW 2.0 environment. They are physically separate places.

The exploration facility can be frozen for periods of time if necessary. It can include external data if warranted. The exploration facility typically contains subsets of data taken from the DW 2.0 environment. Only rarely is it a direct copy of the DW 2.0 environment, or even parts of the DW 2.0 environment. The exploration facility takes in data at its lowest level of granularity. In addition, it typically takes in huge amounts of historical data. Both detail and history are needed to satisfy the requirements of the exploration analyst. The data structure of the exploration facility is mixed. Some data is put in tables on disk storage. Still other data is flattened into a file.

The flattened file is often optimal for doing statistical analysis. Exploration facilities usually contain a large volume of fairly homogeneous data. There is often a low degree of data diversity and a large number of data records found in an exploration facility.

2.3.10 Sources for exploration processing

The exploration facility can draw from many sources of data—from the Integrated Sector, the Archival Sector, and the Near Line Sector.

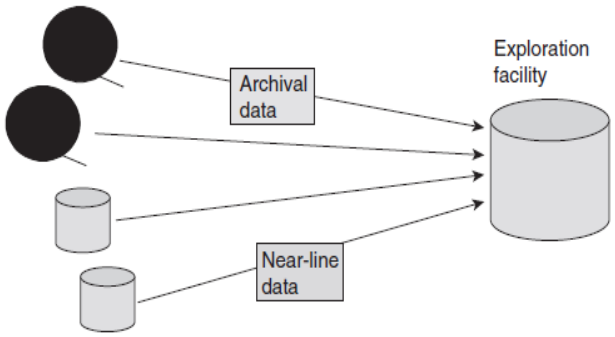


FIGURE 2.3.8 Archival data and near-line data can also be sent to the exploration facility.

Figure 2.3.8 shows that the exploration facility can draw from the Archival and Near Line Sectors. However, the DW 2.0 Integrated Sector is generally the primary source of data for the exploration facility. On occasion the exploration facility can draw data from the Interactive Sector. However, if it does draw data from that sector, then there are several cautions, the first being the disruption of service in the Interactive Sector. If exploration data is to be drawn from the Interactive Sector, it must be taken with great attention to the performance levels in that sector. The second caution is that if data is taken from the Interactive Sector into the exploration facility, it must be understood that the data is not auditable. For example, a unit of data taken

from the Interactive Sector at 10:31 AM may not exist at 10:32 AM. If these cautions are considered, then the Interactive Sector can be used to provide data to the exploration facility.

2.3.11 Refreshing exploration data

Figure 2.3.9 depicts the refreshment of exploration facility data with data from the DW 2.0 environment. The refreshment cycle of data into the exploration facility is deliberate. In other parts of DW 2.0 data flows as rapidly as it is available. Data flows into the exploration facility only when the exploration analyst wants the data to flow. This may be daily, weekly, or monthly, depending on the needs of the exploration analyst.

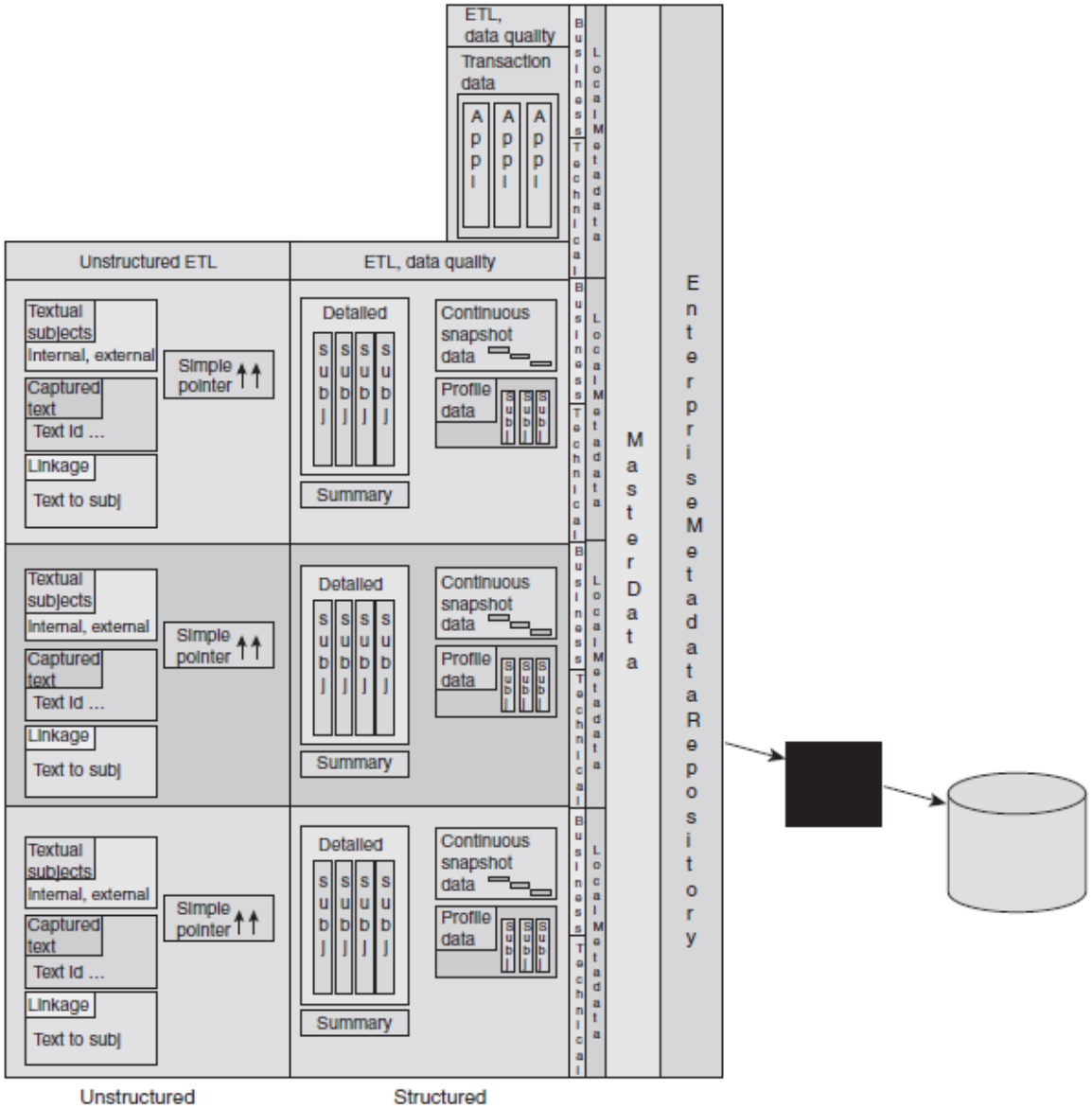


FIGURE 2.3.9 Periodically data is refreshed from the DW 2.0 environment to the exploration facility

2.3.12 Project based data

Exploration facilities are project based as a rule. This usually means that a particular issue needs to be studied by management. Data relevant to the issue is gathered, an analysis is made,

and then the results are sent to management. Once the results are sent to management and the study is finished, the data is either discarded or mothballed. The project based exploration facility is not a permanent structure.

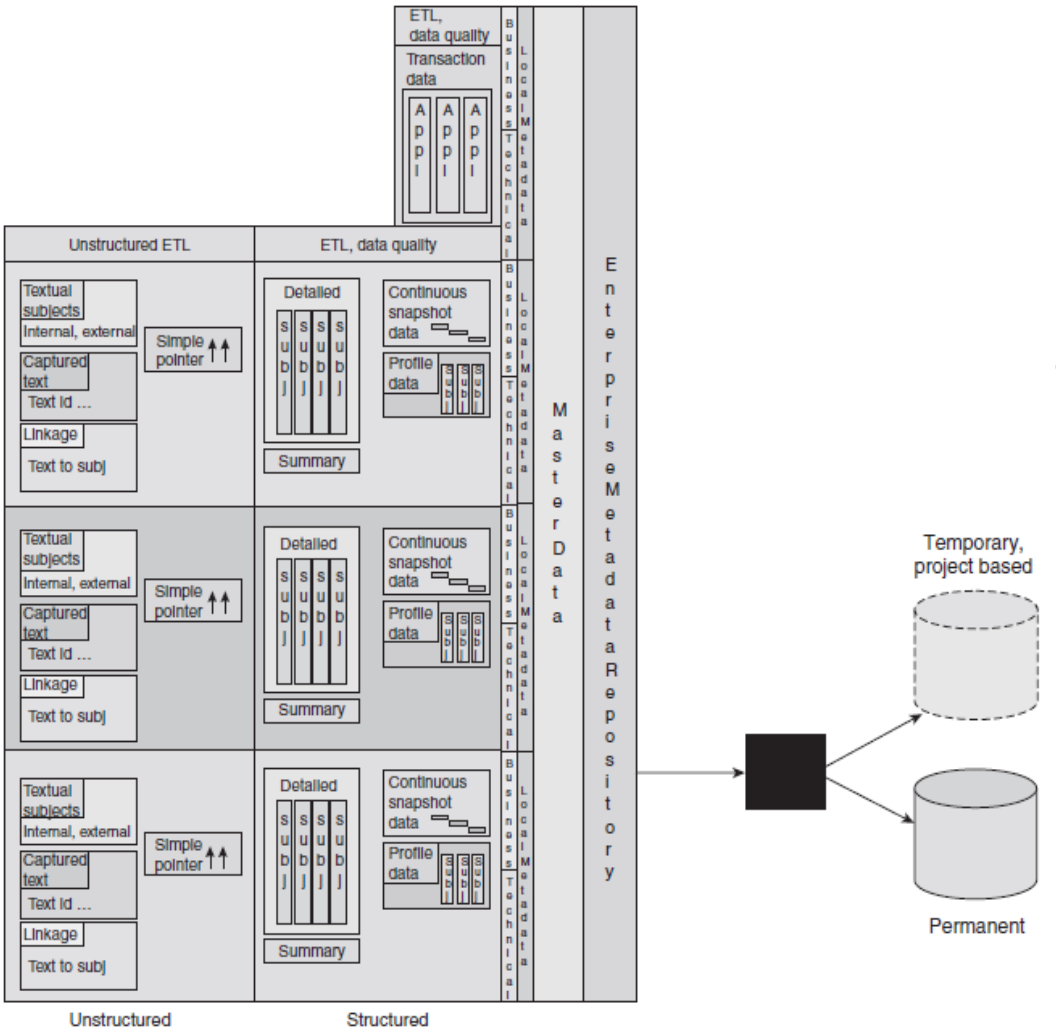


FIGURE 2.3.10 The exploration facility can be either a permanent structure or a temporary structure.

However, some organizations elect to have a permanent exploration facility. When this is the case, the exploration facility is available any time an analysis needs to be done. The detailed data is often changed inside the permanent exploration facility. Figure 2.3.10 illustrates that the data in an exploration facility can be permanent or temporary.

2.3.13 Data marts and exploration facility

Analysts often think that the exploration facility is the same as or very similar to a data mart. In fact, a data mart and an exploration facility are very different. Some of the major differences between a data mart and an exploration facility are:

■ The exploration facility holds detailed data; the data mart holds summarized or aggregated data.
■ The exploration facility is used for the purpose of discovery; the data mart is used for the purpose of easy dissemination of information.
■ The exploration facility attracts mathematicians; the data mart attracts business analysts.
■ The exploration facility is usually a fl at file; the data mart is OLAP based.
■ The exploration facility may be and often is temporary; the data mart is almost always permanent.
■ The exploration facility is acted on by statistical software; the data mart is acted on by business intelligence software.

These are only the primary differences between an exploration facility and a data mart. Figure 2.3.11 suggests that an exploration facility and a data mart are very different components of the architecture.

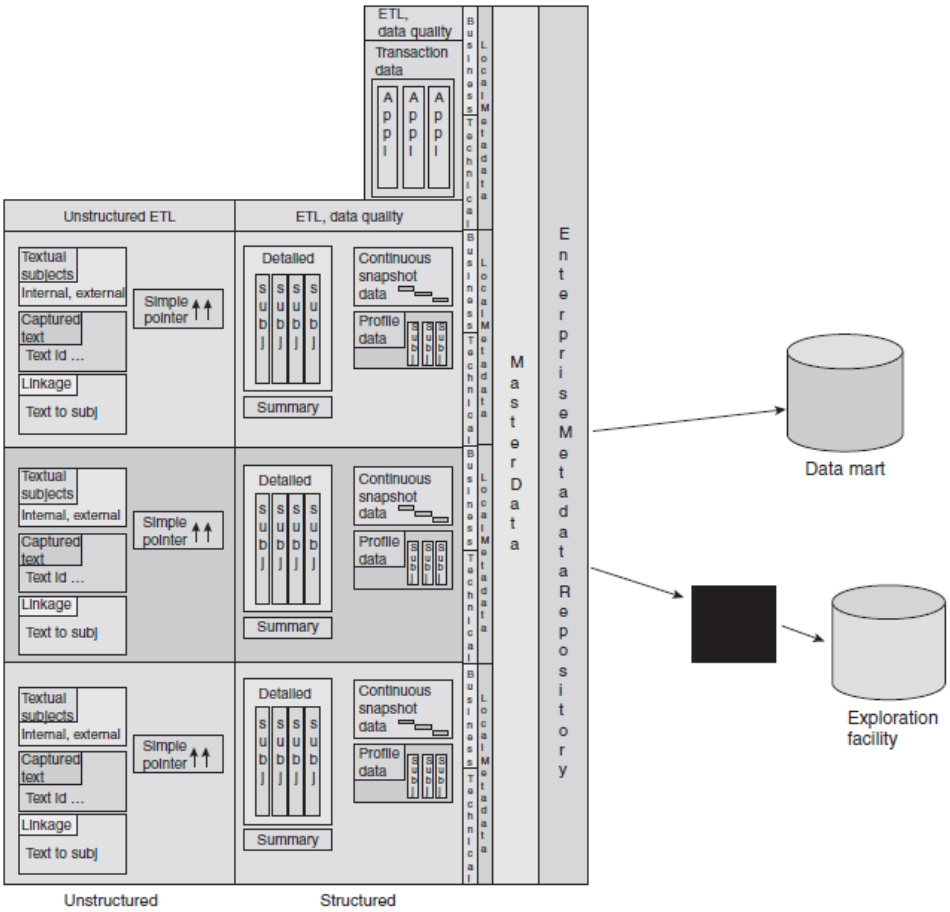


FIGURE 2.3.11 There are significant differences between a data mart and an exploration facility.

2.3.14 A backflow of data

from the exploration facility back into the DW 2.0 environment. Indeed it is permissible for data to flow from the exploration facility into the DW 2.0 environment, but there are certain conditions that must be met beforehand. Some of the conditions are:

■ The data that is output from the exploration facility needs to be used in multiple places throughout the corporate environment. If the output data is to be used in only one or two places, then it does not make sense to place it in the DW 2.0 environment.

■ There needs to be an audit trail of data and calculations associated with any exploration facility data placed in the DW 2.0 environment

If exploration facility data is to be placed in the DW 2.0 environment, and if the exploration facility is a project-based facility, then the expectation for the exploration data to be placed in the DW 2.0 environment must also be that it is a limited, one-time-only supply of data. In other words, if data is to be placed in the DW 2.0 environment from a temporary source, it is not reasonable to expect that source to become a permanent supplier of data to the DW 2.0 data warehouse.

Figure 2.3.12 depicts the potential feedback of data from the exploration facility to the DW 2.0 data warehouse, under the right circumstances.

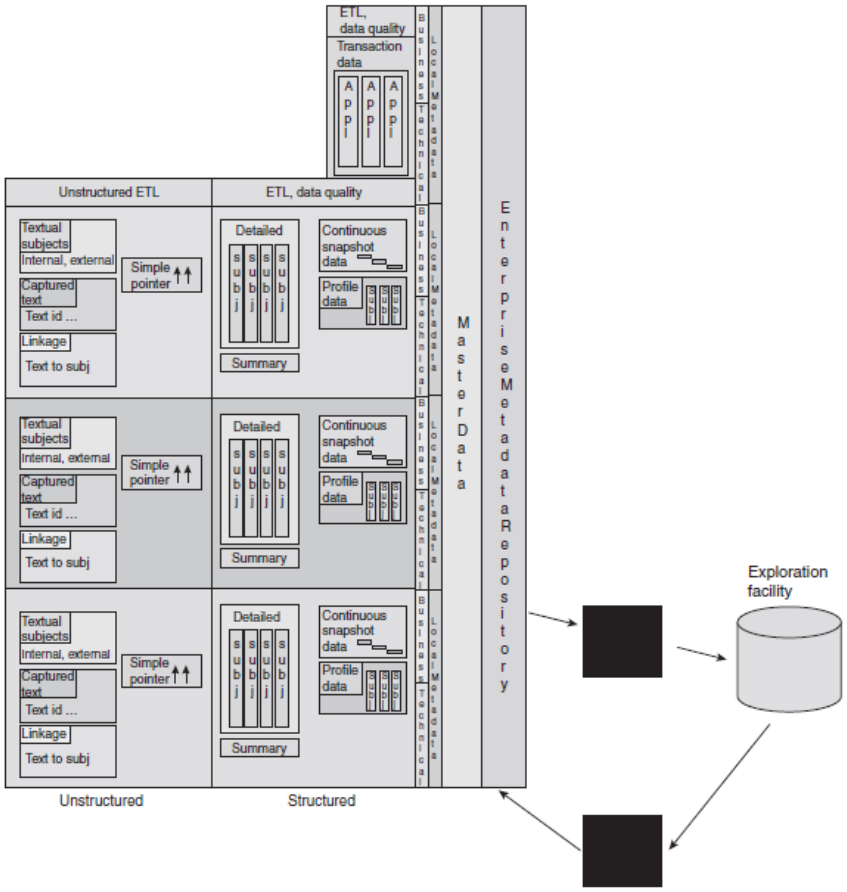


FIGURE 2.3.12 It is possible to have data flow from the exploration facility into DW 2.0.

2.3.15 Using exploration data internally

A word of caution about exploration facilities. In most cases, exploration facilities are used to produce analysis for internal consumption only. This is because the data that goes into the exploration facility normally is not subjected to the same rigorous ETL processing as the data flowing into and through the DW 2.0 environment. Therefore, when it comes to providing reports and data to auditors and examiners, it just does not make sense to use data from the exploration facility. Instead, only "official" data should be used as the basis for official reports. It is well worth remembering that information on reports often ends up also appearing in financial statements or even in the press. It is very unwise to allow reports based on exploration facility data to be used in a public manner, because these reports may not be
Calculated properly and may contain misleading data.

2.3.16 perspective of business analyst

Many businesses do not take advantage of the statistical processing of the data that they own. As such they are not taking full advantage of their information resources. But there are some businesses that have long recognized the value of information and the statistical processing that can be done with the data. Typical of these businesses are insurance actuaries and research engineers. The truth of the matter is that, in insurance and in engineering product development, statistical processing plays a very important role. Furthermore that role is recognized by the businesses that employ these types of people. For a company that traditionally has not used statistical analysis widely to start to do so, there needs to be some obvious successes. These successes do not happen magically. Usually these statistical analyses happen as the result of "skunk-work" projects. Because there has been no past success, the business is loath to fund a statistical analysis. So one or two interested and experimental workers get together and do a "see what happens" project. These projects are almost always small because they are not formally funded. Assuming that the analysts find something interesting and useful, the skunk-work project then makes its way into the corporate mainstream. Once established, the statistical analysis function grows thereafter.
The type of individual who conducts these types of projects must have some sort of mathematical background. The background may be formal or informal, but to do statistical analysis properly requires a way of thinking that entails an understanding of mathematics.

2.3.17 LET US SUM UP

There are two types of queries—analytical queries and exploration queries. Exploration queries access large amounts of data and can take a long time. Exploration queries require granular and historical data. Exploration processing typically makes use of statistical techniques. Occasionally exploration data needs to be frozen. The freezing of exploration data occurs when heuristic processing is being done. In heuristic processing, the next step of analysis depends entirely on the results obtained from the most current level of analysis.

An exploration facility can be built exclusively for the purpose of supporting exploration processing. Whether or not there is a need for an exploration facility depends entirely on the frequency of statistical analysis. If only infrequent statistical analysis is occurring, then there is no need for a separate exploration facility. If there is frequent statistical analysis that occurs, then there may be a need for a separate exploration facility. Exploration facilities are built on a project-by-project basis. There is no need to keep the exploration facility after the project that required it is completed.

2.3.18 List of References

- DW2.0 The architecture for Next Generation of Datawarehousing W.H. Inmon, Derek Strauss, Genia Neushloss, ELSEVIER.
- Building the data warehouse, W.H.Inmon, third Edition, Wiley.
- Datawarehousing, S. Mohanty, TMH .
- The Data Warehouse Lifecycle toolkit", Ralph Kimball ,John Wiley.

2.3.19 Unit End Exercises

1) What is Exploration facility? List its sources of data.
2) Write difference between a data mart and an exploration facility?
3) Define data marts and exploration facility.
4) What is project based data?

UNIT-III: Data models and DW

Unit Structure

3.1.0 OBJECTIVES

In this chapter you will learn about:

- Data model and how it will be shaping business.
- Levels of data model-ERD, midlevel, low level etc.
- Corporate data model and transformation of models.
- Data models that can be applied to understand data.

3.1.1 INTRODUCTION

The world of DW 2.0 is a complex one. There are many aspects and many facets. It is very easy to become entwined in details and quickly lose your direction. It is important to keep one's perspective when dealing with DW 2.0.

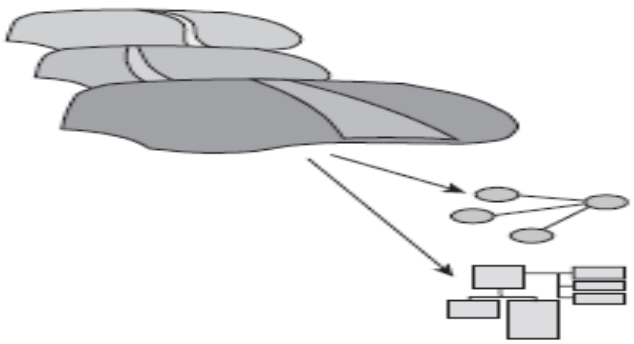


Figure 3.1.1 A data model serves as an intellectual road map

There are many reasons for a road map in DW 2.0. But perhaps the best reason is that DW 2.0 is not built all at once. Instead DW 2.0 is built a step at a time, over a long period of time. In addition DW 2.0 is built by many people, not just a single person. To coordinate the efforts of these people over time and across different audiences, it is mandatory that there be a road map a data model that describes how the different parts of DW 2.0 fit together. Without a data model the DW 2.0 development efforts are scattered and disjointed, resulting in a mess.

3.1.2 The data model and business

The data model is built from the business itself. It mimics the different components of the business.

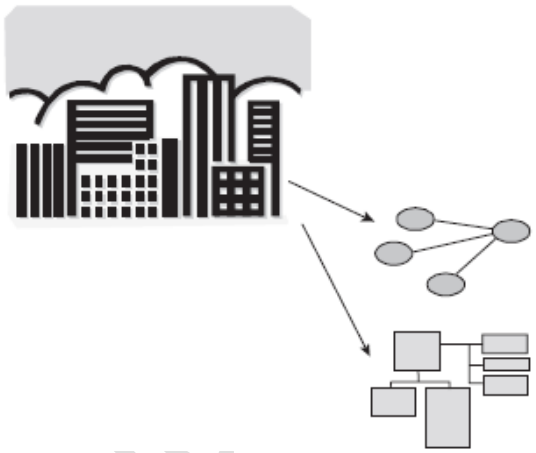


Figure 3.1.2 shows that the data model is built from the business

3.1.3 The scope of integration

The first step in the building of the data model is the definition of the scope of integration. The scope of integration is the statement of what is to be in the data model and what is not to be in it. The scope of integration is necessary because without it the data model can go on endlessly. It can be extended to include the universe. And when the data model includes the universe, the model is never finished.

Figure 3.1.3 The first step in building the data model is to define the scope of integration.

Figure 3.1.3 shows that the definition of the scope of integration is the starting point for the data model. The data model is based on the data found in the corporation. And in most organizations there is ample data. If the analyst is not careful, the data model will go on endlessly even when the scope of integration is defined, unless there is a distinction made between granular data and summarized or aggregated data. Granular data is data at its lowest level of meaning. A person's name is granular. A person's date of birth is granular. The salary of a person at a moment in time is granular.

3.1.4 Making the distinction between granular and summarized data

The reasons there needs to be a distinction made between granular data and summarized data are that

■ there is much more summarized data than granular data;
■ summarized data changes faster than it can be modeled;
■ summarized data carries with it an algorithm describing how the summarization is to be made. If summarized or aggregated data is included in the data model, the model will never be finished. Figure 3.1.4 shows that granular data is the fiber of the data model.

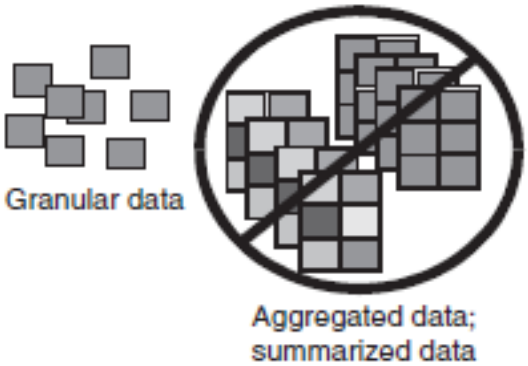


Figure 3.1.4 shows that granular data is the fiber of the data model.

3.1.5 Levels of the data model

There are different levels of the data model. In a standard data model there are
■ The ERD—entity relationship level—the highest level of the data model;
■ The midlevel mode—the dis, or data item set;
■ The low-level model—the physical model—the lowest level of data modeling.

The ERD is at a high level and can be constructed quickly. The ERD shows the major subject areas of the business of the corporation, and the relationships between those subject areas. The midlevel model— the data item set—shows the keys, attributes, and relationships of the details of the data model. The low-level model shows the physical characteristics of the data model, such as physical attributes of data, indexes, foreign keys, and the like. The lower the level of the model, the greater the level of detail. The higher the level of the model, the more complete the model. Figure 3.1.5 shows the different levels of the data model. The fact that there are multiple levels of modeling for something as complex as DW 2.0 is not a new or strange technique. Consider maps of the world, as seen in Figure 3.1.6

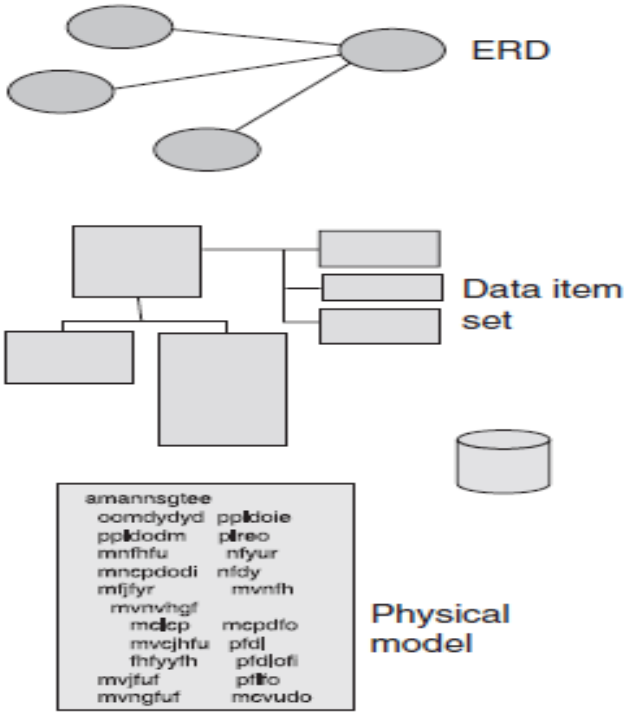


FIGURE 3.1.5 There are different levels of the data model

In Figure 3.1.6 it is seen that there is a map of the United States, a map of Texas, and map of how to get to a house in Dallas, Texas. Each map has a relationship to each of the other maps. Texas can be found inside the United States. Dallas can be found inside Texas. So there is a relationship between each of the maps. There is a different level of detail found on each map. The U.S. interstate highway system is found on the map of the United States. Texas 285 is found on the map of Texas. And Grapevine road in Denton is found in the city map of the Dallas vicinity. So the level of detail goes

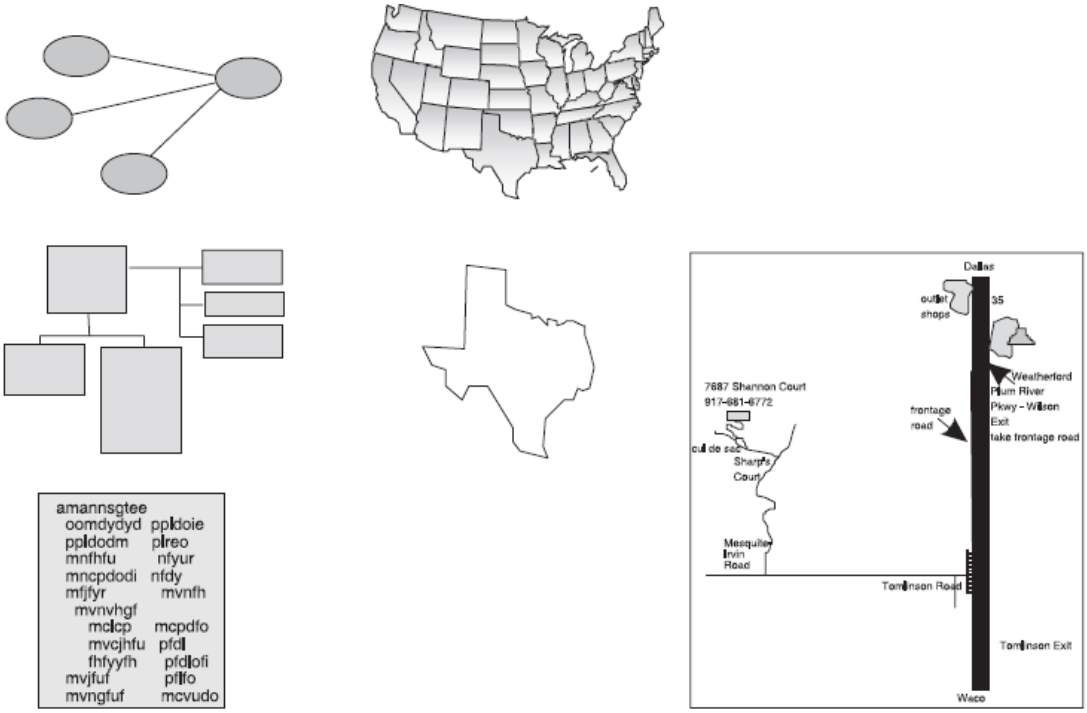


FIGURE 3.1.6 How the different levels of the data model relate.

In the same fashion, the data model is knitted together so that there is meaning and order to all of the systems that constitute the DW 2.0 environment. Figure 3.1.7 shows the data model and the order it brings to information systems.

There are many different models that are found in the DW 2.0 environment. It is a mistake to think that there is one model for DW 2.0.

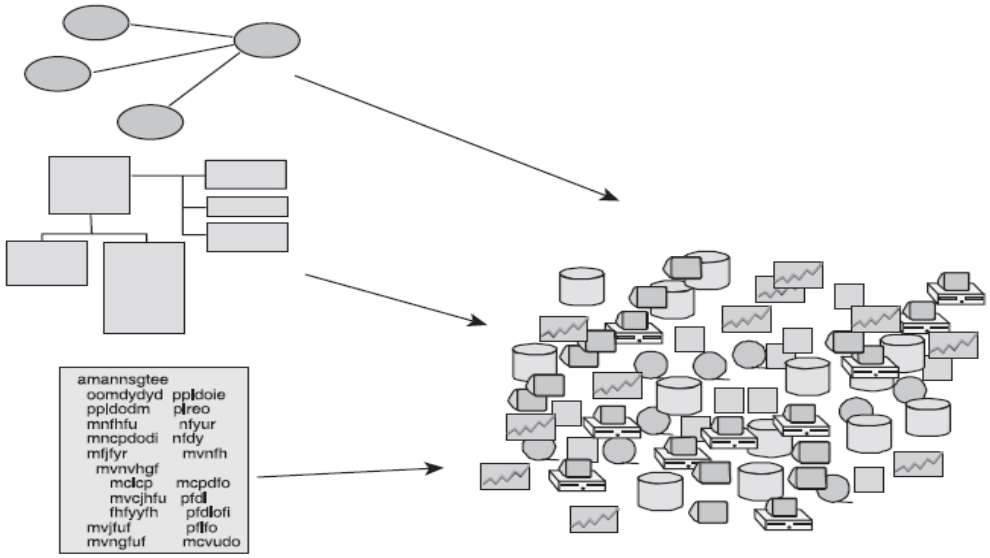


FIGURE 3.1.7 The data model is used to bring order out of chaos.

3.1.6 Data models and interactive sector

The first set of models is found in the applications that are in the Interactive Sector. As a rule there is a separate data model for each application. The application data models are shaped by the application requirements. One major consideration of the application data models is the need for performance. Throughout the application environment, especially where there are OLTP transactions, the data models are shaped by the need for performance. When performance considerations are applied to a data model, the result is a streamlined data model, in which high performance is achieved by a streamlined flow through the system. And a streamlined flow though the system is abetted by a model that positions data together. Figure 3.1.8 depicts applications and their models. In Figure 3.18 it is noteworthy that there are different application models for each application

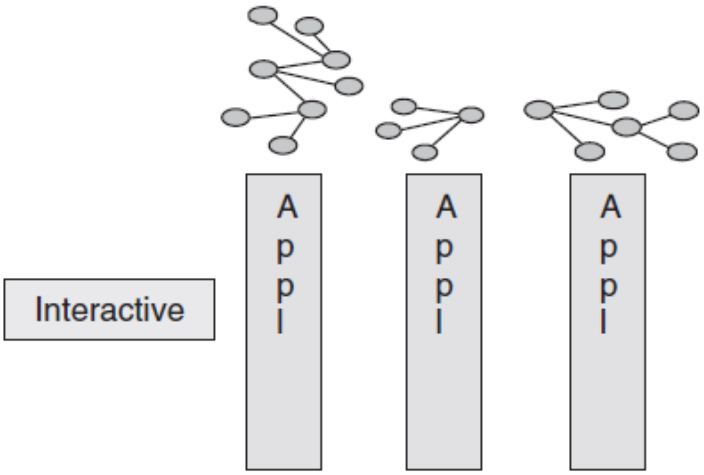


FIGURE 3.1.8 An application data model shapes each of the applications.

