Data Mining and Warehouse
Unit-II

ETL Process and OLAP

By:- Dr. D.R.Patil

Outline

- Major steps in ETL Process,
- Data Extraction Techniques,
- Data Transformation: Basic tasks, Major transformation type.
- Data Loading: Applying Data, OLTP Vs OLAP, OLAP Definition, Dimensional Analysis, Hypercubes.
- OLAP Operations: Drill Down, Roll Up, Slice, Dice and Rotation.
- OLAP models: MOLAP, ROLAP.

- **Major steps in ETL Process**
  - Each of these major steps breaks down into a set of activities and tasks.
  - Use this figure as a guide to come up with a list of steps for the ETL process of your data warehouse.
  - The following list enumerates the types of activities and tasks that compose the ETL process.
  - This list is by no means complete for every data warehouse, but it gives a good insight into what is involved to complete the ETL process.

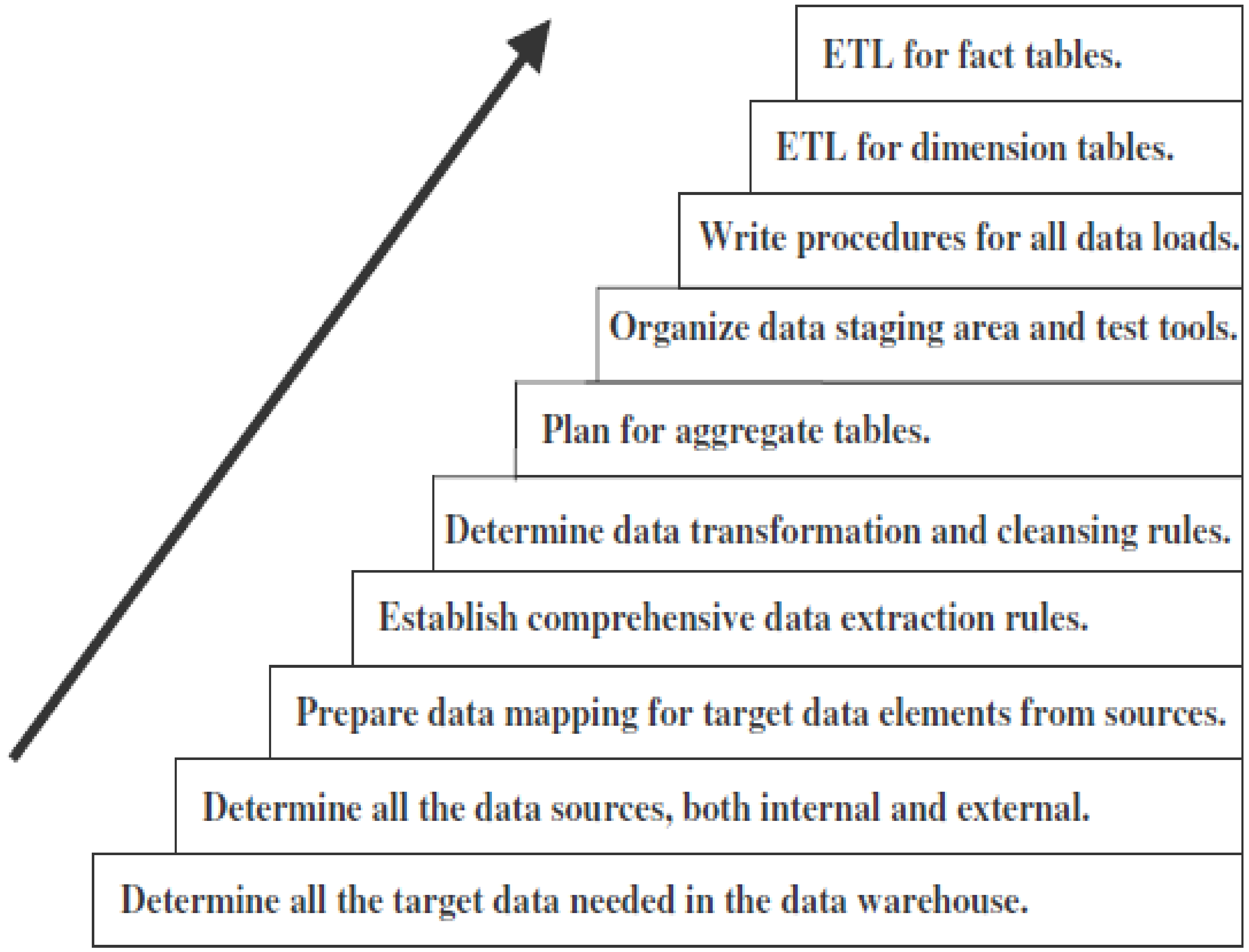

ETL for fact tables.
ETL for dimension tables.
Write procedures for all data loads.
Organize data staging area and test tools.
Plan for aggregate tables.
Determine data transformation and cleansing rules.
Establish comprehensive data extraction rules.
Prepare data mapping for target data elements from sources.
Determine all the data sources, both internal and external.
Determine all the target data needed in the data warehouse.

Figure 12-1 Major steps in the ETL process.

- **Major steps in ETL Process**
  - Combine several source data structures into a single row in the target database of the data warehouse.
  - Split one source data structure into several structures to go into several rows of the target database.
  - Read data from data dictionaries and catalogs of source systems.
  - Read data from a variety of file structures including flat files, indexed files (VSAM), and legacy system databases (hierarchical/network).
  - Load details for populating atomic fact tables.

- **Major steps in ETL Process**
  - Aggregate for populating aggregate or summary fact tables.
  - Transform data from one format in the source platform to another format in the target platform.
  - Derive target values for input fields (example: age from date of birth).
  - Change cryptic values to values meaningful to the users (example: 1 and 2 to male and female).

- Data Extraction Techniques
  - **Two major factors** differentiate the data extraction for a new operational system from the data extraction for a datawarehouse.
  - **First,** for a datawarehouse, you have to extract data from many disparate sources.
  - **Next,** for a data warehouse, you have to extract data on the changes for ongoing incremental loads as well as for a one-time initial full load.
  - For operational systems, all you need is one-time extractions and data conversions.

- **Data Extraction Techniques**
  - These two factors **increase the complexity** of data extraction for a data warehouse and, therefore,warrant the use of third-party data extraction tools in addition to in-house programs or scripts.
  - Third-party tools are generally more expensive than in-house programs, but they record their own metadata.
  - On the other hand, in-house programs increase the cost of maintenance and are hard to maintain as source systems change.

- **Data Extraction Techniques**
  - Third-party tools usually provide built-in flexibility.
  - All you have to do is to change the input parameters for the third-party tool you are using.
  - Effective data extraction is a key to the success of your data warehouse.
  - **Here is a list of data extraction issues:**
    - **Source identification**—identify source applications and source structures.
    - **Method of extraction**—for each data source, define whether the extraction process is manual or tool-based.

- Data Extraction Techniques
  - **Extraction frequency**—for each data source, establish how frequently the data extraction must be done: daily, weekly, quarterly, and so on.
  - **Time window**—for each data source, denote the time window for the extraction process.
  - **Job sequencing**—determine whether the beginning of one job in an extraction job stream has to wait until the previous job has finished successfully.
  - **Exception handling**—determine how to handle input records that cannot be extracted.

- **Data Extraction Techniques**
  - Figure 12-2 describes a stepwise approach to source identification for order fulfillment.
  - Source identification is not as simple a process as it may sound. It is a critical first process in the data extraction function.
  - You need to go through the source identification process for every piece of information you have to store in the data warehouse.
  - As you might have already figured out, source identification needs thoroughness, lots of time, and exhaustive analysis.

- **Data Extraction Techniques**
    - These two factors **increase the complexity** of data extraction for a data warehouse and, therefore,warrant the use of third-party data extraction tools in addition to in-house programs or scripts.
    - Third-party tools are generally more expensive than in-house programs, but they record their own metadata.
    - On the other hand, in-house programs increase the cost of maintenance and are hard to maintain as source systems change.

