CHAPTER 1

The Chess Pieces

All of the authors of this book worked together at Metaphor Computer Systems over a period that spanned more than ten years, from 1982 to 1994. Although the real value of the Metaphor experience was the building of hundreds of data warehouses, there was an ancillary benefit that we sometimes find useful. We are really conscious of *metaphors*. How could we avoid metaphors, with a name like that?

A useful metaphor to get this book started is to think about studying the chess pieces very carefully before trying to play the game of chess. You really need to learn the shapes of the pieces and what they can do on the board. More subtly, you need to learn the strategic significance of the pieces and how to wield them in order to win the game. Certainly, with a data warehouse, as well as with chess, you need to think way ahead. Your opponent is the ever-changing nature of the environment you are forced to work in. You can't avoid the changing user needs, the changing business conditions, the changing nature of the data you are given to work with, and the changing technical environment. So maybe the game of data warehousing is something like the game of chess. At least it's a pretty good metaphor.

If you intend to read this book, you need to read this chapter. We are fairly precise in this book with our vocabulary, and you will get more out of this book if you know where we stand. We begin by briefly defining the basic elements of the data warehouse. As we remarked in the introduction, there is not universal agreement in the marketplace over these definitions.

But our use of these words is as close to mainstream practice as we can make them. Here in this book, we will use these words precisely and consistently, according to the definitions we provide in the next section.

We will then list the data warehouse processes you need to be concerned about. This list is a declaration of the boundaries for your job. Perhaps the biggest insight into your responsibilities as a data warehouse manager is that this list of data warehouse processes is long and somewhat daunting.

BASIC ELEMENTS OF THE DATA WAREHOUSE

As you read through the definitions in this section, please refer to Figure 1.1. We will move through Figure 1.1 roughly in left to right order.

Source System

An operational system of record whose function it is to capture the transactions of the business. A source system is often called a "legacy system" in a mainframe environment. The main priorities of the source system are uptime and availability. Queries against source systems are narrow, "account-based" queries that are part of the normal transaction flow and severely restricted in their demands on the legacy system. We assume that the source systems maintain little historical data and that management reporting from source systems is a burden on these systems. We make the strong assumption that source systems are not queried in the broad and unexpected ways that data warehouses are typically queried. We also assume that each source system is a natural stovepipe, where little or no investment has been made to conform basic dimensions such as product, customer, geography, or calendar with other legacy systems in the organization. Source systems have keys that make certain things unique, like product keys or customer keys. We call these source system keys *production keys,* and we treat them as attributes, just like any other textual description of something. We *never* use the production keys as the keys within our data warehouse. (Hopefully that got your attention. Read the chapters on data modeling.)