3.1.7 The corporate data model

The corporate data model sits between the interactive layer and the integrated layer. It is *the* data model of the DW 2.0 environment, if there is such a thing. The corporate data model depicts all the data of the corporation in an integrated manner. As a simple example of the need for integration at the corporate level, suppose there are three applications. Application A has data at the daily level, where dollars are U.S. dollars, and the gender of people is M/F. Application B has information stored at the weekly level, where dollars are Canadian, and the gender of people is stored as MALE/ FEMALE. Application C has data stored by the hour, where dollars are stored in Australian dollars, and where gender is designated as X/Y. The corporate view is at the daily level, where dollars are stored in euros, and where gender is specified as MEN/WOMEN. The data model reflects the true corporate view of information, which is a unified view of information.

3.1.9 A transformation of models

Figure 3.1.9 shows that a fundamental transformation of data is made as data passes from the Application/Interactive Sector to the Integrated Sector. It is noteworthy that as data passes into the Integrated Sector it is stored by subject area. As data passes into the Near Line Sector, there is no transformation or change of data models. Because the near-line environment needs to mimic the interactive environment as much as possible, the data model for the near-line environment is exactly the same as the data model for the Interactive Sector. Figure 3.1.10 shows that the data model does not change as data passes into the Near Line Sector.

Finally data passes into the Archival Sector. As it does so, there may or may not be a change of data models. In some cases data passes into the Archival Sector in exactly the same state as it was in the Integrated Sector. In this case there is no change of data models. But on other occasions, data undergoes a fundamental change as it passes into the Archival Sector.

In this case data passes into what can be termed an inverted list format. When data passes into an inverted list format, it is fundamentally rearranged into a series of simple lists. The archival analyst may want such a transformation, because it can make the data in the archival environment easier to find and analyze. And of course, data in the archival environment can be placed in *both* a corporate data model format and an inverted list format. Figure 8.11 shows this movement of data into the archival environment.

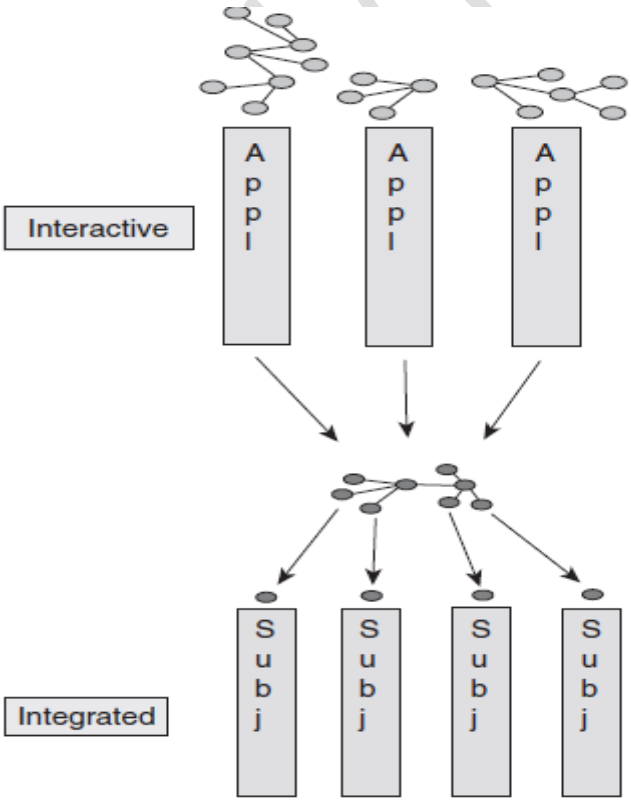


FIGURE 3.1.9 A fundamental transformation of data is made as data passes from application data to corporate data. The corporate data model is used to guide the transformation.

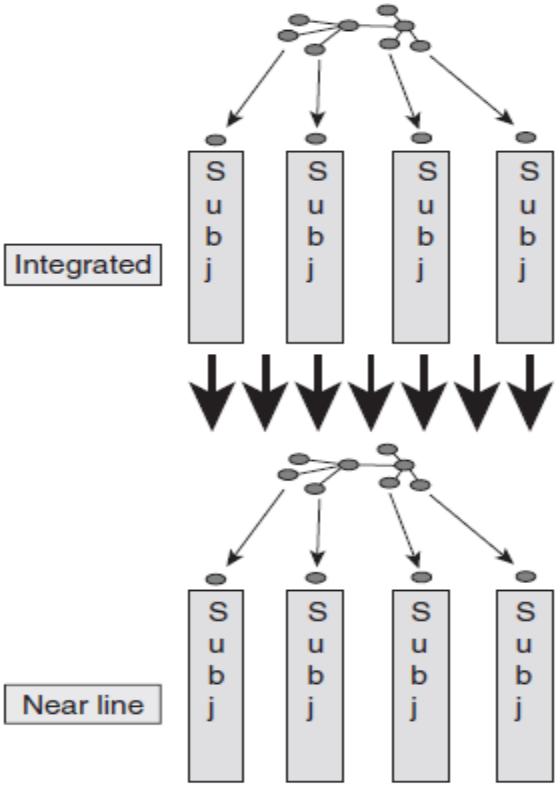


FIGURE 3.1.10 As data passes from the Integrated Sector to the Near Line Sector, no changes are made at all to the data model.

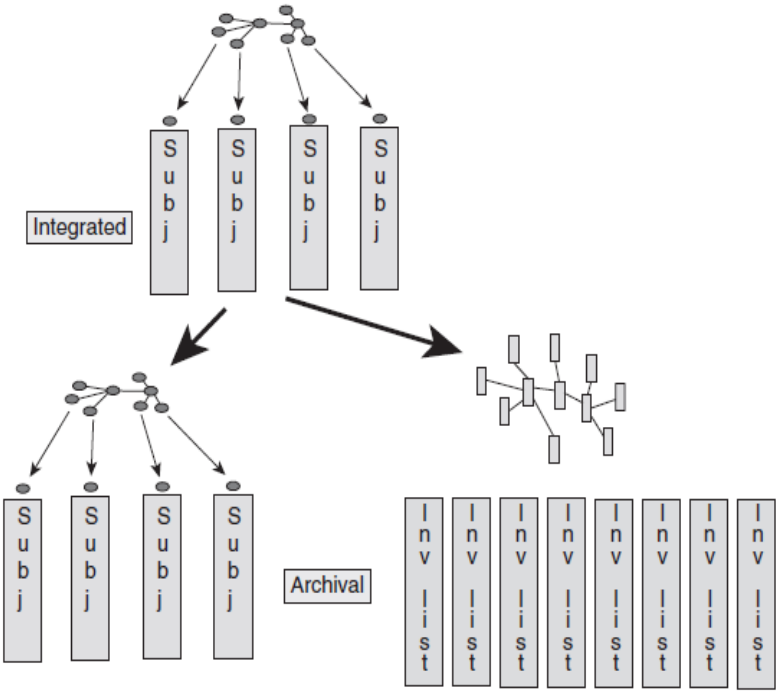


FIGURE 3.1.11 As data passes into the Archival Sector, the data model may be preserved, or the data may be transformed into an inverted list data model, or both.

3.1.10 Data models and unstructured data

Data models are appropriate for and useful to the structured side of DW 2.0. But there is some applicability of data models to the unstructured side of DW 2.0. It is not surprising that the data model does not play as big a role in the unstructured world as it does in the structured world.

The first place where a data model is found in the unstructured component of DW 2.0 is in terms of external taxonomies. External taxonomies are used to group and classify data during the process of terminology normalization or rationalization. Figure 3.1.12 shows that data models can be used to shape an external taxonomy.

The second place where data models can be found in the unstructured environment is in the creation of an internal data model. An internal data model is used to describe the contents of and the structure of a body of text, usually a large body of text. First the unstructured data is gathered. Then the unstructured data is organized into themes. From the themes an SOM (Self-Organizing Map) can be created. Once the SOM is created, the major themes of the unstructured text and the relationships between those themes are formed. From that basic information, an internal data model can be created.
Figure 3.1.13 shows the creation of an internal data model from the themes that occur in the body of unstructured text.

FIGURE 3.1.12 An external data model is used to shape the external taxonomies found in the unstructured environment.

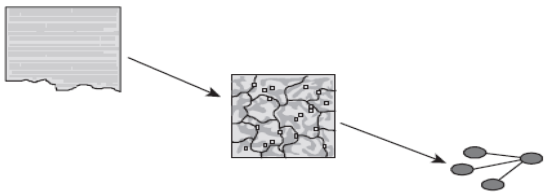


FIGURE 3.1.13 A document can be reduced to internal themes. In turn the internal themes can be used to create an internal data model.

3.1.11 From the perspective of business user

The business user is essential to the data modeling process. Indeed, from an intellectual standpoint, the data model is the embodiment of how the end user perceives the data found in DW 2.0. Stated differently, if the data model comes from any other source than the end user—or if the end user does not at least look at and acknowledge the validity of the data model—then the contents of DW 2.0 are shaped inappropriately.

The implication here is that the end user needs to be involved from the beginning, because it is at the beginning of the building of DW 2.0 that the data model is built. Building the data model at the beginning and then showing the data model to the end user at a later point in time risks having major aspects of DW 2.0 built incorrectly.

The end user does not have to become an expert in data modeling techniques. (Ironically some end users become enamored of the modeling process and do—in fact—become experts in data modeling.) Instead the data modeling process is usually handled by an outsider who is a professional in data modeling. Over time the business model will change. The business user is as involved in the changes to the data model as he/she was in the original creation of the business model.

3.1.12 LET US SUM UP

Thus, we have studied data models form the intellectual road map for the DW 2.0 environment. DW 2.0 is large and complex and will be built over a period of time by a large number of developers. It is the data model that allows one development effort to be connected to another development effort. The data model is shaped from the business requirements of the corporation. It is built for the most granular of data, not summarized data or aggregated data. There are three levels of the data model—the ERD level, the midlevel, and the low level. The Interactive Sector is shaped by an application model. The Integrated Sector is shaped by the corporate data model.

3.1.13 List of References

- DW2.0 The architecture for Next Generation of Datawarehousing W.H. Inmon, Derek Strauss, Genia Neushloss, ELSEVIER.
- Building the data warehouse, W.H.Inmon, third Edition, Wiley.
- Datawarehousing, S. Mohanty, TMH .
- The Data Warehouse Lifecycle toolkit", Ralph Kimball ,John Wiley.

3.1.14 Unit End Exercises

1) What is a granular data?
2) Explain different levels of data model with diagram.
3) Explain corporate data model.
4) Write a note on transformation of data model across different sectors.

UNIT-III: Monitoring the DW environment

Unit Structure

3.2.0 OBJECTIVES

In this chapter you will learn about:

- Three types of monitor are needed in DW environment:-Transaction, Data quality, and Data ware house monitor
- The transaction monitor is especially concerned with data warehouse.
- Data quality monitor.
- Data warehouse monitor is focused on integrated sector.

3.2.1 INTRODUCTION

The DW 2.0 environment is complex and dynamic. There are many complex interactions between the various components. Data flows from one component to another, transactions are executed, and transformations of data are done.

In many ways the DW 2.0 environment is like a black box. Data is put in one place and taken out of another, and mysterious things happen in between. Unfortunately, if the DW 2.0 environment is treated like an opaque black box, it is a good bet that over time, things will happen inside the black box that are untoward—data starts to collect where it shouldn't, transaction response time turns bad, data is not placed where it ought to be placed, or worse.

The DW 2.0 environment should not be like a black box. There needs to be periodic "checks underneath the hood" to ensure that the DW 2.0 environment is operating as it should be.

3.2.2 Monitoring DW environment

To that end it is strongly recommended that regular monitoring of the DW 2.0 environment be conducted. At the very least, a stethoscope should be inserted into the black box to find out what is going on. When adjustments need to be made to the DW 2.0 environment or any of its components, those adjustments can be made proactively rather than reactively.

3.2.3 The transaction monitor

There are at least three types of monitoring that need to occur in the DW 2.0 environment. The first is monitoring of the transactions that are run in the Interactive Sector of DW 2.0. A transaction monitor ensures that there is good and consistent response time.

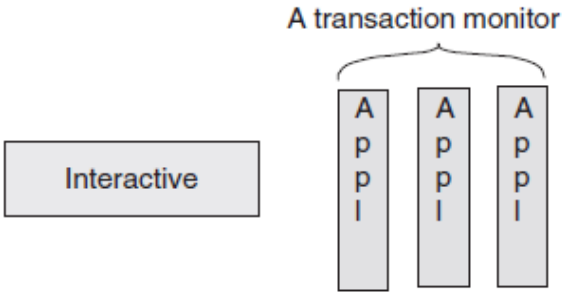


FIGURE 3.2.1 A transaction monitor is one form of monitoring found in the DW 2.0 environment

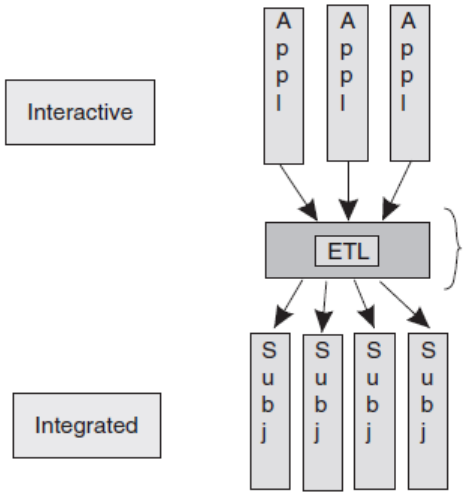


FIGURE 3.2.2 A data quality monitor at the moment of ETL is also found in the DW 2.0 environment.

3.2.4 Monitoring data quality

The second type of monitoring that needs to occur in the DW 2.0 environment is ETL monitoring for data quality. There needs to be a monitor that is dedicated to verifying the quality of data passing through the transformation components of DW 2.0. If low-quality data is being passed into DW 2.0, then the data analyst needs to be alerted, at the very least.

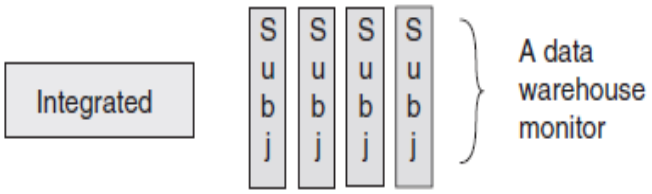


FIGURE 3.2.3 A data warehouse monitor is an essential component of DW 2.0.

3.2.5 A data warehouse monitor

The third type of monitor that needs to be a part of the DW 2.0 environment is a data warehouse monitor. This monitor looks at the data in the data warehouse. While the data warehouse monitor serves many different purposes, its main purpose is to measure the usage frequency of data. From the usage frequency of data, it can be determined if any data has gone dormant.

The management of dormant data is one of the most important aspects of the management of the DW 2.0 environment. Figure 3.2.3 illustrates the data warehouse monitor. Each of these types of monitors for the DW 2.0 environment will be addressed in greater depth.

3.2.6 The transaction monitor

The primary purpose of the transaction monitor is to ensure that there is good, consistent response time. Unfortunately, there are many aspects to system processing that have an effect on system performance. When system performance is mentioned, it is usually a reference to response time. Response time in the 2- to 3-second range is normally considered acceptable. There may be a few periods during the course of the day when response time starts to drift higher, but as long as those periods are short and infrequent, and as long as the response times do not drift too high, then the system will be deemed to be running in a satisfactory manner. Usually acceptable response time parameters are defined in a Service Level Agreement.

Some of the characteristics and features of a transaction monitor include:

■ **Transaction queue monitoring:** The transaction queue is the place where transactions are stored prior to execution. When the system becomes very busy, transactions can become stalled in the transaction queue awaiting execution. If the system becomes really busy, this wait in the transaction queue can become the single largest obstacle to performance.

■ **Application monitoring:** Applications that process transactions inside the computer need to be monitored. When a transaction goes into execution, it occupies system resources. The length of time those resources are dedicated to running the code that constitutes the transaction being executed is the single most important measurement of system throughput and performance.

■ **Transaction record monitoring:** The number of records needed to complete a transaction also impacts system performance. Often many resources are consumed by a single business transaction. But the most revealing indicator of transaction processing performance is the number of records the transaction needs for execution. Simply stated, a transaction that requires a few records will execute much more quickly than a transaction that entails processing many records.

There are other measurements of performance, but these measurements are the most important. Figure 3.2.4 illustrates some of the components to which transaction monitors can be applied for maximum benefit.

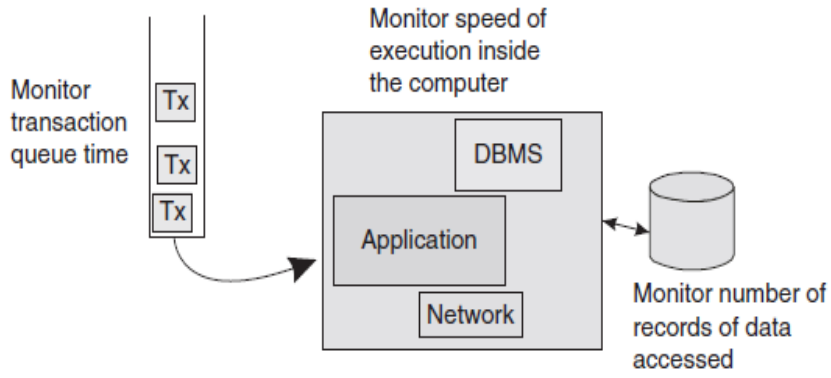


FIGURE 3.2.4 The basic activities of a transaction monitor.

3.2.7 Peak period processing

One important metric that comes from the monitoring of transactions is the measurement of how close the system comes to having all of its resources consumed during peak-period processing.

There are periods of low activity and periods of high activity in every transaction processing environment. The periods of high activity are called the " peak periods. " The system operates smoothly as long as there is capacity for all processing. But during peak periods, if the system's demand for resources starts to exceed the resources that are available, then the system starts to slow down, in most cases dramatically.

Therefore, it behooves every organization to monitor how closely peak-period processing comes to exhausting the available system resources. If the resources used in peak periods are steady, then there is no need to add capacity. If or when there is a steady increase in the resources needed for peak period processing, that is a warning sign that more system resources

are needed, or a different allocation of resources is called for. Figure 3.2.5 shows the tracking of peak-period resources.

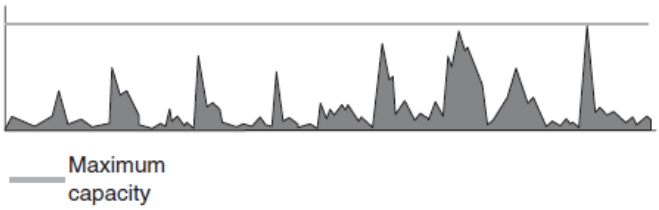
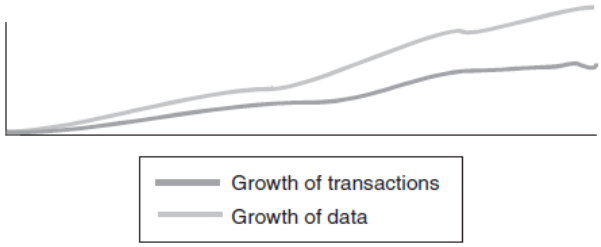


FIGURE 3.2.5 The burning question—When will capacity used reach maximum capacity and what will happen when that point is reached?

Another important parameter typically tracked by the transaction monitor is the rate of growth of the system. Typical indicators of system growth that can be tracked over time are the number of transactions and the volume of data in the system. The number of transactions is a good indicator of the rate at which a system is growing and its capacity is being consumed. By extrapolating and projecting the number of transactions a system processes, the systems analyst can determine when a hardware upgrade will be needed.

The objective is to predict when an upgrade will be needed and to enable the organization to respond in a proactive manner before performance problems begin. Operating in a reactive manner invariably means that the organization will suffer periodic " meltdowns. " Meltdowns cause an untold amount of grief due to their negative impact on the operations of the company. Figure 3.2.6 illustrates the typical result of tracking transaction volume and data quantity growth over time.



There are many other aspects of the operations of transaction processing that can be monitored. For example, the juxtaposition of transaction volume growth and associated response time degradation over time can reveal and forecast when it will become critical for the organization to increase its transaction processing capacity. Figure 3.2.7 depicts this kind of comparative performance measurement.

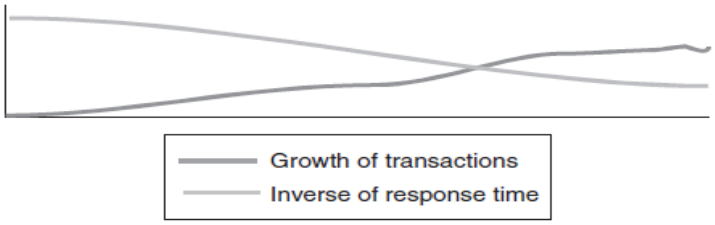


FIGURE 3.2.7 Tracking the growth of transactions matched against the inverse of response time.

3.2.8 The ETL data quality monitor

The ETL data quality monitor examines data as it passes from one DW 2.0 sector to another or as the data initially enters the system. The purpose of the ETL data quality monitor is to evaluate the quality of data as it is being transformed. The ETL data quality monitor looks at many aspects of data. Among other things, it examines

■ Domain of data: Suppose gender is defined as " M/F. " If gender data is entered as " MALE, " the ETL data quality monitor registers a discrepancy.

■ Unmatched foreign key: For example, a foreign key is presumed missing and unmatched if there is a reference to " John Jones " in the data, but the customer data base has no John Jones. Outlying range: The normal age of a customer is between 15 and 80. If a customer comes into the system attributed with an age of 234 years, there is obviously an age range data quality problem that needs to be detected and reported.

■ Null values: Every data key that is specified needs to be present. If a record is passed to the data warehouse with no key, it must be detected and reported.

■ Mangled character: A name spelled "Mar[_ _ *] " could find its way into an ETL routine. Unless the person has a really unusual name, like the Artist Formerly Known as Prince, there is a good chance that there is a quality discrepancy in this data.

These are just a few examples of many data quality conditions that need to be detected and reported by an ETL data quality monitor. One of the most interesting data quality questions is what to do once an error condition has been detected. One option is to reject the data, but this is generally a bad solution because

■ Other parts of the record are rejected when they may be perfectly fine;
■ Some means of correction will be needed. Manual correction is absolutely the last choice, because manually correcting large amounts of incorrect data can take an enormous amount of time and significantly delay project progress. Another solution is to create default data. Although this works well, the data that has been determined to be incorrect is permanently lost to the system. Yet another solution is to let the bad data pass into the system and flag it as being incorrect. Flagging incorrect data warns the end user that there is a problem with the data. Figure 3.2.8 depicts the place and role of an ETL data quality monitor.

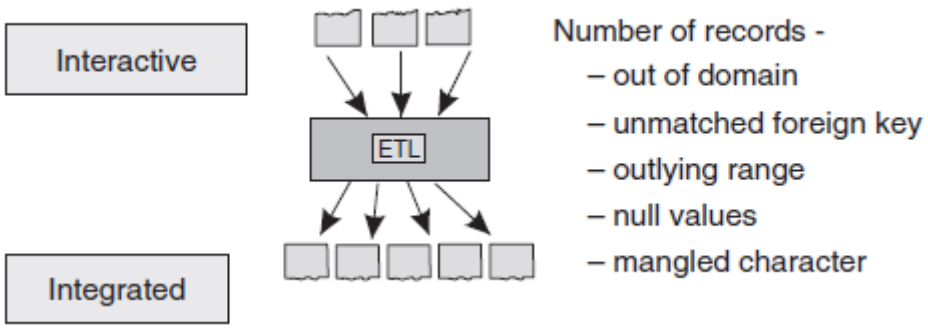


FIGURE 3.2.8 The data quality monitor.

3.2.9 The data warehouse monitor

The data warehouse monitor is a software tool that monitors what data in the data warehouse is being used and what is not being used. If a unit of data goes long enough without being used, it is considered "dormant" data. Good data warehouse monitors are designed to detect and report dormant data.

The normal way that data warehouse data monitoring is done is through the interception of SQL code submitted to the data warehouse system. By gathering the SQL passed into the system, the analyst can determine what data is being accessed inside the data warehouse and what data is not being accessed. Usually the SQL is intercepted as a result of "sniffing" the lines of communications.

One way to arrange a sniffer is by placing it outside the data warehouse computer. Figure 3.2.9 depicts a sniffer located outside the computer that manages the data warehouse. The other place the sniffer can be located is inside the computer where the data warehouse is being managed. Figure 9.10 depicts the placement of a sniffer inside the data warehouse computer.

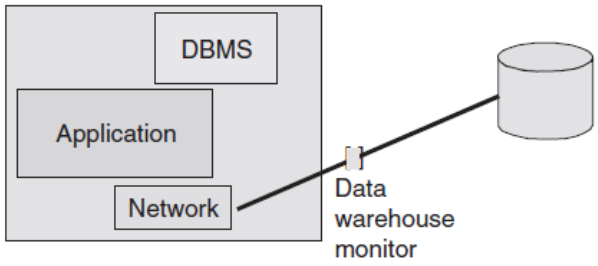


FIGURE 3.2.9 The data warehouse monitor is placed outside the computer on the network.

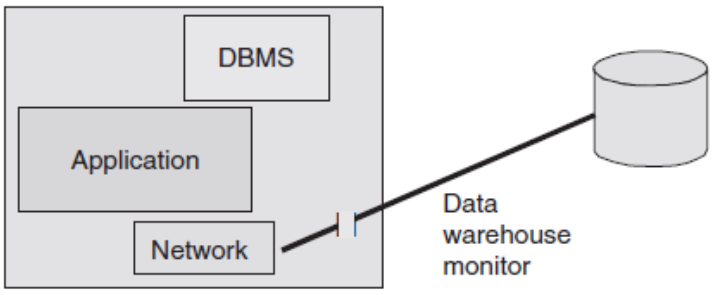


FIGURE 3.2.10 The data warehouse monitor is placed inside the computer

.

As a rule, it is much more efficient to sniff SQL code from outside the computer that hosts the data warehouse. The overhead of sniffing can become a large factor when the sniffer is allowed to be part of or interact directly with the data warehouse DBMS.

3.2.10 Dormant data

There are lots of reasons for having a dormant data monitor in the data warehouse. The primary reason is that when data goes dormant, it needs to be moved to alternate storage. Alternate storage is much less expensive than high-performance disk storage. In addition, dormant data "clogs the arteries" of high-performance disk storage. There are two good reasons that moving dormant data to an alternate form of storage makes sense:

■ It saves money—potentially lots of money.

■ It improves performance.

Dormant data creeps into a system silently. Figure 3.2.11 shows how dormant data grows inside a data warehouse.

Newly built and implemented data warehouses typically do not contain a lot of data and therefore do not contain much dormant data. As the volume of data grows in a data warehouse, the percentage of data that is dormant grows as well. When there is a significant amount of data in a data warehouse, there is almost always a significant amount of data that has gone dormant. One alternative is simply to leave the dormant data in the data warehouse, but doing so is expensive and slows down the system considerably. The other alternative is to move the dormant data to either near-line storage or archival storage. Figure 3.2.12 depicts the periodic transfer of dormant data to near-line or archival storage.

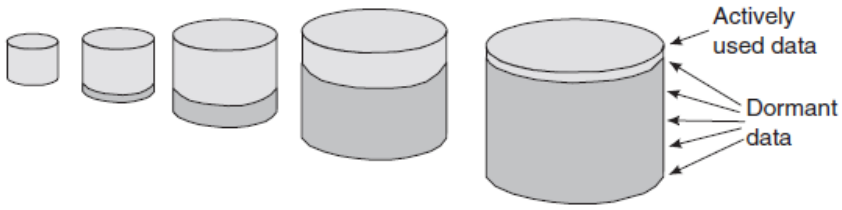


FIGURE 3.2.11 As the volume of data increases, the percentage of dormant data grows.

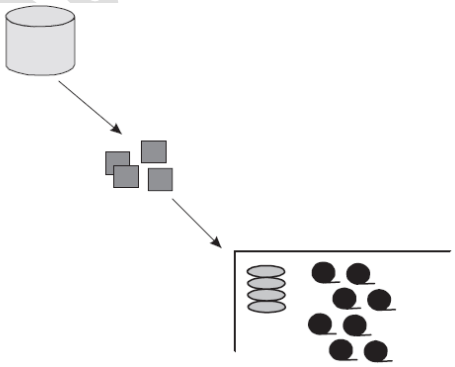


FIGURE 3.2.12 There is a burning question—What data needs to be placed into near-line storage?

The data warehouse monitor is used to tell when data has gone dormant.

3.2.11 LET US SUM UP

Thus we studied; types of monitors are needed in the DW 2.0 environment:

■ Transaction monitor
■ Data quality monitor
■ Data warehouse monitor

The transaction monitor addresses the Interactive Sector and is focused on transaction response time and capacity planning.

The data warehouse monitor is focused on the Integrated Sector of the DW 2.0 data warehouse and addresses dormant data. It looks at data and determines what data is being used and what is not being used.

The best data warehouse monitors are those that operate outside of the data warehouse DBMS. The use of SQL sniffers is the least obtrusive and the least time-consuming technology for monitoring the activity that goes on inside the data warehouse.

3.2.12 List of References

- DW2.0 The architecture for Next Generation of Datawarehousing W.H. Inmon, Derek Strauss, Genia Neushloss, ELSEVIER.
- Building the data warehouse, W.H.Inmon, third Edition, Wiley.
- Datawarehousing, S. Mohanty, TMH .
- The Data Warehouse Lifecycle toolkit", Ralph Kimball ,John Wiley.

3.2.13 Unit End Exercises

1) Explain in brief ETL data quality monitor.
2) Explain in brief how to monitor DW environment.
3) Explain transaction monitor with its features.
4) Explain dormant data monitor in data warehouse.
5) Explain briefly the types of monitors used to monitor DW 2.0 environment.

UNIT-III: DW and security

Unit Structure

3.3.0 OBJECTIVES

In this chapter you will learn about:

- Security is a requirement for the entire DW environment.
- Types of security.
- Data protection to move data offline this prevents access to data from online hacks.
- Database dump is a form of security breach.

3.3.1 INTRODUCTION

There is no information systems built today that can ignore the issue of security. The DW 2.0 next-generation data warehouse is no different from any other type of system in this regard. Given the sweeping nature of data warehousing—covering transaction processing systems to archival systems—it is no surprise that the security measures take many different forms and address security from many different perspectives in the DW 2.0 environment.

3.3.2 Protecting access to data

There are many ways to address security of data and security of systems. Figure 3.3.1 suggests one of the simplest ways to protect data. Figure 3.3.1 represents the barriers that may be put up to prevent people from gaining unwanted or unauthorized access to data. These barriers take many different forms, such as passwords, special transactions, and software intervention. Barriers are useful throughout the DW 2.0 environment—from the Interactive Sector to the Integrated Sector, the Near Line Sector, and the Archival Sector.

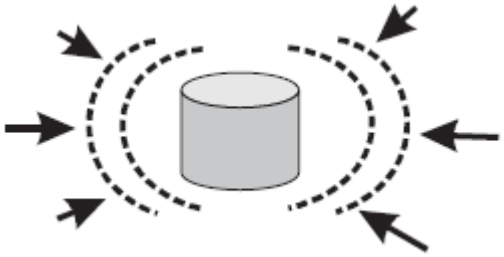


FIGURE 3.3.1 One way to protect data is to be very careful who has access to it.

3.3.3 Encryption

There are other ways to protect data. Figure 10.2 depicts another technique. When data is encrypted it is rewritten in a form that is different from its original form. While anyone may be able to access encrypted data, only those who know how to decrypt the data can make sense of it. The protection of data by encryption does not lie in protecting it from access. Encryption protects data by restricting the decryption mechanisms to only authorized people. Encryption is useful in some cases in the DW 2.0 environment, but due to the disadvantages of encryption, very careful use of it must be made in DW 2.0. These two security techniques—protection from the access of data and data encryption—are found in different places in different forms throughout the DW 2.0 environment.

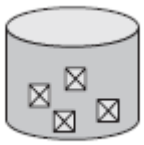


FIGURE 3.3.2 Another way to protect data is to encrypt it.

3.3.4 Drawbacks

There are drawbacks to both types of security. Protecting data from unauthorized access requires a technological infrastructure all its own. There is an administrative cost to keeping the protective infrastructure up to date.

There are also definite costs associated with data encryption. When data is encrypted, the encryption disables several important aspects of the data warehouse system. Encrypted data cannot be indexed effectively. It requires that the person accessing the data encrypt the parameters in the query before the query can be executed. And, it cannot be used for logic or numerical calculations or comparisons. In short, there are many disadvantages to the encryption of data.

Figure 3.3.3 illustrates one such disadvantage. Because of the disadvantages of both types of security, no one type is sufficient for the DW 2.0 data warehouse. Instead, a mixture of security types is found in the DW 2.0 environment.

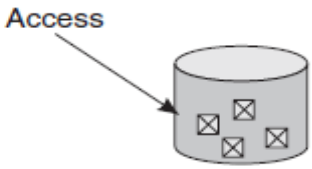


FIGURE 3.3.3 When data is encrypted, the person accessing the data must be aware of the encryption prior to conducting a search.

3.5 Firewall

Perhaps the most well-known type of security is the firewall. Firewalls are used where the Internet connects the outside world to the systems of the corporation. Figure 3.3.4 depicts a firewall regulating the transactions that are entering from the Internet. It is worth noting that firewall protection applies only to access to the Interactive Sector, because the Interactive Sector is the place in the DW 2.0 architecture where active transaction processing occurs

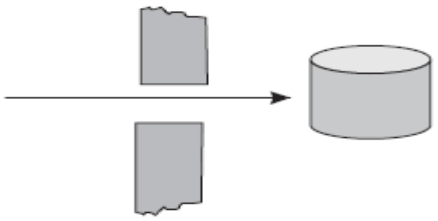


FIGURE 3.3.4 A basic form of protection for online security is the firewall.

3.3.6 Moving data offline

Figure 10.5 shows that it is only the Interactive Sector that has an interface with the Internet environment.

This figure shows that the Integrated Sector, the Near Line Sector, and the Archival Sector of the DW 2.0 data warehouse must not interface directly with the Internet. Completely insulating these key data warehouse sectors from the Internet means that even the best hacker cannot get to the data contained in their data bases. Only the interactive data is at risk.

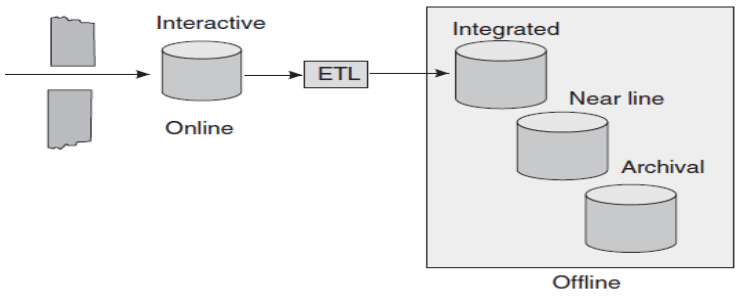


FIGURE 3.3.5 A basic form of protection is to keep the non interactive data in DW 2.0 offline.