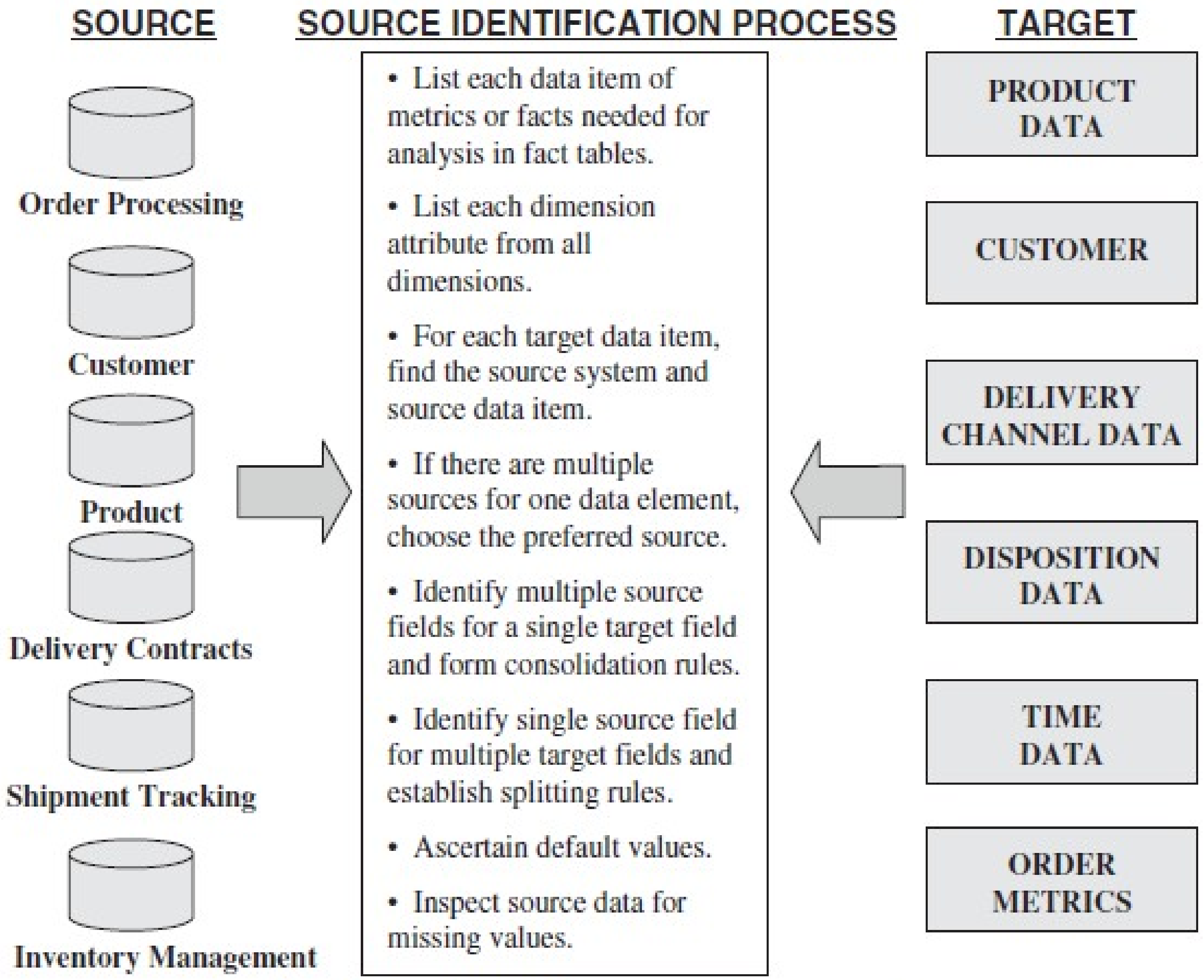

SOURCE
SOURCE IDENTIFICATION PROCESS
TARGET
Order Processing
Customer
Product
Delivery Contracts
Shipment Tracking
Inventory Management
• List each data item of metrics or facts needed for analysis in fact tables.
• List each dimension attribute from all dimensions.
• For each target data item, find the source system and source data item.
• If there are multiple sources for one data element, choose the preferred source.
• Identify multiple source fields for a single target field and form consolidation rules.
• Identify single source field for multiple target fields and establish splitting rules.
• Ascertain default values.
• Inspect source data for missing values.
PRODUCT DATA
CUSTOMER
DELIVERY CHANNEL DATA
DISPOSITION DATA
TIME DATA
ORDER METRICS

Figure 12-2 Source identification: a stepwise approach.

- Data Extraction Techniques
  - **Data in Operational Systems**
  - These source systems generally store data in two ways.
  - Operational data in the source system may be thought of as falling into two broad categories.
  - The type of data extraction technique you have to use depends on the nature of each of these two categories.
  - **Current Value**
  - Most of the attributes in the source systems fall into this category. Here the stored value of an attribute represents the value of the attribute at this moment of time.

- **Data Extraction Techniques**
  - The values are transient or transitory.
  - As business transactions happen, the values change.
  - There is no way to predict how long the present value will stay or when it will get changed next.
  - Customer name and address, bank account balances, and outstanding amounts on individual orders are some examples of this category.
  -

- **Data Extraction Techniques**
  - **Periodic Status**
  - This category is not as common as the previous category.
  - In this category, the value of the attribute is preserved as the status every time a change occurs.
  - At each of these points in time, the status value is stored with reference to the time when the new value became effective.
  - This category also includes events stored with reference to the time when each event occurred.

- **Data Extraction Techniques**
  - **Periodic Status**
    - For operational data in this category, the history of the changes is preserved in the source systems themselves.
    - Therefore, data extraction for the purpose of keeping history in the data warehouse is relatively easier.
    - Whether it is status data or data about an event, the source systems contain data at each point in time when any change occurred.
    - Study Figure 12-3 for an understanding of the two categories of data stored in the operational systems. Pay special attention to the examples.

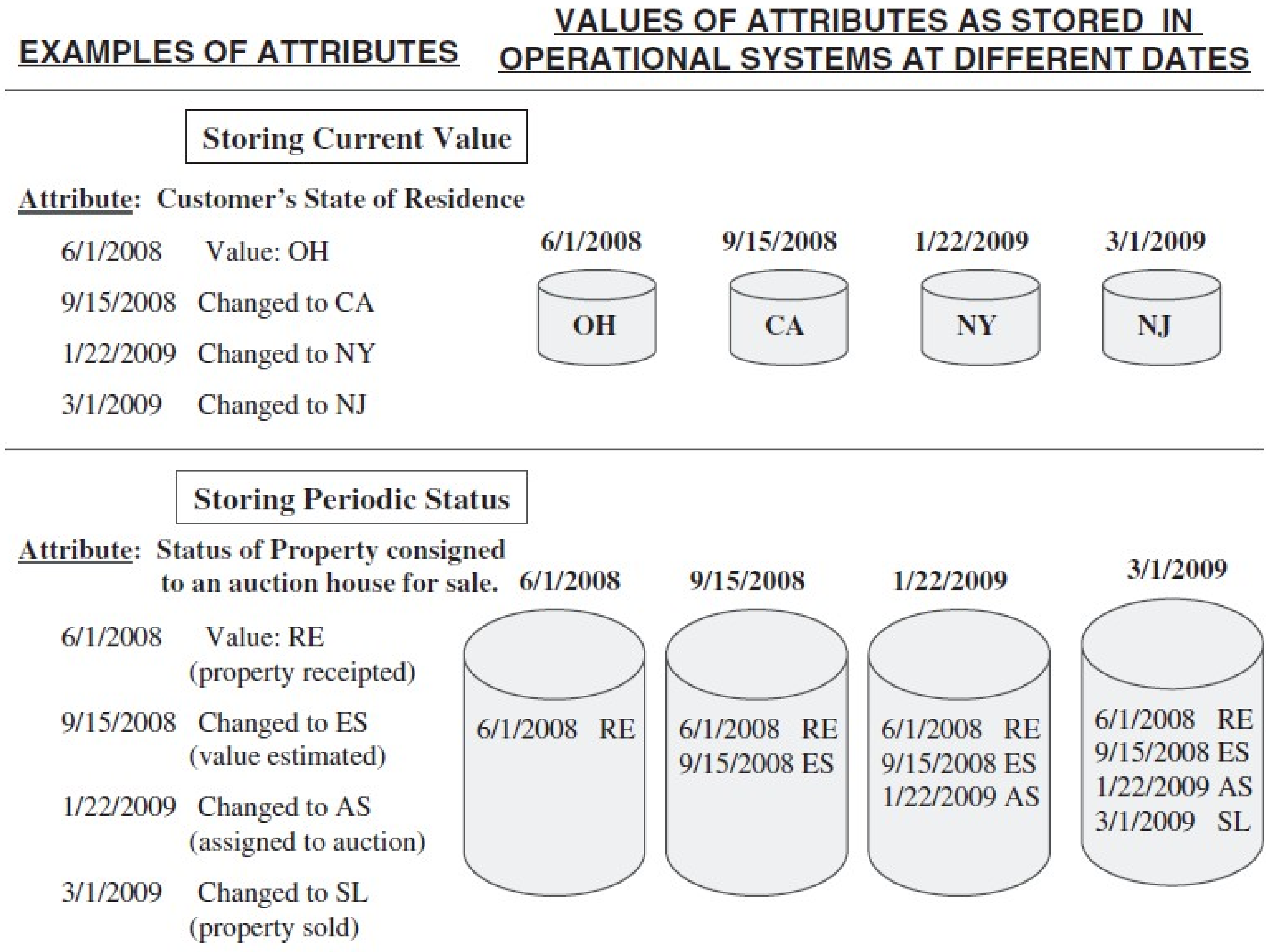

EXAMPLES OF ATTRIBUTES
VALUES OF ATTRIBUTES AS STORED IN OPERATIONAL SYSTEMS AT DIFFERENT DATES
Storing Current Value
Attribute: Customer's State of Residence
6/1/2008 Value: OH
9/15/2008 Changed to CA
1/22/2009 Changed to NY
3/1/2009 Changed to NJ
6/1/2008
OH
9/15/2008
CA
1/22/2009
NY
3/1/2009
NJ
Storing Periodic Status
Attribute: Status of Property consigned to an auction house for sale.
6/1/2008 Value: RE (property receipted)
9/15/2008 Changed to ES (value estimated)
1/22/2009 Changed to AS (assigned to auction)
3/1/2009 Changed to SL (property sold)
6/1/2008
6/1/2008 RE
9/15/2008
6/1/2008 RE
9/15/2008 ES
1/22/2009
6/1/2008 RE
9/15/2008 ES
1/22/2009 AS
3/1/2009
6/1/2008 RE
9/15/2008 ES
1/22/2009 AS
3/1/2009 SL

Figure 12-3 Data in operational systems.

- **Data Extraction Techniques**
  - Having reviewed the categories indicating how data is stored in the operational systems, we are now in a position to discuss the common techniques for data extraction.
  - When you deploy your data warehouse, the initial data as of a certain time must be moved to the data warehouse to get it started.
  - This is the initial load.
  - After the initial load, your data warehouse must be kept updated so the history of the changes and statuses are reflected in the data warehouse.