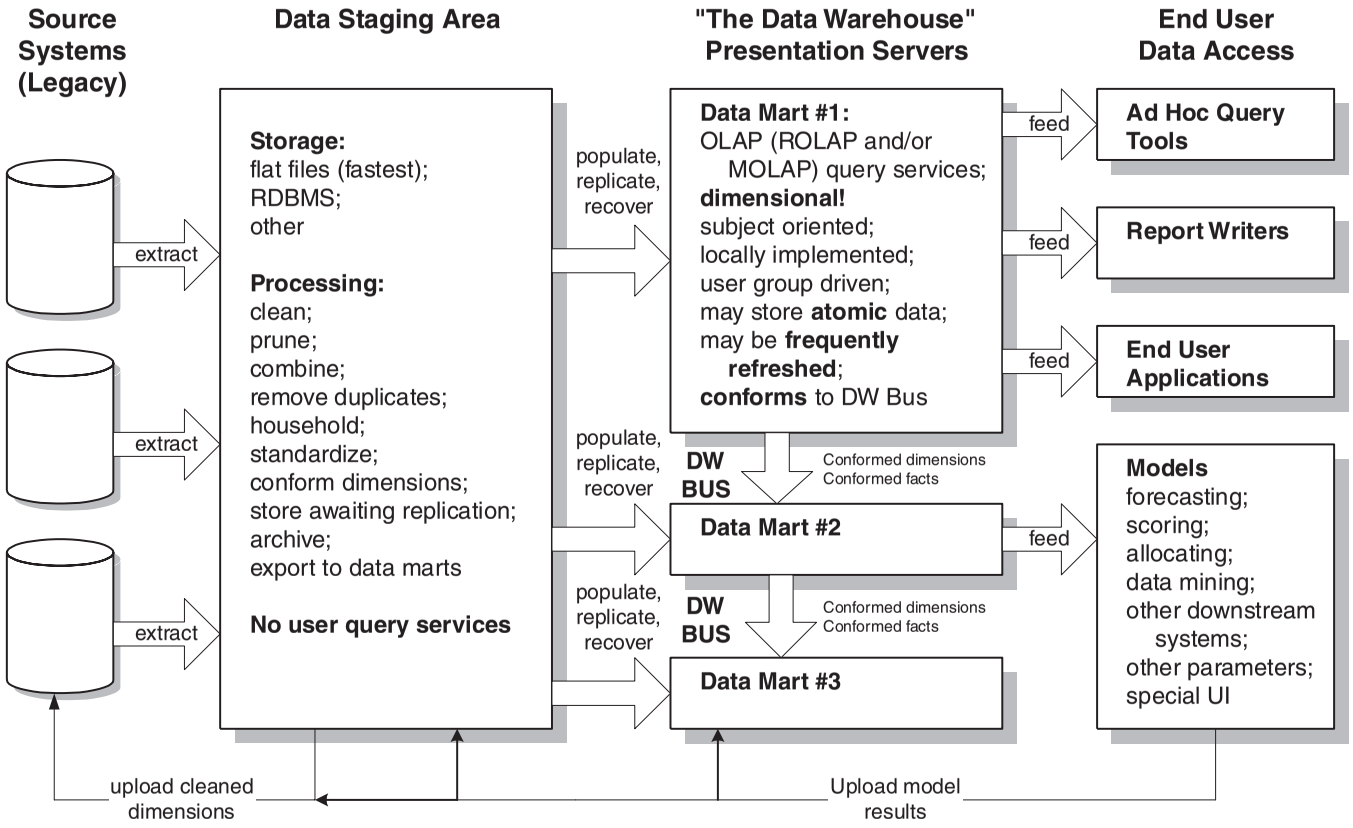

FIGURE 1.1 The basic elements of the data warehouse.

Data Staging Area

A storage area and set of processes that clean, transform, combine, deduplicate, household, archive, and prepare source data for use in the data warehouse. The data staging area is everything in between the source system and the presentation server. Although it would be nice if the data staging area were a single centralized facility on one piece of hardware, it is far more likely that the data staging area is spread over a number of machines. The data staging area is dominated by the simple activities of sorting and sequential processing and, in some cases, the data staging area does not need to be based on relational technology. After you check your data for conformance with all the one-to-one and many-to-one business rules you have defined, it may be pointless to take the final step of building a full blown entity-relation-based physical database design.

However, there are many cases where the data arrives at the doorstep of the data staging area in a third normal form relational database. In other cases, the managers of the data staging area are more comfortable organizing their cleaning, transforming, and combining steps around a set of normalized structures. In these cases, a normalized structure for the data staging storage is certainly acceptable. The key defining restriction on the data staging area is that *it does not provide query and presentation services*. As soon as a system provides query and presentation services, it must be categorized as a presentation server, which is described next.

Presentation Server

The target physical machine on which the data warehouse data is organized and stored for direct querying by end users, report writers, and other applications. In our opinion, three very different systems are required for a data warehouse to function: the source system, the data staging area, and the presentation server. The source system should be thought of as outside the data warehouse, since we assume we have no control over the content and format of the data in the legacy system. We have described the data staging area as the initial storage and cleaning system for data that is moving toward the presentation server, and we made the point that the data staging area may well consist of a system of flat files. It is the presentation server where we insist that the data be presented and stored in a dimensional framework. If the

presentation server is based on a relational database, then the tables will be organized as star schemas. If the presentation server is based on nonrelational *on-line analytic processing* (OLAP) technology, then the data will still have recognizable dimensions, and most of the recommendations in this book will pertain. At the time this book was written, most of the large data marts (greater than a few gigabytes) were implemented on relational databases. Thus, most of the specific discussions surrounding the presentation server are couched in terms of relational databases.