This is a very simple yet very effective security measure. The only Interactive Sector data that is allowed to pass to the Integrated Sector must first be processed through the DW 2.0 ETL interface. Data in the Integrated, Near Line, and Archival Sectors can still be accessed; however, to access them requires offline processing or authorized access to the organization's internal network. Figure 10.6 shows that DW 2.0 data can still be accessed offline. Data in the DW 2.0 environment can in fact be encrypted. However, because it is not practical or effective to encrypt huge amounts of data, only selected pieces of data should be encrypted.

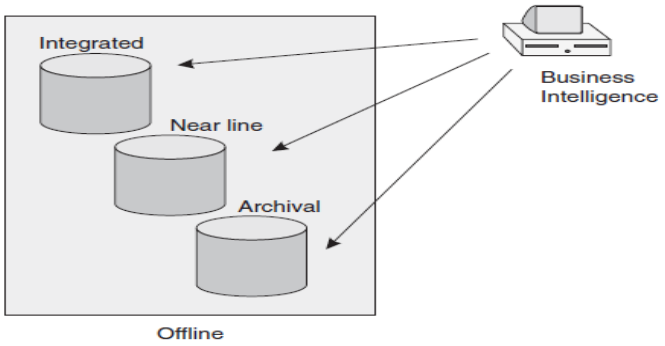


FIGURE 3.3.6 Offline data is accessed by a separate network

Data in the DW 2.0 environment can in fact be encrypted. However, because it is not practical or effective to encrypt huge amounts of data, only selected pieces of data should be encrypted.

3.3.7 Limiting encryption

Figure 3.3.7 depicts encryption of only a few fields of data inside a record. Even though it is technically possible to index encrypted data, data is not easy to retrieve after it has been encrypted. By encrypting only a small part of the data in a record, the disadvantages that come with encryption are minimized.

FIGURE 3.3.7 One approach is to encrypt only part of the data in a record.

There needs to be a means by which the actual values that have been encrypted can be restored. Figure 3.3.8 shows that actual values represented by the encrypted data can be determined by either going through an index or by passing data into an algorithm. The techniques of

encryption are best used in the DW 2.0 environment in the Integrated Sector. If they are used at all, encryption techniques should be used sparingly, due to the disadvantages that accompany encrypted data. When encryption techniques are used in the Interactive Sector, special attention should be paid as to whether performance will be negatively and severely impacted. It is questionable if encrypted techniques should be used in the Archival Sector.

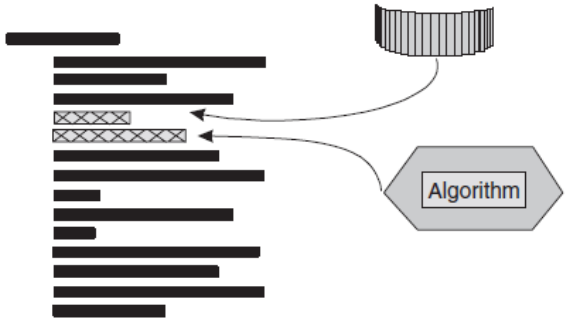


FIGURE 3.3.8 The way encrypted data is accessed.

3.3.8 A direct dump

As effective as barriers to unauthorized data access are, security protection can be bypassed by simply doing a system dump of the data and then reading the data manually. Figure 3.3.9 depicts a system dump of a DW 2.0 data base.

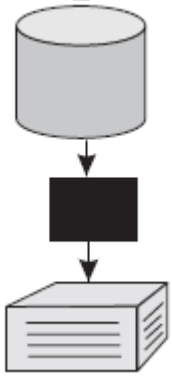


FIGURE 3.3.9 One way to bypass all security is to do a system dump.

After the system data dump is done and the data is free of any protection other than encryption, the dump can then be read manually, as indicated by Figure 3.3.10. Reading a dump manually is not something that most people would do, either for pleasure or for work. There is also an arcane set of rules that need to be followed if a dumped data set is to be read in its entirety.

However, textual data contained in a data base dump can easily be read without the aid of special skills or tools. Even a non technician can read a data dump if all that is required is picking out text that the dump contains. Because reading a dump bypasses all other forms of protection, at least some form of encryption is encouraged. This protection is especially encouraged for DW 2.0's Integrated Sector.

FIGURE 3.3.10 After the dump is made, the data is read manually

3.3.9 The data warehouse monitor

The data warehouse monitor is a standard recommendation for the DW 2.0 environment. But there is another good reason the data warehouse monitor should be used, namely to determine who is looking at what data. Identifying who is sending queries and to what data bases is one of the most useful things that a data warehouse monitor can do. Data warehouse monitors and the transaction monitor both produce transaction logs.

A transaction log is just a register of the activity that has been detected by the monitor. Figure 3.3.11 shows a DW 2.0 environment that has been monitored, resulting in the creation of a transaction log. It is a fairly simple matter to read the log after it has been generated by the monitor. The analyst can tell who has been querying what data by reading the access transactions logged by the data warehouse monitor. This form of security is passive, because it does not prevent an unauthorized access to the data warehouse data bases. But it is security nevertheless, and if there have been unauthorized accesses they will show up in the log.

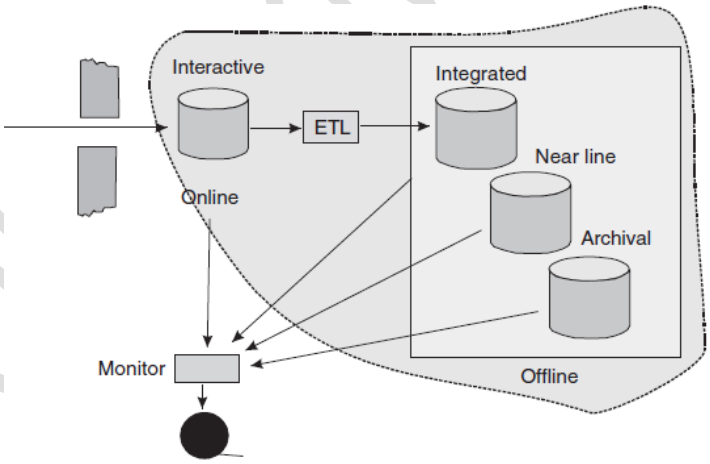


FIGURE 3.3.11 The access log can be used to see who is looking at what data.

3.3.10 Sensing an attack

Detecting an attack coming from outside the DW 2.0 environment before it becomes a serious problem, or better yet before it happens, is **FIGURE 3.3.10** After the dump is made, the data is read manually another good way to protect and secure the data warehouse. Prompt identification of invalid or unauthorized access enables the organization
to sense and stop an attack. For example, occasionally a system will want to enter another system without knowing the password.

A fairly effective way to overcome this problem and carry out an unauthorized access is to flood the system with many different passwords. As soon as the assault yields a password that grants entry to the system, the attacking routine stores the good password for future use. This kind of attack can be sensed by keeping track of then acceptable passwords that are sent to the system for authorization. If there is a sudden flurry of passwords that are invalid, the system senses an attack.

The system can then selectively shut down until the attack is over. This kind of attack is most likely to happen in the Interactive Sector of DW 2.0 as transactions flow in via the Internet. Figure 3.3.12 depicts a password flooding attack.

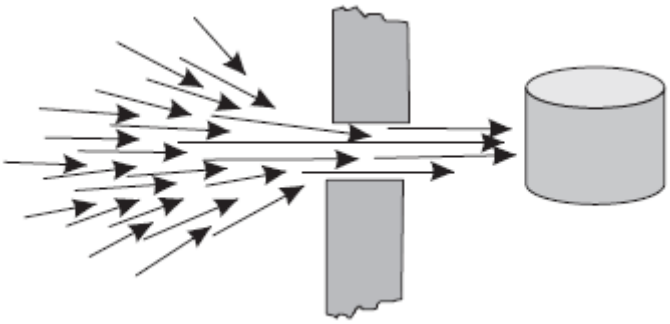


FIGURE 3.3.12 If too many unsuccessful access requests come through the firewall in rapid succession, the system senses that an attack may be occurring.

Security protection on the unstructured side of the DW 2.0 data warehouse mimics the security of the structured side, but there is an added twist. If incoming documents are protected and the system detects that protection, then the documents are never brought into the DW 2.0 environment at all. This ensures that external protection mechanisms are honored and data that has been protected by others never even enters the unstructured side of DW 2.0. Figure 3.3.13 shows that external security is honored at the point of entry into DW 2.0.

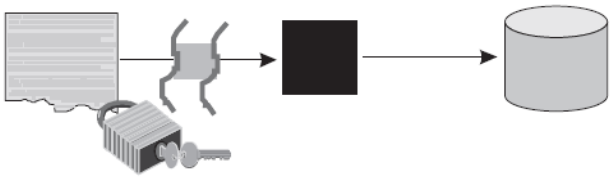


FIGURE 3.3.13 If documents are protected they are not entered into the unstructured data base

3.3.11 Security for near line data

The Near Line Sector is the environment that requires the least protection. The Near Line Sector is merely an extension of the Integrated Sector. Therefore, any security measures that apply to the Integrated Sector also apply to the Near Line Sector. Stated differently, under normal operating procedures, the Near Line Sector cannot be accessed by itself; it consequently requires the least protection. Finally there is protection of archival data. Archival data can be protected in the standard ways—through access authorization and through encryption.

However, because archival data is often placed on storage media other than disk storage, there are usually even more possibilities for protecting it. Many archival environments include check-in/check-out processing. If a unit of data is to be protected, not allowing the data to be checked out is a good approach that adds an extra level of protection. Figure 3.3.14 shows that archival data offers even more opportunities for security.

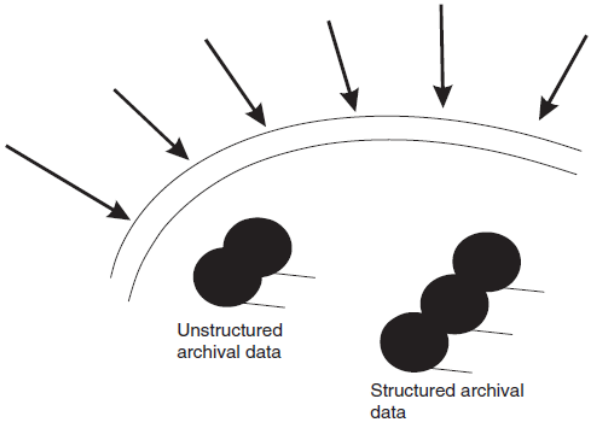


FIGURE 3.3.14 Often times the archival environment can have its own separate levels of security.

3.3.12 LET US SUM UP

Security is a requirement for the entire DW 2.0 environment. There are two basic types of security: (1) barrier security, in which blocks are placed in front of people to prevent unauthorized access; and (2) data encryption security, in which anyone can access the data, but only authorized people can make sense of the data.

There is passive security, in which no attempt is made to stop people from accessing data, but a log record is maintained of all data that is accessed. Upon sensing that an unauthorized hacking has occurred, the passive monitor reports what data has been accessed and by whom. One technique for protecting data is to move as much data offline as possible. This prevents access to the data from online hacks. Another form of security is an attack monitor that checks to see if an unusual number of unauthorized accesses to protected data are being made.

3.3.13 List of References

- DW2.0 The architecture for Next Generation of Datawarehousing W.H. Inmon, Derek Strauss, Genia Neushloss, ELSEVIER.
- Building the data warehouse, W.H.Inmon, third Edition, Wiley.
- Datawarehousing, S. Mohanty, TMH .
- The Data Warehouse Lifecycle toolkit", Ralph Kimball ,John Wiley.

3.3.14 Unit End Exercises

1) Explain the importance of encryption process.
2) How data access can be protected in DW?
3) Write a short note on attack sensing.
4) Discuss the different security measures employed for data warehouse.

UNIT-IV: Time variant data

Unit Structure

4.1.0 OBJECTIVES

In this chapter you will learn about:

- How data in data warehouse is relative to some moment in time.
- Transaction in integrated sector
- Continuous time span data without overlapping records.
- Time relevancy in DW environment.

4.1.1 INTRODUCTION

One of the essences of the DW 2.0 environment is the relationship of data to time. Unlike other environments where there is no relationship between data and time, in the DW 2.0 environment all data—in one way or another—is relative to time.

4.1.2 All data in DW

Figure 4.1.1 shows that all data in DW 2.0 is relative to time. This fact means that when you access any given unit of data, you need to know at what time the data is accurate. Some data will represent facts from 1995. Other data will represent information from January. And other data will represent data from this morning. In DW 2.0 then, whether it is explicit or implicit, all data has a moment in time that depicts its accuracy and relevancy. The data structure at the record level that is commonly used to make this depiction is seen by Figure 4.1.2. In Figure

4.1.2 there are two record types. One record type is used for a snapshot of data at a single moment in time. This record type on the left has DATE and TIME as part of the key structure.

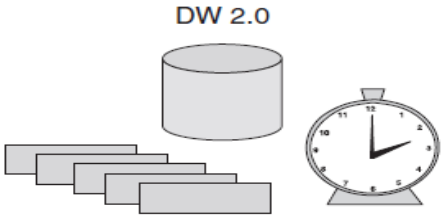


FIGURE 4.1.1 In one way or another, all data in the DW 2.0 environment is relative to time

The other type of record is shown on the right. It is a record that has a FROM date and a TO date. The implication is that a block of time—not a point in time—is being represented. Note that in both cases the element of time is part of the key structure. The key is a compound key and the date component is the lower part of the compound key.

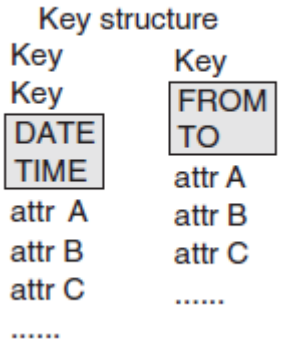


FIGURE 4.1.2 Time shows up as part of the key structure of the data.

4.1.3 Time relativity in the interactive sector

In the Interactive Sector, the time relevancy of data is somewhat different. In this sector, data values are assumed to be current as of the moment of access. For example, suppose you walk into a bank and inquire as to your balance in an account. The value that is returned to you is taken to be accurate as of the moment of access. If the bank teller says to you that you have $3971 in the bank, then that value is calculated up to the moment of access. All deposits and all withdrawals are taken into account. Therefore, because interactive data is taken to mean accurate as of the moment of access, there is no date component to interactive data. Figure 11.3 shows a banking transaction occurring in which interactive, up-to-the-second data is being used. But in all other sectors of DW 2.0—in the Integrated Sector, the Near Line Sector, and the Archival Sector—data explicitly has a moment in time associated with the data.

FIGURE 4.1.3 In the Interactive Sector, data is correct as of the moment of usage.

4.1.4 Data relativity elsewhere in DW

Figure 4.1.4 shows that each record in the Integrated Sector, the Near Line Sector, and the Archival Sector represents either a point in time or a span of time. This notion of data being relative to time produces some very different ways of doing processing. In the interactive environment, update of data is done. In this case the update of data refers to the actual changing of the value of data.

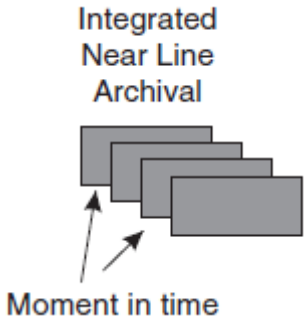


FIGURE 4.1.4 In all other sectors of DW 2.0, each record represents a moment in time.

Figure 4.1.5 shows that a banking transaction is done and the value of the data is changed in the interactive environment. At 10:31 AM there is $2000 in an account. A transaction adding $500 to the account occurs. The transaction is executed against the data in the data base and at 10:32 AM, the bank account has a balance of $2500. The data has changed values because of the transaction.

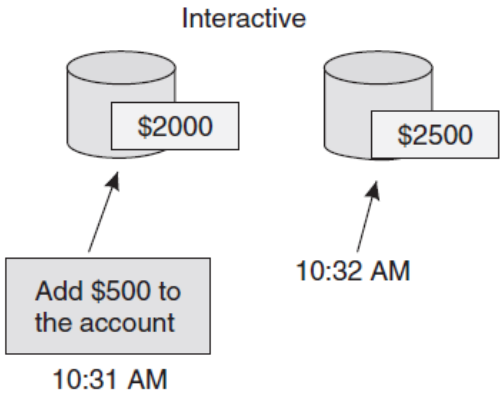


FIGURE 4.1.5 In the Interactive Sector, the value of data is changed as a result of an activity.

4.1.5 Transactions in integrated sector

Now let us consider a similar scenario in the Integrated Sector. At 10:31 AM there is a value of $2000 sitting in the integrated data base. A transaction is executed. At 10:32 a new record is placed in the data base. Now there are two records in the data base showing the different data at different moments in time. Figure 4.1.6 shows the execution of a transaction in the Integrated Sector.

The different data found in Figures 4.1.5 and 4.1.6 make it clear that because of the difference in the way data relates to time, the content of data in the different environments is very different. There are terms for these different types of data. Figure 4.1.7 shows those terms. Where there is just a point in time, the data is called discrete data. Where there is a FROM date and a TO date, the data is called continuous time span data. These two types of data have very different characteristics.

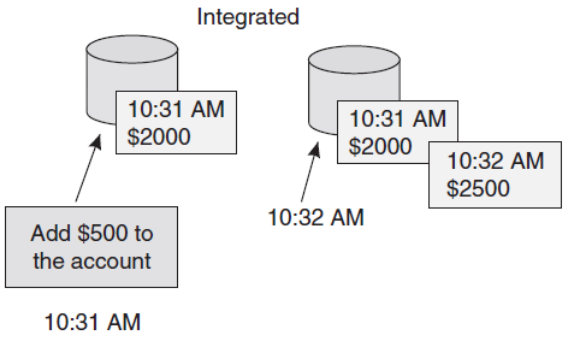


FIGURE 4.1.6 In the Integrated Sector, a historical record is kept for all activities

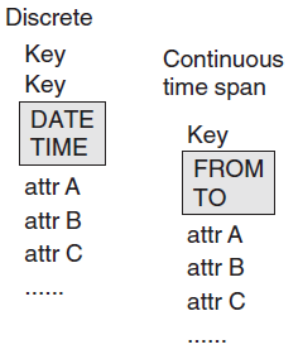


FIGURE 4.1.7 The two common forms of time-variant data.

4.1.6 Discrete data

Discrete data is good for lots of variables that quickly change. As an example, consider the Dow Jones Industrial average. The Dow Jones is typically measured at the end of the day, not when a stock that is part of the Dow is bought or sold. The variables that are captured in the discrete snapshot include variables that are measured at the same moment in time. Other than that one coincidence, there is nothing that semantically ties the attributes of data to the discrete record. Figure 4.1.8 shows some of the characteristics of the discrete structuring of data.

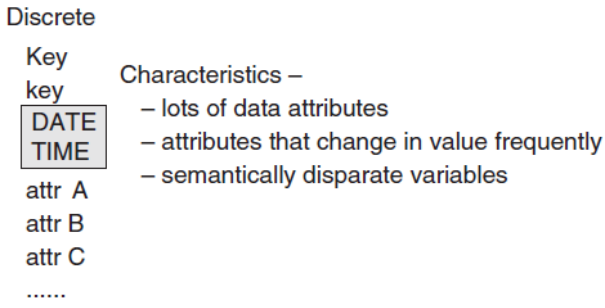


FIGURE 4.1.8 Some characteristics of discrete time-variant data.

4.1.7 Continuous time span data

Continuous time span data has a different set of characteristics. Typically, continuous time span data has very few variables in the record. And the variables that are in the record do not change often. The reason for these characteristics is that a new continuous time span record must be written every time a value changes. For example, suppose that a continuous time span record contains the following attributes:

Name
Address
Gender
Telephone Number

A new record must be written every time one of these values changes. Name changes only when a woman marries or divorces, which is not often. Address changes more frequently, perhaps as often as every 2 to 3 years. Gender never changes, at least for most people. Telephone Number changes with about the same frequency as Address changes.

Thus it is safe to put these attributes into a continuous time span record. Now consider what would happen if the attribute Job Title were put into the record. Every time the person changed jobs, every time the person was promoted, every time the person transferred jobs, every time there was a corporate reorganization, it is likely that Job Title would change. Unless there were desires to create many continuous time span records, it would not be a good idea to place Job Title with the other, more stable, data.

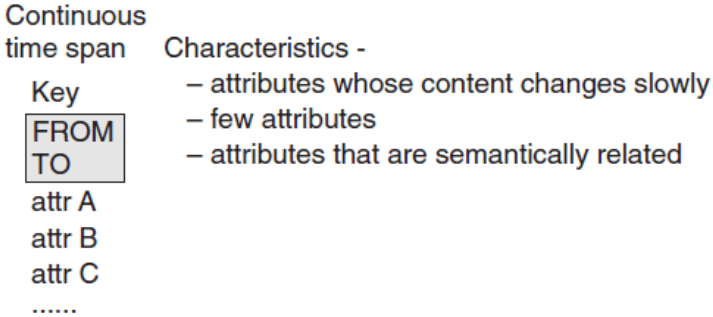


FIGURE 4.1.9 Some characteristics of continuous time span time-variant data.

Figure 4.1.9 shows some of the characteristics of the continuous time span records. Great care must be taken in the design of a continuous time span record because it is possible to create a real mess if the wrong elements of data are not laced together properly. As a simple example, Figure 4.1.10 shows some typical attributes that have been placed in a continuous time span record. Figure 4.1.10 shows that the attributes Name, Date of Birth, Address, and Gender have been placed in a continuous time span record.

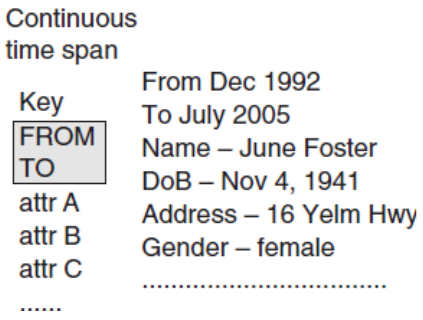


FIGURE 4.1.10 The implications of a continuous time span record

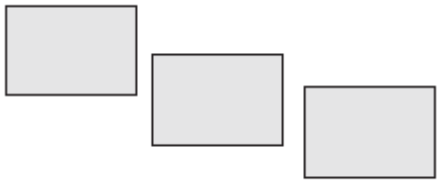


FIGURE 4.1.11 A series of records are combined to form a continuous record over time.

These elements of data are appropriate because
■ They are slow to change in terms of content;
■ They all relate to descriptive information about an individual.

Whereas a single continuous time span record is useful, multiple continuous time span records can be strung together to logically form a much bigger record of continuity. Figure 4.1.11 shows several continuous time span records strung together.

4.1.8 A sequence of records,

The records form a continuous sequence. For example, one record ends on January 21, 2007, and the next record begins on January 22, 2007. In doing so, the records logically form a continuous set. As a simple example, June Foster's address was on Yelm Highway until July 20, 2002. One record indicates that value.

Then June moved to Apartment B, Tuscaloosa, Alabama, and her official move date was July 21, 2002. A new record is formed. Together the two records show the date and time of her change of addresses and shows a continuous address wherever she was at. Although continuous time span records are allowed to form a continuous record, they are not allowed to overlap. If there were an overlap of records, there would be a logical inconsistency. For example, if two records had address information for June Foster and they overlapped, they would show that June lived in two places at once.

4.1.9 Non overlapping records

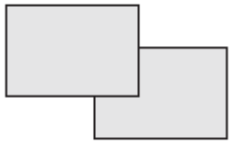


FIGURE 4.1.12 Overlapping records are not allowed.

Figure 4.1.12 shows that continuous time span record overlap is not allowed. Although continuous time span records are not allowed to overlap, there can be periods of discontinuity. In 1995, June Foster sailed around the world. During that time she had no mailing address. The records of her address would show an address up until the moment
she sailed and would show an address for her when she returned from her sailing voyage, but while she was on the voyage, there was no permanent address for her.

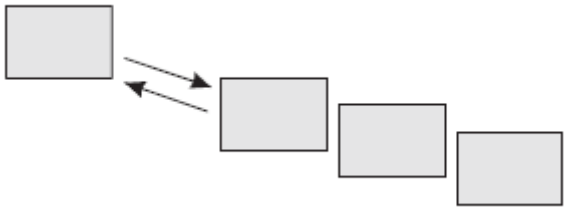


FIGURE 4.1.13 Undefined gaps of time are allowed.

Figure 4.1.13 shows that gaps of discontinuity are allowed if they match the reality of the data. When it comes to adding new records, the new record is added as of the moment in time when the business was transacted or concluded. Depending on how the records are constructed, it may be necessary to adjust the ending record. Figure 4.1.14 shows the update of a new record into a sequence of continuous records.

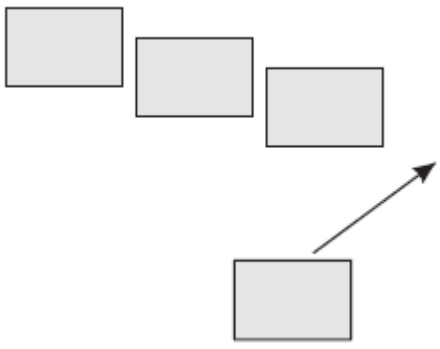


FIGURE 4.1.14 When it comes time to update a set of time span records, a new record is added.

4.1.10 Beginning and ending a sequence of records

There are a variety of options for beginning and ending the sequence of continuous time span records. For example, suppose that the most current record is from May 1999 to the present. Suppose there is an address change in April 2007. A new record is written whose FROM date is April 2007. But to keep the data base in synch, the previous current record has to have the TO date adjusted to show that the record ends on March 2007. To that end, a sequence of records can begin and end anywhere. The FROM date for the first record in the sequence may have an actual date. Or the FROM date may be minus infinity. When the FROM date is minus infinity, the implication is that the record covers data from the beginning of time. Where there is a FROM date specified for the first record in a sequence, for any time before the FROM date, there simply is no definition of the data. The ending sequence operates in much the same manner. The ending record in a continuous time span sequence may have a value in the

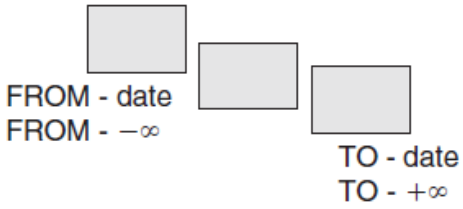


FIGURE 4.1.15 There are several options for start date and stop date.

TO field, or the TO value may be plus infinity. When the value is plus infinity, the implication is that the record contains values that will be applied until such time as a new record is written. For example, suppose there is a contract who's TO value is plus infinity. The implication is that the contract is valid until such time as notification is given that the contract is over. Figure 4.1.15 shows some of the options for starting and stopping a sequence of continuous time span records.

4.1.11 Continuity of data

One of the limitations of discrete data is that there is no continuity between two measurements of data. For example suppose the NASDAQ closes at 2540 on Monday and at 2761 on Tuesday. Making the assumption that the NASDAQ was at a high of 2900 sometime on Tuesday is an assumption that cannot be made. In fact, *no* assumptions about the value of the NASDAQ can be made, other than at the end of the day when the measurements are made.

Figure 4.1.16 shows the lack of continuity of the discrete measurements of data. Continuous time span data does not suffer from the same limitations. With continuous time span data you can make a judgment about the continuity of data over time.

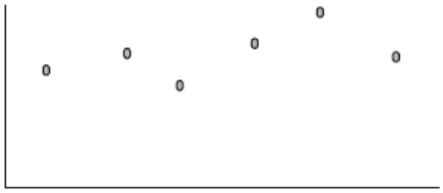


FIGURE 4.1.16 With discrete data, there are no implications of continuity.

Figure 4.1.17 shows the continuity of data that can be inferred with continuous time span data. Whereas discrete data and continuous time span data are the most popular forms of data, they are not the only forms of time-variant data in DW 2.0. Another form of time-variant data is time-collapsed data.

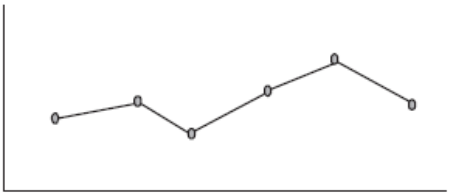


FIGURE 4.1.17 With continuous time span data, there are definitely implications of continuity.

4.1.12 Time-collapsed data

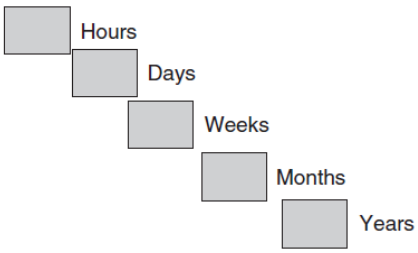


FIGURE 4.1.18 Time-collapsed data—another form of time-variant data.

Figure 4.1.18 shows a simple example of time-collapsed data. In time-collapsed data, there are several forms of measurement of data. When data enters the system it is measured in hours. Then at the end of the day, the 24 hours are added up to produce a recording of a day's worth of activities. The 24-hour measurements are then reset to zero. At the end of a week, the week's totals are created. Then the daily totals are reset to zero. At the end of the month, the month's totals are created.

Then the weekly totals are reset to zero. At the end of the year, the year's totals are created. Then the monthly totals are reset to zero. When this is done there is only one set of hourly totals, one set of daily totals, one set of weekly totals, and so forth? There is a tremendous savings of space. The collapsing of time-variant data works well on the assumption that the fresher the data the more detail there needs to be. In other words, if someone wants to look at today's hourly data, they can find it readily. But if someone wants to find hourly data from 6

months ago, they are out of luck. In many cases the assumptions hold true and collapsing of data makes sense. But where the assumptions do not hold true, then collapsing data produces an unworkable set of circumstances.

4.1.13 Time variance in the archival sector

The last place where time variance applies to the DW 2.0 environment is in the Archival Sector. It is a common practice to store data by years. One year's worth of data is stored, then another year's worth of data is stored. There are many good reasons for the segmentation of data in this way. But the best reason is that the semantics of data have the habit of varying slightly each year. One year a new data element is added. The next year a data element is defined differently.

The next year a calculation is made differently. Each year is slightly different from each preceding year. Figure 11.19 shows that each year there are slight changes to the semantics of data.

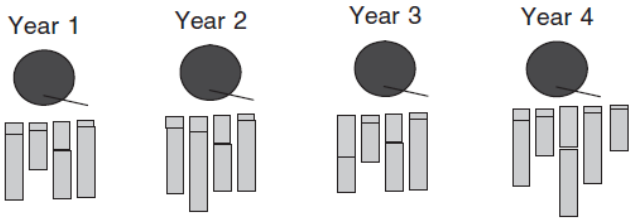


FIGURE 11.19 Archival data is stored on a year-by-year basis. Note that the data from one year to the next is never quite the same semantically.

4.1.14 LET US SUM UP

In one form or another, all data in DW 2.0 is relative to some moment in time. Interactive data is current. It is accurate as of the moment of access. Other forms of data in DW 2.0 have time stamping of the record. Time stamping takes two general forms. There is data that has a date attached. Then there is data that has a FROM and a TO field attached to the key. The first type of data is called discrete data. The second type is called continuous time span data. Continuous time span data can be sequenced together over several records to form a long time span. The time span that is defined by continuous time span records can have gaps of discontinuity. But there can be no overlapping records. There are other forms of time relativity in the DW 2.0 environment. There is time-collapsed data. Time-collapsed data is useful when only current data needs to be accessed and analyzed in detail. Over time the need for detail diminishes. The other form of time relevancy in the DW 2.0 environment is that of archival data. As a rule archival data is organized into annual definitions of data. This allows for there to be slight semantic changes in data over time.

4.1.15 List of References

- DW2.0 The architecture for Next Generation of Datawarehousing W.H. Inmon, Derek Strauss, Genia Neushloss, ELSEVIER.
- Building the data warehouse, W.H.Inmon, third Edition, Wiley.
- Datawarehousing, S. Mohanty, TMH .
- The Data Warehouse Lifecycle toolkit", Ralph Kimball ,John Wiley.

4.1.16 Unit End Exercises

1) Briefly explain discrete data.
2) Write a short note on time variant data.
3) Explain non-overlapping records.
4) Explain time-collapsed data.
5) Explain Discrete and Continuous time span data with an example of each. What is significance of these?

UNIT-IV: Flow of data in DW

Unit Structure

4.2.0 OBJECTIVES

In this chapter you will learn about:

- How data flows throughout the DW environment.
- Data entering the interactive sector.
- Data flows to the integrated sector through the ETL process.
- Data flows from the Interactive Sector to the Near Line Sector or the Archival Sector

4.2.1 INTRODUCTION

There are many components of the DW 2.0 architecture. A next generation DW 2.0 data warehouse includes many technologies. Building a DW 2.0 data warehouse environment is not like building a house. It is not even like building a town. It is much more like building a megalopolis. Because of the enormity and complexity of DW 2.0's scope, it is easy to get lost. It is easy to focus in on and dwell on just one aspect of DW 2.0. If this is done, the larger picture of the architecture becomes lost. It sometimes is useful to stand back from the details of the architecture and look at the larger picture.

4.2.2 Flow of data throughout the architecture

There is a flow of data throughout the DW 2.0 architecture. In many ways, this flow is as important as the flow of blood in the human body. The flow of data feeds everything else that is accomplished by DW 2.0. The flow of data begins as data is sent to the interactive environment. Data can be entered directly into the Interactive Sector, or data can flow into the sector through an ETL process. Exactly how data enters the Interactive Sector depends entirely on the application or applications that are found in DW 2.0.

4.2.3 Entering the interactive sector

Data enters the Interactive Sector as application-oriented data. After it has found its way into the Interactive Sector, data is sent to the Integrated Sector. Figure 4.2.1 shows the basic flow of data entering the DW 2.0 environment.

One of the interesting aspects about the flow of data is the speed and volume at which the flow is realized. Data flows into the Interactive Sector quickly, sometimes just milliseconds after it has been transacted in the external application environment. Incoming transaction data of this nature can be considered real-time data. And of course, if transactions are executed directly from the Interactive Sector, the data has no latency.

In other cases, it may be an hour or a day before the external application transaction data enters the interactive environment. The amount of time the data is delayed in entering the interactive environment is determined entirely by the business requirements for the data. If there is a legitimate business case for having the data enter the interactive environment immediately, then the data should be entered immediately. If there is no legitimate business case for the speedy entry of data into the Interactive Sector, then the entry of the data should not be hurried.