- **Data Extraction Techniques**
  - Broadly, there are two major types of data extractions from the source operational systems: **“as is” (static) data and data of revisions.**
  - **“As is” or static** data is the capture of data at a given point in time.
  - It is like taking a snapshot of the relevant source data at a certain point in time.
  - For current or transient data, this capture would include all transient data identified for extraction.

- **Data Extraction Techniques**
  - **Data of revisions** is also known as incremental data capture.
  - Strictly, it is not incremental data but the revisions since the last time data was captured.
  - If the source data is transient, the capture of the revisions is not easy.
  - For periodic status data or periodic event data, the incremental data capture includes the values of attributes at specific times.

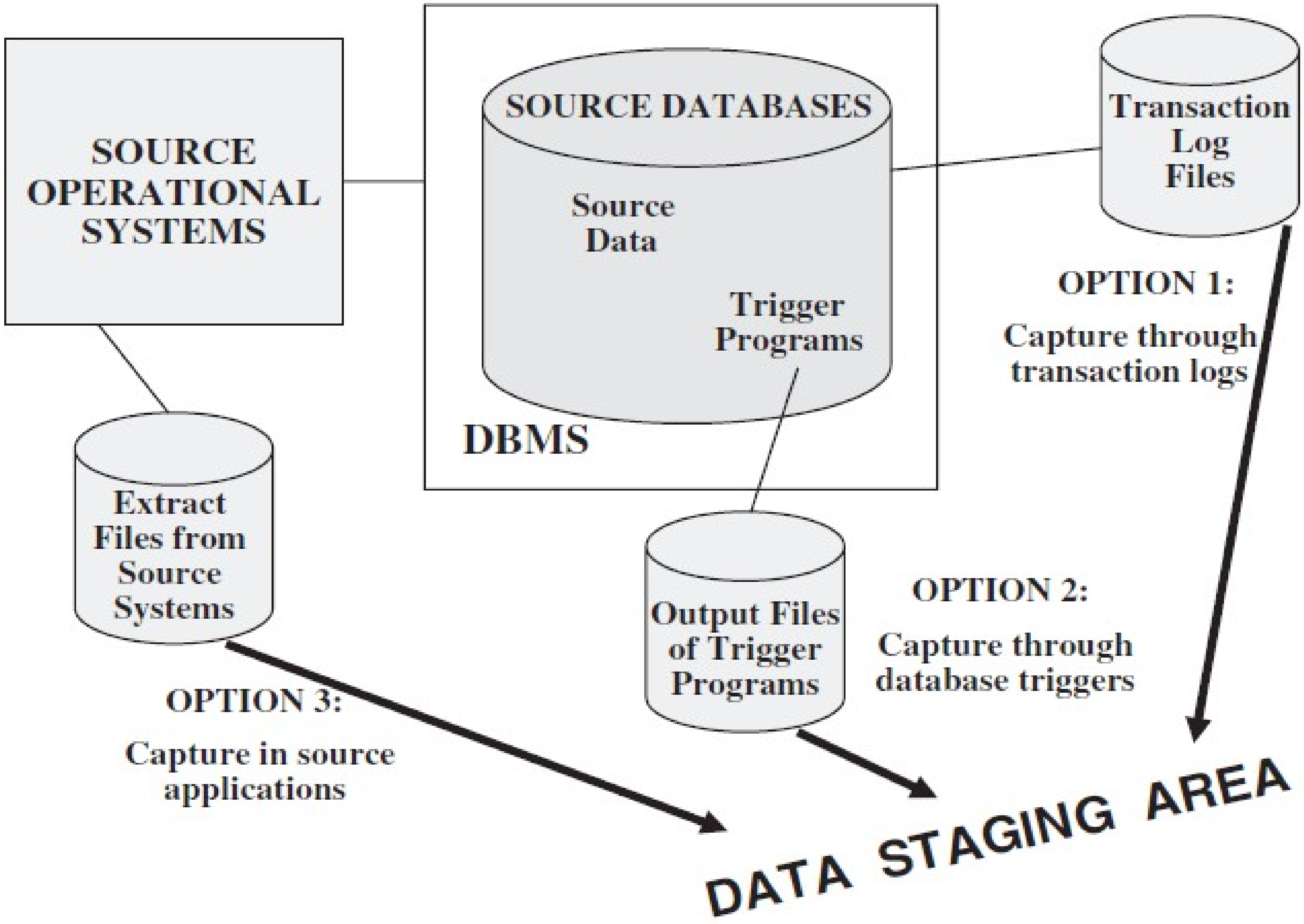

SOURCE OPERATIONAL SYSTEMS
SOURCE DATABASES
Source Data
Trigger Programs
DBMS
Transaction Log Files
OPTION 1:
Capture through transaction logs
Extract Files from Source Systems
Output Files of Trigger Programs
OPTION 2:
Capture through database triggers
OPTION 3:
Capture in source applications
DATA STAGING AREA

Figure 12-4 Options for immediate data extraction.

- **Data Extraction Techniques**
  - Now let us go into some details about the three options for immediate data extraction.
  - **Capture through Transaction Logs**
  - This option uses the transaction logs of the DBMSs maintained for recovery from possible failures.
  - As each transaction adds, updates, or deletes a row from a database table, the DBMS immediately writes entries on the log file.
  - This data extraction technique reads the transaction log and selects all the committed transactions.

- **Data Extraction Techniques**
  - Now let us go into some details about the three options for immediate data extraction.
  - **Capture through Transaction Logs**
  - This option uses the transaction logs of the DBMSs maintained for recovery from possible failures.
  - As each transaction adds, updates, or deletes a row from a database table, the DBMS immediately writes entries on the log file.
  - This data extraction technique reads the transaction log and selects all the committed transactions.

- **Data Extraction Techniques**
  - If all of your source systems are **database applications**, there is no problem with this technique.
  - But if some of your source system data is on **indexed and other flat files,** this option will not work for these cases. There a**re no log files** for these non-database applications.
  - You will have to devise some other data extraction technique for these cases.
  - **Data replication** is simply a method for creating copies of data in a distributed environment.
  - Figure 12-5 illustrates how replication technology can be used to capture changes to source data.

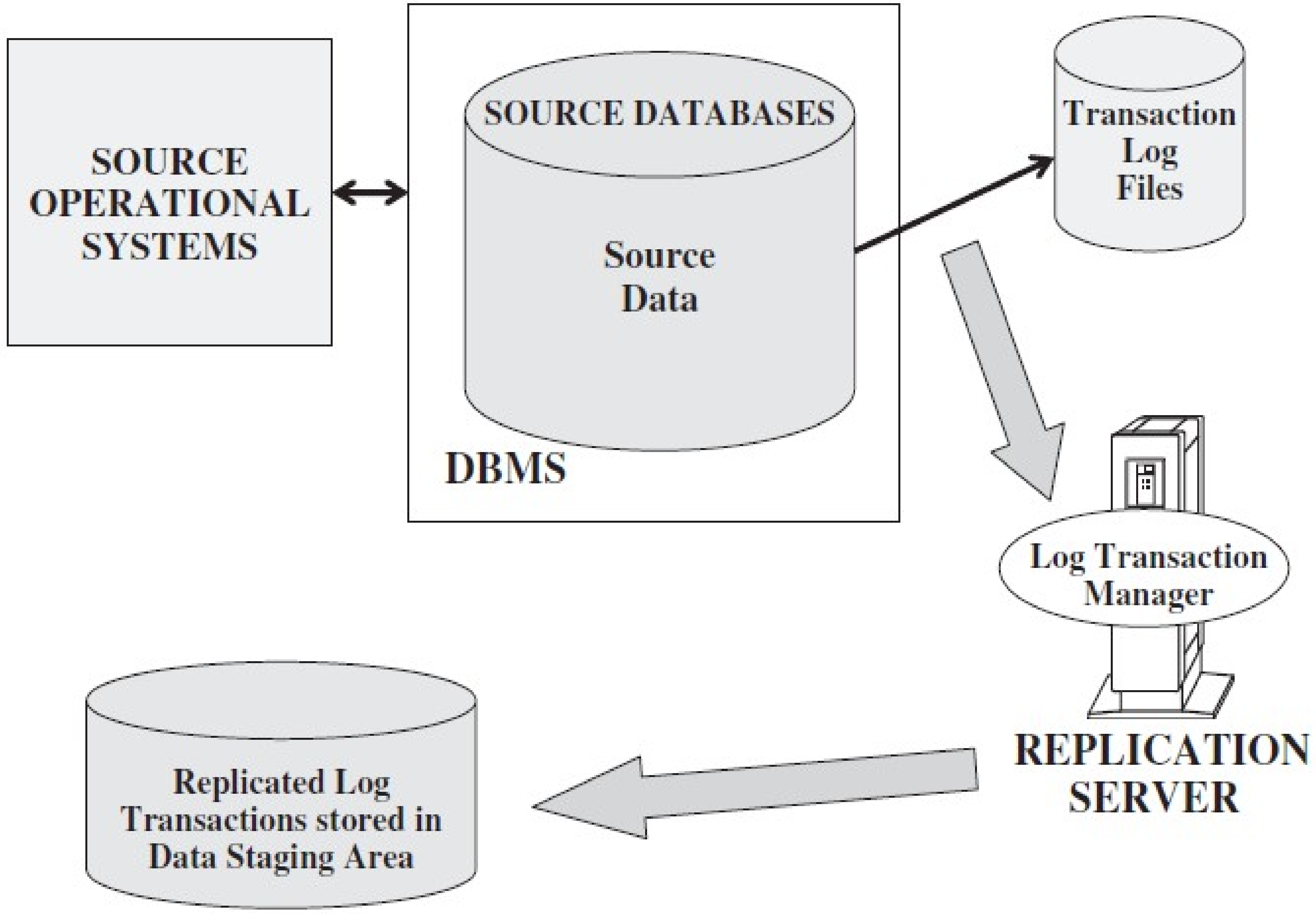


Figure 12-5 Data extraction using replication technology.