Dimensional Model

A specific discipline for modeling data that is an alternative to entity-relationship (E/R) modeling. A dimensional model contains the same information as an E/R model but packages the data in a symmetric format whose design goals are user understandability, query performance, and resilience to change. The rationale for dimensional modeling is presented in Chapter 5.

This book and its predecessor, *The Data Warehouse Toolkit*, are based on the discipline of dimensional modeling. We, the authors, are committed to this approach because we have seen too many data warehouses fail because of overly complex E/R designs. We have successfully employed the techniques of dimensional modeling in hundreds of design situations over the last 15 years.

The main components of a dimensional model are fact tables and dimension tables, which are defined carefully in Chapter 5. But let's look at them briefly.

A *fact table* is the primary table in each dimensional model that is meant to contain measurements of the business. Throughout this book, we will consistently use the word *fact* to represent a business measure. We will reduce terminology confusion by not using the words *measure* or *measurement*. The most useful facts are numeric and additive. Every fact table represents a many-to-many relationship and every fact table contains a set of two or more foreign keys that join to their respective dimension tables.

A *dimension table* is one of a set of companion tables to a fact table. Each dimension is defined by its primary key that serves as the basis for referential integrity with any given fact table to which it is joined. Most dimension tables contain many textual *attributes* (fields) that are the basis for constraining and grouping within data warehouse queries.

Business Process

A coherent set of business activities that make sense to the business users of our data warehouses. This definition is purposefully a little vague. A business process is usually a set of activities like "order processing" or "customer pipeline management," but business processes can overlap, and certainly the definition of an individual business process will evolve over time. In this book, we assume that a business process is a useful grouping of information resources with a coherent theme. In many cases, we will implement one or more data marts for each business process.

Data Mart

A logical subset of the complete data warehouse. A data mart is a complete "pie-wedge" of the overall data warehouse pie. A data mart represents a project that can be brought to completion rather than being an impossible galactic undertaking. A data warehouse is made up of the union of all its data marts. Beyond this rather simple logical definition, we often view the data mart as the restriction of the data warehouse to a single business process or to a group of related business processes targeted toward a particular business group. The data mart is probably sponsored by and built by a single part of the business, and a data mart is usually organized around a single business process.

We impose some very specific design requirements on every data mart. Every data mart must be represented by a dimensional model and, within a single data warehouse, all such data marts must be built from conformed dimensions and conformed facts. This is the basis of the Data Warehouse Bus Architecture. Without conformed dimensions and conformed facts, a data mart is a stovepipe. Stovepipes are the bane of the data warehouse movement. If you have any hope of building a data warehouse that is robust and resilient in the face of continuously evolving requirements, you must adhere to the data mart definition we recommend. We will show in this book that, when data marts have been designed with conformed dimensions and conformed facts, they can be combined and used together. (Read more on this topic in Chapter 5.)

We do not believe that there are two "contrasting" points of view about top-down vs. bottom-up data warehouses. The extreme top-down perspective is that a completely centralized, tightly designed master database must be completed before parts of it are summarized and published as individual data marts. The extreme bottom-up perspective is

that an enterprise data warehouse can be assembled from disparate and unrelated data marts. Neither approach taken to these limits is feasible. In both cases, the only workable solution is a blend of the two approaches, where we put in place a proper architecture that guides the design of all the separate pieces.

When all the pieces of all the data marts are broken down to individual physical tables on various database servers, as they must ultimately be, then the only physical way to combine the data from these separate tables and achieve an integrated enterprise data warehouse is if the dimensions of the data mean the same thing across these tables. We call these conformed dimensions. This Data Warehouse Bus Architecture is a fundamental driver for this book.