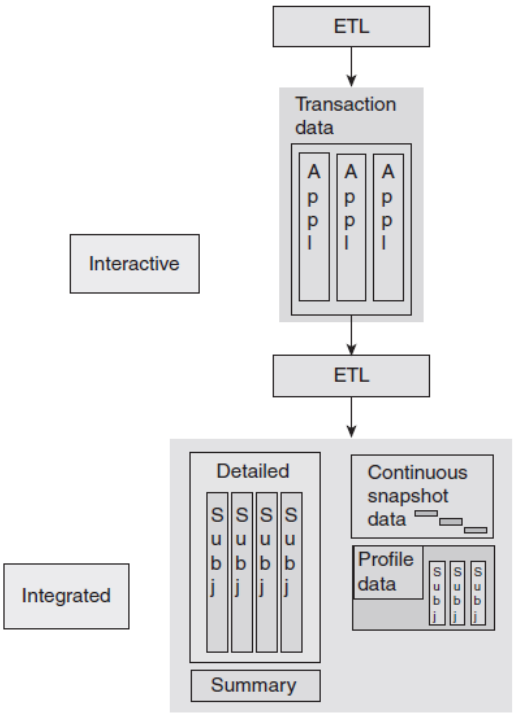


FIGURE 4.2.1 The flow of data into the Interactive Sector and from it to the Integrated Sector.

It is important to point out that the faster the data needs to be entered into the Interactive Sector, the more complex and the more expensive the technology that is required to accomplish the goal of rapid data transfer will be.

4.2.4 Role of ETL

The data coming from the external applications normally enters the Interactive Sector by means of ETL processing. It is possible, but not very common, for data to be placed in the Interactive

Sector by means of a simple file transfer. It is much more likely that the data will enter through standard ETL technology. Data passing into the Integrated Sector comes from the Interactive Sector. It is possible to have data enter the Integrated Sector directly, without passing through the Interactive Sector.

Normal processing occurs as data passes into the Integrated Sector from the Interactive sector. The normal way that data enters the Integrated Sector is through ETL processing. The ETL process reorganizes the data from an application-oriented structure into a corporate data structure.

4.2.5 Data flow into integrated sector

The flow of data into the Integrated Sector is more relaxed in terms of speed compared to the flow of data to the Interactive Sector from the Integrated Sector. Data flows into the integrated environment on a periodic basis overnight, weekly, monthly, or even quarterly in some cases. Figure 4.2.2 depicts the speed of the flow of data into the integrated Sector.

Data flows into the Interactive Sector in small, fast spurts. Data records enter this sector one at a time. It is not normal for an entire file or large set of records to be passed into the Interactive Sector all at once. Data enters the Interactive Sector like a rainfall, not a flood. The event that sets the stage for the entry of data into the Integrated Sector is the execution of a transaction. As soon as a transaction is completed, the data it contains is ready to be entered into the integrated Sector. Of course, transaction records can be batched together and entered en masse. But the gathering and queuing of the transactions prior to entry into the Interactive Sector slow transaction immeasurably and defeat one of the goals of the interactive process. The trigger for the transfer of data into the Integrated Sector is simple: the passing of time. In some cases, data needs to be entered into the Integrated Sector quickly, on an hourly basis.

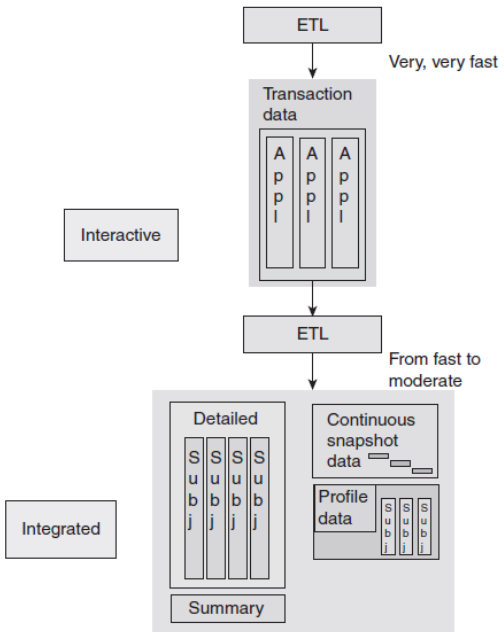


FIGURE 4.2.2 The rates of flow.

In other cases, data can wait to be transferred on an overnight basis. As a rule, if the data required for a report needs to be up to the second, the report should be created using the data found in the Interactive Sector. Normally it is a mistake to generate reports or analyses that require immediacy or real-time data from the Integrated Sector. Reports and analyses based on integrated data are of a strategic nature and should not depend on up-to-the-second data accuracy. Therefore, the movement of data from the Interactive Sector to the Integrated Sector can be done on a somewhat more relaxed schedule. Figure 4.2.3 shows the triggers that cause movement of data from external applications to the Interactive Sector and from the Interactive Sector to the Integrated Sector.

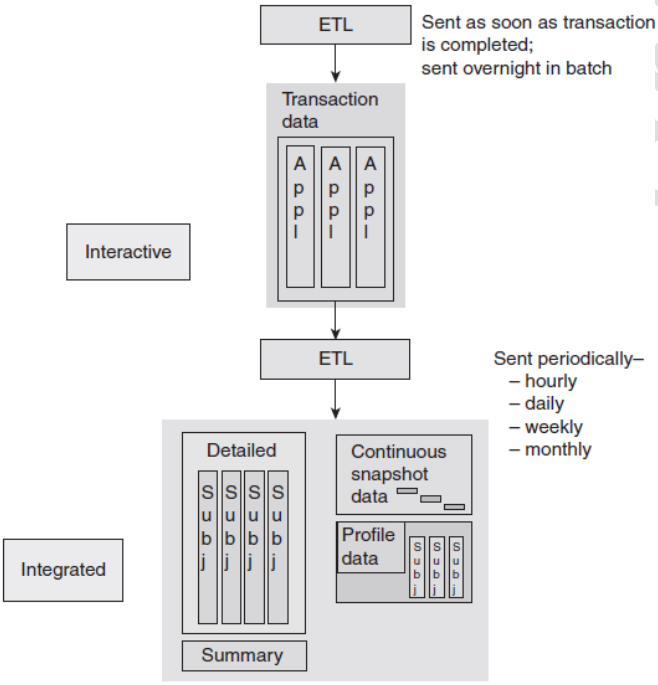


FIGURE 4.2.3 The triggers that determine when data moves to different sectors.

4.2.6 Near line

As important as the flow of data from the external applications to the Interactive Sector and that from the Interactive Sector to the Integrated Sector are, they are not the only major data flows in the DW 2.0 data warehouse. The flow of data from the Integrated Sector to the Near Line Sector is another important data flow in the DW 2.0 environment. This flow is optional and occurs when

- there is a lot of data in the Integrated Sector;
- some portion of the data in the Interactive Sector is dormant;
- where there is a desire for access to data in the Integrated Sector.

If data in the Integrated Sector does not meet these criteria, then there is no need for moving it to the Near Line Sector. In many ways, the Near Line Sector acts like a cache for the data in the Integrated Sector.

Data is placed in the Near Line Sector when it is not likely to be needed very frequently. The Near Line Sector is based on non disk storage. Consequently, near-line data storage is able to be stored much less expensively, and this sector is capable of holding vast amounts of data. Figure 4.2.4 illustrates the flow of data from the Integrated Sector to the Near Line Sector. The flow of the data from the Integrated Sector is slow.

Data is generally removed from the Integrated Sector only in large chunks, periodically—once a month, once a quarter. A decrease in the probability of data access is the sole criterion for moving data from the Integrated Sector to the Near Line Sector. Moving data to the Near Line Sector reduces the volume of data in the Integrated Sector. This lowers the cost of the data warehouse environment and improves performance. Ridding the integrated environment of data that is not accessed very often frees disk storage in the integrated environment for data that has a high probability of access.

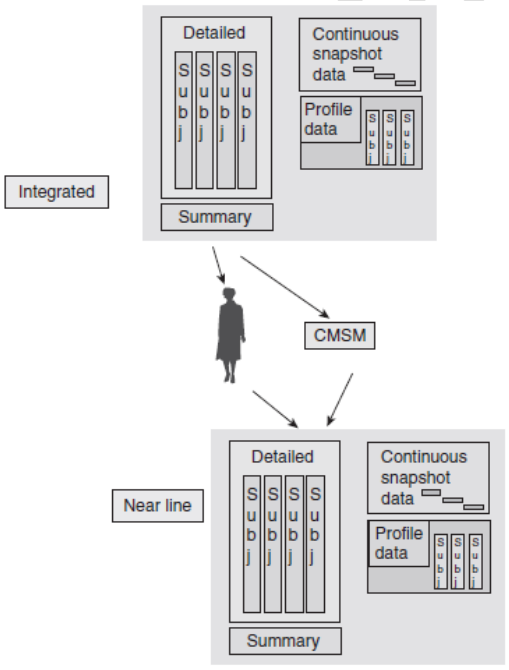


FIGURE 4.2.4 The flow of data from the Integrated Sector to the Near Line Sector.

4.2.7 Archival sector

Data also moves from the Integrated Sector to the Archival Sector. There is a key difference between data being moved from the Integrated Sector to the Archival Sector and integrated data being moved to the Near Line Sector.

When data is moved to the Near Line Sector, the structure and format of the data are preserved. This means data can be moved quickly and smoothly back to the Integrated Sector from the Near Line Sector when necessary. The Near Line Sector is designed to support access of data

in the Integrated Sector. In contrast, when data is moved to the Archival Sector, there is no intention of quickly moving the data back to the Integrated Sector.

The purpose of the Archival Sector is to preserve data for a long period of time. At some point in the future, data may need to be retrieved from the archival environment and restored in a place for a special analysis, or may even be sent back to the integrated environment, but there is no intention of immediate and detailed support of the integrated environment by the archival environment. Immediate and exclusive support of the Integrated Sector is the role of the Near Line Sector. Data is removed to the Archival Sector for the purpose of eliminating data that has a low probability of access. Figure 4.2.5 illustrates the movement of data from the Integrated Sector to the Archival Sector. The speed of data flow from the integrated environment to the Archival Sector is slow. Integrated data is typically retired to the Archival Sector on a quarterly or even annual cycle.

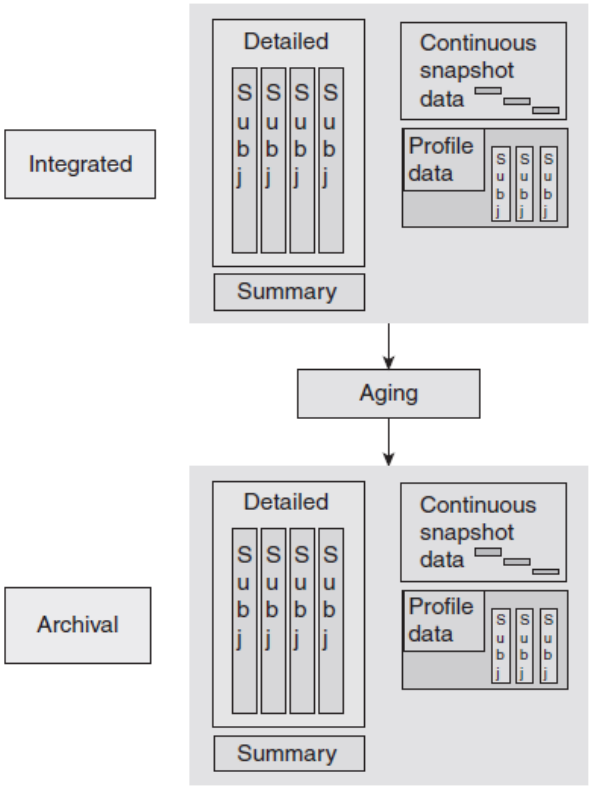


FIGURE 4.2.5 The flow of data from the Integrated Sector to the Archival Sector.

4.2.8 Falling probability of data access

The trigger for the movement of data from the Integrated Sector to the Archival Sector is when the probability of access of the integrated data has dropped. There are two basic ways to determine that the probability of data access has fallen. One way is based on date. For example, all data older than 3 years is removed from the Integrated Sector.

Another way to determine the probability of data access is by using a data warehouse monitor. Data warehouse monitors examine the accesses made to data in the Integrated Sector. Between the two approaches to determine the probability of access, the usage of a data warehouse monitor is by far the most accurate approach. While the data movements that have been

discussed can be considered the normal day-to-day migration of data in the DW 2.0 data warehouse environment, two other types of data movement are worthy of note.

4.2.9 Exception flow of data

The next two movements of data in the DW 2.0 environment are executed only on a limited need-for-data basis. The first non normal movement of data is the transfer of data from the archival environment back to the Integrated Sector. In this case there is data that, for whatever reason, has been deemed to be of use in standard analytical processing. This means, among other things, that the probability of access of the data has been elevated, and because the probability of access is high, the data properly belongs in the Integrated Sector.

Usually data is moved to the Integrated Sector from the Archival Sector as a large block, not a few records at a time. In all cases, the transfer of archival data back to the Integrated Sector is done on an on-demand basis.

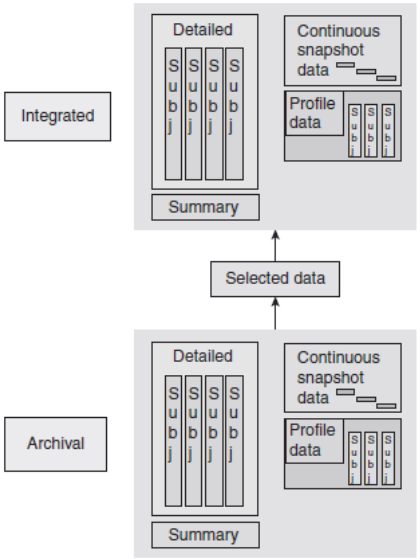


FIGURE 4.2.6 Occasionally there is a need to move data from the archival environment to the integrated environment.

Figure 4.2.6 depicts the movement of data from the Archival Sector to the Integrated Sector. The other non normal movement of data occurs when data from the near-line environment needs to be returned to the integrated environment. This data transfer can be accomplished in two ways. The CMSM software that sits between the two environments can be used to manage the relay of individual records of data. The CMSM tool places single records in the Integrated Sector, and the records are made to appear as if they have always resided in the Integrated Sector. The exchange is done so quickly that there is no serious degradation of system performance. The end user makes a query, and the system automatically senses that some of the requested data is in near-line storage. The system uses the CMSM tool to find, fetch, and place the data in integrated storage, and then the query is fulfilled. The second non normal movement of data, from the Near Line Sector to the Integrated Sector, occurs in bulk batch mode, when whole sections of data are moved. In this case the data may be moved either by the CMSM software or manually. In any case, the older data is moved back to the integrated environment because the anticipated probability of access has risen.

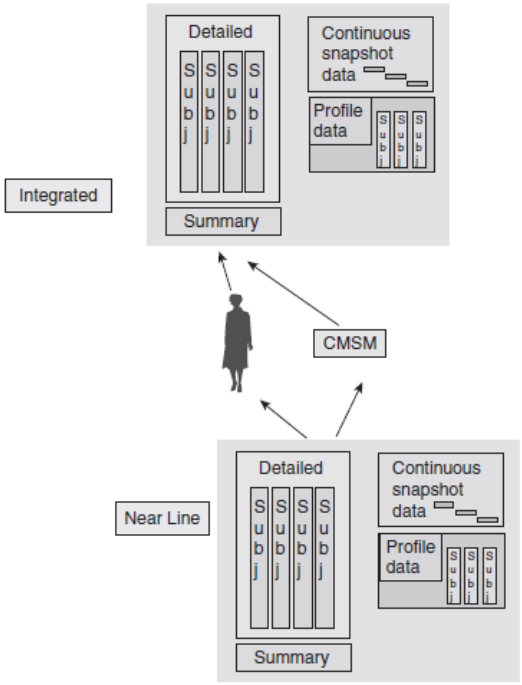


FIGURE 4.2.7 The movement of data into the Integrated Sector from the Near Line Sector.

Figure 4.2.7 depicts the movement of data to the Integrated Sector from the Near Line Sector. There is one final movement of data in the DW 2.0 environment that is worthy of discussion—the movement of data from the Integrated Sector to the Interactive Sector, also known as a data " back flow. "

The movement of data from the Integrated Sector to the Interactive Sector occurs infrequently. Usually the volume of data involved is not large. When this back flow occurs, it must be done without disruption of the online performance that is an essential part of the interactive environment. Figure 4.2.8 depicts the back flow of data from the Integrated Sector to the Interactive Sector.

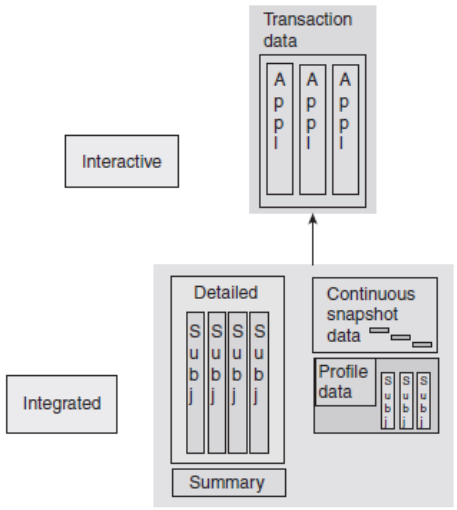


FIGURE 4.2.8 Back flow of data to the Interactive Sector

4.2.10 LET US SUM UP

Data flows throughout the DW 2.0 environment. Data enters the Interactive Sector either directly or through ETL from an external application. Data flows to the Integrated Sector through the ETL process, coming from the Interactive Sector. Data flows from the Interactive Sector to the Near Line Sector or the Archival Sector as it ages. On a limited basis, data may flow from the Archival Sector back to the Integrated Sector, and occasionally data flows from the Near Line Sector to the Integrated Sector.

4.2.11 List of References

- DW2.0 The architecture for Next Generation of Datawarehousing W.H. Inmon, Derek Strauss, Genia Neushloss, ELSEVIER.
- Building the data warehouse, W.H.Inmon, third Edition, Wiley.
- Datawarehousing, S. Mohanty, TMH .
- The Data Warehouse Lifecycle toolkit", Ralph Kimball ,John Wiley.

4.2.12 Unit End Exercises

1) Explain exception flow of data.
2) Explain data flow into the integrated sector.
3) Explain data flow into the archival sector.
4) What are the three exceptional flow of data in DW 2.0? Explain briefly.

UNIT-IV: ETL processing and DW

Unit Structure

4.3.0 OBJECTIVES

In this chapter you will learn about:

- How ETL changes the state of data.
- ETL can run in online or batch mode.
- ETL throughput, metadata, audit trail and data quality.
- Creation and rejection of ETL process.

4.3.1 INTRODUCTION

One of the most important processes in the DW 2.0 environment is ETL extract/transform/load processing. ETL processing gathers, cleanses, and integrates data as it enters the Interactive Sector or as it passes through the Interactive Sector to the Integrated Sector. ETL processing is fundamental to the day-to-day operation of the DW 2.0
Environment.

4.3.2 Changing states of data

ETL is representative of something even more powerful than the mere flow of data. ETL is the mechanism by which data changes its state. As data passes through and is transformed by ETL processing, it undergoes a fundamental change in state. It evolves from an application state to a corporate state. This fundamental change lies at the heart of the rationale and existence of the DW 2.0 environment. ETL does not just accomplish the collection and movement of data. ETL processing is the profound act of changing the state of data, and that is why ETL is an essential component of DW 2.0.

4.3.3 Where ETL fits

Figure 4.3.1 illustrates where the ETL processing fits in DW 2.0. ETL processing does a lot. It gathers old legacy data from technologies that have not been used as a primary DBMS technology for years, such as IMS, VSAM, CICS, IDMS, Model 204, and Adabas. Gathering the old legacy data is a nontrivial act, as each interface to a legacy application environment requires its own technology. After the legacy data is gathered, the real work of data transformation begins. The problem with older legacy systems is that they were never designed to work together.

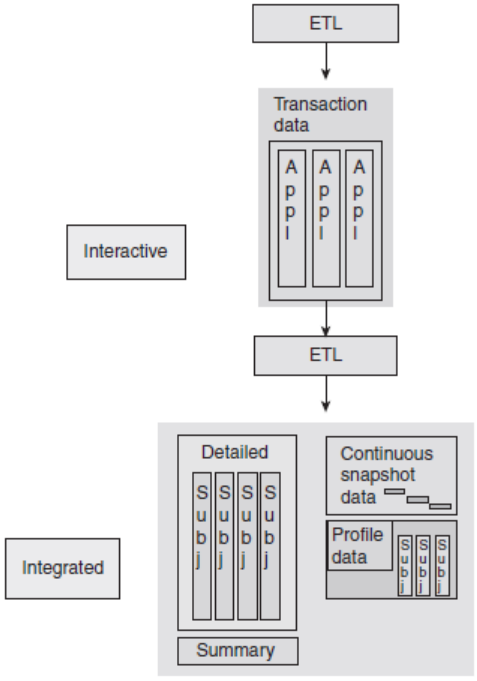


FIGURE 4.3.1 ETL processing is an essential part of the DW 2.0 environment.

There are differences in structure, format, calculations, definitions of data, and other semantic differences. In short, the old legacy application data needs a lot of work before it is fit for the corporate data environment, and it is the job of ETL to make the necessary major corrections to the legacy data.

4.3.4 Application data to corporate data

Figure 4.3.2 depicts the transformation of application data to corporate data by ETL. ETL processing can operate in two basic modes: online, real-time processing and batch processing.

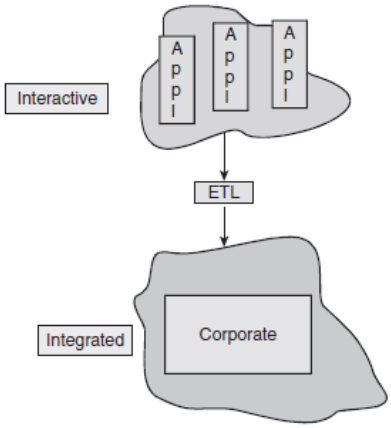


FIGURE 4.3.2 The real function of ETL is to convert application data to corporate data.

4.3.5 ETL in online mode and batch mode

When ETL processing is done in an online mode, the length of time from the moment of execution of the legacy transaction until the transaction is reflected in the DW 2.0 environment is measured in very small units of time, such as milliseconds.

For example, a transaction is executed by a bank teller at 11:32 AM in the legacy systems environment. ETL captures the fact that the transaction has been executed, and at 11:32 AM and 10 milliseconds the transaction finds its way into the DW 2.0 environment. The transaction has been sent to the DW 2.0 environment so quickly that the transaction seems to appear simultaneously in the legacy environment and the DW 2.0 environment.

In reality, the transaction has not been transacted and processed through ETL simultaneously, but it has happened so fast that for practical purposes it has happened simultaneously. The problem with real-time ETL is that speed is generally considered the main success factor. Because speed is the main factor of success, not much transformation can be done to the data coming into DW 2.0. The other mode of ETL processing is batch mode. In batch mode, transactions in the legacy environment are stored and batched together. Then, at a convenient time (perhaps overnight), the batch of legacy transactions is run through the ETL process. This means that from the time the transaction has been processed in the legacy environment until DW 2.0 is aware of the transaction 24 hours or more may have elapsed. One of the advantages of batch ETL is that there is plenty of time for significant data transformation processing.

As a rule, real-time ETL is expensive and not much transformation processing can be done. Batch ETL processing does not move the data through the system as quickly, but is much less expensive and it supports much more data transformation. Whether to use batch or real-time

ETL processing is more of a business decision than a technology decision. Some data—simple transaction data, such as a bank withdrawal—often can be done in real time. But more cumbersome, more complex data cannot be done in real time.

However, in the case of batch ETL transactions, there often is no business justification for a speedy movement through the ETL process. It is more important to take the time to do the transformation properly than it is to do it quickly. Figure 4.3.3 depicts the two modes of ETL processing.

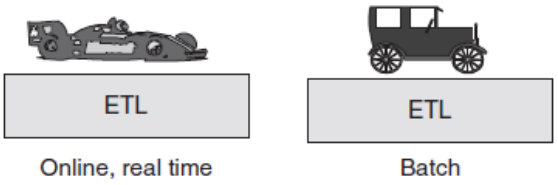


FIGURE 4.3.3 There are two modes that ETL operates in

4.3.6 Source and target

The analyst determines what ETL logic must be applied to source operational data before it enters the data warehouse by building a source to target data map. The mapping is simply a statement of what data must be placed in the DW 2.0 environment, where that data comes from, and the logic or calculation or reformatting that must be done to the data. The source of the data—the operational application systems environment—is called the "source." The place where the data lands in the DW 2.0 environment is called the "target." The source system in which the data originates is also called the "system of records." The following questions must be answered during the source-to-target mapping of each unit of data destined for the DW 2.0 environment:

- What specific units of data in what source system will constitute the data warehouse data?
- How should multiple sources of the same data be resolved?
- What must be done if default values of data are needed?
- What logic must be used to adjust the data to a corporate state?
- What calculations must be applied to reconstitute the data into a corporate state?
- What restructuring or reformatting must be done to create corporate data?

The source-to-target data mapping needs to be complete before the ETL process design commences.

4.3.7 ETL mapping

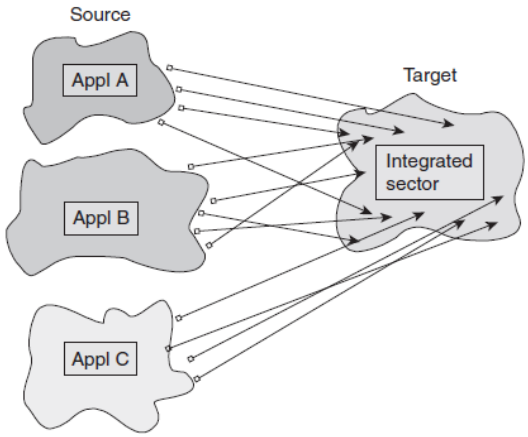


FIGURE 4.3.4 A mapping from the source to the target.

Figure 4.3.4 depicts an ETL data source-to-target mapping. One of the essential components governing the data that is mapped into the DW 2.0 environment is business rules. Business rules have a profound effect on the data that is sent to DW 2.0. Business rules reveal the correctness or quality of data. But, they often are different for different legacy systems. The ETL processing must therefore act like a referee and determine which business rule has precedence over all other business rules for each unit of data.

Figure 4.3.5 indicates that business rules are also necessary input to ETL specifications, along with source-to-target mapping.

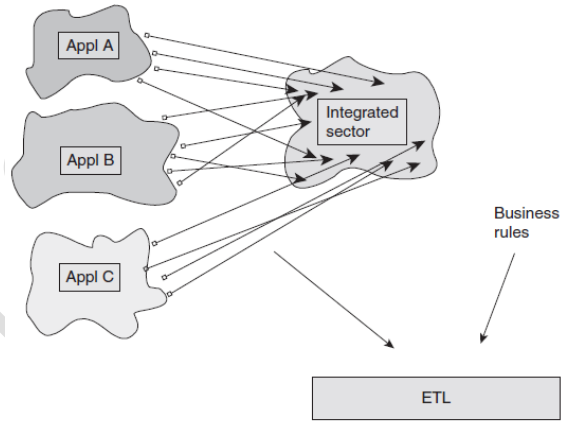


FIGURE 4.3.5 The mapping and business rules are used to determine how ETL processing should be done.

To the uninitiated it is not apparent that anything unusual is happening here. Indeed, in many cases data is merely passed from one environment to the next, because there is no difference between its operational and its corporate state. But in other cases, a significant transformation of data is necessary.

Figure 4.3.6 illustrates some simple transformations. This figure illustrates that application data for the same thing can exist in different forms. There are different kinds of dollars, there are different calculations of interest, and there are different closing dates for accounting. To achieve a corporate understanding of this data, all of these differences must be reconciled.

The organization's business rules might dictate that in the corporate data state, cash is measured in terms of euros, the last day of the month is the closing date for all accounting activities, and interest is not included in certain calculations. It is the ETL process that will make these adjustments as the source data is passed into the data warehouse environment. In some cases people say— " I can't trust DW 2.0 data because I ran an operational report last week, then I ran the same report from the DW 2.0 environment. I got different answers, so that means that DW 2.0 data can't be trusted. " This confusion happens when people do not understand the difference between corporate and application data. The truth is that reports coming from these different sources ought to be different.

To understand why that is the case, imagine that the closing date used in a corporation's accounting system is the 28th of the month. A report is run from this application showing the closing balance as $ 108,076. Now the transaction is run through the ETL process. Suppose the corporate accounting close date is the 31st of the month. Now suppose that a transaction occurred on the 29th of the month. In the application, the transaction will show up in one month's total. In the corporate data the transaction will show up in another month's total. There indeed are differences in data, but the differences are caused by the transformation from application data to corporate data.

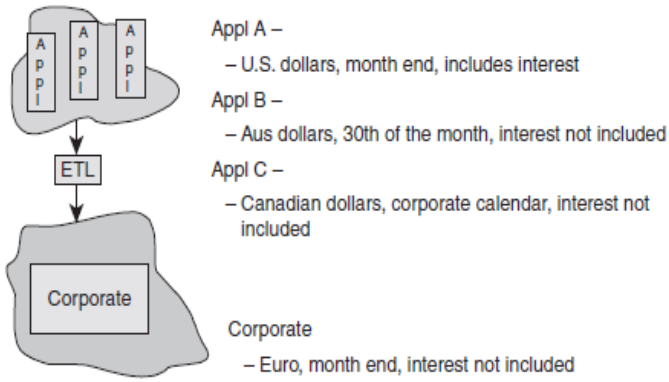


FIGURE 4.3.6 Although there will be some similarity of data between the Interactive Sector and the Integrated Sector, the data is not identical and is not redundant.

4.3.8 More complex transformation

The data transformations depicted in Figure 4.3.6 are actually easy transformations. Transformations become difficult when data keys have to be transformed. Usually when keys have to be transformed, one application is selected as having the proper designation for keys and all other applications have to conform to the key structure. Whereas most key transformations are resolved by selection of one key style and structure, occasionally data keys need to be completely converted to or replaced with an entirely new key format during ETL processing. Regardless of the approach, the rules for data key transformation must be specified during the design of the ETL routines.

4.3.9 ETL and throughput

Data throughput is another concern with ETL processing. Even with the simplest of ETL processing involving only the execution of a single process, if throughput is a problem, the ETL process can be parallelized. When an ETL job stream is parallelized, part of the data passes through one ETL process, and another part of the job stream passes through another copy of the same ETL process. The ETL process and job streams are replicated as many times at it takes to achieve satisfactory ETL throughput. The creation of parallel ETL throughput streams greatly reduces the elapsed time for the execution of the entire data ETL process.

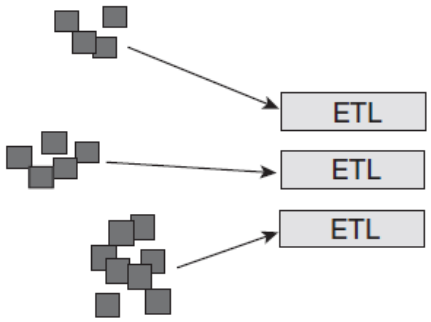


FIGURE 4.3.7 If throughput is an issue, ETL processing can be parallelized.

Figure 4.3.7 suggests that the ETL process can be parallelized. From a theoretical perspective, there are a lot of transformation activities that have to be accomplished by ETL processing. Any one transformation activity is usually not terribly complex, but the fact that *all* of the transformations have to be accomplished in a single pass of the data makes ETL processing truly complex.

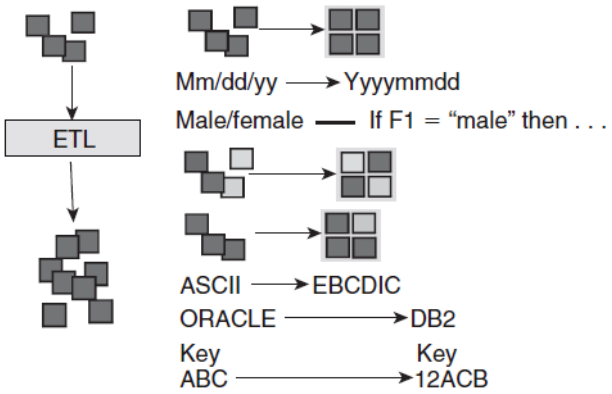


FIGURE 4.3.8 The primary purpose of ETL is transformation.

Figure 4.3.8 illustrates some typical transformations that have to be done by ETL processing. This figure illustrates the many detailed types of data transformation that may occur inside of an ETL process, including:

■ Summarization processing

■ Adjustment of date formats
■ Specification of logic—such as conversion of " Male/Female " to " M/F "
■ Aggregation of data
■ Provision of default values
■ Conversion from ASCII to EBCDIC
■ DBMS conversions
■ Restructure or creation of keys

This short list contains just a *few* of the transformation activities that need to be accomplished by ETL processing.

4.3.10 ETL and metadata

Although the primary purpose of ETL is to create corporate data from application data, ETL processing has a separate and secondary purpose. ETL can also be used to create an auditable record of the transformation of the data—a data transformation audit trail that needs to be placed into metadata.

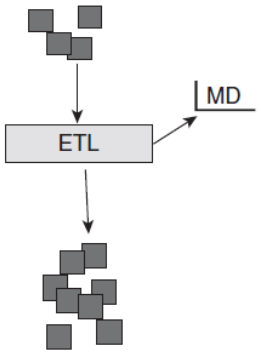


FIGURE 4.3.9 Metadata is a by-product of ETL processing.

Metadata then becomes an important secondary by-product of ETL processing, as suggested in Figure 4.3.9. Metadata is a natural by-product that is easily produced by ETL processing. The source-to-target data mapping that the analyst creates during ETL design is itself a metadata design. In fact, all source-to target data mapping really is nothing but a data transformation audit trail and therefore is metadata. Source-to-target data mapping is "data about data" specifically data about how data is to be transformed on way into the data warehouse, and therefore the design for ETL data transformation tracking. The ETL data transformation metadata can be a very useful analytical tool for the decision support analyst. End users often want to find out more about the data used in an analysis. The metadata by-products of ETL data transformation processing can be a handy first step toward satisfying the end user.

4.3.11 ETL and an audit trail

Just as it is important as capturing and making available metadata when source data passes through ETL processing, there is also a need for an audit trail of ETL processing.

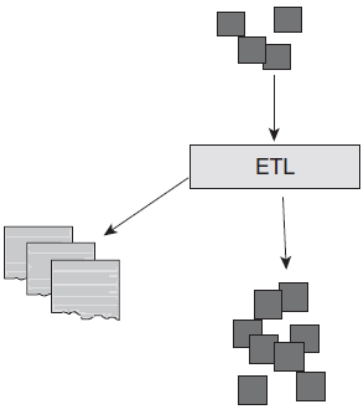


FIGURE 4.3.10 An audit trail is often left by ETL processing.