- **Data Extraction Techniques**
  - The appropriate transaction logs contain all the changes to the various source database tables.
  - Here are the broad steps for using replication to capture changes to source data:
    - Identify the source system database table
    - Identify and define target files in the staging area
    - Create mapping between the source table and target files
    - Define the replication mode
    - Schedule the replication process

- **Data Extraction Techniques**
    - Capture the changes from the transaction logs
    - Transfer captured data from logs to target files
    - Verify transfer of data changes
    - Confirm success or failure of replication
    - In metadata, document the outcome of replication
    - Maintain definitions of sources, targets, and mappings

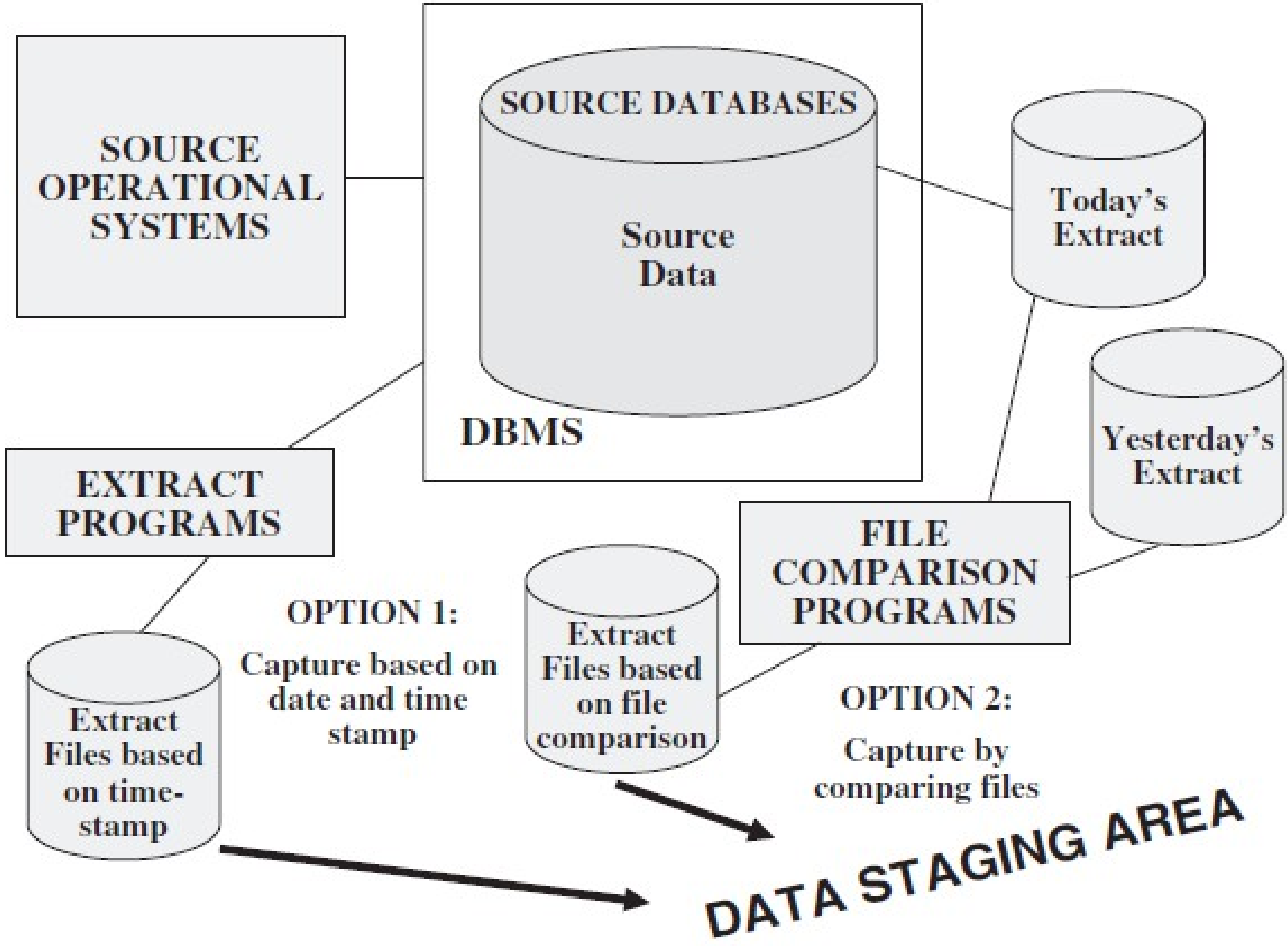
- **Data Extraction Techniques**
  - **Capture through Database Triggers**
  - Again, this option is applicable to your source systems that are database applications.
  - As you know, triggers are special stored procedures (programs) that are stored on the database and fired when certain predefined events occur.
  - You can create trigger programs for all events for which you need data to be captured.
  - The output of the trigger programs is written to a separate file that will be used to extract data for the data warehouse.

- **Data Extraction Techniques**
  - **Capture through Database Triggers**
  - For example, if you need to capture all changes to the records in the customer table, write a trigger program to capture all updates and deletes in that table.
  - However, building and maintaining trigger programs puts an additional burden on the development effort.
  - Also, execution of trigger procedures during transaction processing of the source systems puts additional overhead on the source systems.
  - Further, this option is applicable only for source data in databases.

- **Data Extraction Techniques**
  - **Capture in Source Applications**
  - This technique is also referred to as application assisted data capture.
  - In other words, the source application is made to assist in the data capture for the data warehouse.
  - You have to modify the relevant application programs that write to the source files and databases.
  - You revise the programs to write all adds, updates, and deletes to the source files and database tables.

- **Data Extraction Techniques**
  - **Capture in Source Applications**
  - Then other extract programs can use the separate file containing the changes to the source data.
  - Unlike the previous two cases, this technique may be used for all types of source data irrespective of whether it is in databases, indexed files, or other flat files.
  - But you have to revise the programs in the source operational systems and keep them maintained.
  - Also, this technique may degrade the performance of the source applications because of the additional processing needed to capture the changes on separate files.

- **Data Extraction Techniques**
  - **Deferred Data Extraction**
  - In the cases discussed above, data capture takes place while the transactions occur in the source operational systems.
  - The data capture is immediate or real-time.
  - In contrast, the techniques under deferred data extraction **do not capture the changes in real time**.
  - **The capture happens later**. Figure 12-6 shows the deferred data extraction options. Now let us discuss the two options for deferred data extraction.



SOURCE OPERATIONAL SYSTEMS
SOURCE DATABASES
Source Data
DBMS
Today's Extract
Yesterday's Extract
EXTRACT PROGRAMS
FILE COMPARISON PROGRAMS
OPTION 1:
Capture based on date and time stamp
Extract Files based on time-stamp
Extract Files based on file comparison
OPTION 2:
Capture by comparing files
DATA STAGING AREA

Figure 12-6 Options for deferred data extraction.

- **Data Extraction Techniques**
  - **Capture Based on Date and Time Stamp**
  - Every time a source record is created or updated it may be marked with a stamp showing the date and time.
  - The time stamp provides the basis for selecting records for data extraction.
  - Here the data capture occurs at a later time, not while each source record is created or updated.
  - If you run your data extraction program at midnight every day, each day you will extract only those with the date and time stamp later than midnight of the previous day.

- Data Extraction Techniques
  - **Capture Based on Date and Time Stamp**
    - This technique works well if the number of revised records is small.
    - Of course, this technique presupposes that all the relevant source records contain date and time stamps.
    - Provided this is true, data capture based on date and time stamp can work for any type of source file.
    - This technique captures the latest state of the source data.

- Data Extraction Techniques
  - **Capture by Comparing Files**
  - If none of the **above techniques are feasible for** specific source files in your environment, then consider this technique as the last resort.
  - This technique is **also called the snapshot differential technique** because it compares two snapshots of the source data.

- **Data Extraction Techniques**
  - **Capture by Comparing Files**
  - Suppose you want to apply this technique to capture the changes to your product data.
  - While performing today’s data extraction for changes to product data, you do a full file comparison between today’s copy of the product data and yesterday’s copy.
  - You also compare the record keys to find the inserts and deletes.
  - Then you capture any changes between the two copies.

- Data Extraction Techniques
  - **Capture by Comparing Files**
  - **This technique necessitates** the keeping of prior copies of all the relevant source data.
  - Though simple and straightforward, **comparison of full rows in a large file can be very inefficient.**
  - However, this may be the only feasible option for some legacy data sources that do not have transaction logs or time stamps on source records.

- **Evaluation of the Techniques**
  - To summarize, the following options are available for data extraction:
    - Capture of static data
    - Capture through transaction logs
    - Capture through database triggers
    - Capture in source applications
    - Capture based on date and time stamp
    - Capture by comparing files

Capture of static data

Good flexibility for capture specifications.
Performance of source systems not affected.
No revisions to existing applications.
Can be used on legacy systems.
Can be used on file-oriented systems.
Vendor products are used. No internal costs.