Finally, we do not adhere to the old data mart definition that a data mart is comprised of summary data. Data marts are based on granular data and may or may not contain performance enhancing summaries, which we call "aggregates" in this book.

Data Warehouse

The queryable source of data in the enterprise. The data warehouse is nothing more than the union of all the constituent data marts. A data warehouse is fed from the data staging area. The data warehouse manager is responsible both for the data warehouse and the data staging area.

Please understand that we (and the marketplace) have departed in a number of ways from the original definition of the data warehouse dating from the early 1990s. Specifically, the data warehouse is the *queryable* presentation resource for an enterprise's data and this presentation resource *must not* be organized around an entity-relation model because, if you use entity-relation modeling, you will lose understandability and performance. Also, the data warehouse is *frequently updated* on a controlled load basis as data is corrected, snapshots are accumulated, and statuses and labels are changed. Finally, the data warehouse is precisely the *union of its constituent data marts*.

Operational Data Store (ODS)

The term "operational data store" has taken on too many definitions to be useful to the data warehouse. We have seen this term used to describe everything from the database that underlies the operational system to the data warehouse itself. There are two primary definitions that are

worth exploring in the context of the data warehouse. Originally, the ODS was meant to serve as the point of integration for operational systems. This was especially important for legacy systems that grew up independent of each other. Banks, for example, typically had several independent systems set up to support different products—loans, checking accounts, savings accounts, and so on. The advent of teller support computers and the ATM helped push many banks to create an operational data store to integrate current balances and recent history from these separate accounts under one customer number. This kind of operational lookup is a perfect example of the useful role an ODS can play. In fact, this need for integration has been the driving force behind the success of the client/server ERP business.

Since this kind of ODS needs to support constant operational access and updates, it should be housed outside the warehouse. That is, any system structured to meet operational needs and performance requirements will be hard pressed to meet decision support needs and performance requirements. For example, you don't want someone to launch a complex scoring model that requires full table scans and aggregation of the customer history at the same time 1,000 catalog phone reps are trying to view customer history to support a one-to-one marketing relationship. This would not be good.

In the second definition, the purpose of the ODS has changed to include what sounds like decision support access by "clerks and executives." In this case, the logic seems to be that since the ODS is meant to contain integrated data at a detailed level, we should build one to support the lowest layer of the data warehouse.

In our view, these two definitions are very different. The original ODS is truly an operational system, separate from the data warehouse, with different service levels and performance requirements it must meet. The second ODS is actually the front edge of the kinds of data warehouse we design, really a part of the data warehouse and not a separate system at all.

If you have an operational data store in your systems environment, or in your plans, examine it carefully. If it is meant to play an operational, real-time role, then it truly is an operational data store and should have its own place in the systems world. If, on the other hand, it is meant to provide reporting or decision support, we encourage you to skip the ODS and meet these needs directly from the detailed level of the data warehouse. We provide additional discussion on including this detailed level in the warehouse in Chapter 9.

OLAP (On-Line Analytic Processing)

The general activity of querying and presenting text and number data from data warehouses, as well as a specifically dimensional style of querying and presenting that is exemplified by a number of "OLAP vendors." The OLAP vendors' technology is nonrelational and is almost always based on an explicit multidimensional cube of data. OLAP databases are also known as multidimensional databases, or MDDBs. OLAP vendors' data designs are often very similar to the data designs described in this book, but OLAP installations would be classified as small, individual data marts when viewed against the full range of data warehouse applications. We believe that OLAP-style data marts can be full participants on the data warehouse bus if they are designed around conformed dimensions and conformed facts.

ROLAP (Relational OLAP)

A set of user interfaces and applications that give a relational database a dimensional flavor. This book is highly consistent with both ROLAP and MOLAP approaches, although most of the specific examples come from a ROLAP perspective.

MOLAP (Multidimensional OLAP)

A set of user interfaces, applications, and proprietary database technologies that have a strongly dimensional flavor.