Figure 4.3.10 depicts the audit trail that can be left by metadata passing through an ETL process. The audit trail is very similar to metadata. However, the audit trail contains much more detailed data concerning the passage of data through the ETL process. Typically data transformation audit trail records include:

- Number of records processed in a day
- Successful job completion indicators for a batch run through the ETL process
- Number of ETL failures in a day
- Reasons for any ETL failures
- Description of how the failures will be resolved
- Amount of new records added to the DW 2.0 environment
- Number of records read into the ETL process
- Length of time the ETL process was up and running

4.3.12 ETL and data quality

As data passes through the ETL process, there is an opportunity to do some basic data quality examination and even editing. Because source data must pass through ETL processing in any case on its way to the data warehouse, only a few additional resources are required to assess data quality along the way. For example, the following data quality checks are easily done as part of ETL processing:

- Domain checks
- Range checks
- Reasonability checks

It should be noted that any data quality assurance done during ETL processing is limited to processing that can be done with one record in hand. In other words, if data quality requires multiple records to determine the validity or quality of data, which type of data quality verification/checking is hard to do with ETL? Figure 13.11 suggests that data quality checking can be done as part of ETL processing.

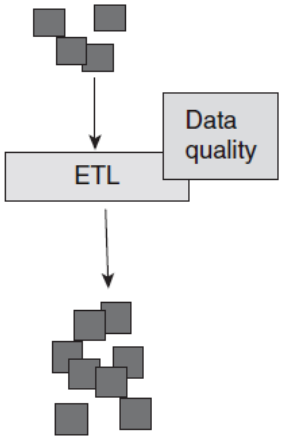


FIGURE 4.3.11 ETL provides an excellent opportunity for data quality checks.

4.3.13 Creating ETL

There are two basic ways that ETL programs can be created. One method is to create ETL programs from scratch using a programming language or tool such as VB.Net, C, or C _ _ . The other approach is to purchase a third-party vendor's software package designed to enable ETL process development, such as Essential, Informatics, or Talent.
There are pros and cons to both approaches.

The advantage of ETL processing with home-grown software is that it can perform any kind of logic that is desired and specified by the organization. Anything that can be programmed can be included in an ETL program. The disadvantage of home-grown code is that most ETL programs are very standard routines that do not require any special processing. Furthermore, maintaining code over time can be an undesirable, expensive, resource-consuming approach. The advantage of third-party software is that ETL processes based on third-party software can be created much more quickly than homegrown software. In addition, third-party software can be maintained much more easily than home-grown code. The problem is that almost always a few special data transformation algorithms are needed that are so complicated that they are difficult to incorporate into any preprogrammed third-party technology.

A good solution is to build 95% of the needed ETL transformation programs using a third-party tool and build the remaining 5% or less of the required ETL programs with home-grown code.

4.3.14 Code creation or parametrically driven ETL

Some ETL operates on the basis of code creation, and other ETL code operates on the basis of a parametrically fed program that does not generate special code. Figure 4.3.12 depicts the choices in the way ETL is to be implemented.

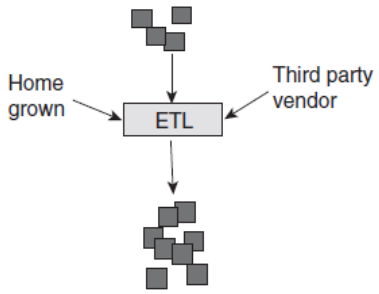


FIGURE 4.3.12 ETL can be homegrown, from a third-party vendor, or both.

4.3.15 ETL and rejects

One of the major challenges of ETL processing is how to handle rejected data. What happens when a source system record contains a value that is incorrect? There are several approaches for handling of this circumstance. Each approach has its advantages and disadvantages.

The first approach is not to let rejected data pass into the data warehouse environment at all. Although this approach maintains pristine data in the data warehouse, it can prevent the inclusion of other data that might be correct and useful. For example, suppose a transaction has 10 elements of data. If one element of data is incorrect, does that warrant exclusion of the other nine valid pieces of data from the data warehouse?

A second approach is to supply default values for data that is known to be incorrect. This approach allows all of the accompanying valid data to be entered into the data warehouse. The important caveat is that this approach may produce very irregular or questionable results if/when data summaries are made or other data analysis is done. The third approach is to place invalid data in the data warehouse and then flag it so analysts know that the data is invalid. A fourth approach is to create a reject file. This is a good approach as long as the reject file can be reconciled in an automated manner. But if the reject record has to be reconciled manually, then this is usually a very poor approach and is not recommended.
Figure 4.3.13 depicts the creation of a reject file.

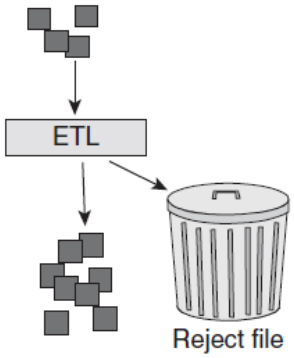


FIGURE 4.3.13 Reject files must be handled carefully.

4.3.16 Changed data capture

A special form of data input into the DW 2.0 Interactive Sector is “changed data capture,” or CDC. CDC is done when a log tape is created as a result of making transactions. It is much more efficient and accurate to capture transaction activities off of a log tape than it is to search a file looking for possible activities. But there are several problems with using log tapes for this purpose.

The first problem is that operations are often unwilling to part with what amounts to one of their essential cornerstone technologies. Log tapes are needed during backup and recovery. Should operations lose one of these tapes, the results could be disastrous. The second problem is that backup log files are created for the purpose of data recovery, not for input to the data warehouse environment. The format of log tapes is arcane, at best, and is difficult to decipher.

Figure 4.3.14 depicts the usage of a log tape as input to the ETL process.

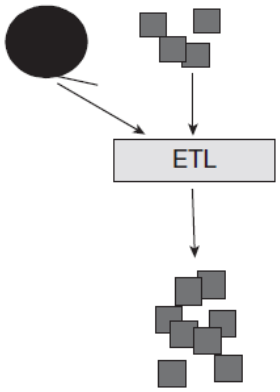


FIGURE 4.3.14 Changed data capture logs are sometimes very useful.

4.3.17 ETL

Extract/load/transform (ELT) processing is closely related to but not the same thing as ETL. The difference between ETL and ELT is that in ETL, data is extracted, transformed, and then loaded into the data warehouse, whereas in ELT processing data is extracted and loaded into the data warehouse and then transformed. In other words, with ELT, the data is not transformed until after it actually arrives in the data warehouse. There are some fundamental problems with ELT. The first problem is the cleanliness of data. It is entirely possible to extract and load data and then forget to do the transformation process. This, of course, is not acceptable.

The ELT approach also invites data integrity problems when data transformation is done after the data has been loaded into the data warehouse. At one moment in time there is a value of 100 for some unit of data. At the next moment there is a value of 35. At this point the data warehouse has lost its credibility. Stated differently, with ELT there is no integrity of data in

the data warehouse environment. Figure 13.15 depicts an ELT process. Because of its inherent problems, ELT processing is not a good architectural choice.

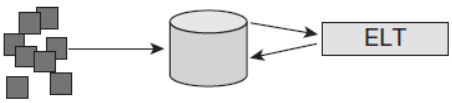


FIGURE 13.15 Another form of ETL is ELT.

4.3.18 LET US SUM UP

ETL is the process used to change the state of data. Data changes from an application state to a corporate state after passing through ETL. ETL processing occurs as data passes into the Interactive Sector and again as data passes from the Interactive Sector to the Integrated Sector. ETL can be run in an online mode or a batch mode. When it is run in an online mode the emphasis is on the movement of data. When it is run in a batch mode, the emphasis is on the transformation of the data. The place where data comes from is called the source. The place where data goes to is called the target. The logic that shows how to get from the source to the target is called the mapping. The collective source of the data for all data is called the system of record. ETL processes can be run in a parallel manner if there is a lot of data that needs to go into the target.

4.3.19 List of References

- DW2.0 The architecture for Next Generation of Datawarehousing W.H. Inmon, Derek Strauss, Genia Neushloss, ELSEVIER.
- Building the data warehouse, W.H.Inmon, third Edition, Wiley.
- Datawarehousing, S. Mohanty, TMH .
- The Data Warehouse Lifecycle toolkit", Ralph Kimball ,John Wiley.

4.3.20 Unit End Exercises

1) Explain audit trail with respect to ETL.
2) Define role of ETL in brief.
3) Explain ETL in batch mode.
4) Explain ETL processing in online mode.
5) Explain different approaches for handling rejected data in ETL processing

Unit V : Data warehousing

Chapter 1 : DW 2.0 and the granularity manager

1.0 Introduction: -

Nearly all of the data that passes from external sources to the Interactive Sector, or occasionally directly into the Integrated Sector, passes through an ETL process, for good reason. But on rare occasions, there is need for another type of process to move data from external sources.

1.1 The Granularity Manager

This type of processor is called a granularity manager. The granularity manager does something quite different from what an ETL program does.

Consider what an ETL process does. Figure 1.1 depicts normal ETL processing at a conceptual level.

Figure 1.1 represents the customary storage of different kinds of data by subject area following normal ETL processing. ETL reads a source record and then sends different parts of the record to different destinations based on the subject area to which they relate. For example, suppose a transaction is executed and produces a record about the sale of an item. The item sales data reaches the ETL process as a single record. However, in the record are elements of information about very different kinds of data, such as information about revenue, about products, and about a customer. The ETL process knows that these different types of

information need to be sent to different destinations. The ETL process turns the sales application transaction data into corporate, subject-oriented data when it grooms the source data to be loaded into the target data warehouse. This is one of the primary, most important functions of ETL processing.

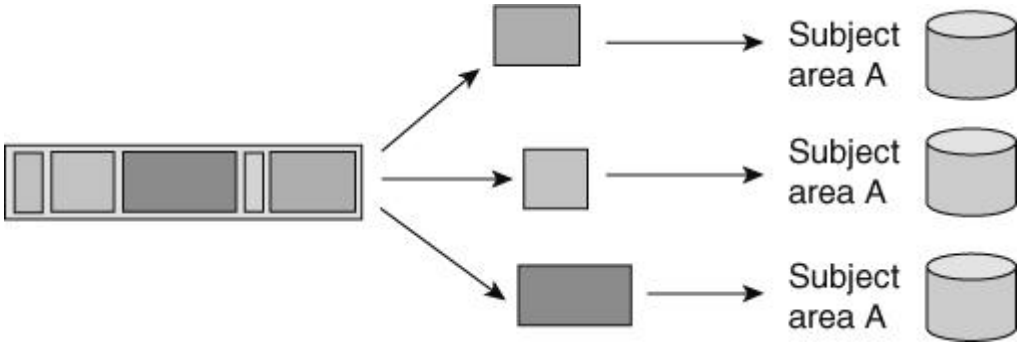


FIGURE 1.1 The normal progression of data.

1.2 Raising The Level Of Granularity

Data granularity management is quite different. Instead of separating data contained in a single record into subject-oriented units of data, a granularity manager actually brings data together. The need to combine or unify data does not occur very often, but it does occur. Data needs to be unified on the rare occasions when the source data state is at too low a level of detail in the external world. Click-stream data produced during Internet processing is a good example of when there is a need for a granularity manager.

When web site processing is tracked, every movement of the cursor, every movement to another page, and every entry into a new area on the screen generates a click-stream record. The activity that occurs on the web site is tracked down to the lowest level of detail. The problem with this very low-level click-stream data tracking is that most of it has no business value whatsoever.

It has been estimated that more than 90% of the data that ends up as click-stream data is useless. The nature of click-stream data tracking is that it generates a lot of this data and the data gets in the way. The useless data found in the click-stream data can create an enormous and totally unnecessary overhead.

Another example of source data at too low a level of granularity occurs when manufacturing data is collected by an analog computer. Most of the analog data is specious, but some of the analog data is really important. The important data is at too low a level of

granularity, but it is needed in the DW 2.0 environment, so it must be processed by a granularity manager before entry.

1.3 Filtering Data-

Figure 1.2 illustrates a large volume of data entering the work stream through a mechanism such as click-stream processing and then being "filtered," aggregated, or summarized.

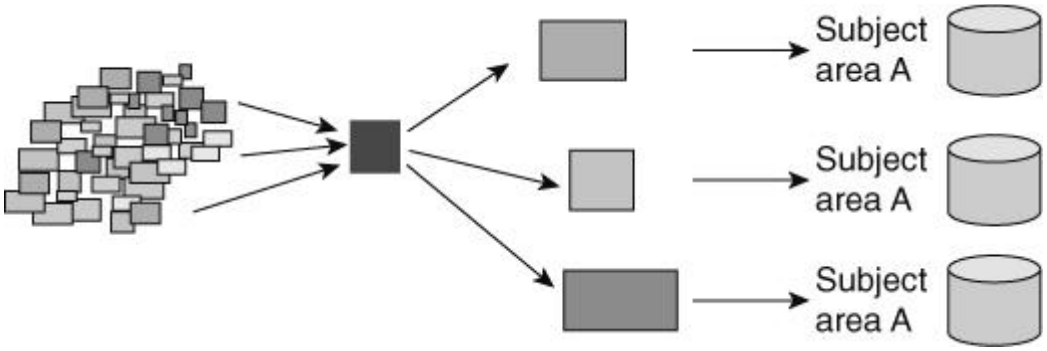


FIGURE 1.2 There are some types of data the granularity of which is too low for the data warehouse.

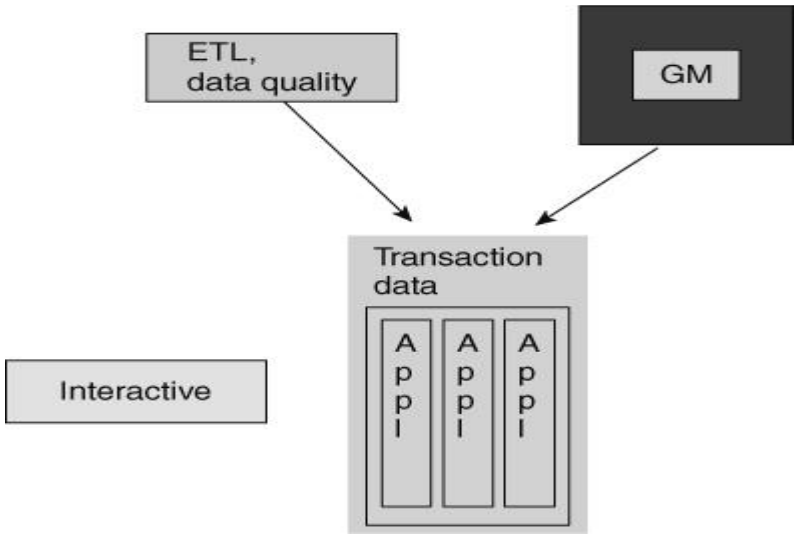
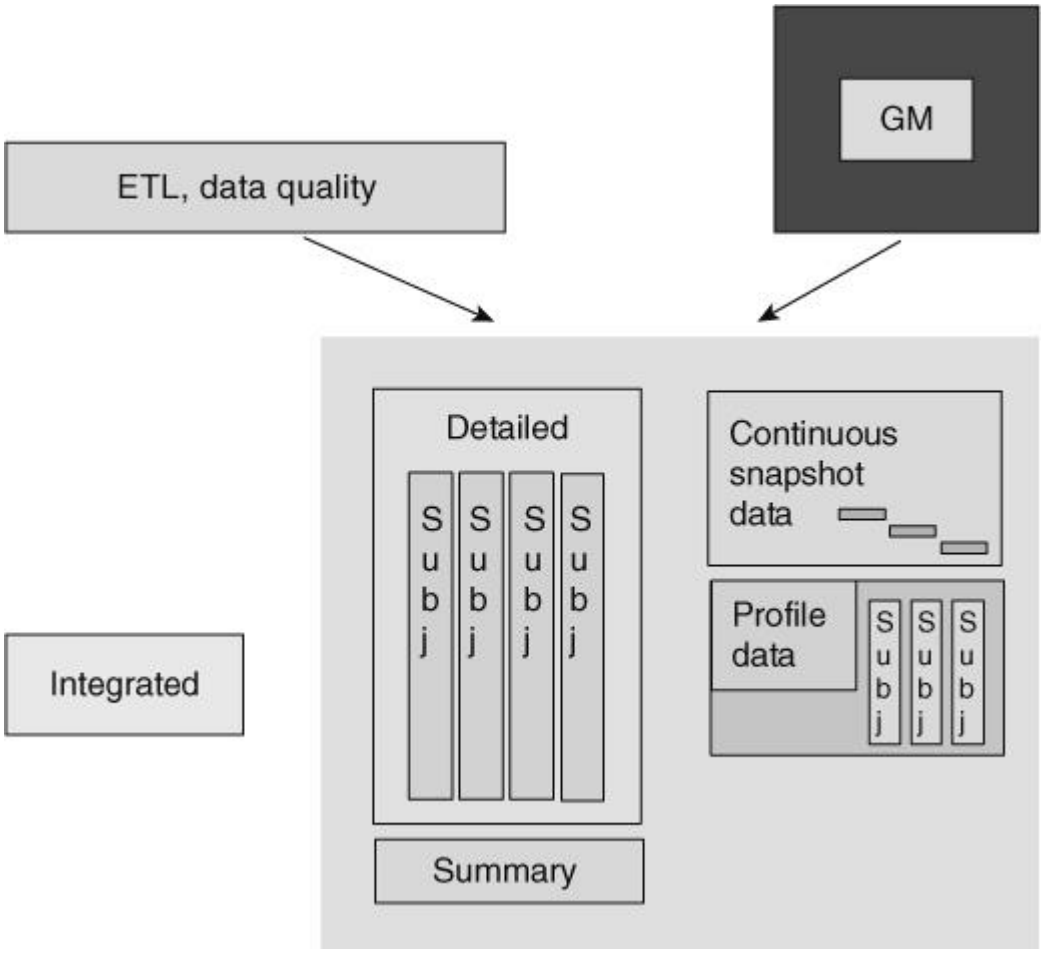


FIGURE 1.3 One place the granularity manager fits.

The granularity manager can be located when needed in several places in the DW 2.0 environment. Figure 1.3 illustrates one place where the granularity manager may be found as data enters the Interactive Sector.

A second place where a granularity manager may be required is where data enters the Integrated Sector directly. This is the case when there is no reason for data to reside in the Interactive Sector before it enters the Integrated Sector. Figure 1.4 depicts a data granularity manager positioned to process data that passes directly into the Integrated Sector.

There is one other place where the granularity manager is occasionally used, and that is when data is passed into the archival environment.



■ **FIGURE 1.4** Another place the granularity manager fits.

Figure 1.5 depicts the granularity manager's role as data enters into the archival environment.

The need for a granularity manager as data enters the archival environment is a rare occurrence. It makes sense to use a granularity manager here only when massive amounts of transactions occur in the Integrated Sector and the details of those transactions are never going to be needed for analysis.

1.4 The Functions Of The Granularity Manager

The functions that a granularity manager accomplishes are very straightforward. The granularity manager does, at the least, the following:

■ Eliminate unneeded data: Records in the input stream that will have no future value to the corporation are discarded. This may be as much as 90% of the incoming data.

- Summarize: Data that will have future value to the corporate analysts can often be summarized from multiple records into a single record.
- Aggregate: Occasionally aggregation of different kinds of data into a single record makes more sense than data summarization.
- Recast data: When data is recast it enters as input in one format and structure and exits in a different format and structure. Recasting data that originates at too low a level of granularity is a very common thing to do.

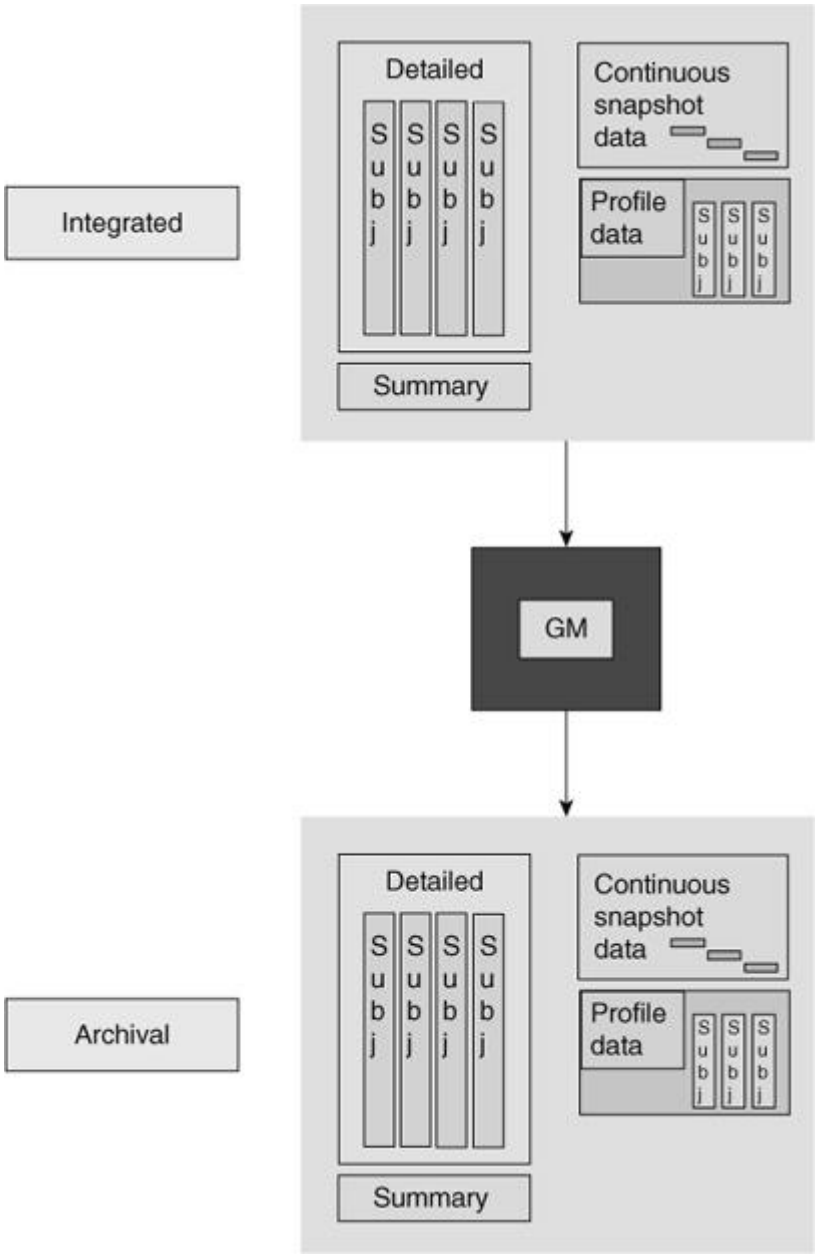


FIGURE 1.5 Yet another place the granularity manager is found.

These are the activities that are accomplished by a granularity manager. The net result is a tremendous compaction of data and a weeding out of useless data. Figure 14.6 illustrates the functions of granularity management.

1.5 Home-Grown Versus Third-Party Granularity Managers

The data granularity management module may be created by third-party software vendors or may be home grown. There is a strong case for using third-party ETL software. The rationale for using a third-party granularity management software package is not nearly so strong. There are far fewer cases in which granularity management will be needed compared to when ETL will be needed. Customized processing requirements for granularity management are very common. Consequently, it is not unusual for home-grown software to be used for granularity management.

Figure 1.7 illustrates the home-grown and third-party options for granularity management.

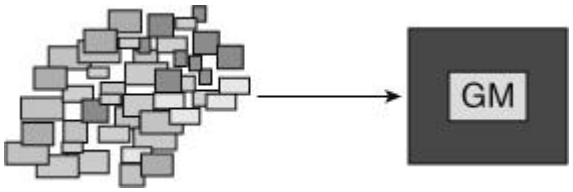


FIGURE 1.6 Granularity manager functions: eliminate, summarize, aggregate, recast.

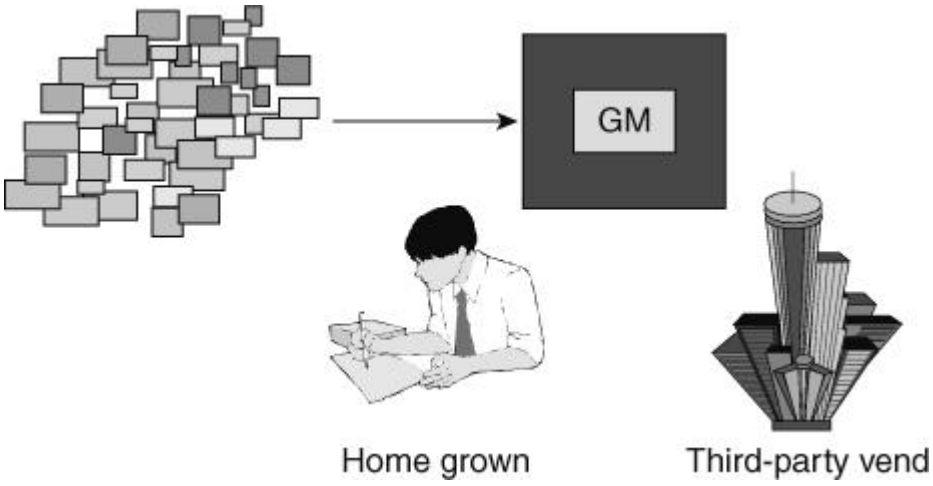


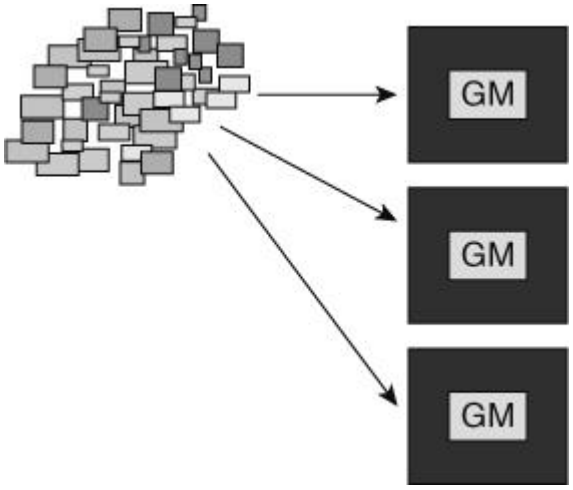
FIGURE 1.7 There are two basic ways the granularity manager is created.

1.6 Parallelizing The Granularity Manager

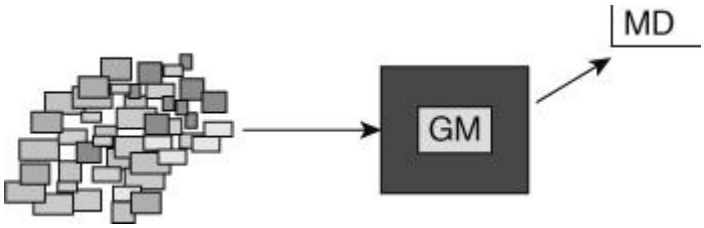
On occasion there will be *huge* amounts of data that will need to be processed by a data granularity manager. It is possible to run granularity management software in a parallel fashion to accommodate a large processing load. By running two or more instances of the data granularity management software in parallel, the elapsed time needed for processing can be dramatically reduced.

Figure 1.8 depicts data granularity management routines running in parallel.

In addition to compacting data into a meaningful and useful size, granularity management can also be used for the production of metadata. Figure 14.9 depicts metadata as a by-product of granularity management.



■ **FIGURE 1.8** If warranted, the granularity manager can be run in parallel.



■ **FIGURE 1.9** It is useful to produce metadata as a by-product of processing in the granularity manager.

The metadata by-products of data granularity processing may include information about the following:

- What data has been discarded
- What data has been summarized and what the summary record looks like

■ What data has been aggregated and what the aggregate record looks like

■ How data has been recast and what the recast record looks like

The metadata summarizes the results of granularity management processing.

1.7 Summary: -

On occasion data exists outside of DW 2.0 at too low a level of granularity. When this is the case the data needs to be passed through a granularity manager to raise its level of granularity before it is loaded into the data warehouse.

The data granularity manager weeds out, summarizes, aggregates, or otherwise restructures data as it passes into or between the DW 2.0 sectors.

Data granularity managers can be run in parallel. The granularity manager can be home-grown software or third-party software.

Metadata is a by-product of processing data through a granularity manager.

1.8 References and Bibliography

- ✓ DW2.0 The architecture for Next Generation of Datawarehousing W.H. Inmon, Derek Strauss, Genia Neushloss, ELSEVIER.

1.9 Exercise

1. Write a short note on Granularity manager.
2. Explain Raising the level of granularity.

Unit V : Data warehousing

Chapter 2 : DW and performance

2.0 Introduction,

2.1 Online response time

2.0 Introduction: -

An information system that does not yield adequate performance is an abomination. Most people abhor waiting, especially when waiting is unnecessary and unproductive. Peoples' attitudes toward waiting are no different in the DW 2.0 environment.

An information system that does not provide good performance is simply ineffective. If performance is bad enough, the information system is useless. It is simply assumed that good performance is a feature of a good information system.

Yet good performance is not a feature that can simply be added into a system like a new function. Good performance has many facets and must be holistically designed and built into the system from the start.

Good performance is essential for an effective DW 2.0 data warehouse, and it is necessary throughout the DW 2.0 environment.

There are different kinds of performance applicable to the DW 2.0 environment. In the Interactive Sector there is online response performance, or OLTP—online transaction processing performance. OLTP performance is measured in seconds. In the Integrated Sector there is analytical performance. Depending on the analytical activity, response time may be measured in seconds or minutes. In the Archival Sector response time may be measured in terms of days. Response time is relative. Depending on what is being done and where, there is a different set of response-time measurement and expectations. Yet response time is important throughout the DW 2.0 environment.

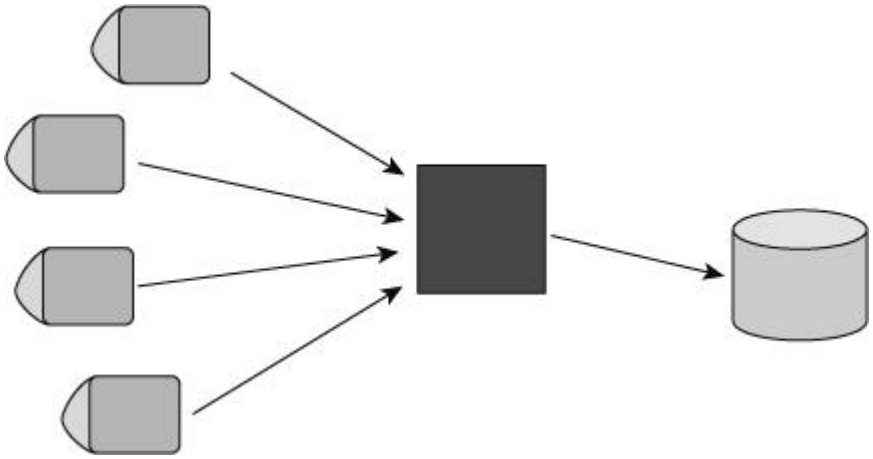


FIGURE 2.1 Transaction response time—2 to 3 seconds consistently.

2.1 Online Response Time-

The normal expectation for online response time is 2 to 3 seconds. Online response time is often called real-time response time. Online response time is achieved in the Interactive Sector.

Figure 2.1 depicts online response time.

Online response time is measured from the moment the user presses the key that sends the transaction to the computer until the first of the response is returned to the user. A lot of processing happens from the time the transaction is entered until the first of the response is returned. The transaction is delivered to the network. The network then passes the transaction into the system. After passing through security protocols, the transaction enters a queue. It waits in the queue until system resources are available for its execution. The transaction then enters the application area of the computer where the transaction is processed. Computer code is executed and calls to the data base are done through the DBMS. The data is gathered from the data base(s) and returned to the application area. The application performs logic using the transaction, makes decisions, and may write more data into the data base. Finally, when the transaction is complete, it leaves the application area, and the final results are sent back through the network where they are displayed to the user who initiated the transaction. All of these activities must be completed in seconds to achieve good response-time performance.

The enemies of good transaction performance include:

- Lengthy activities, such as traveling up and down a network
- Input/output (I/O) into and out of a data base
- Wait times in queues or elsewhere
- Large volumes of competing transactions

2.2 Analytical Response Time

The other type of performance consideration for the DW 2.0 environment can be called the analytical performance measurement. The analytical performance issues are relevant to the Integrated Sector, the Archival Sector, and occasionally the Near Line Sector.

Analytical response-time expectations are generally in the 10 seconds to 1 hour time range. Given the large amounts of time that performance is measured in for analytical processing, it is often assumed that performance in the analytical environment is easy to achieve. That is not the case at all.

The analytical environment is very different from the transaction environment. In the analytical environment

- the decisions being made are much more strategic than tactical;
- there is often *much* more data needed for analytical processing than is needed for transaction processing;

- there is a much less predictable flow of data throughout the day.

These fundamental differences between the transaction environment and the analytical environment allow for a much more relaxed approach to analytical response-time performance.

2.3 The Flow of Data

The flow of data through the system is generally the same as that described for transactional processing. There are two places where there are key differences in data flow:

- The volume of data required by analytical processing is significantly greater than that needed for transaction processing. Gathering large amounts of data for analysis requires lots of data input/output activity. I/O activity slows the computer down significantly, because I/O operates at mechanical speeds instead of electronic speeds like other operations inside the computer.
- The queue time for an analytical transaction is dependent on the other analytical activities that are in the queue waiting to be executed. In the case of analytical activities, the queue time is both unpredictable and potentially long.

Response time for analytical activities are therefore often long and unpredictable.

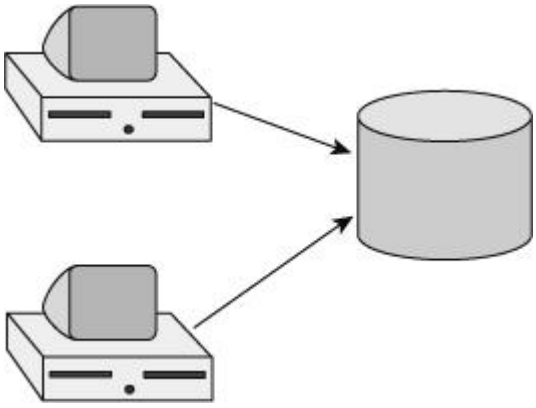


FIGURE 2.2 Analytical response time—15 seconds to 24 hours.

Figure 2.2 illustrates analytical activities being done against data.

Any discussion of performance must address the impact of poor performance. The impacts of poor performance on transactional processing and analytical processing are quite different.