Capture in source applications

Good flexibility for capture specifications.
Performance of source systems affected a bit.
Major revisions to existing applications.
Can be used on most legacy systems.
Can be used on file-oriented systems.
High internal costs because of in-house work.

Capture through transaction logs

Not much flexibility for capture specifications.
Performance of source systems not affected.
No revisions to existing applications.
Can be used on most legacy systems.
Cannot be used on file-oriented systems.
Vendor products are used. No internal costs.

Capture based on date and time stamp

Good flexibility for capture specifications.
Performance of source systems not affected.
Major revisions to existing applications likely.
Cannot be used on most legacy systems.
Can be used on file-oriented systems.
Vendor products may be used.

Capture through database triggers

Not much flexibility for capture specifications.
Performance of source systems affected a bit.
No revisions to existing applications.
Cannot be used on most legacy systems.
Cannot be used on file-oriented systems.
Vendor products are used. No internal costs.

Capture by comparing files

Good flexibility for capture specifications.
Performance of source systems not affected.
No revisions to existing applications.
May be used on legacy systems.
May be used on file-oriented systems.
Vendor products are used. No internal costs.

Figure 12-7 Data capture techniques: advantages and disadvantages.

- **DATA TRANSFORMATION**
  - Irrespective of the variety and complexity of the source operational systems, and regardless of the extent of your data warehouse, you will find that most of your data transformation functions break down into a few basic tasks.
  - Let us go over these basic tasks so that you can view data transformation from a fundamental perspective.
  - Here is the set of basic tasks:

- DATA TRANSFORMATION
  - **Selection.**
  - This takes place at the beginning of the whole process of data transformation.
  - You select either whole records or parts of several records from the source systems.
  - The task of selection usually forms part of the extraction function itself.
  - However, in some cases, the composition of the source structure may not be amenable to selection of the necessary parts during data extraction.

- DATA TRANSFORMATION
  - **Splitting/Joining.**
  - This task includes the types of data manipulation you need to perform on the selected parts of source records.
  - Sometimes (uncommonly), you will be splitting the selected parts even further during data transformation.
  - Joining of parts selected from many source systems is more widespread in the data warehouse environment.

- DATA TRANSFORMATION

  - **Conversion.**
  - This is an all-inclusive task.
  - It includes a large variety of rudimentary conversions of single fields for two primary reasons—one to standardize among the data extractions from disparate source systems, and the other to make the fields usable and understandable to the users.

  - **Summarization.** Sometimes you may find that it is not feasible to keep data at the lowest level of detail in your data warehouse.
  - It may be that none of your users ever need data at the lowest granularity for analysis or querying.

- DATA TRANSFORMATION
  - **Enrichment.** This task is the rearrangement and simplification of individual fields to make them more useful for the data warehouse environment.
  - You may use one or more fields from the same input record to create a better view of the data for the data warehouse.
  - This principle is extended when one or more fields originate from multiple records, resulting in a single field for the data warehouse.

- **Major Transformation Types**
  - **Format Revisions.** You will come across these quite often.
  - These revisions include changes to the data types and lengths of individual fields.
  - In your source systems, product package types may be indicated by codes and names in which the fields are numeric and text data types.
  - Again, the lengths of the package types may vary among the different source systems.
  - It is wise to standardize and change the data type to text to provide values meaningful to the users.

- **Major Transformation Types**
  - **Decoding of Fields.** This is also a common type of data transformation.
  - When you deal with multiple source systems, you are bound to have the same data items described by a plethora of field values.
  - The classic example is the coding for gender, with one source system using 1 and 2 for male and female and another system using M and F.
  - Also, many legacy systems are notorious for using cryptic codes to represent business values.

•

- **Major Transformation Types**
  - **Calculated and Derived Values.** What if you want to keep profit margin along with sales and cost amounts in your data warehouse tables?
  - The extracted data from the sales system contains sales amounts, sales units, and operating cost estimates by product.
  - You will have to calculate the total cost and the profit margin before data can be stored in the datawarehouse.
  - Average daily balances and operating ratios are examples of derived fields.

-

- **Major Transformation Types**
  - **Splitting of Single Fields.** Earlier legacy systems stored names and addresses of customers and employees in large text fields.
  - The first name, middle initials, and last name were stored as a large text in a single field.
  - Similarly, some earlier systems stored city, state, and zip code data together in a single field.
  - You need to store individual components of names and addresses in separate fields in your data warehouse for two reasons.
  - First, you may improve the operating performance by indexing on individual components.
  - Second, your users may need to perform analysis by using individual components such as city, state, and zip code.

-

- **Major Transformation Types**
  - **Merging of Information.** This is not quite the opposite of splitting of single fields.
  - This type of data transformation does not literally mean the merging of several fields to create a single field of data.
  - For example, information about a product may come from different data sources.
  - The product code and description may come from one data source.
  - The relevant package types may be found in another data source.
  - The cost data may be from yet another source.
  - In this case, merging of information denotes the combination of the product code, description, package types, and cost into a single entity.

-

- **Major Transformation Types**
  - **Character set conversion.** This type of data transformation relates to the conversion of character sets to an agreed standard character set for textual data in the data warehouse.
  - If you have mainframe legacy systems as source systems, the source data from these systems will be in EBCDIC characters.
  - If PC-based architecture is the choice for your data warehouse, then you must convert the mainframe EBCDIC format to the ASCII format.
  - When your source data is on other types of hardware and operating systems, you are faced with similar character set conversions.
  -

- **Major Transformation Types**
  - **Conversion of Units of Measurements.** Many companies today have global branches.
  - Measurements in many European countries are in metric units.
  - If your company has overseas operations, you may have to convert the metrics so that the numbers are all in one standard unit of measurement.
  - **Date/Time Conversion.** This type relates to representation of date and time in standard formats.
  - For example, the American and the British date formats may be standardized to an international format.
  - The date of October 11, 2008 is written as 10/11/2008 in the U.S. format and as 11/10/2008 in the British format.
  - This date may be standardized to be written as 11 OCT 2008.

- **Major Transformation Types**
  - **Summarization**. This type of transformation is the creating of summaries to be loaded in the data warehouse instead of loading the most granular level of data.
  - For example, for a credit card company to analyze sales patterns, it may not be necessary to store in the data warehouse every single transaction on each credit card.
  - Instead, you may want to summarize the daily transactions for each credit card and store the summary data instead of storing the most granular data by individual transactions.
  -

- **Major Transformation Types**
  - **Key Restructuring**. While extracting data from your input sources, look at the primary keys of the extracted records.
  - You will have to come up with keys for the fact and dimension tables based on the keys in the extracted records.
  - See Figure 12-8. In the example shown in the figure, the product code in this organization is structured to have inherent meaning.
  - If you use this product code as the primary key, there will be problems.
  - If the product is moved to another warehouse, the warehouse part of the product key will have to be changed. This is a typical problem with legacy systems.
  - When choosing keys for your data warehouse database tables, avoid such keys with built-in meanings.
  - Transform such keys into generic keys generated by the system itself**. This is called key restructuring.**
  -

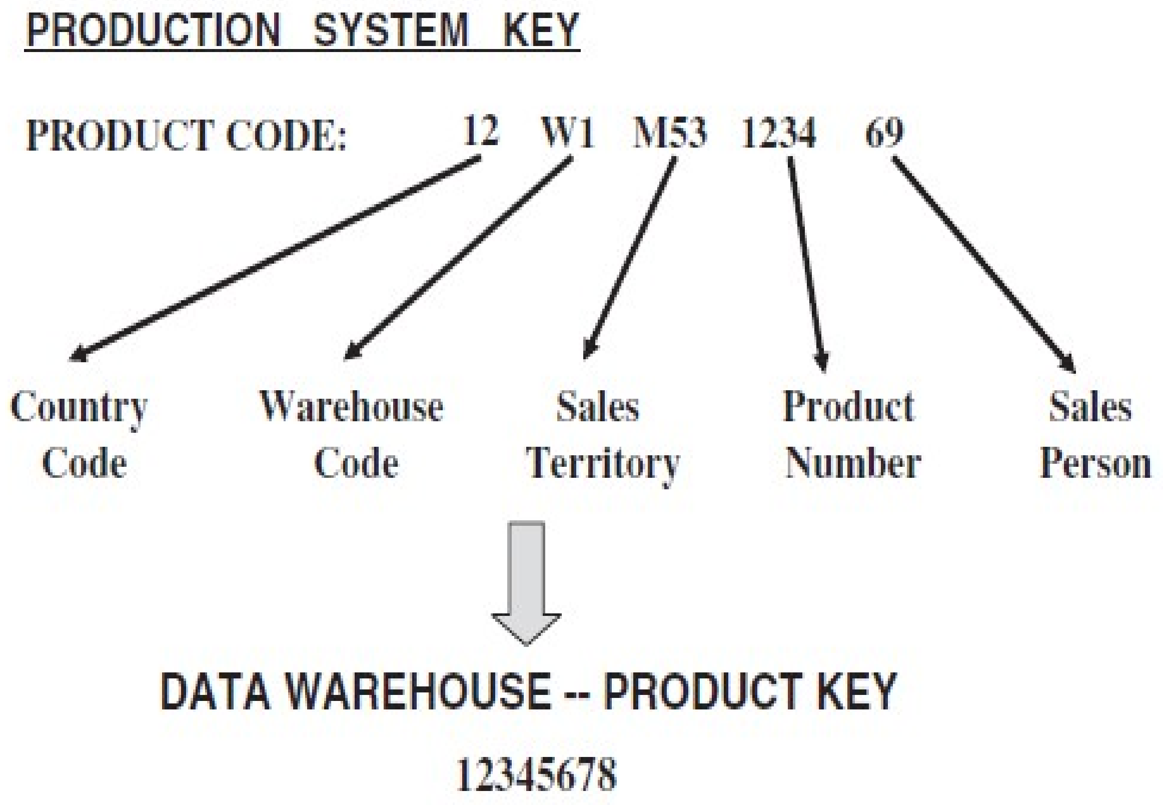


Figure 12-8 Data transformation: key restructuring.