End User Application

A collection of tools that query, analyze, and present information targeted to support a business need. A minimal set of such tools would consist of an end user data access tool, a spreadsheet, a graphics package, and a user interface facility for eliciting prompts and simplifying the screen presentations to end users.

End User Data Access Tool

A client of the data warehouse. In a relational data warehouse, such a client maintains a session with the presentation server, sending a stream of separate SQL requests to the server. Eventually the end user data access tool is done with the SQL session and turns around to

present a screen of data or a report, a graph, or some other higher form of analysis to the user. An end user data access tool can be as simple as an ad hoc query tool, or can be as complex as a sophisticated data mining or modeling application. A few of the more sophisticated data access tools like modeling or forecasting tools may actually upload their results into special areas of the data warehouse.

Ad Hoc Query Tool

A specific kind of end user data access tool that invites the user to form their own queries by directly manipulating relational tables and their joins. Ad hoc query tools, as powerful as they are, can only be effectively used and understood by about 10 percent of all the potential end users of a data warehouse. The remaining 90 percent of the potential users must be served by pre-built applications that are much more finished "templates" that do not require the end user to construct a relational query directly. The very best ROLAP-oriented ad hoc tools improve the 10 percent number to perhaps 20 percent.

Modeling Applications

A sophisticated kind of data warehouse client with analytic capabilities that transform or digest the output from the data warehouse. Modeling applications include:

- *Forecasting models* that try to predict the future
- *Behavior scoring models* that cluster and classify customer purchase behavior or customer credit behavior
- *Allocation models* that take cost data from the data warehouse and spread the costs across product groupings or customer groupings
- Most *data mining* tools

Metadata

All of the information in the data warehouse environment that is not the actual data itself. We take an aggressive and expansive view of metadata in this book. Chapter 11 enumerates all the forms of metadata we can think of and tries to give you some guidance about how to recognize, use, and control metadata. You should catalog your metadata, version stamp your metadata, document your metadata, and backup your

metadata. But don't expect your metadata to be stored in one central database. There is too much that is metadata, and its formats and uses are too diverse.

BASIC PROCESSES OF THE DATA WAREHOUSE

Data staging is a major process that includes, among others, the following subprocesses: extracting, transforming, loading and indexing, and quality assurance checking.

- **Extracting.** The extract step is the first step of getting data into the data warehouse environment. We use this term more narrowly than some consultants. Extracting means reading and understanding the source data, and copying the parts that are needed to the data staging area for further work.
- **Transforming.** Once the data is extracted into the data staging area, there are many possible transformation steps, including
  - Cleaning the data by correcting misspellings, resolving domain conflicts (such as a city name that is incompatible with a postal code), dealing with missing data elements, and parsing into standard formats
  - Purging selected fields from the legacy data that are not useful for the data warehouse
  - Combining data sources, by matching exactly on key values or by performing fuzzy matches on non-key attributes, including looking up textual equivalents of legacy system codes
  - Creating surrogate keys for each dimension record in order to avoid a dependence on legacy defined keys, where the surrogate key generation process enforces referential integrity between the dimension tables and the fact tables
  - Building aggregates for boosting the performance of common queries
- **Loading and Indexing.** At the end of the transformation process, the data is in the form of load record images. Loading in the data warehouse environment usually takes the form of replicating the dimension tables and fact tables and presenting these tables to the bulk loading facilities of each recipient data mart. Bulk loading is a very important capability that is to be contrasted with record-at-a-time loading, which is far slower. The target data mart must then

index the newly arrived data for query performance, if it has not already done so.