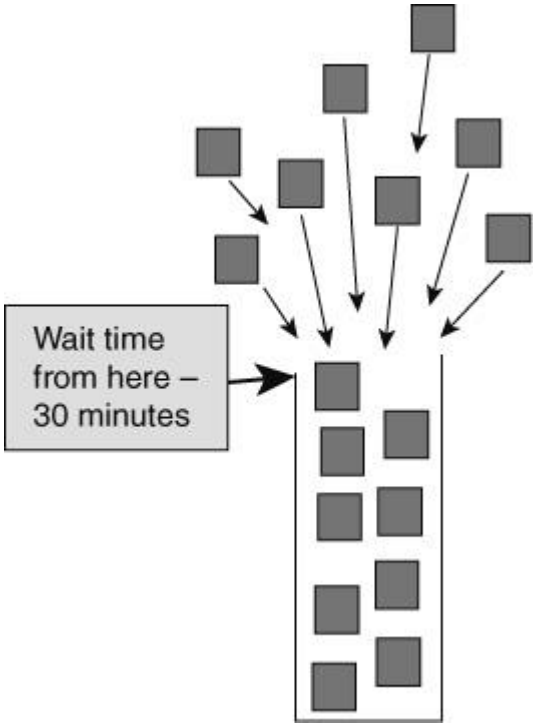
2.4 Queues

When the performance of transactional processing goes bad, there is an impact on the business of the corporation, at the place where customers meet the corporation. Waiting queues form. Consider a bank or an airline check-in counter. Under normal operation, when customers arrive at the bank or the airport, there may be a few people in line in front of them—on a good day there will be just two or three people in line. On a bad day at the bank or airport, there may be 20 people in line. The speed with which customers move up in the line and receive service is a function of the length of the waiting line and their position in it.

To achieve smooth operations, the bank clerk or the airlines reservation clerk must have good response time as the clerk is transacting the business of the customer. Response time must be almost immediate and must be consistent. When transaction response time is good, the clerk can serve each customer in a short amount of time, and the customer waiting line advances at a steady, reasonable rate.

But consider what happens when the bank or the airline clerk does not get good system-response time. It takes a long time to serve any single customer, which is bad for the customer who has the clerk's attention. But it is even worse for the customers who are waiting in line. The queue builds and builds until there is an unacceptably large number of people in the waiting line.

So the real negative impact on transaction response time is not the execution of any given transaction, but rather the cumulative effect on the transactions that build up in the queue waiting to reach the position of execution. The worst part of poor transaction performance and long queue times is the negative impact that is felt directly by the customers of the enterprise. Figure 2.3 illustrates the intolerably long queue that can form when basic transaction processing goes bad.



■ **FIGURE 2.3** There is a very negative impact on the business when OLTP response time goes bad.

Poor performance in the analytical environment has an equally negative impact. It affects the productivity of the analyst preparing information for management.

2.5 Heuristic Processing

The negative impact of poor performance on the analytical community is related to the way in which analytical processing is done. Figure 15.4 depicts the heuristic nature of analytical processing.

When processing is done heuristically, there can be very little planning of the analytical activity, because each step in a heuristic analysis depends entirely on the results achieved in the previous step.

An organization sets out to do heuristic analysis. The first results are obtained. The results are analyzed. Only then does it become clear what the next step of analysis needs to be. The second step is done and more results are achieved. Now the results of the second step are analyzed, and the activities for the third step of analysis become clear. The process continues until the final results are achieved.

The fact is that in heuristic analytical processing, there can be no organized, planned path through the analysis.

2.6 Analytical Productivity and Response Time

The speed with which heuristic analysis can be done depends entirely on how fast an analytical process can be done. Stated differently, the faster analytical processing can be done, the faster the final results can be achieved.

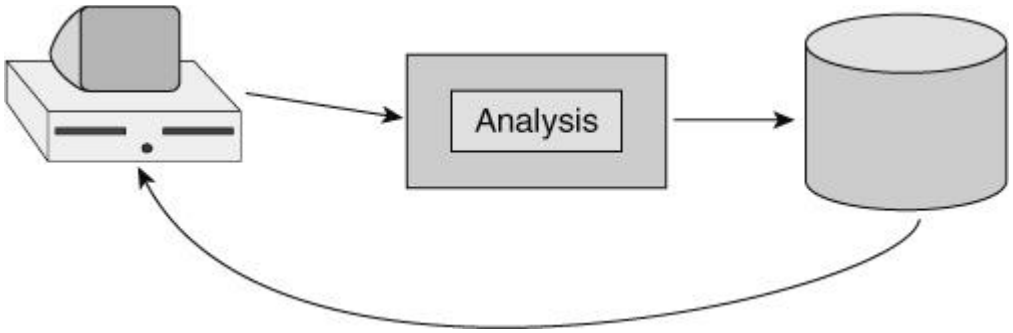


FIGURE 2.4 Heuristic processing is the essence of analytic processing.

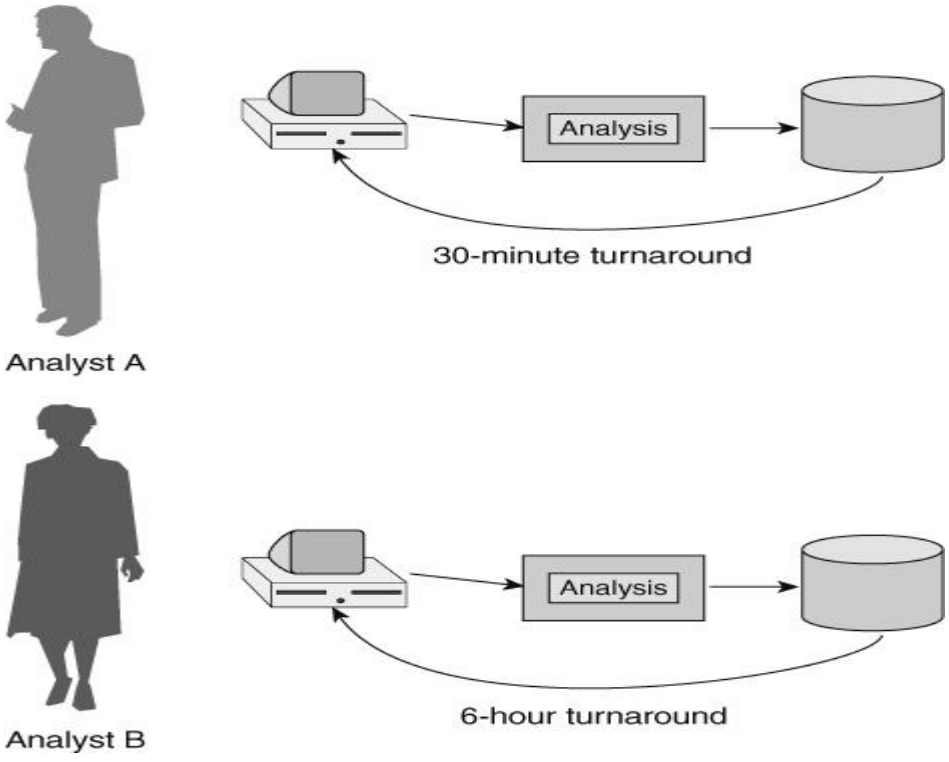


FIGURE 2.5 Consider the productivity of two analysts whose only difference is in the speed of their analytical turnaround.

Figure 2.5 contrasts the performance of two analysts.

The only real difference between the two analysts shown in Figure 2.5 is the speed with which analysis can be done. Analyst A is able to turn his heuristic analysis around in 30 minutes. Analyst B needs 6 hours to turn her heuristic analysis around.

There is another major difference in poor performance between transactional processing and analytical processing. In transactional processing poor results are felt by the customers of the corporation. In contrast, poor analytical results are felt internally, by the analysts of the corporation. Poor transaction processing response times are much more devastating in the short term for the corporation.

2.7 Many Facets to Performance

There are many facets to achieving good performance. A holistic approach to performance is required. Performance must be addressed on many fronts, all at the same time. The following sections describe what needs to be done to achieve good long-term performance in the DW 2.0 environment.

2.8 Indexing

One of the simplest design solutions for data warehouse performance is to create an index of the data. If this is done, the system does not have to do a sequential search of all of the data to locate and retrieve requested information.

Figure 2.6 depicts an indexing of data.

A data base without an index is shown at the top of Figure 2.6. When a person accesses this data base, the entire data base must be searched to find a record.

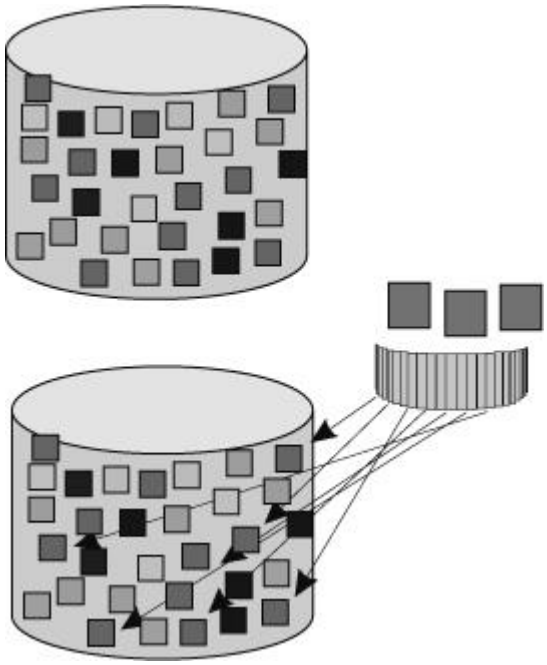


FIGURE 2.6 Searching for data without an index and with an index.

The data base on the bottom of Figure 2.6 has an index. Now when a person wants to find something in the data base, the index is consulted first. If the data can be found in the index, it is accessed directly. The result is a tremendous savings in the time required for the access of data.

It would seem then that indexing all data in a data base is a good thing to do. There is, however, a cost of indexing data. The index requires space, and if the data is constantly being updated in the data base, the indexes have to constantly be adjusted. When an online transaction occurs, the overhead of adjusting a lot of indexes can become prohibitive.

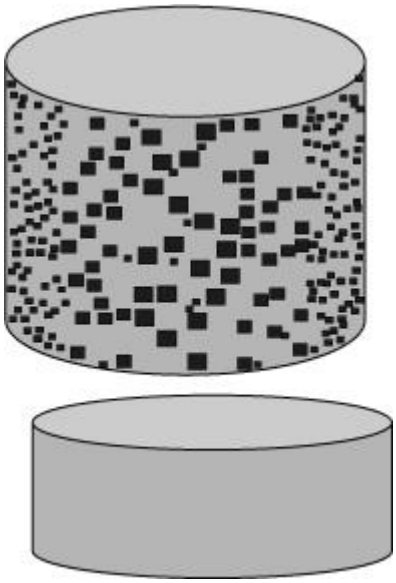
Therefore, there is a trade-off to be made. Although indexes can save tremendous time in accessing data, additional processing and data storage overhead are associated with the creation and maintenance of the indexes as well.

2.9 Removing Dormant Data

A second great way to enhance data warehouse performance is to remove dormant data. Dormant data is data that is no longer accessed or that has a very low probability of being accessed. Dormant data affects performance negatively by clogging up the machinery. It acts like cholesterol in the body. The more cholesterol there is in the body, the harder the heart has to pump just to keep the body alive. The less cholesterol there is, the more efficient the circulatory system is.

Every computer system has some amount of dormant data in it. It is only when there becomes an overwhelming amount of dormant data in the system that performance is hurt.

Figure 2.7 depicts two data bases. The data base on top contains a lot of dormant data. The data base on the bottom contains very little dormant data. It is *much* more efficient to access the data base on the bottom than it is to access the data base on top, with all of the dormant data in it.



■ **FIGURE 2.7** Accessing a data base with no dormant data is much more efficient than accessing a data base with dormant data.

It is then a very good practice to remove dormant data from the data base environment. Whereas the removal of dormant data is applicable to the Interactive Sector, the removal of dormant data is especially relevant to the Integrated Sector.

2.10 End-User Education

There is another very simple idea that can produce really good results in the enhancement of performance. That idea is to educate the end users before they are allowed to use the data warehouse. There are two aspects to the end-user education that should be emphasized:

- What data is in the data warehouse, and what the structure and format of the data are?
- How to create efficient data queries?

A little end-user education goes a long way toward ensuring efficient data warehouse use and processing performance.

End-user education is applicable throughout the DW 2.0 environment, but is especially applicable to the Integrated Sector and the Archival Sector.

2.11 Monitoring The Environment

Another technique for achieving good performance throughout the DW 2.0 environment is to monitor the environment. It is amazing how many people have important data bases and technology infrastructures that go unmonitored. A performance monitor is an excellent tool for detecting and diagnosing problems when they happen, as they inevitably will. If a problem cannot be diagnosed, then the necessary cure is only a guess.

There are two kinds of monitors that are relevant to the DW 2.0 environment: a transaction monitor for the Interactive Sector and a data warehouse monitor for the Integrated Sector.

The transaction monitor examines the speed of transaction processing, the resources used during transaction processing, and transaction queue wait times. The data warehouse monitor looks at dormant data and at the data users are accessing. Figure 2.8 depicts the monitoring of data warehouse activity and data.

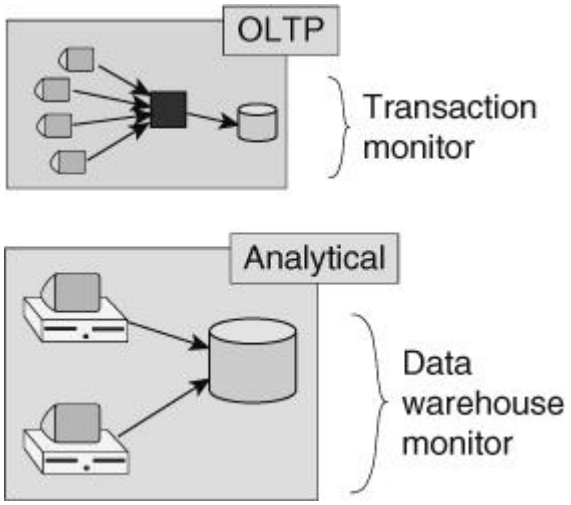


FIGURE 2.8 Monitoring the environment is a fundamentally good way of enhancing performance.

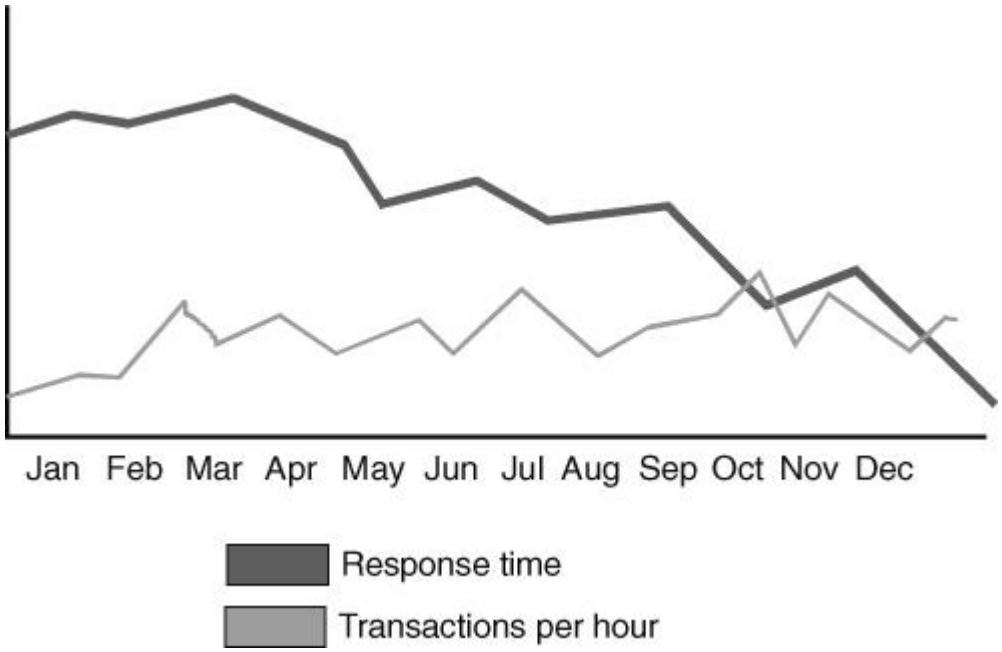
2.12 Capacity Planning

Capacity planning naturally accompanies transaction performance and data usage monitoring. The objective of capacity planning is to predict proactively when performance may go bad so that remedial steps can be taken before it happens.

Without capacity planning, performance is great until, one day, the end of the capacity is reached. Acquiring new equipment and upgrading the technology are not things that can be done quickly, so the entire organization suffers when performance goes bad and until more

capacity can be acquired and implemented. Therefore it is in the organization's best interests to be proactive, before the end of the data warehouse's capacity is reached.

Whereas capacity planning is important across the DW 2.0 environment, it is especially relevant to the Interactive Sector. Figure 2.9 illustrates how data warehouse performance monitoring can provide predictive information that is extremely useful for proactive capacity planning purposes.



■ **FIGURE 2.9** Capacity planning puts the organization in a proactive stance when it comes to performance.

Capacity planning invariably raises the issues of hardware and software upgrades. There are occasions when the capacity of the hardware and/or software running a part of the DW 2.0 environment is simply exceeded. When this happens, more raw horsepower needs to be added to the environment. Data warehouse hardware capacity upgrades may involve many facets of the hardware environment, such as:

- More main memory
- More caching
- Faster internal speeds
- Parallelization of storage management
- Additional data storage

Increasing the capacity of a data warehouse may also involve software upgrades as well as hardware upgrades. Some typical features of software upgrades include:

- Support for newer and faster operating systems
- Support for parallelized software

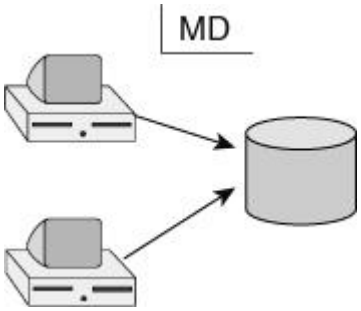
■ Support for new features in later releases of the software

There are plenty of reasons that upgrading both hardware and software can enhance the performance of a system. The task of keeping hardware and software on current releases is relevant across all sectors of DW 2.0. System upgrades are a regular part of the strategy for maintaining good data warehouse performance.

2.13 Metadata

A solid metadata infrastructure is an essential component of good data warehouse performance. The metadata infrastructure describes where data resides in the DW 2.0 environment.

Figure 2.10 depicts metadata in the DW 2.0 environment.



■ **FIGURE 2.10** The metadata infrastructure is essential to the performance of a system.

It may not be obvious that metadata has a relationship to performance, but it turns out that metadata has a very real and positive relationship to performance. Metadata is the key to reusability. Without reusability, everything must be built from scratch every time a problem or a question arises. With reusability, when an organization has already produced a result or done an analysis, it can reuse it rather than recalculating or redeveloping it. There is absolutely *nothing* that enhances performance like *not* having to do a major amount of work over again. If a major analysis does not have to be done because it has already been done, a huge amount of data warehouse resources are saved.

The reuse of existing analysis applies across the sectors of DW 2.0, but applies especially to the Integrated and Archival Sectors.

2.14 Batch Parallelization

Parallelization of processing is a really good way to enhance performance. Suppose a job stream takes 12 hours to run. If two computers can be used to do the processing, then as few as 6 hours are required. If three computers can be used, then as few as 4 hours may be required.

By parallelizing a job stream, the elapsed time required to do the job is decreased. The decrease is proportional to the number of processors. If two processors are used to do the job, then the decrease in the elapsed time is 1/2. If 10 processors are used to do a job, then the decrease is 1/10. If there are *n* processors that the job stream can be spread across, then the maximum elapsed time benefit is 1/*n*.

Note that, whereas many job streams can be parallelized, there are some job streams that cannot be parallelized.

Parallelization of job streams occurs as an approach to performance enhancement across the DW 2.0 environment. Figure 2.11 depicts parallelization to reduce elapsed processing time in the DW 2.0 environment.

The parallelization that is shown in Figure 2.11 applies to parallelization across multiple unrelated machines. This certainly is a way to achieve strong throughput, but there are other forms of parallelization that may also be useful.

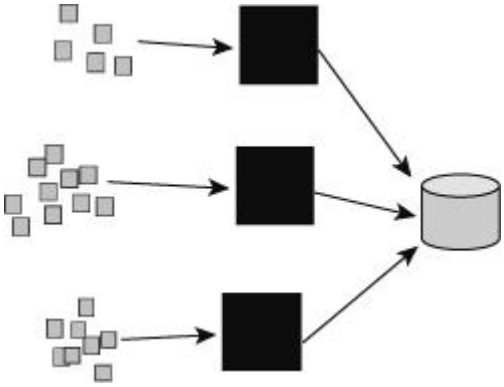


FIGURE 2.11 Parallelization of a batch job stream reduces elapsed time.

2.15 Parallelization For Transaction Processing

Parallelization works for transaction processing, not just for job stream processing. When parallelization occurs for transaction processing, the transactions are gathered and managed centrally, while the data and processing power needed to handle the processing are managed separately. The operating systems that manage this type of processing are called "shared-nothing environments." In a shared-nothing environment, each processor has and manages its own data. Each transaction is managed entirely separately from the execution of all other

transactions. The result is really fast throughput. This form of parallelization is traditionally found in the Interactive Sector of DW 2.0. Figure 2.12 shows a shared-nothing transaction processing environment.

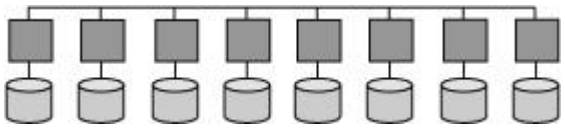


FIGURE 2.12 Parallelization of storage management is another approach.

2.16 Workload Management

The workload that passes through a system is an important factor when it comes to good and swift performance. To understand how performance relates to the workload running through the system, consider a roadway and the cars that run on the roadway. Now consider a question—how fast can a Porsche go on the roadway? With rare exceptions, the answer is not 185 miles per hour. The Porsche can only go as fast as the vehicle in front of it. If the roadway is in Mexico City and it is rush hour, then the Porsche can go about 2 miles per hour. If the roadway is the German Autobahn, then the Porsche may be able to go 185 miles per hour. But if there is a lot of traffic and there are a lot of semitrailers on the roadway, then the speeds that are achievable by all types of vehicles are not very high.

Transaction processing streams work the same way. The types of transactions, the sizes of the transactions, and the numbers of transactions that share the same processing routes all have a direct bearing on the overall data warehouse speeds that are achievable.

In the Interactive Sector, only very small transactions are allowed to execute. This means that there are fast execution times. In the Integrated Sector there is a mixture of various sizes of transactions that are typically allowed to execute. This means that mixed response time is generally expected for the Integrated Sector. Archival Sector transactions are typically very large, so response-time performance in the archival environment is often poor.

Figure 2.13 illustrates how the makeup of the workload relates to response time.

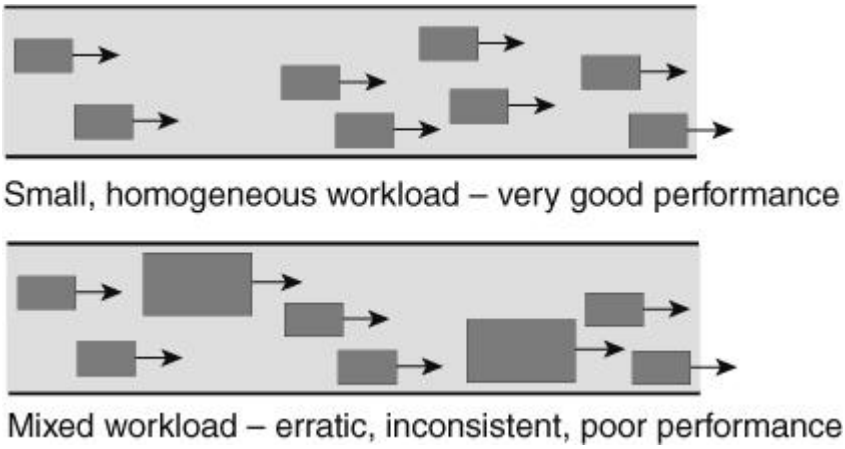


FIGURE 2.13 The makeup of the workload is a big factor when it comes to achieving good and consistent performance.

From a strategic standpoint, the movement of data to a data mart can greatly enhance performance.

2.17 Data Marts

A data mart is a collection of data that is used to satisfy the analytical needs of a group of users. Usually data marts are built along organizational lines, for example, financial, sales, and marketing data marts. Data marts are usually designed to serve a discrete group of people who all have fundamentally the same way of looking at the data.

By creating a data mart, the analytical processing can be shifted from one environment to the next. In other words, when a data mart is created, the amount of processing that is done against the DW 2.0 environment is reduced, and some of the processing load transfers to the data mart. Creation of a data mart introduces the opportunity to do something else that is very beneficial for data warehouse performance. Moving data mart processing to a physically separate processor can significantly reduce the cost of processing in the enterprise data warehouse. Given the amount of processing that occurs in the DW 2.0 environment, the processor(s) that may be required to support it can be very expensive. Moving analytical cycles out of the DW 2.0 enterprise data warehouse environment can reduce processing costs because data marts usually require significantly less hardware support. So there are other powerful motivations for building data marts than just the enhancement of performance of the DW 2.0 environment.

Building of data marts applies almost exclusively to the Integrated Sector of the DW 2.0 environment. Figure 2.14 depicts the creation of data marts as an approach to the enhancement of performance.

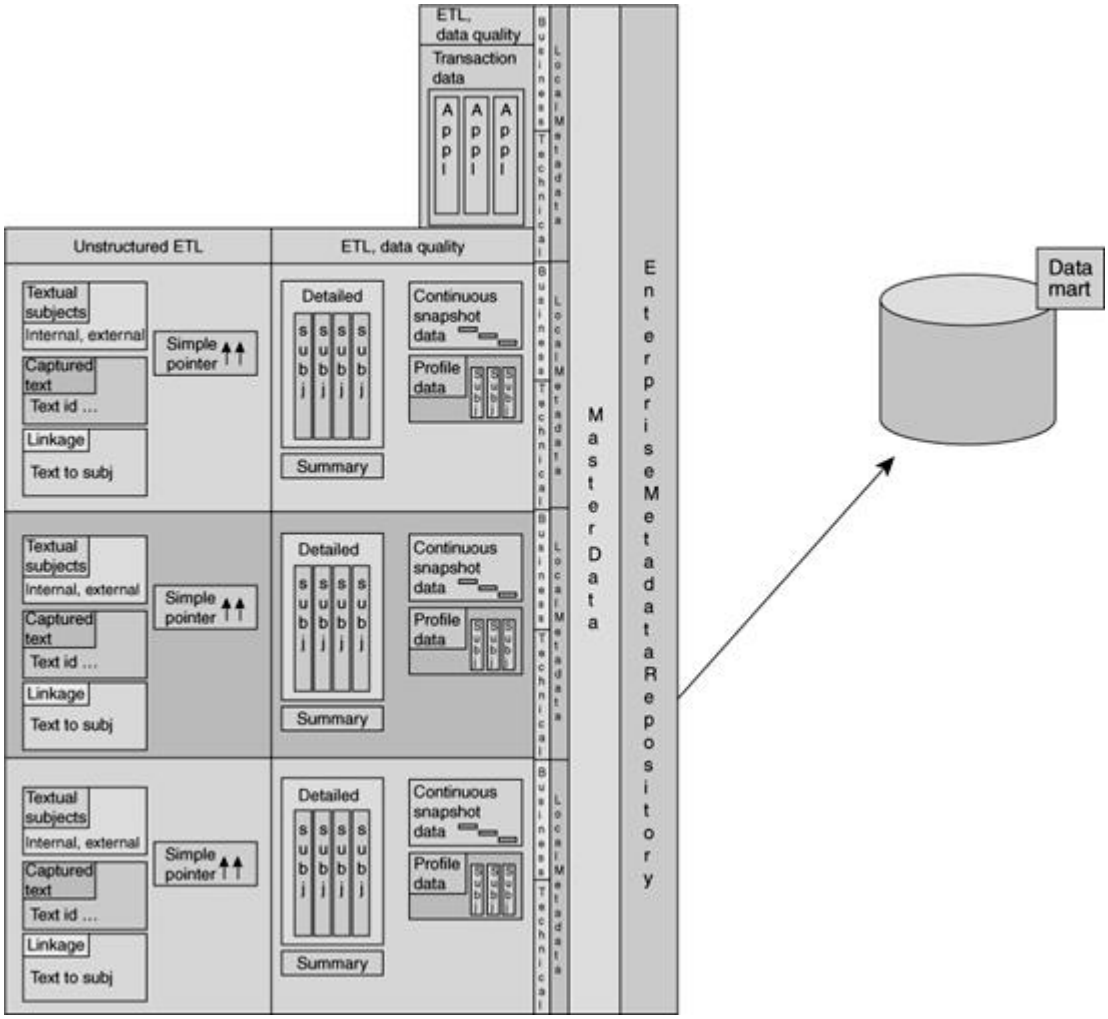


FIGURE 2.14 Moving data to a data mart is another way to enhance performance.

2.18 Exploration Facilities

The creation of exploration facilities also enhances DW 2.0 performance in much the same ways as data marts. Figure 2.15 depicts the creation of exploration facilities to enhance the performance of the DW 2.0 enterprise data warehouse environment.

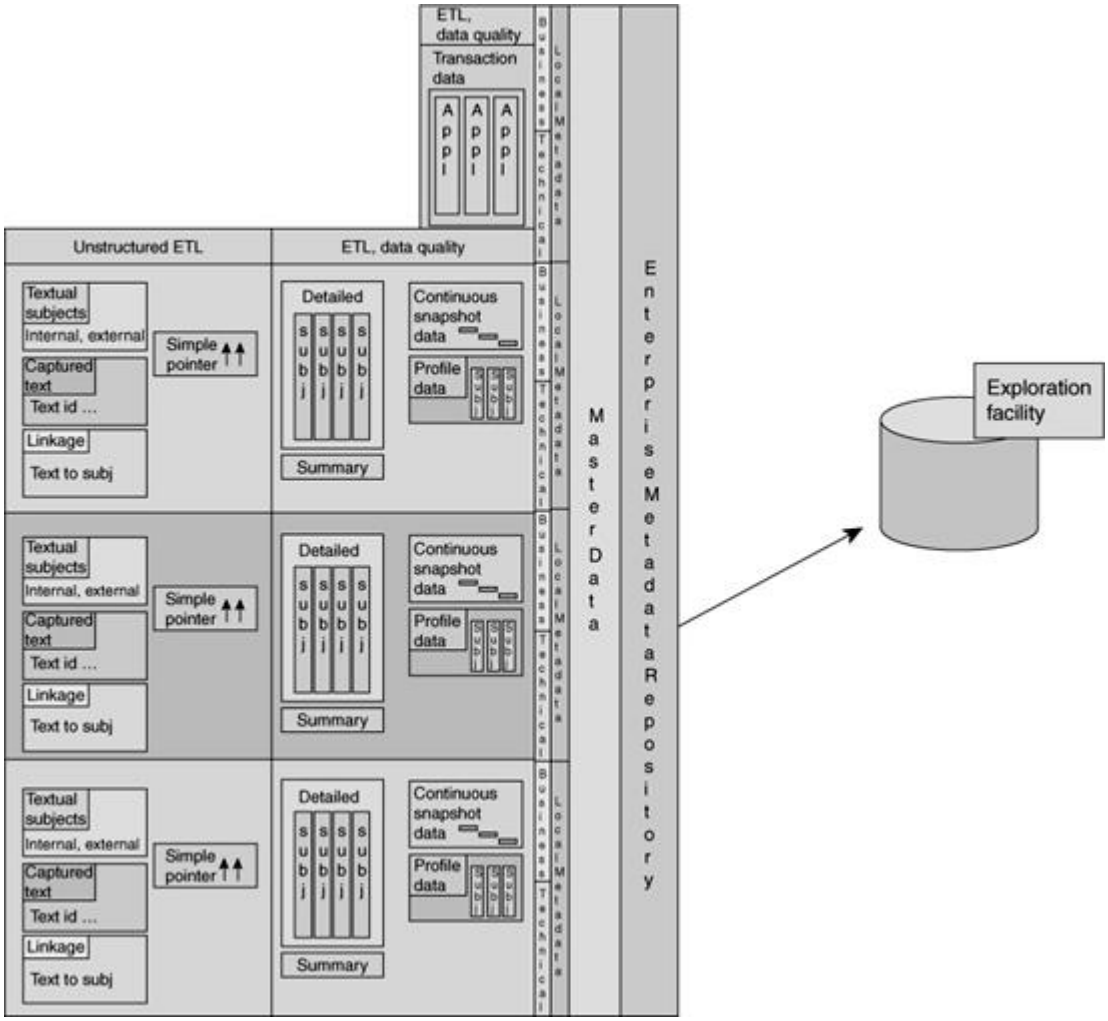


FIGURE 2.15 Moving data to an exploration facility is yet another way to enhance performance.

2.19 Separation of Transactions Into Classes

The separation of transactions into different classes is another good way to enhance data warehouse performance. Figure 2.16 illustrates the separation of transactions into two different classes.

One of the interesting questions about the technique of the separation of transactions into different classes is, How does an organization determine if a transaction is going to be fast or slow? As a general rule, the more data a transaction accesses the slower it is going to be. If a transaction is going to access a whole data base, then the transaction is likely to be slow. If a transaction accesses only a small amount of data, then the transaction is likely to run in a short amount of time.

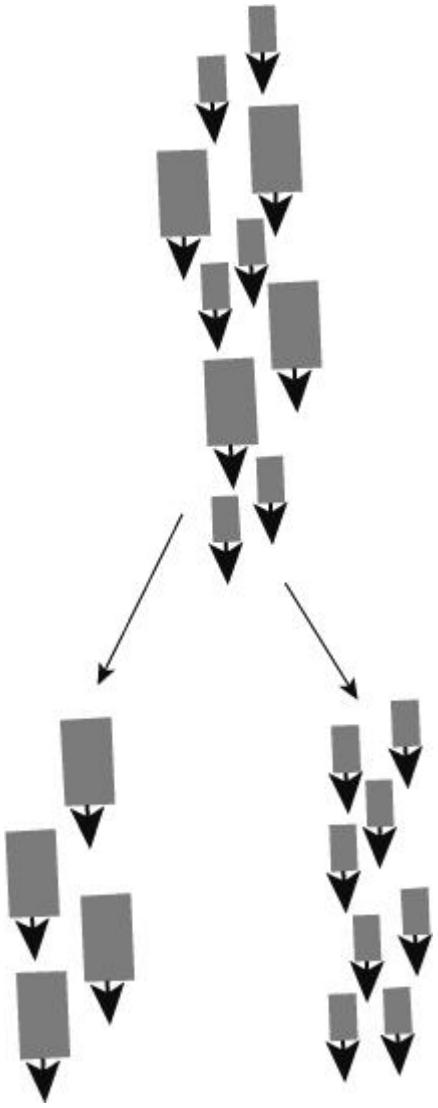


FIGURE 2.16 Dividing the work stream into processes that are going to take a lot of resources and processes that will not take a lot of resources is a very good thing to do.