- **Major Transformation Types**
  - **Deduplication.** In many companies, the customer files have several records for the same customer.
  - Mostly, the duplicates are the result of creating additional records by mistake.
  - In your data warehouse, you want to keep a single record for one customer and link all the duplicates in the source systems to this single record.
  - **This process is called deduplication** of the customer file.
  - Employee files and, sometimes, product master files have this kind of duplication problem.
  -

- **Data Integration and Consolidation**
  - The real challenge of ETL functions is the pulling together of all the source data from many disparate, dissimilar source systems.
  - Even today, many data warehouses get data extracted from a combination of legacy mainframe systems and old minicomputer applications, in addition to newer client/server systems.
  - Most of these source systems do not conform to the same set of business rules.
  - Very often they follow different naming conventions and varied standards for data representation.
  - Figure 12-9 shows a typical data source environment.
  - Notice the challenging issues indicated in the figure.

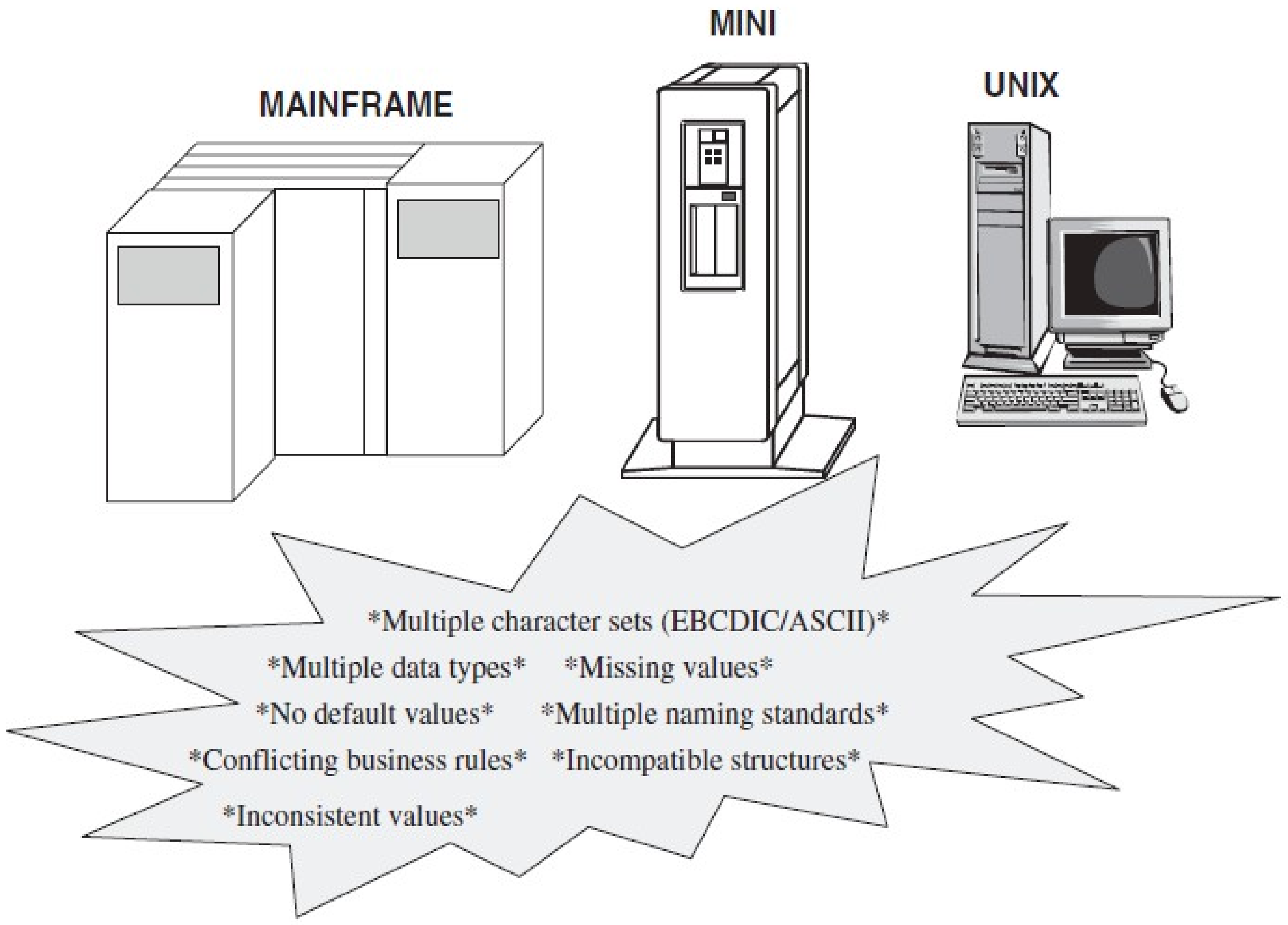

MAINFRAME
MINI
UNIX
Multiple character sets (EBCDIC/ASCII)
Multiple data types
Missing values
No default values
Multiple naming standards
Conflicting business rules
Incompatible structures
Inconsistent values

Figure 12-9 Typical data source environment.

- **How to Implement Transformation**
  - The complexity and the extent of data transformation strongly suggest that manual methods alone will not be enough.
  - You must go beyond the usual methods of writing conversion programs when you deployed operational systems.
  - The types of data transformation are by far more difficult and challenging.
  - The methods you may want to adopt depend on some significant factors.
  - If you are considering automating most of the data transformation functions, first consider if you have the time to select the tools, configure and install them, train the project team on the tools, and integrate the tools into the data warehouse environment.

-

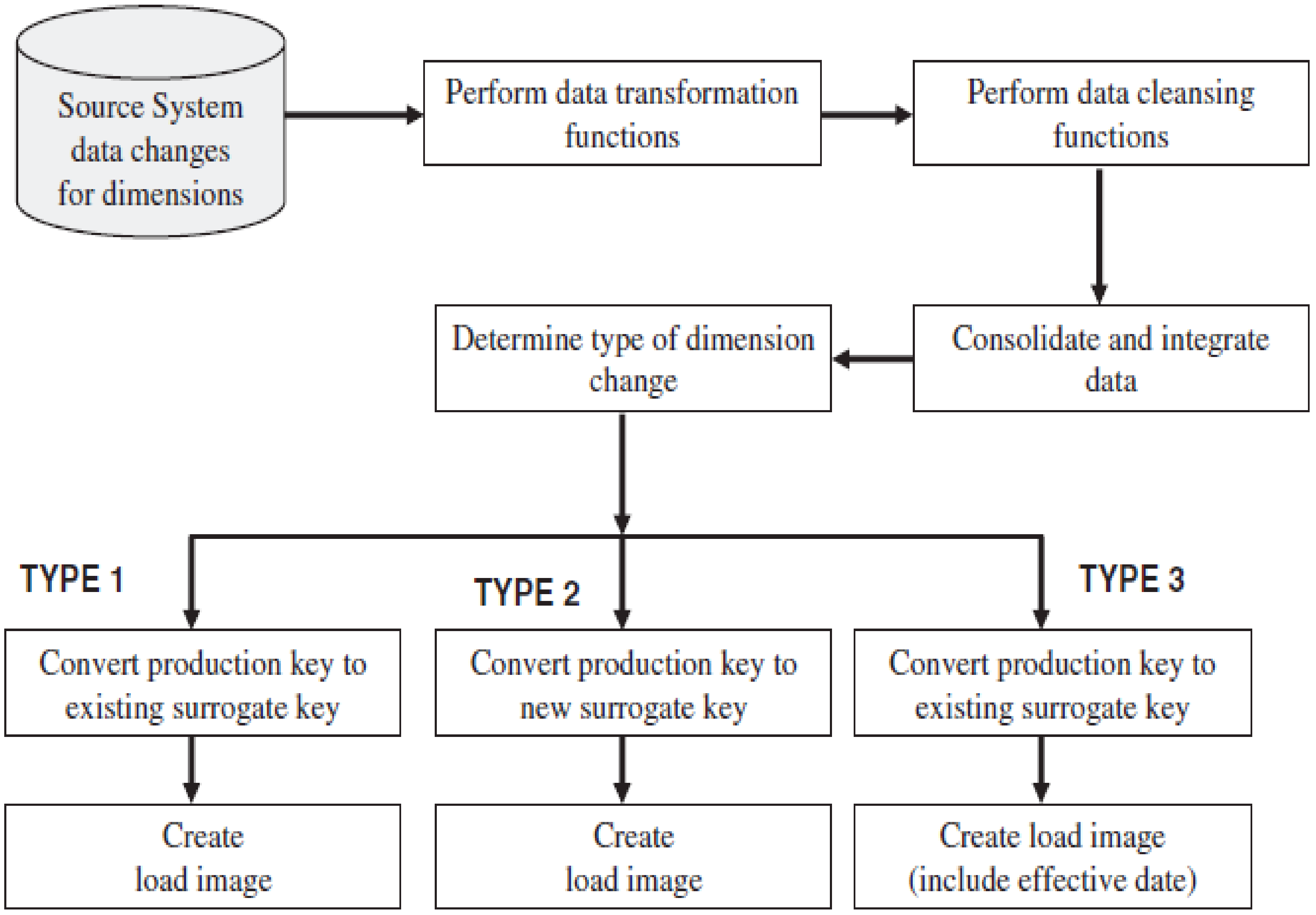

Source System data changes for dimensions
Perform data transformation functions
Perform data cleansing functions
Consolidate and integrate data
Determine type of dimension change
TYPE 1
TYPE 2
TYPE 3
Convert production key to existing surrogate key
Convert production key to new surrogate key
Convert production key to existing surrogate key
Create load image
Create load image
Create load image (include effective date)

Figure 12-10 Transformed for dimension changes.