- **Quality Assurance Checking.** When each data mart has been loaded and indexed and supplied with appropriate aggregates, the last step before publishing is the quality assurance step. Quality assurance can be checked by running a comprehensive exception report over the entire set of newly loaded data. All the reporting categories must be present, and all the counts and totals must be satisfactory. All reported values must be consistent with the time series of similar values that preceded them. The exception report is probably built with the data mart's end user report writing facility.
- **Release/Publishing.** When each data mart has been freshly loaded and quality assured, the user community must be notified that the new data is ready. Publishing also communicates the nature of any changes that have occurred in the underlying dimensions and new assumptions that have been introduced into the measured or calculated facts.
- **Updating.** Contrary to the original religion of the data warehouse, modern data marts may well be updated, sometimes frequently. Incorrect data should obviously be corrected. Changes in labels, changes in hierarchies, changes in status, and changes in corporate ownership often trigger necessary changes in the original data stored in the data marts that comprise the data warehouse, but in general these are "managed load updates," not transactional updates.
- **Querying.** Querying is a broad term that encompasses all the activities of requesting data from a data mart, including ad hoc querying by end users, report writing, complex decision support applications, requests from models, and full-fledged data mining. Querying never takes place in the data staging area. By definition, querying takes place on a data warehouse presentation server. Querying, obviously, is the whole point of using the data warehouse.
- **Data Feedback/Feeding in Reverse.** There are two important places where data flows "uphill" in the opposite direction from the traditional flow we have discussed in this section. First, we may upload a cleaned dimension description from the data staging area to a legacy system. This is desirable when the legacy system recognizes the value of the improved data. Second, we may upload the results of a complex query or a model run or a data mining analysis back into a data mart. This would be a natural way to capture the value

of a complex query that takes the form of many rows and columns that the user wants to save.

- **Auditing.** At times it is critically important to know where the data came from and what were the calculations performed. In Chapter 6, we discuss a technique for creating special audit records during the extract and transformation steps in the data staging area. These audit records are linked directly to the real data in such a way that a user can ask for the audit record (the lineage) of the data at any time.
- **Securing.** Every data warehouse has an exquisite dilemma: the need to publish the data widely to as many users as possible with the easiest-to-use interfaces, but at the same time protect the valuable sensitive data from hackers, snoopers, and industrial spies. The development of the Internet has drastically amplified this dilemma. The data warehouse team must now include a new senior member: the data warehouse security architect. Data warehouse security must be managed centrally, from a single console. Users must be able to access all the constituent data marts of the data warehouse with a single sign-on. In Chapter 12, we present an in-depth discussion of security issues in the data warehouse and what you should do about them.
- **Backing Up and Recovering.** Since data warehouse data is a flow of data from the legacy systems on through to the data marts and eventually onto the users' desktops, a real question arises about where to take the necessary snapshots of the data for archival purposes and disaster recovery. Additionally, it may be even more complicated to back up and recover all of the metadata that greases the wheels of the data warehouse operation. In Chapter 9, we discuss the various kinds of backup activities, and what a realistic recovery operation would entail.

THE BIG DATA WAREHOUSE DEBATES

At the time of writing this book, the data warehouse market is in the middle of a number of evolutionary changes. As an industry, we have thousands of working data marts and data warehouses under our belts. We must now revisit some of the original assumptions and restrictions we placed on ourselves in the late 1980s and early 1990s. And of course, we have very different technology to work with. In early 1998, $10,000

could buy a machine with twin 300 MHz processors, 512 MB of random access memory, and 50 GB of fast disk drive. This machine can sit on a fast Ethernet system and run any of the major relational databases, even DB2. Although many data marts need a bigger machine than this, one wonders if terabyte data marts on PC class machines are just around the corner.

At the same time, the data warehouse market has reacted strongly to the difficulty of planning and implementing a single, undifferentiated, master data warehouse for the whole enterprise. This job is just too overwhelming for most organizations and most mortal designers to even think about.

The future of data warehousing is modular, cost effective, incrementally designed, distributed data marts. The data warehouse technology will be a rich mixture of large monolithic machines that grind through massive data sets with parallel processing, together with many separate small machines (i.e., maybe only terabyte data marts!) nibbling away on individual data sets that may be granular, mildly aggregated, or highly aggregated. The separate machines will be tied together with navigator software that will serve as switchboards for dispatching queries to the servers best able to respond.