2.20 Service Level Agreements

One way to govern the execution of transactions once the speed of the transaction has been determined is to create what are called "service level agreements," or SLAs.

An SLA is a statement of the expected levels of service. Normally the SLA is for transaction response time and system availability.

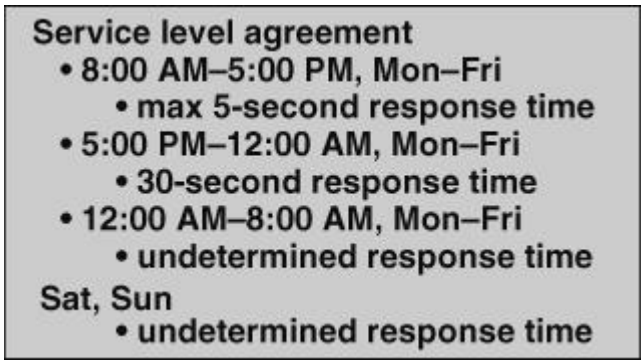
Figure 2.17 shows a service level agreement.

It is noteworthy that there are different service level agreements for the different sectors of the DW 2.0 environment. For example, there is an OLTP response-time SLA for the Interactive Sector. There is a class of transaction response-time SLA for the Integrated Sector. There is a transfer of data SLA for the Near Line Sector, and so forth. As such there is a measurable set

of operating parameters that outline the end-user expectation for performance and availability. The service level agreement defines the operating boundaries between the IT organization and the end user.

2.21 Protecting The Interactive Sector

Another way to protect the DW 2.0 environment performance is by protecting the Interactive Sector. Data in the Interactive Sector should not be removed to the Integrated Sector until it is sure that the probability of access has abated. Consider what happens when data is removed prematurely from the Interactive Sector. The request for the data is routed to the Integrated Sector, where online response time is *not* guaranteed. It may take a while to find the data in the Integrated Sector. Therefore, if there is any doubt about the probability of data needing to be accessed in the Interactive Sector, then the data should remain in the Interactive Sector of DW 2.0.



■ **FIGURE 2.17** A service level agreement is a good way to measure service in the transaction and the analytical environments.

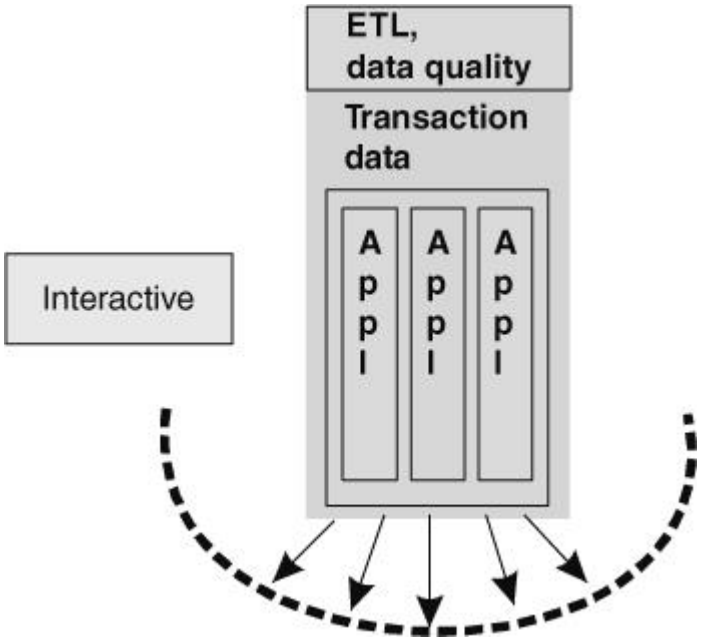


FIGURE 2.18 Make sure data does not leave the Interactive Sector until it is ready.

Figure 2.18 illustrates that data needs to remain in the Interactive Sector while the probability of accessing it is still high.

2.22 Partitioning Data

Partitioning data requires that data be divided up into physically distinct sectors when stored. Partitioning of data ensures that different sets of data can be handled separately, which greatly increases the flexibility of data.

To illustrate the benefits of physically partitioning data, consider a data base that is 10 terabytes in size. One day some data needs to be added or recalculated for some small part of the data base. The entire data base may be made unavailable while this activity occurs. Now consider what happens if the data base is partitioned into 1-terabyte chunks. Only a fraction of the data is paralyzed while repair work goes on. The impact is that major amounts of data remain free to fulfill processing demands that otherwise would have been unavailable.

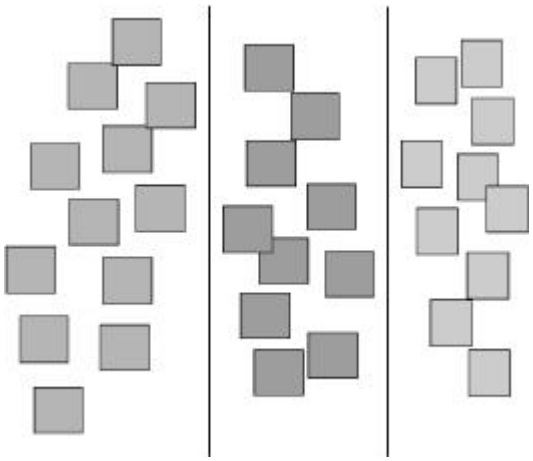


FIGURE 2.19 Partition data in storage.

Partitioning applies to anywhere in the DW 2.0 environment there are large stores of data. Figure 2.19 depicts partitioning.

2.23 Choosing The Proper Hardware

Choosing the proper hardware and software for the DW 2.0 environment also has a big impact on performance (Figure 2.20).

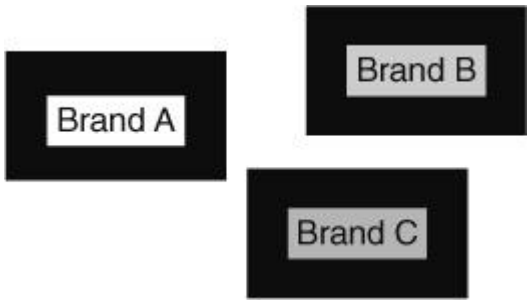


FIGURE 2.20 Choose the technology with performance capabilities that are suited to the performance needs.

2.24 Separating Farmers And Explorers

Another way to optimize performance is to separate the processing workload based on the different ways that data can be used. To this end, users can be roughly divided into two classes—farmers and explorers.

Farmers are end users who do analytical activities with regularity and predictability. Farmers know what they want before they set out to look for it. Explorers are people who are hard to

predict. Explorers may go 6 months without doing any analytical activity and then suddenly need to do a lot of analytical processing in the same week.

Explorers do not know exactly what they want before they embark on a search.

Both farmers and explorers are legitimate users of the DW 2.0 environment. Both deserve resources, and both make valuable contributions to the decision-making process of the corporation. But, farmers and explorers need to be separated for a very good reason. The analytical activities performed by these two different types of data warehouse users are radically different, like oil and water. Performance of the entire data warehouse environment is enhanced when the analytical activities of these two different user communities are not mixed together. Figure 15.21 reinforces the message that farmers and explorers do not mix well in the same environment and need to be separated.

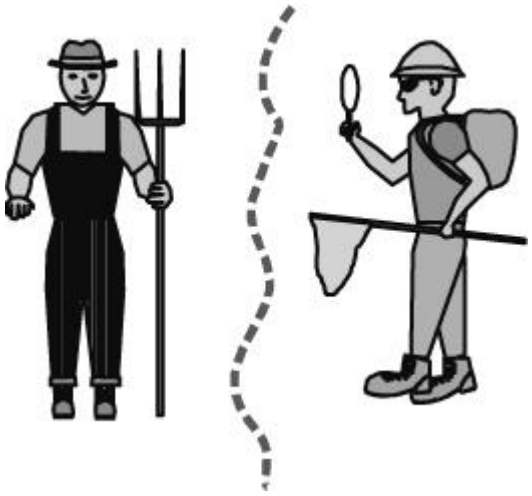
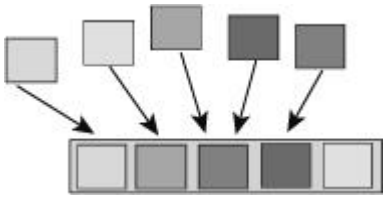


FIGURE 2.21 Separate farmers from explorers.

2.25 Physically Group Data Together

Another very fundamental activity that should be done to enhance data warehouse performance is to group data together physically when it is used in groups by a vast majority of the users of that data.

Figure 2.22 depicts five different kinds of data by color coding. If the analyst determines that 95% of the time the elements of data are accessed together, it makes sense for the data base designer to group the data together physically by the way it is most often accessed. By grouping the data together in this manner, the system can very efficiently retrieve the data.



■ **FIGURE 2.22** When data is used together, it should be physically grouped together, even though it is denormalized.

Grouping data according to access and usage patterns like this is recognized as data "denormalization." As a rule denormalization of data is not a good data architecture practice for an enterprise data warehouse and is discouraged. The data in the company's primary enterprise data warehouse should not be denormalized in this manner unless an extremely compelling, consistent, and enduring pattern of data subset access is identified.

2.26 Summary

Performance throughout the DW 2.0 environment is an essential characteristic.

There are two types of performance—transaction and analytical. When transaction performance suffers, the operational activities of the corporation suffer. When analytical performance suffers, the analytical capabilities of the corporation suffer.

Good performance is a result of many factors, including but not limited to the following:

- Choosing proper indexes
- Removing dormant data as soon as possible
- Educating the end user in how to distinguish between good and bad code
- Monitoring the transaction and the data warehouse environment so that when performance goes bad, there is a starting point to determine what has gone wrong
- Capacity planning, so that the organization can be in a proactive state when it comes to running out of resources
- Upgrades, making sure the latest releases of hardware and software are being used
- Metadata, so reusability minimizes the work that needs to be done
- Batch parallelization, to decrease elapsed time
- Transaction parallelization, to handle a large workload efficiently
- Workload management, to ensure that jobs do not conflict with each other because of size
- Data marts, to offload whole bodies of analytical processing from the central data warehouse
- Exploration facilities, which move statistical processing to another location
- Separation of transactions into classes based on the resources that a transaction will consume

- Service level agreements, which establish quantified goals for the measurement of performance
- Protection of the Interactive Sector to minimize resource contention
- Partitioning of data into different classes for separate management
- Choosing the right hardware and software for the achievement of performance
- Separating the work done by farmers and explorers
- Denormalizing data and placing data that is commonly accessed together in the same physical location
- Examining the code that is automatically created by tools, such as business intelligence tools

This is just a sample of the techniques and approaches that can be used to enhance performance in the DW 2.0 environment.

2.27 References and Bibliography

- ✓ DW2.0 The architecture for Next Generation of Datawarehousing W.H. Inmon, Derek Strauss, Genia Neushloss, ELSEVIER.

2.28 Exercise

1. Explain Capacity Planning.
2. Explain Online Response Time.

Unit V: Data warehousing

Chapter 3 : Migration

3.0 Introduction: -

The DW 2.0 environment is a large complex environment. It requires many resources over a long period of time to build the DW 2.0 environment. Figure 16.1 indicates that the DW 2.0 environment is more like a city than a house.

3.1 Migration in A Perfect World

In a perfect world, the order of the building of a DW 2.0 data warehouse mimics the flow of data into and through the DW 2.0 environment.

Figure 3.1 depicts a "perfect world" implementation of the DW 2.0 architecture.

This figure depicts the order in which the DW 2.0 data warehouse would be built if there were no other existing data warehouse infrastructure. As each level of processing is built, the foundation is set for the next level.

But the sequence shown in Figure 3.1 is a theoretical sequence. A DW 2.0 data warehouse is almost never built in top-down sequential steps as depicted. The primary reason a DW 2.0 data warehouse is not built in this "natural order" is that almost everyone who builds a DW 2.0 data warehouse already has an existing data warehouse environment in place.

Figure 3.2 depicts the infrastructure that almost everyone starts with, including a legacy applications environment, ETL processing, and a data base or data warehouse. These are the bare-bone components in the infrastructure that are found in most corporations.

3.2 Adding Components Incrementally

One of the really good pieces of news about the DW 2.0 architecture is that most of its components can be added incrementally and independently, on an as-needed basis. This ability to add components independently and incrementally means that companies can migrate and evolve to the DW 2.0 environment in an orderly manner. Nowhere in the migration to the DW 2.0 architecture is there a call to uproot and discard existing systems. Instead, the path to DW 2.0 is one by which the DW 2.0 infrastructure components can be built on top of an already-existing data warehouse.

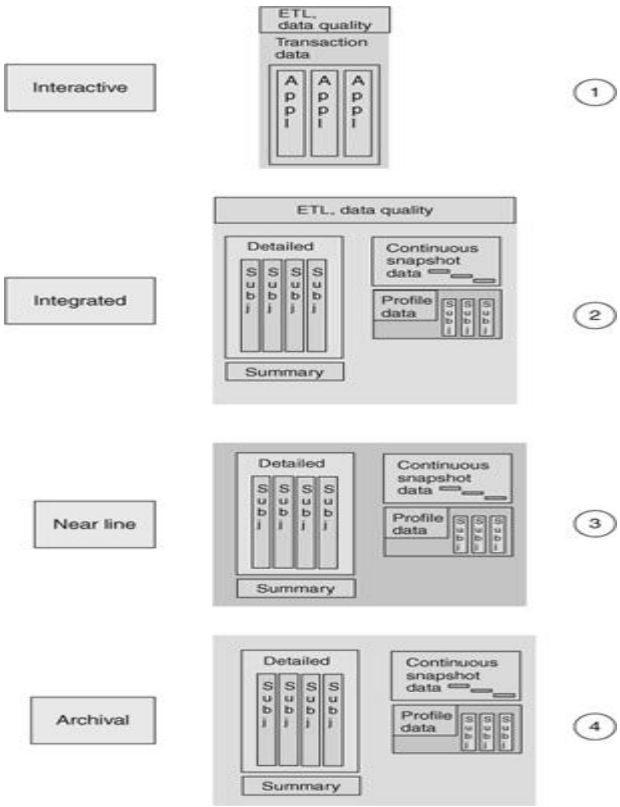


FIGURE 3.1 The "natural" order in which DW 2.0 is built.

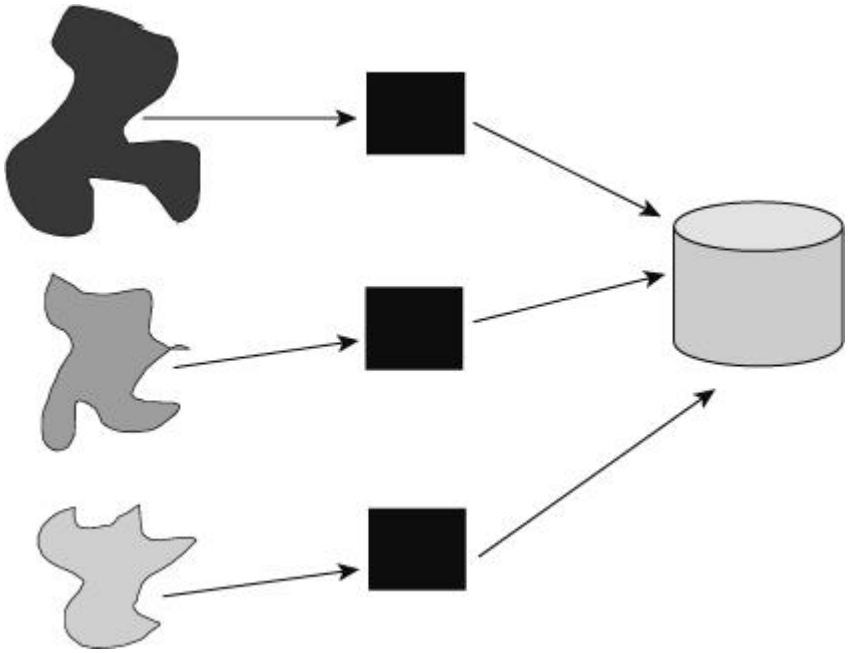


FIGURE 3.2 Where most organizations begin.

Adding a near-line storage component to an existing data warehouse infrastructure is a good example of incremental migration to the DW 2.0 architecture. Although near-line storage is optional and is not for every corporation, when it is needed, there is nothing else like it. Adding near-line storage to a first-generation data warehouse environment is an architecturally easy thing to do. No special work or preparation is needed to attach new near-line storage facilities to a first-generation data warehouse.

Figure 3.3 depicts the addition of near-line storage to an existing data warehouse environment.

3.3 Adding Components Incrementally

Next consider the Archival Sector. The Archival Sector can also be built with no advance preparation. One day the archival facility is not there, the next day it is, and nothing special had to be done to the first-generation data warehouse in the process.

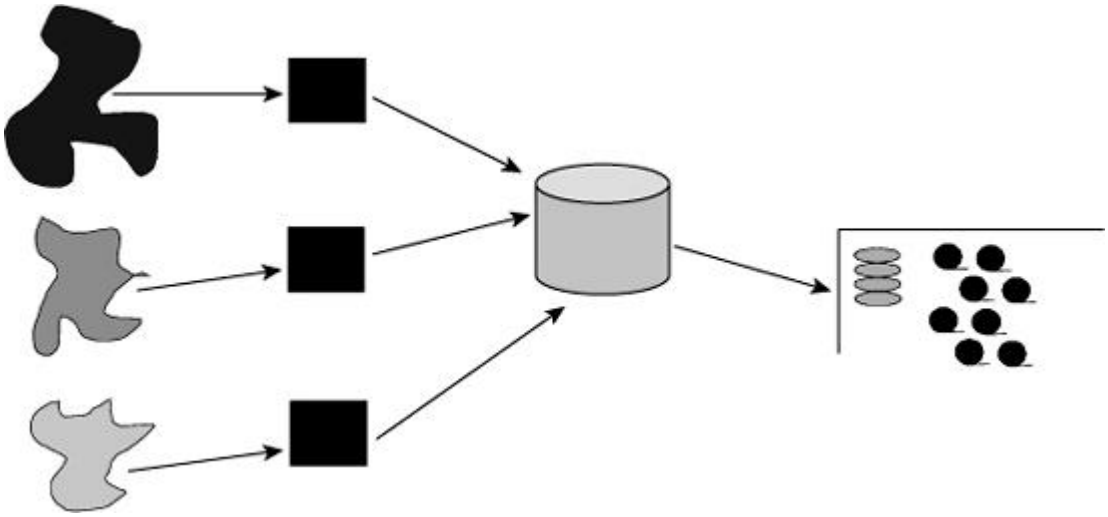


FIGURE 3.3 Adding near-line storage is incremental.

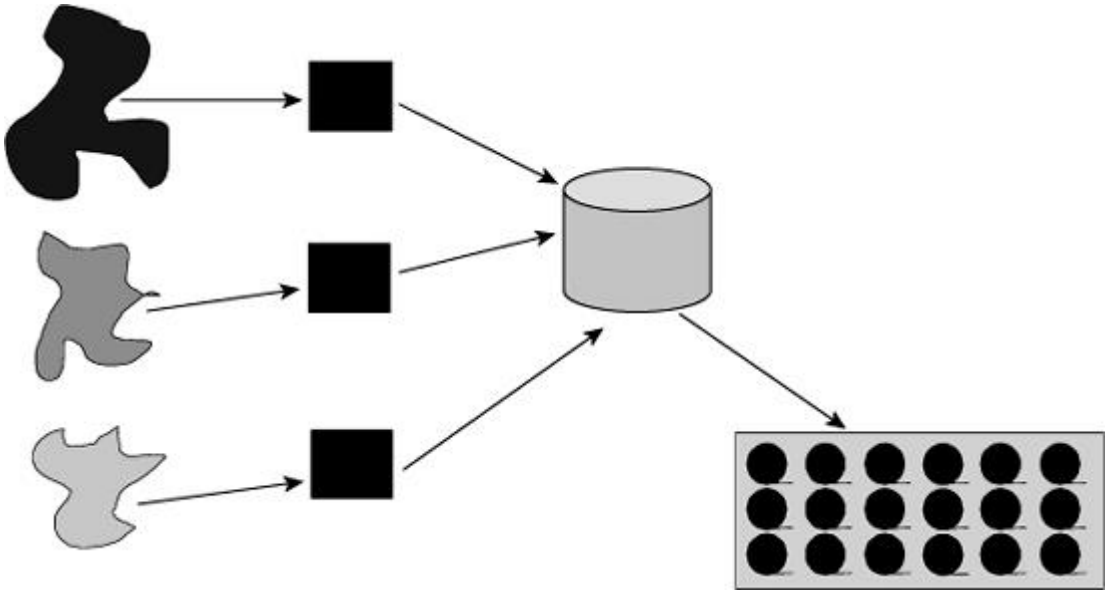


FIGURE 3.4 Adding archives is relatively easy to do.

Figure 16.5 depicts the addition of an Archival Sector to an existing first-generation data warehouse environment.

3.4 Creating Enterprise Metadata

The same considerations are true for the DW 2.0 metadata facility. As a rule, local metadata is already in place. Whether it is being used or not, the vendors that supply technology often provide a facility for the local storage and management of metadata, such as ETL metadata, business intelligence metadata, and DBMS metadata. So the local foundation of metadata is usually already in place. What needs to be added is enterprise metadata. Building enterprise metadata usually consists of three steps:

- Building the enterprise metadata repository
- Moving local metadata into the enterprise metadata repository
- Reconciling the local metadata with an enterprise metadata format

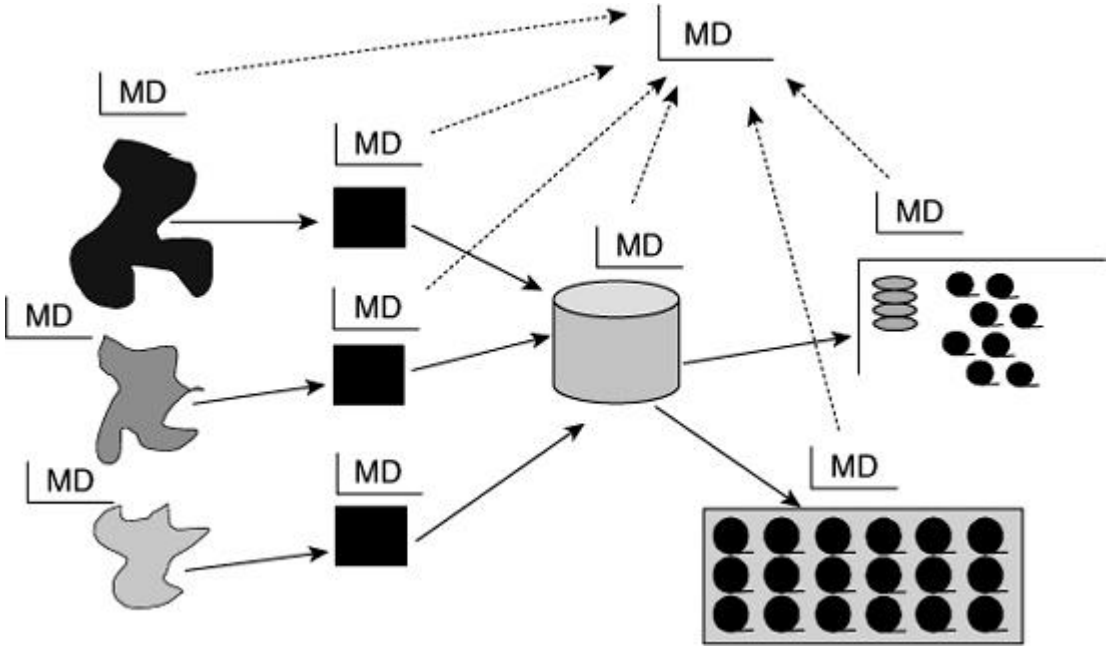


FIGURE 3.5 Metadata has to be gathered from many sources to form the enterprise metadata repository.

The last of these steps is always the hardest. Revising local metadata to conform to a corporate, enterprise format and structure is a difficult task to do.

3.5 Building The Metadata Infrastructure

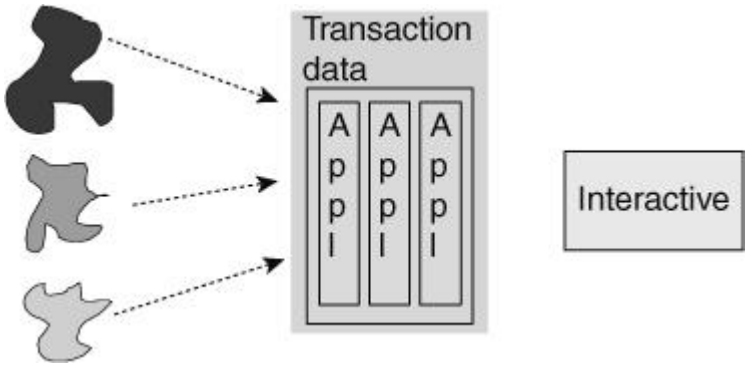
At no point does building an enterprise-level metadata repository require tearing down or discarding the existing environment. Instead the DW 2.0 metadata infrastructure is built over any already existing data warehouse infrastructure.

Figure 3.5 depicts the building of the enterprise metadata infrastructure over an existing first-generation data warehouse.

3.6 "Swallowing" Source Systems

If there is any place where there is likely to be some amount of decommissioning of systems in the existing operational application environment, it is legacy applications that need to be consumed by the Interactive Sector of the DW 2.0 environment. In many cases the Interactive

Sector will "swallow up" the old source applications. In other cases, the source applications will be left "as is" and simply continue to contribute data to the Interactive Sector.



■ **FIGURE 3.6** The applications morph into the Interactive Sector.

In the cases in which a source application is swallowed up by the Interactive Sector, it is a good bet that the application was an old out-of-date legacy system. The legacy application that is swallowed up was designed to satisfy business requirements from a long time ago—requirements that have long since changed. If the Interactive Sector had not come along, then the legacy application would have needed to be refurbished in any case.

Figure 3.6 depicts the absorption of some legacy applications into the Interactive Sector.

3.7 ETL As A Shock Absorber

ETL processing acts like a shock absorber for the entire data warehouse evolution and migration process. A drastic change can occur in the operational source application world, and through ETL transformation the effect on the Interactive Sector is minimized. Likewise, a major change may occur in the Interactive Sector, and through ETL transformation the effect on the Integrated Sector is nonexistent or at worst minimal.

Figure 3.7 shows ETL acting as a shock absorber between the different sectors.

3.8 Migration to The Unstructured Environment

The unstructured data domain is one of the newest and most important features of the DW 2.0 data warehouse environment. In many DW 2.0 environments, unstructured data is the added component that unlocks the door to many new kinds of analytical and decision-support processing.

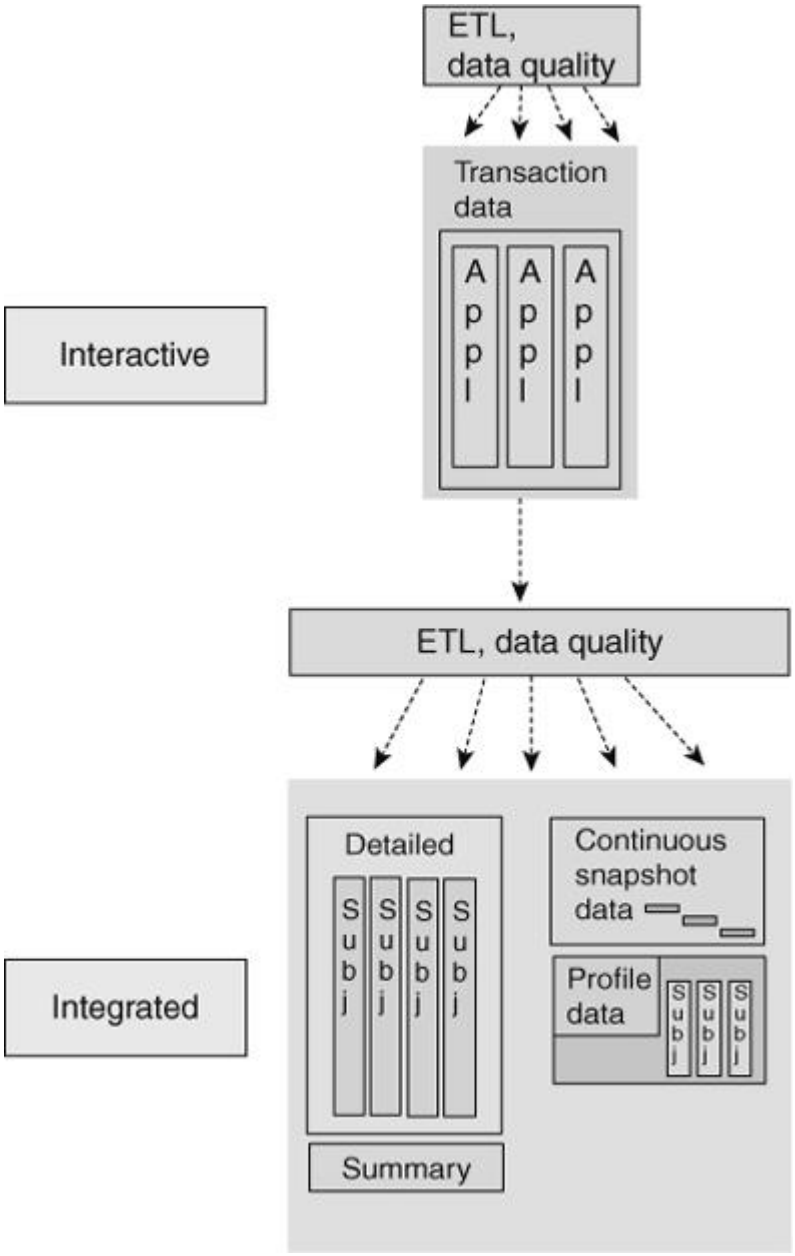


FIGURE 3.7 ETL processing acts like a shock absorber.

The migration to the DW 2.0 environment for unstructured data is quite different from the migration for structured data. Whereas the structured environment almost always exists in the form of a first-generation data warehouse, the same cannot be said for the unstructured component. There almost never is preexisting unstructured data that can be added to a DW 2.0 data warehouse environment.

Figure 3.8 shows that unstructured data is almost always captured in its entirety from its textual sources and is passed through a new unstructured data ETL routine into the unstructured side of the DW 2.0 data warehouse.

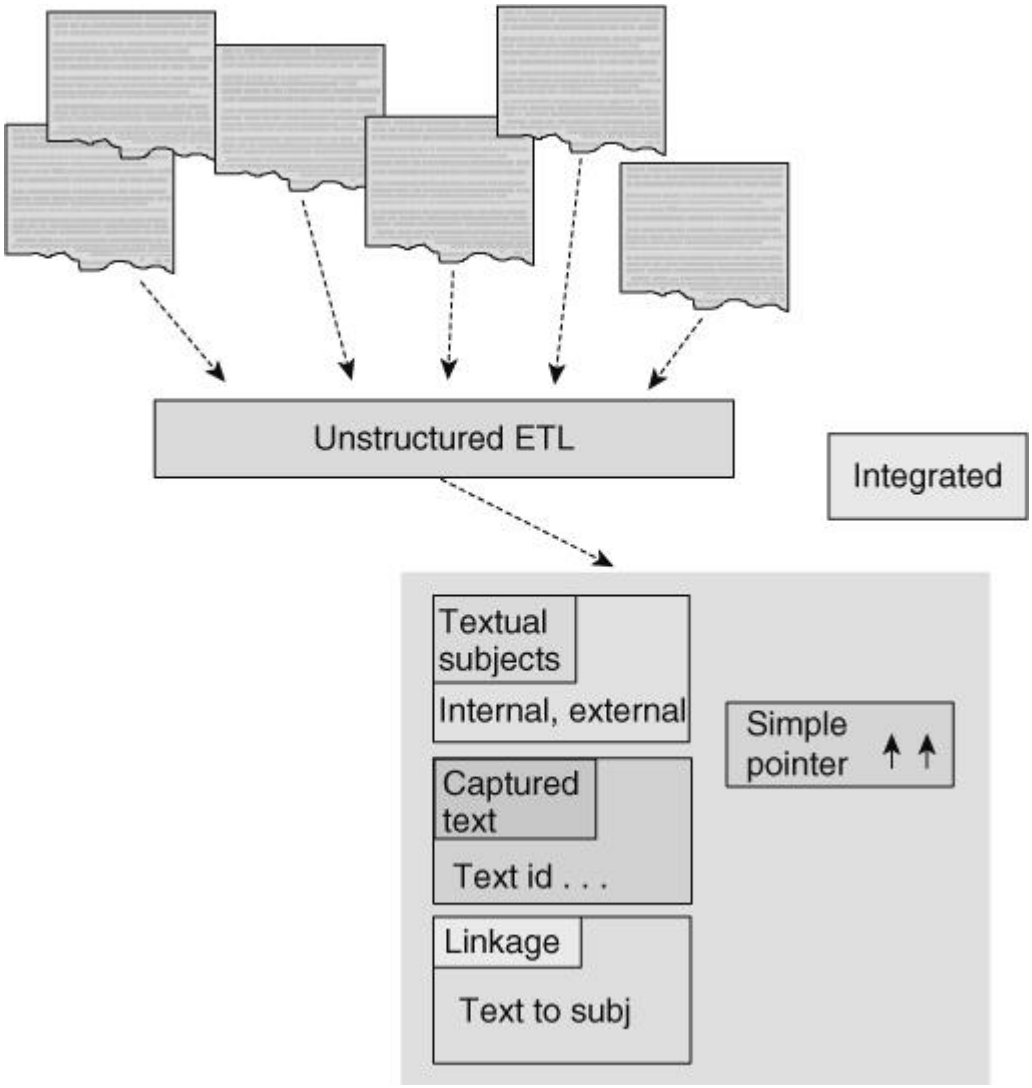


FIGURE 3.8 Unstructured data is entered from text and other forms of unstructured data.

After unstructured data has been processed into the DW 2.0 data warehouse, linkages are formed between structured data and unstructured data. Figure 16.10 depicts the formation of linkages between the unstructured and the structured data domains within a DW 2.0 sector.

Over time, as unstructured data ceases to be used, the unstructured data migrates to the DW 2.0 Archival Sector's unstructured data domain. There is more information in Chapter 19 about unstructured data.