- DATA LOADING
  - It is generally agreed that transformation functions end as soon as load images are created.
  - The next major set of functions consists of the ones that take the prepared data, apply it to the data warehouse, and store it in the database there.
  - You create load images to correspond to the target files to be loaded in the data warehouse database.
  - The whole process of moving data into the data warehouse repository is referred to in several ways.
  - You must have heard the phrases applying the data, loading the data, and refreshing the data.
  -

- DATA LOADING
  - For the sake of clarity we will use the phrases as indicated below:
    - **Initial load**—populating all the data warehouse tables for the very first time.
    - **Incremental load**—applying ongoing changes as necessary in a periodic manner.
    - **Full refresh**—completely erasing the contents of one or more tables and reloading with fresh data (initial load is a refresh of all the tables).

- **Applying Data: Techniques and Processes**
  - Earlier in this section, we defined three types of application of data to the data warehouse: **initial load, incremental load, and full refresh.**
  - Consider how data is applied in each of these types.
  - Let us take the example of product data.
  - **For the initial load**, you extract the data for all the products from the various source systems, integrate and transform the data, and then create load images for loading the data into the product dimension table.
  - **For an incremental load**, you collect the changes to the product data for those product records that have changed in the source systems since the previous extract, run the changes through the integration and transformation process, and create output records to be applied to the product dimension table.

- **Applying Data: Techniques and Processes**
  - Earlier in this section, we defined three types of application of data to the data warehouse: **initial load, incremental load, and full refresh.**
  - Consider how data is applied in each of these types.
  - Let us take the example of product data.
  - **For the initial load**, you extract the data for all the products from the various source systems, integrate and transform the data, and then create load images for loading the data into the product dimension table.
  - **For an incremental load**, you collect the changes to the product data for those product records that have changed in the source systems since the previous extract, run the changes through the integration and transformation process, and create output records to be applied to the product dimension table. **A full refresh is similar to the initial load.**

- **Applying Data: Techniques and Processes**
  - How can you apply the data to the warehouse? What are the modes?
  - Data may be applied in the following four different modes:
    - load,
    - append,
    - destructive merge, and
    - constructive merge.
  - Study Figure 12-11 for an understanding of the effect of applying data in each of these four modes.
  - Let us explain how each mode works.

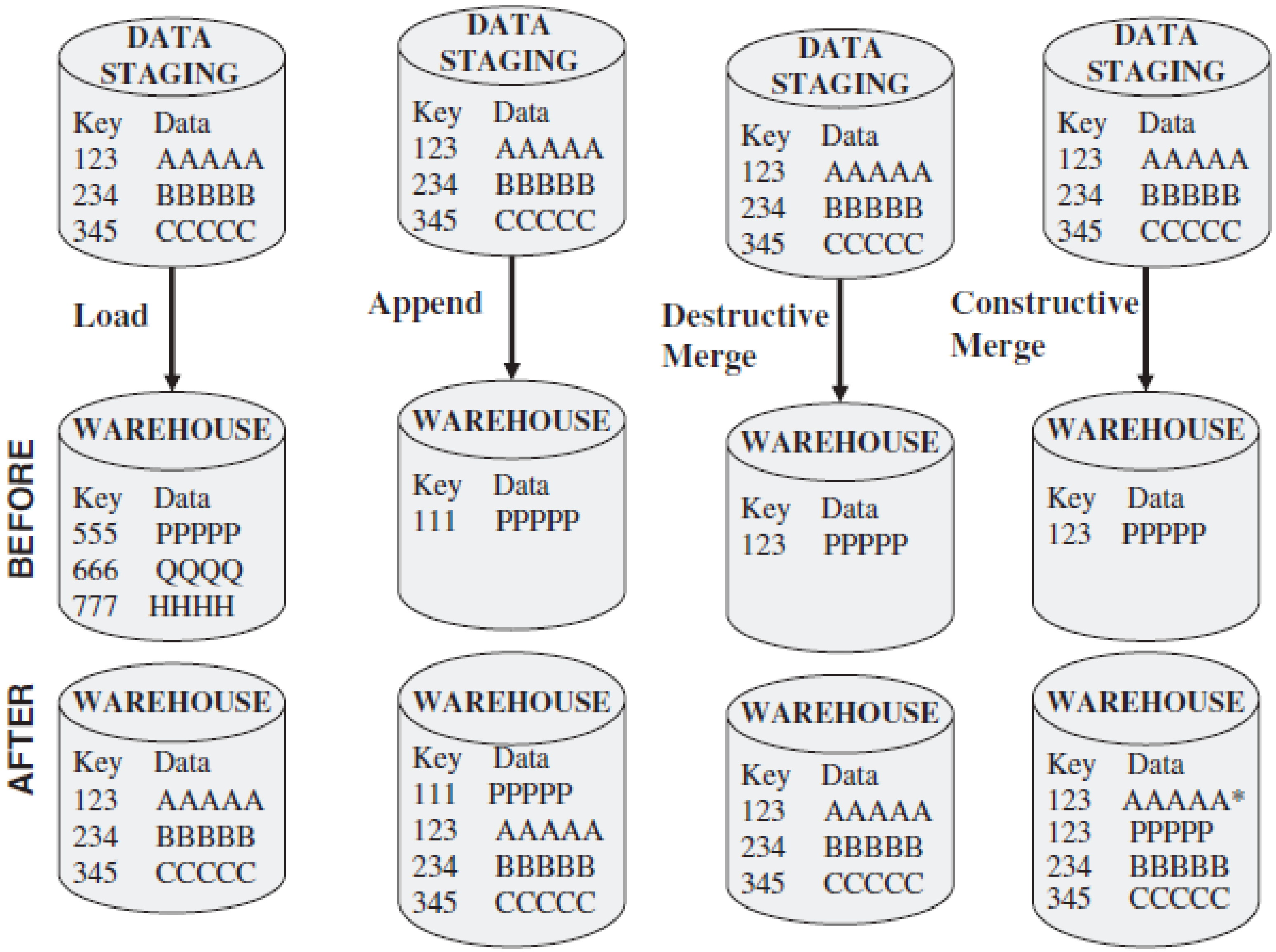


Figure 12-11 Modes of applying data.

- **Applying Data: Techniques and Processes**
  - **Load:** If the target table to be loaded already exists and data exists in the table, the load process wipes out the existing data and applies the data from the incoming file.
  - If the table is already empty before loading, the load process simply applies the data from the incoming file.
  - **Append:** You may think of the append as an extension of the load.
  - If data already exists in the table, the append process unconditionally adds the incoming data, preserving the existing data in the target table.
  - When an incoming record is a duplicate of an already existing record, you may define how to handle an incoming duplicate. The incoming record may be allowed to be added as a duplicate. In the other option, the incoming duplicate record may be rejected during the append process

- **Applying Data: Techniques and Processes**
  - **Destructive Merge:**
  - In this mode, you apply the incoming data to the target data.
  - If the primary key of an incoming record matches with the key of an existing record, update the matching target record.
  - If the incoming record is a new record without a match with any existing record, add the incoming record to the target table.

- **Applying Data: Techniques and Processes**
  - **Constructive Merge:** This mode is slightly different from the destructive merge.
  - If the primary key of an incoming record matches with the key of an existing record, leave the existing record, add the incoming record, and mark the added record as superseding the old record.
  - Let us now consider how these modes of applying data to the data warehouse fit into the three types of loads.

- **Applying Data: Techniques and Processes**
  - **Initial Load:** Let us say you are able to load the whole data warehouse in a single run.
  - As a variation of this single run, let us say you are able to split the load into separate subloads and run each of these subloads as single loads.
  - **In other words, every load run creates the database tables from scratch. In these cases, you will be using the load mode discussed above.**
  - If you need more than one run to create a single table, and your load runs for a single table must be scheduled to run on several days, then the approach is different.
  - For the first run of the initial load of a particular table, use the load mode. All further runs will apply the incoming data using the append mode.

- **Applying Data: Techniques and Processes**
  - **Initial Load:** Let us say you are able to load the whole data warehouse in a single run.
  - As a variation of this single run, let us say you are able to split the load into separate subloads and run each of these subloads as single loads.
  - **In other words, every load run creates the database tables from scratch. In these cases, you will be using the load mode discussed above.**
  - If you need more than one run to create a single table, and your load runs for a single table must be scheduled to run on several days, then the approach is different.
  - For the first run of the initial load of a particular table, use the load mode. All further runs will apply the incoming data using the append mode.

- **Applying Data: Techniques and Processes**
    - **Initial Load:** Let us say you are able to load the whole data warehouse in a single run.
    - As a variation of this single run, let us say you are able to split the load into separate subloads and run each of these subloads as single loads.
    - **In other words, every load run creates the database tables from scratch. In these cases, you will be using the load mode discussed above.**
    - If you need more than one run to create a single table, and your load runs for a single table must be scheduled to run on several days, then the approach is different.
    - For the first run of the initial load of a particular table, use the load mode. All further runs will apply the incoming data using the append mode.