The future of data warehousing is in software advances and design discipline. Although the largest machines will continue to be even more effective at parallel processing, the smallest machines will become proportionally more powerful due to hardware advances. The biggest gains in performance, analysis power, and user interface effectiveness, however, will come from better algorithms and tighter, more predictable data designs. By adhering to the discipline of dimensional modeling, a data warehouse will be in a much better position to ride the advances being made in database software technology.

At the time of this writing, the most visible discussions in data warehousing included the topics listed in the next section. We will not develop the full arguments in this chapter, but we make our summary positions clear.

Data Warehouse Modeling

As we have already remarked several times, we believe strongly in dimensional modeling for the presentation phase of the data warehouse. Chapter 5 leads off with a detailed justification for this approach. To

summarize, dimensional modeling should be used in all the presentation servers of a data warehouse because, compared to entity-relation (E/R) modeling, this approach yields predictable, understandable designs that users can use and assimilate and that can be queried with high performance. Understandability and performance are the twin, nonnegotiable requirements of the data warehouse. The dimensional approach, unlike the E/R approach, does not require the database to be restructured or the queries to be rewritten when new data is introduced into the warehouse or when the relationships among data elements must be revised. A dimensional data mart, unlike the E/R data mart, does not need to anticipate the user's queries and is very resilient to changes in user analysis patterns.

Data Marts and Data Warehouses

Again, as we have already described, the data warehouse is nothing more than the union of its constituent data marts. These data marts avoid being stovepipes by being organized in a Bus Architecture around conformed dimensions and conformed facts. The main data design task for the data warehouse team is identifying and establishing these conformed dimensions and facts. The opposite perspective, which we disagree with, is that the data warehouse is a nonqueryable, E/R structured, centralized store of data and that data marts are disjoint and incomplete summarization of the central data warehouse that are spun off when the users demand a particular kind of analysis.

As a historical footnote, the idea that a data warehouse can be built incrementally from a series of data marts with conformed dimensions was fully described by Ralph Kimball in a DBMS magazine article in August 1996. Other descriptions of this technique, notably the "Enterprise Data Mart Architecture" with "common dimensions" and "common facts," appeared in the literature a year later. These descriptions are virtually identical to Kimball's work. The original terms "conformed dimensions" and "conformed facts" were described by Nielsen Marketing Research to Ralph Kimball in 1984, and referred to Nielsen's practice at that time of tying together syndicated scanner data with customers' internal shipments data. The terms "dimension" and "fact" originated from developments conducted jointly by General Mills and Dartmouth University in the late 1960s. It is clear that these ideas for combining data marts had been invented and introduced into the commercial mar-

ketplace long before the current generation of industry experts and consultants, even if we didn't call them data marts.

Distributed versus Centralized Data Warehouses

We feel that the tide has been coming in for some time in this industry. The idea that an organization's data warehouse is supported by a single, centralized mainframe-class machine is about as realistic as the 1950s idea that you only need one computer in an organization. At the personal computing level, we already have tens of thousand of computers in large organizations. The data warehouse is already following suit. Future data warehouses will consist of dozens or hundreds of separate machines with widely different operating systems and widely different database systems, including all flavors of OLAP. If designed correctly, these machines will share a uniform architecture of conformed dimensions and conformed facts that will allow them to be fused into a coherent whole.

We think that these last two topics, namely data warehouses consisting of many data marts and the enterprise data warehouse being a distributed system, will fuse together into a single architectural view. This view allows both the "hub and spoke" view of an overall data warehouse as well as a fully distributed view of the warehouse. We don't in any way oppose the idea of a large monolithic machine at the middle of a data warehouse operation. Some organizations will find that this makes most sense for them. Inside that monolithic machine will be hundreds of tables, organized by subject areas. We will call these groups of tables "data marts," and they will only function as a seamless whole if they possess conformed dimensions.

SUMMARY

We have defined all the parts of the data warehouse environment shown in Figure 1.1, and we have described how they work together. We have briefly touched on the big discussions taking place in the data warehouse industry today. In the next chapter it is time to turn our attention to the Business Dimensional Lifecycle, which is the framework for the rest of the book.