Migration is something the business user is indirectly involved in. The business user determines what new subject areas are to be included in DW 2.0. He/she determines when data should be placed in archives and near-line storage. He/she also determines the transformations that occur as data passes from one sector to another.

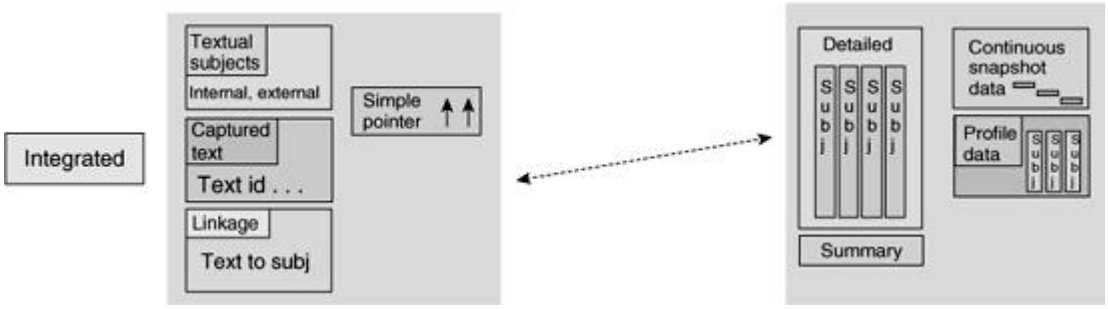


FIGURE 3.9 The unstructured environment is linked to the structured environment.

But at the end of the day, the business user is only tangentially involved in the migration of data that occurs as the DW 2.0 environment is being built.

3.9 Summary

There is a natural migration order to a DW 2.0 data warehouse architecture. The natural migration order follows the same path on which data flows—first to the Interactive Sector, next to the Integrated Sector, then to the Near Line Sector, and finally to the Archival Sector. Although the natural order is well defined, it is only theoretical.

In reality, the DW 2.0 environment evolves from the first-generation data warehouse one component at a time. The archival environment can be added independently. The near-line environment can be added independently. So can the enterprise metadata infrastructure and the unstructured data domain.

The different components of the DW 2.0 environment are added in response to business needs. The only preexisting systems components that are sometimes destroyed and replaced during migration to the DW 2.0 architecture are legacy application systems. On occasion, a legacy systems environment is so out of date and so fragile that it is easier to rewrite the system than it is to integrate data from the old system.

3.10 References and Bibliography

- ✓ DW2.0 The architecture for Next Generation of Datawarehousing W.H. Inmon, Derek Strauss, Genia Neushloss, ELSEVIER.

3.11 Exercise

1. Write a short note on ETL
2. Explain Data warehousing in detail.

Unit VI : Data warehousing

Chapter 1 : Implementation And Maintenance

1.0 Introduction: -

As an IT professional, you are familiar with logical and physical models. You have probably worked with the transformation of a logical model into a physical model. You also know that completing the physical model has to be tied to the details of the platform, the database software, hardware, and any third-party tools.

As you know, in an OLTP system you have to perform a number of tasks for completing the physical model. The logical model forms the primary basis for the physical model. But, in addition, a number of factors must be considered before you can get to the physical model. You must determine where to place the database objects in physical storage.

What is the storage medium and what are its features? This information helps you define the storage parameters. Then you have to plan for indexing, an important consideration. On which columns in each table must the indexes be built? You need to look into other methods for improving performance. You have to examine the initialization parameters in the DBMS and decide how to set them. Similarly, in the data warehouse environment, you need to consider many different factors to complete the physical model.

We have considered the logical model for the data warehouse in sufficient detail. You have mastered the dimensional modeling technique that helps you design the logical model.

In this chapter, we will use the logical model of a data warehouse to develop and complete the physical model. Physical design gets the work of the project team closer to implementation and deployment. Every task so far has brought the project to the grand logical model. Now, physical design moves it to the next significant phase.

1.1 The Physical Design Process

Figure 1-1 is a pictorial representation of the steps in the physical design process for a data warehouse. Note the steps indicated in the figure. In the following subsections, we will broadly describe the activities within these steps. You will understand how at the end of the process you arrive at the completed physical model. After the end of this section, the rest of the chapter elaborates on all the crucial aspects of the physical design.

Develop Standards

Many companies invest a lot of time and money to prescribe standards for information systems. The standards range from how to name the fields in the database to how to conduct interviews with the user departments for requirements definition. A group in IT is designated to keep the standards up-to-date. In some companies, every revision must be updated and authorized by the CIO. Through the standards group, the CIO ensures that the standards are followed correctly and strictly. Now the practice is to publish the standards on the company's intranet. If your IT department is one of the progressive ones giving due attention to standards, then be happy to embrace and adapt the standards for the data warehouse.

In the data warehouse environment, the scope of the standards expands to include additional areas. Standards ensure consistency across the various areas. If you have the same way of indicating names of the database objects, then you are leaving less room for ambiguity. Let us say the standards in your company require the name of an object to be a concatenation of multiple words separated by dashes and that the first word in the group indicates the business subject. With these standards, as soon as someone reads an object name, that person can know the business subject. Standards take on greater importance in the data warehouse environment. This is because the usage of the object names is not confined to the IT department. The users will also be referring to the objects by names when they formulate and run their own queries.

As standards are quite significant, we will come back to them a little later in this chapter. Now let us move on to the next step in the physical design.

Create Aggregates Plan

Let us say that in your environment more than 80% of the queries ask for summary information. If your data warehouse stores data only at the lowest level of granularity, every such query has to read through all the detailed records and sum them up. Consider a query looking for total sales for the year, by product, for all the stores. If you have detailed records keeping sales by individual calendar dates, by product, and by store, then this query needs to read a large number of detailed records. So what is the best method to improve performance

in cases like this? If you have higher levels of summary tables of products by store, the query could run faster. But how many such summary tables must you create? What is the limit?

In this step, review the possibilities for building aggregate tables. You get clues from the requirements definition. Look at each dimension table and examine the hierarchical levels. Which of these levels are more important for aggregation? Clearly assess the tradeoff. What you need is a comprehensive plan for aggregation. The plan must spell out the exact types of aggregates you must build for each level of summarization. It is possible that many of the aggregates will be present in the OLAP system. If OLAP instances are not for universal use by all users, then the necessary aggregates must be present in the main warehouse. The aggregate database tables must be laid out and included in the physical model. We will have some more to say about summary levels in a later section.

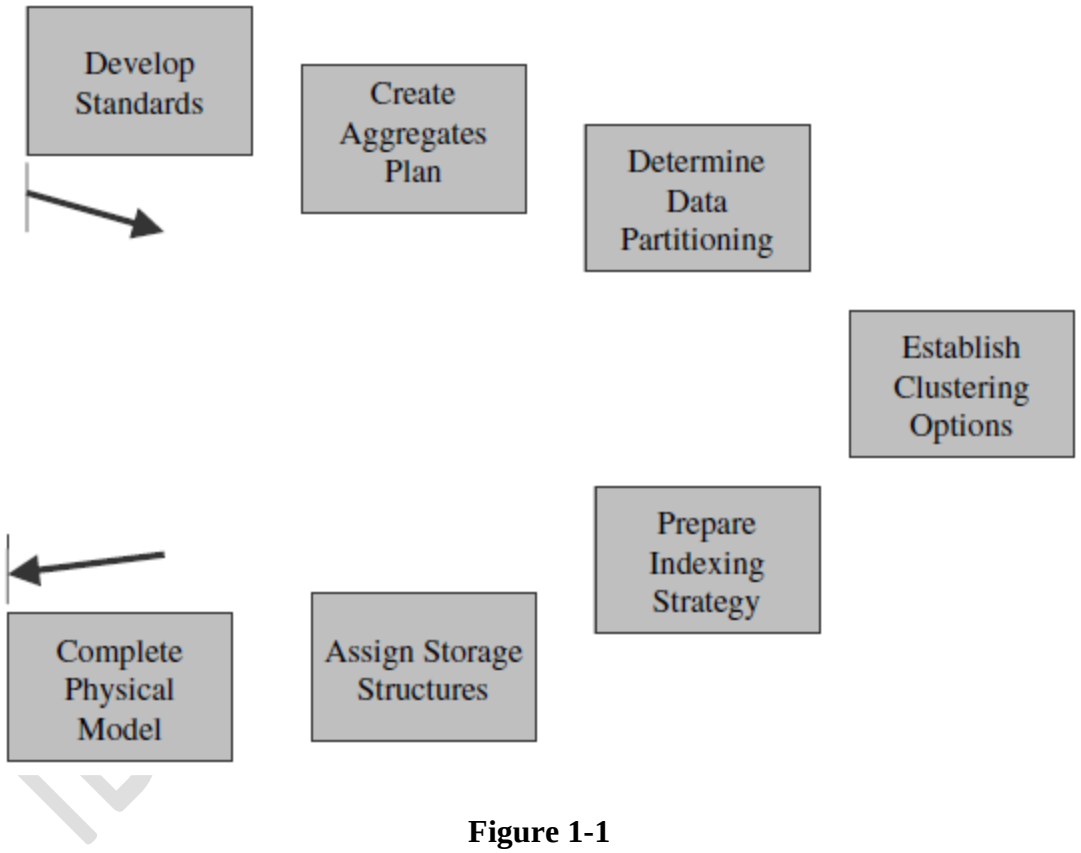


Figure 1-1

Determine the Data Partitioning Scheme

Consider the data volumes in the warehouse. What about the number of rows in a fact table? Let us make some rough calculations. Assume there are four dimension tables with 50 rows each on average. Even with this limited number of dimension table rows, the potential number of fact table rows exceeds six million. Fact tables are generally very large.

Large tables are not easy to manage. During the load process, the entire table must be closed to the users. Again, back up and recovery of large tables pose difficulties because of their sheer sizes. Partitioning divides large database tables into manageable parts. Always consider partitioning options for fact tables. It is not just the decision to partition that counts. Based on your environment, the real decision is about how exactly to partition the fact tables. Your data warehouse may be a conglomerate of conformed data marts. You must consider partitioning options for each fact table. Should some be partitioned vertically and the others horizontally? You may find that some of your dimension tables are also candidates for partitioning. Product dimension tables are especially large. Examine each of your dimension tables and determine which of these must be partitioned. In this step, come up with a definite partitioning scheme. The scheme must include:

- ✓ The fact tables and the dimension tables selected for partitioning
- ✓ The type of partitioning for each table—horizontal or vertical
- ✓ The number of partitions for each table
- ✓ The criteria for dividing each table (for example, by product groups)
- ✓ Description of how to make queries aware of partitions

Establish Clustering Options

In the data warehouse, many of the data access patterns rely on sequential access of large quantities of data. Whenever you have this type of access and processing, you will realize much performance improvement from clustering. This technique involves placing and managing related units of data in the same physical block of storage. This arrangement causes the related units of data to be retrieved together in a single input operation.

You need to establish the proper clustering options before completing the physical model. Examine the tables, table by table, and find pairs that are related. This means that rows from the related tables are usually accessed together for processing in many cases. Then make plans to store the related tables close together in the same file on the medium. For two related tables, you may want to store the records from both files interleaved. A record from one table is followed by all the related records in the other table while storing in the same file.

Prepare an Indexing Strategy

This is a crucial step in the physical design. Unlike OLTP systems, the data warehouse is query-centric. As you know, indexing is perhaps the most effective mechanism for improving performance. A solid indexing strategy results in enormous benefits. The strategy must lay down the index plan for each table, indicating the columns selected for indexing.

The sequence of the attributes in each index also plays a critical role in performance. Scrutinize the attributes in each table to determine which attributes qualify for bit-mapped indexes. Prepare a comprehensive indexing plan. The plan must indicate the indexes for each table. Further, for each table, present the sequence in which the indexes will be created. Describe the indexes that are expected to be built in the very first instance of the database. Many indexes can wait until you have monitored the data warehouse for some time. Spend enough time on the indexing plan.

Assign Storage Structures

Where do you want to place the data on the physical storage medium? What are the physical files? What is the plan for assigning each table to specific files? How do you want to divide each physical file into blocks of data? Answers to questions like these go into the data storage plan.

In an OLTP system, all data resides in the operational database. When you assign the storage structures in an OLTP system, your effort is confined to the operational tables accessed by the user applications. In a data warehouse, you are not just concerned with the physical files for the data warehouse tables. Your storage assignment plan must include other types of storage such as the temporary data extract files, the staging area, and any storage needed for front-end applications. Let the plan include all the types of storage structures in the various storage areas.

Complete Physical Model

This final step reviews and confirms the completion of the prior activities and tasks. By the time you reach this step, you have the standards for naming the database objects. You have determined which aggregate tables are necessary and how you are going to partition the large tables. You have completed the indexing strategy and have planned for other performance options. You also know where to put the physical files.

All the information from the prior steps enables you to complete the physical model. The result is the creation of the physical schema. You can code the data definition language statements (DDL) in the chosen RDBMS and create the physical structure in the data dictionary.

1.2 Data warehouse deployment

Let us continue from the end of the construction phase. As you observe, a large number of critical activities have been completed. All the components have been tested. The pieces are in place. Now please look at Figure 19-1 showing the activities in the deployment phase. Observe the primary tasks in each box representing the activities in this phase. In the deployment phase, establish a feedback mechanism for the users to let the project team know how the deployment is going. If this is the initial rollout, most of your users will be new to the processes. Although the users must have received training, substantial handholding is essential in this phase. Be prepared to provide the support. Let us inspect each major activity in the deployment phase. As we move along, please pick up tips and adapt them to your environment.

Complete User Acceptance

Proper acceptance of the system by the users is not just a formality in the deployment phase, it is an absolute necessity. Do not proceed to force the deployment before the key user representatives express their satisfaction about the data warehouse. Some organizations have procedures for a formal sign-off. Others conduct a series of user acceptance tests in which each function is accepted. It does not matter how the user acceptance activity is completed but do complete it in the manner in which such acceptances are usually done in your environment.

Who should do the acceptance testing from the side of the users? Remember the users who are already on the project team? Start with these people. If you have a user liaison manager in your project team, then this person must be made responsible. Get the end- user applications specialists to perform the acceptance testing for their own areas. In addition to the user representatives on the project team, include a few other users for a few final test sessions.

How should the user acceptance testing be conducted? Which particular users should be testing at this final phase of deployment? Here are a few tips:

- ✓ In each subject area or department, let the users select a small number of typical queries and reports, ones for which they can verify the results without too much work or difficulty, but substantial ones involving combinations of dimension table constraints. Let the users run the queries and produce the reports. Then produce reports from the operational systems for verification. Compare the reports from the operational systems to the results from the data warehouse. Resolve and account for all seeming discrepancies. Verify and make sure that the reports from the operational systems are free of errors before matching them up with the results from the warehouse.
- ✓ This is a good time to test some of the predefined queries and reports. Have each user group pick a small number of such queries and reports and test the executions.

- ✓ Have the users test the OLAP system. Create the multidimensional cubes needed for the test and store the cubes in the OLAP multidimensional database if you are adopting the MOLAP approach. Let the users select about five typical analysis sessions to try out. Again, verify the results with reports from the operational systems.
- ✓ As you know, in almost every warehouse, users need to learn about and be comfortable with the functioning of the new front-end tools. The majority of the users must be able to use the tools with comparative ease. Design acceptance tests for the user representatives to sign-off on the usability of the tools. Of course, most of this type of testing would have been done at the time of tool selection. But at that time, the testing would have been done at vendor sites or in the system development environment. Now the verification is being done in the production environment. This is a big difference.
- ✓ If your data warehouse is Web-enabled, have the users test the Web features. If Web technology is used for information delivery, let the users test this aspect.
- ✓ No user acceptance testing is complete without acceptance of the system performance. The project must have set the expectation of the users at an agreed level of performance. Query response time expectations are usually at the level of about 3 to 5 seconds. In fact, individual queries may deviate from the average, and that is understandable. The users will be able to accept such variations provided that these are exceptional and not the norm.
- ✓ Remember, the acceptance test is useful if conducted in the production environment. You may conduct all the previous individual module testing and the overall system testing in the development environment. When all the acceptance testing is completed successfully, get the sign-off formally or by any other acceptable method. This is a signal that the project is ready for full deployment.

Perform Initial Loads

In Chapter 12, we discussed the loading of the data warehouse in sufficient depth. We reviewed how initial loads are done and went over the methods for incremental loads. We also covered the four different modes for applying data to the warehouse repository. By the time the project arrives at the deployment phase, the team must have tested sample initial loads and mock incremental loads. Now is the time to do the complete initial load.

Also, now the time is also close for doing the first incremental load, which normally takes place in 24 fours after the deployment. Recalling what we studied in Chapter 12 as background information, let us review the steps of the complete initial load. If you need to go back to Chapter 12 for a brief refresher, please do so now. Especially review how load images

are created for dimension and fact tables. The initial load process picks up these load images that are already in the format of the table records themselves.

Here are the major steps of the complete initial load:

- ✓ Drop the indexes on the data warehouse relational tables. As you know, index building during the loading consumes an enormous chunk of time. Remember that the initial load deals with very large volumes of data, hundreds of thousands and even millions of rows. You cannot afford to have anything slowing down the load process.
- ✓ As you know, each dimension table record is in a one-to-many relationship with the corresponding fact table records. That means referential integrity is enforceable by the DBMS on this relationship. But we assume that the load images have been created carefully and that we can suspend this restriction to speed up the load process. This is up to each team, based on the confidence level for the creation of the load images.
- ✓ In some cases, the initial loads may run for a number of days. If your initial load aborts after a few days of processing because of some system failure, then you have a disaster on your hands. What is the solution? Should you go back to the beginning and start all over again? No. Make sure you have proper checkpointing so that you can pick up from the latest checkpoint and continue.
- ✓ Load the dimension tables first for the reasons given in Chapter 12. Remember how the keys are built for the dimension table records. Recall how the keys for the fact table records are formed from those of the dimension table records. That is why you need to load the dimension tables first and then the fact tables. Some data warehouse teams opt to load the smaller dimension tables first and verify the load process before starting the loads of the larger tables.
- ✓ Load the fact tables next. The keys for the fact table records would already have been resolved before creating load images in the staging area.
- ✓ Based on your already established plan for aggregate or summary tables, create the aggregate tables based on the records in the dimension and fact tables. Sometimes, load images are created for the aggregate tables beforehand in the staging area. If so, apply these load images at this time to create the aggregate tables.
- ✓ You had suspended index creation during the loads; now build the indexes.
- ✓ If you had opted not to suspend the enforcement of referential integrity, all the referential violations would have recorded on the system load during the load process. Examine the log files and resolve the load exceptions.

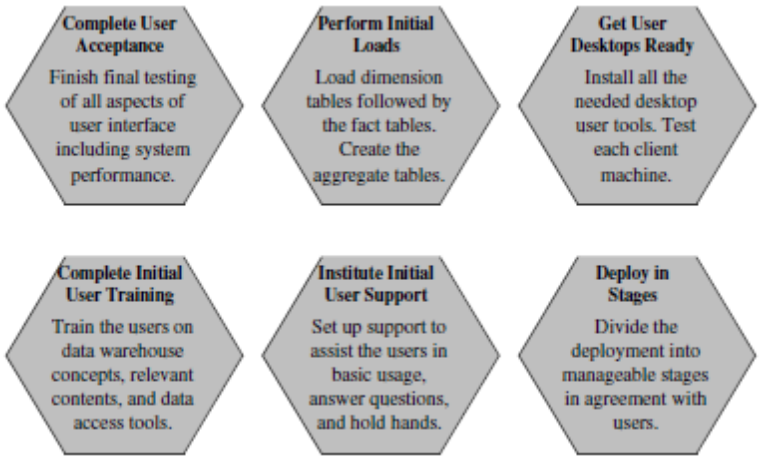


Figure 1-2

Get User Desktops Ready

Getting the user machines ready for the data warehouse is a comparatively small part in terms of the overall effort from beginning to end. Although the effort may be much less than 10% of the total activities, what the users see and experience at their desktops is what counts for them. The set of desktop tools is the data warehouse for the users. Therefore, pay special attention to the installation of the data access tools, the network connections that link up the user machines to the servers, and the configuration of the middle tiers.

Depending upon the method of deployment, consider allocating enough time for getting
the desktops ready. Before starting this activity, make a list of configuration needs for the client machines, all the information delivery software to be installed, the hardware setup for the desktop
machines, and the entire spectrum of requirements for network connections. Let us itemize a few practical suggestions:

- ✓ Remote deployment of the data access tools for the client machines is a faster method. The data warehouse administrators are able to install the software on the various machines from a central location, thus avoiding individual visits to the user workstations. On the other hand, if you plan to install and test the access tools on the client machines one by one, plan for longer lead time.
- ✓ Irrespective of whether the deployment method is by remote installation or by individual visits to user areas, this is a unique opportunity to upgrade the workstations with other relevant types of software that may be lacking at the user sites.

- ✓ Desktop tools cannot function without the appropriate server and middle-tier components. Plan for proper timing, installation, and testing of these other components.
- ✓ Test each client machine to ensure that all components are properly installed and work well together.
- ✓ Completion of the desktop readiness activity means that the users can get to their machines and start accessing the data warehouse information. This activity necessarily includes establishing and acquiring the user passwords and logon user-id's. Ensure this is done and tested.

Complete Initial User Training

The importance of training and orientation for the users cannot be overemphasized. IT professionals may think of the separate components of data, applications, and tools. From an IT department's point of view, training is thought of as training about these three components. But to the users, it is all one. They do not distinguish between applications and tools. The training program must be designed from the users' point of view. An essential difference exists between training of users in operational system implementations and data warehouse implementations. The capabilities offered in the data warehouse have a much wider potential. Users are not aware of how much they can really do with the tools in the data warehouse.

To get started, plan to train the users in the following areas:

- ✓ Basic database and data storage concepts
- ✓ Fundamental features of a data warehouse
- ✓ Contents of the data warehouse as applicable to each user group
- ✓ Browsing through the warehouse contents
- ✓ Use of data access and retrieval tools
- ✓ Applicability of Web technology for information delivery
- ✓ Set of predefined queries and reports
- ✓ Types of analysis that can be performed
- ✓ Query templates and how to use them
- ✓ Report scheduling and delivery
- ✓ Data load schedule and currency of the data
- ✓ The support structure, including first line of contact

Institute Initial User Support

In the early days of post deployment, every member of the data warehouse support staff is usually very busy. Users may have a wide range of questions from basic sign-on to performing complex drill-down analysis. Many questions may just relate to the hardware issues.

The users need an extensive degree of handholding, at least during the initial stages. Figure 1-3 depicts a set-up for the initial user support. Note the basic support centers. The user representative in each department is the first point of contact. This person must have been trained well enough to answer most of the questions on the applications and data content. The user representative is also knowledgeable about the end-user tools on the desktops. The hotline support comes after the user representative is unable to provide answers. At least in the initial deployment, this procedure seems to work well. Also, note the type of support provided by the technical support group.

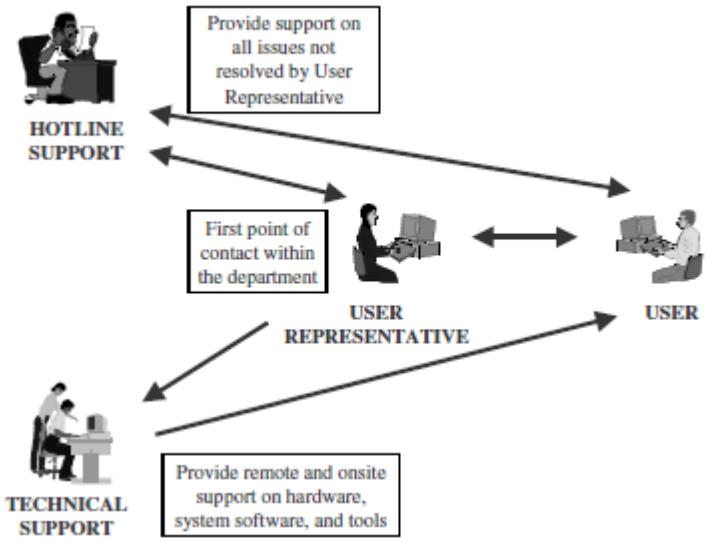


Figure 1-3

Deploy in Stages

Building and deploying a data warehouse is a major undertaking for any organization. This is a project that calls for many different types of skills. The data warehouse encompasses several different technologies. You are faced with new techniques of design. Dimensional modeling is a very different approach not previously used by designers in any operational systems. The process of data extraction, transformation, and loading is tedious and laborintensive. Users receive information in vastly new ways. The effort is big, possibly bigger than most of the previous information projects any enterprise has ever launched before.

Under these circumstances, what is a reasonable method for deploying your data warehouse? Most decidedly, if you parcel the deployment into manageable parts, you can bring the data warehouse in at a comfortable pace. Make the deployment happen in stages.
Clearly chalk out the stages. Plan a schedule that is most effective from the point of view of the users as well as the project team.

The topdown approach started out with a corporate normalized data warehouse feeding several departmental data marts. The bottom-up approach relates to building a group of data marts without the idea of melding the data marts together. Then the practical approach leads to a set of supermarts that form a conglomerate of conformed data marts.

Irrespective of the approach your project team is adopting for whatever reasons most plausible for your environment, divide and stage your deployment. For all the approaches, the overall requirements definition, the architecture, and the infrastructure take the corporate view. You plan for the entire enterprise but you deploy the pieces in well-defined stages. Please refer to Figure 1-4 showing the staging of the deployment. Notice the suggested stages under the different approaches. Note how you build the overall enterprisewide data warehouse first in the top-down approach. Then the dependent data marts are deployed in a suitable sequence. The bottom-up approach is less structured and less refined.

In the practical approach, you deploy one data mart at a time. You remember how conforming the dimensions is key to the success of the practical approach. Let us reconsider this important concept. The reason for this requirement is to be able to preserve the cohesion of all the data marts that comprise the enterprise data warehouse. At the fundamental level, conforming of dimensions simply means this: when we say “product” in two different data marts, it means the same thing. In other words, the product dimension table in every subsequently deployed data mart is the same as the product dimension table in the first data mart. The tables must be identical in terms of all the attributes and keys.

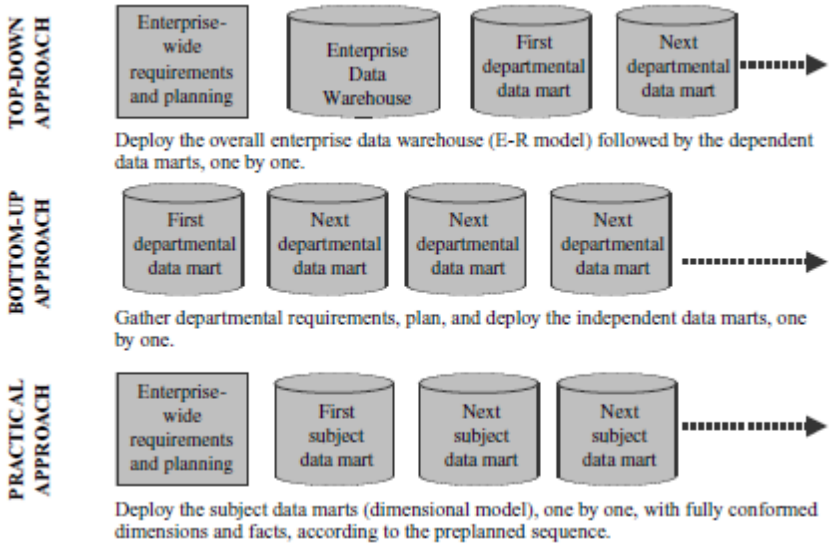


Figure 1-4

1.3 Growth and Maintenance

MONITORING THE DATA WAREHOUSE

When you implement an OLTP system, you do not stop with the deployment. The database administrator continues to inspect system performance. The project team continues to monitor how the new system matches up with the requirements and delivers the results.

Monitoring the data warehouse is comparable to what happens in an OLTP system, except for one big difference. Monitoring an OLTP system dwindles in comparison with the monitoring activity in a data warehouse environment. As you can easily perceive, the scope of the monitoring activity in the data warehouse extends over many features and functions. Unless data warehouse monitoring takes place in a formalized manner, desired results cannot be achieved. The results of the monitoring gives you the data needed to plan for growth and to improve performance.

Figure 1-5 presents the data warehousing monitoring activity and its usefulness. As you can observe, the statistics serve as the life-blood of the monitoring activity. That leads into growth planning and fine-tuning of the data warehouse.

Collection of Statistics

What we call monitoring statistics are indicators whose values provide information about data warehouse functions. These indicators provide information on the utilization of the hardware and software resources. From the indicators, you determine how the data warehouse performs. The indicators present the growth trends. You understand how well the servers function. You gain insights into the utility of the end-user tools.

How do you collect statistics on the working of the data warehouse? Two common methods apply to the collection process. Sampling methods and event-driven methods are generally used. The sampling method measures specific aspects of the system activity at regular intervals. You can set the duration of the interval. If you set the interval as 10 minutes for monitoring processor utilization, then utilization statistics are recorded every 10 minutes. The sampling method has minimal impact on the system overhead.

The event-driven methods work differently. The recording of the statistics does not happen at intervals, but only when a specified event takes place. For example, if you want to monitor the index table, you can set the monitoring mechanism to record the event

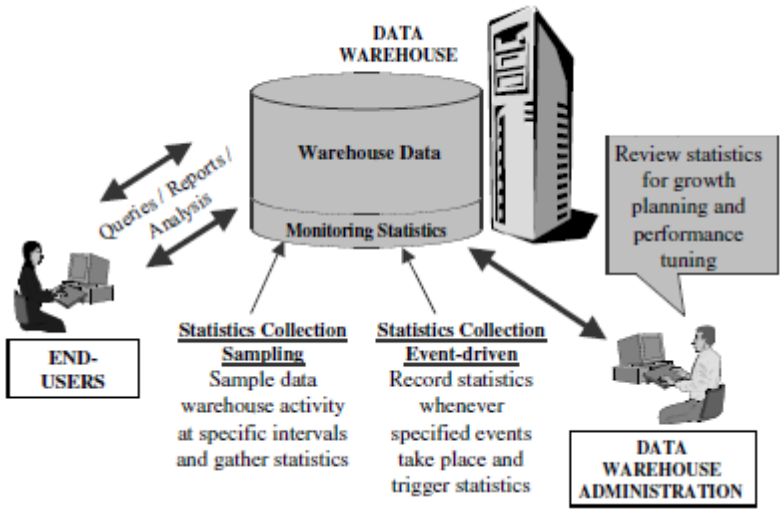


Figure 1-5

when an update takes place to the index table. Event-driven methods add to the system overhead but are more thorough than sampling methods.

Which tools collect statistics? The tools that come with the database server and the host operating system are generally turned on to collect the monitoring statistics. Over and above these, many third-party vendors supply tools especially useful in a data warehouse environment. Most tools gather the values for the indicators and also interpret the results. The data collector component collects the statistics; the analyzer component does the interpretation. Much of the monitoring of the system occurs in real time.

Let us now make a note of the types of monitoring statistics that are useful. The following is a random list that includes statistics for different uses. You will find most of these applicable to your environment. Here is the list:

- ✓ Physical disk storage space utilization
- ✓ Number of times the DBMS is looking for space in blocks or causes fragmentation
- ✓ Memory buffer activity
- ✓ Buffer cache usage
- ✓ Input–output performance
- ✓ Memory management
- ✓ Profile of the warehouse content, giving number of distinct entity occurrences (example: number of customers, products, etc.)
- ✓ Size of each database table
- ✓ Accesses to fact table records
- ✓ Usage statistics relating to subject areas
- ✓ Numbers of completed queries by time slots during the day
- ✓ Time each user stays online with the data warehouse
- ✓ Total number of distinct users per day
- ✓ Maximum number of users during time slots daily
- ✓ Duration of daily incremental loads
- ✓ Count of valid users
- ✓ Query response times
- ✓ Number of reports run each day
- ✓ Number of active tables in the database

Using Statistics for Growth Planning

As you deploy more versions of the data warehouse, the number of users increases and the complexity of the queries intensifies, you then have to plan for the obvious growth. But how do you know where the expansion is needed? Why have the queries slowed down?

Why have the response times degraded? Why was the warehouse down for expanding the table spaces? The monitoring statistics provide you with clues as to what is happening in the data warehouse and how you can prepare for the growth.

We indicate below the types of action that are prompted by the monitoring statistics:

- ✓ Allocate more disk space to existing database tables
- ✓ Plan for new disk space for additional tables
- ✓ Modify file block management parameters to minimize fragmentation

- ✓ Create more summary tables to handle large number of queries looking for summary information
- ✓ Reorganize the staging area files to handle more data volume
- ✓ Add more memory buffers and enhance buffer management
- ✓ Upgrade database servers
- ✓ Offload report generation to another middle tier
- ✓ Smooth out peak usage during the 24-hour cycle
- ✓ Partition tables to run loads in parallel and to manage backups

Using Statistics for Fine-Tuning

The next best use of statistics relates to performance. You will find that a large number of monitoring statistics prove to be useful for fine-tuning the data warehouse. In a later section, we will discuss this topic in more detail. For now, let us indicate below the data warehouse functions that are normally improved based on the information derived from the statistics:

- ✓ Query performance
- ✓ Query formulation
- ✓ Incremental loads
- ✓ Frequency of OLAP loads
- ✓ OLAP system
- ✓ Data warehouse content browsing
- ✓ Report formatting
- ✓ Report generation

Publishing Trends for Users

This is a new concept not usually found in OLTP systems. In a data warehouse, the users must find their way into the system and retrieve the information by themselves. They must know about the contents. Users must know about the currency of the data in the warehouse.

When was the last incremental load? What are the subject areas? What is the count of distinct entities? The OLTP systems are quite different. These systems readily present the users with routine and standardized information. Users of OLTP systems do not need the inside view.

In your data warehouse is Web-enabled, use the company's intranet to publish the statistics for the users. Otherwise, provide the ability to inquire into the dataset where the statistics are kept.

1.3 Summary

- ✓ Immediately following the initial deployment, the project team must conduct review sessions.
- ✓ Ongoing monitoring of the data warehouse requires collection of statistics on a variety of indicators. Use the statistics for growth planning and for fine-tuning.
- ✓ User training function consists of determining the content of the needed training, preparation of the training program, and delivering the training program.
- ✓ User support function needs to have multiple tiers to deliver the appropriate support relating to data content, applications, and tools.
- ✓ Ongoing management and administration includes the following: platform upgrades, managing data growth, storage management, ETL management, data model revisions, information delivery enhancements, and ongoing fine-tuning.

1.5 References and Bibliography

- ✓ Paulraj Ponnian, "Data Warehousing Fundamentals", John Wiley. (Unit VI)

1.5 Exercise

1. List any four of the physical design steps. For two of the selected four, describe the activities.
2. Name the physical design objectives. Which objective do you think ranks as the most important?