- **Applying Data: Techniques and Processes**
  - **Incremental Loads:** These are the applications of ongoing changes from the source systems.
  - Changes to the source systems are always tied to specific times, irrespective of whether or not they are based on explicit time stamps in the source systems.
  - Therefore, you need a method to preserve the periodic nature of the changes in the data warehouse.
  - **Full Refresh:** This type of application of data involves periodically rewriting the entire data warehouse.
  - Sometimes, you may also do partial refreshes to rewrite only specific tables.
  - Partial refreshes are rare because every dimension table is intricately tied to the fact table.

- **Applying Data: Techniques and Processes**
  - **Data Refresh Versus Update**
  - After the initial load, you may maintain the data warehouse and keep it up to date by using two methods:
  - **Update**—application of incremental changes in the data sources.
  - **Refresh**—complete reload at specified intervals.
  - Technically, refresh is a much simpler option than update.
  - To use the update option, you have to devise the proper strategy to extract the changes from each data source.
  - Then you have to determine the best strategy to apply the changes to the data warehouse.
  - The refresh option simply involves the periodic replacement of complete data warehouse tables

- Applying Data: Techniques and Processes
  - Data Refresh Versus Update
  - **But refresh jobs can take a long time to ru**n. If you have to run refresh jobs every day, you may have to keep the data warehouse down for unacceptably long times.
  - The case worsens if your database has large tables.
  - Is there some kind of a guideline as to when refresh is better than update or vice versa?
  - Figure 12-12 shows a graph comparing refresh with update. The cost of refresh remains constant irrespective of the number of changes in the source systems. If the number of changes increases, the time and effort for doing a full refresh remain the same.
  - O**n the other hand, the cost of update varies with the number of records to be updated.**

After the initial load, the data warehouse is kept up-to-date by

REFRESH -	complete reload at specified intervals
UPDATE -	application of incremental changes

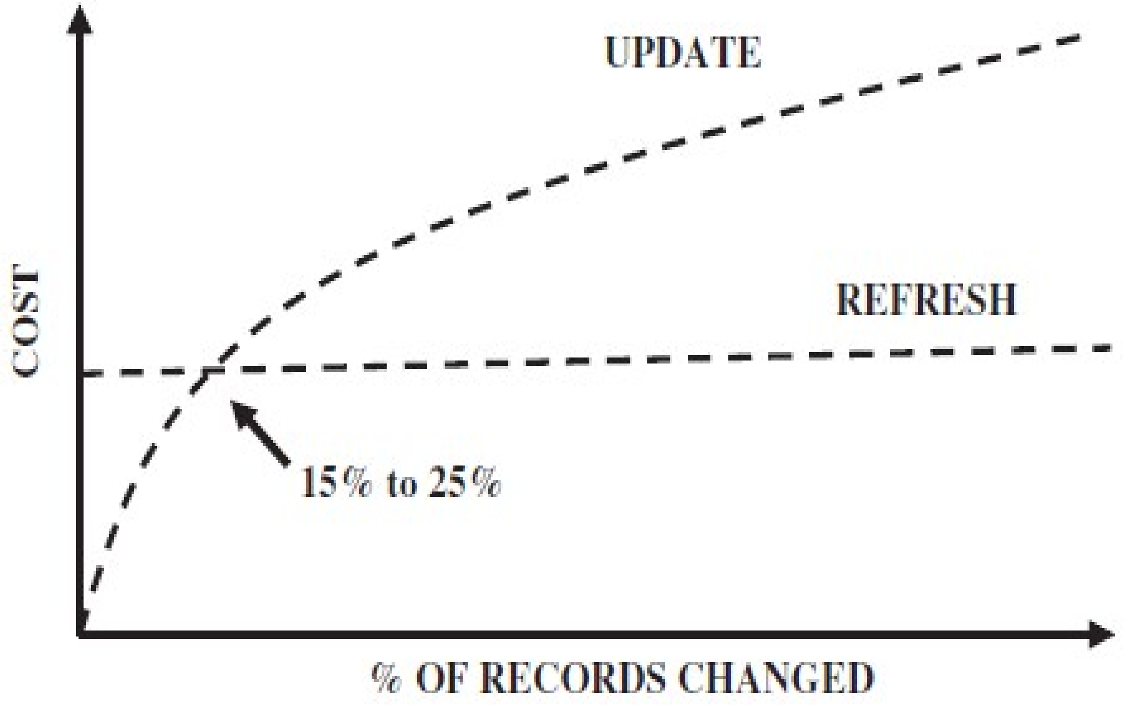
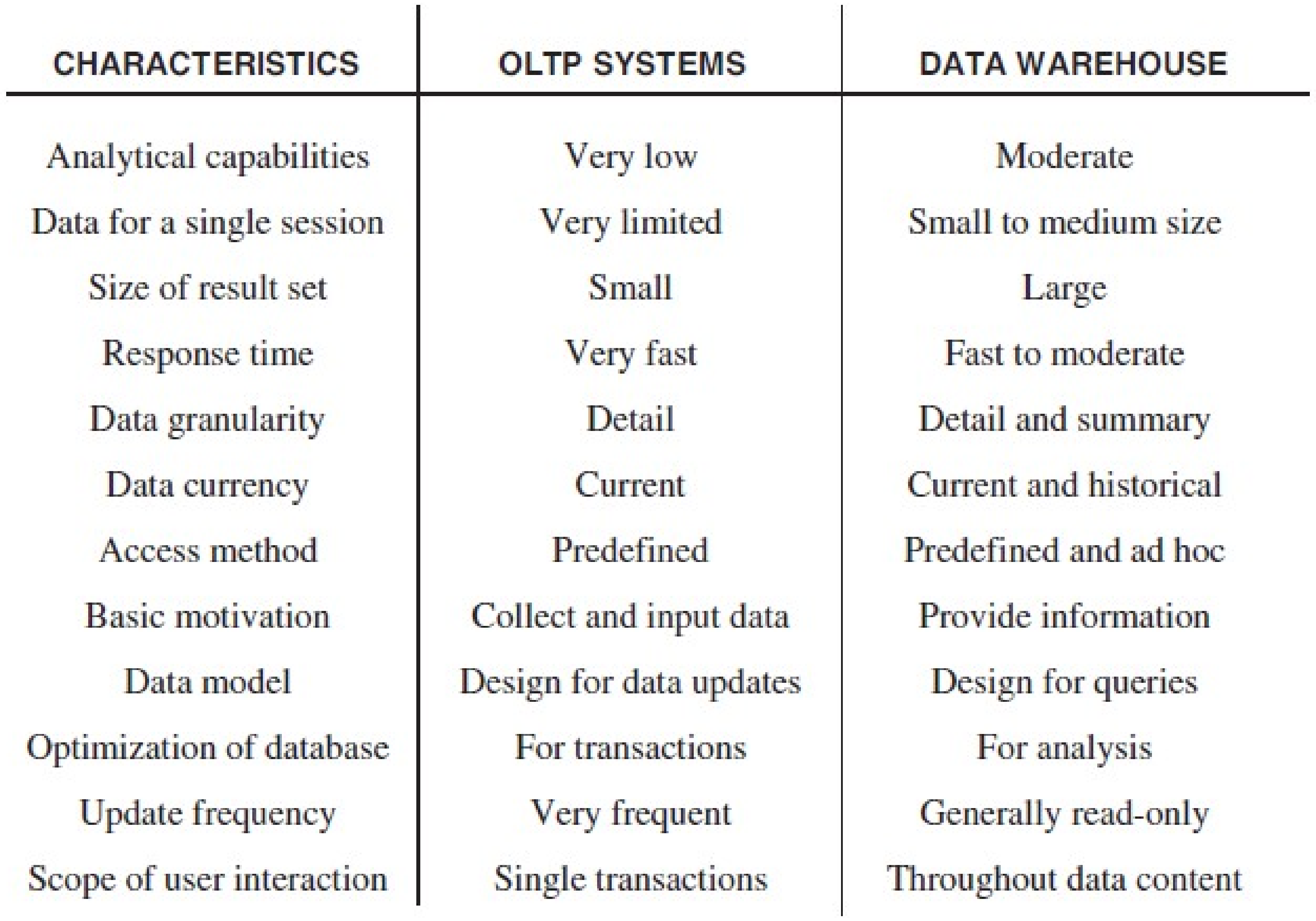


Figure 12-12 Refresh versus update.

- **Applying Data: Techniques and Processes**
  - **Data Refresh Versus Update**
  - If the number of records to be updated falls between 15% and 25% of the total number of records, the cost of loading per record tends to be the same whether you opt for a full refresh of the entire data warehouse or to do the updates.
  - This range is just a general guide. If more than 25% of the source records change daily, then seriously consider full refreshes.
  - **Generally, data warehouse administrators use the update process.**
  - **Occasionally, you may want to redo the data warehouse with a full refresh when some major restructuring or similar mass changes take place.**

- OLTP Vs OLAP



CHARACTERISTICS	OLTP SYSTEMS	DATA WAREHOUSE
Analytical capabilities	Very low	Moderate
Data for a single session	Very limited	Small to medium size
Size of result set	Small	Large
Response time	Very fast	Fast to moderate
Data granularity	Detail	Detail and summary
Data currency	Current	Current and historical
Access method	Predefined	Predefined and ad hoc
Basic motivation	Collect and input data	Provide information
Data model	Design for data updates	Design for queries
Optimization of database	For transactions	For analysis
Update frequency	Very frequent	Generally read-only
Scope of user interaction	Single transactions	Throughout data content

Figure 15-2 OLTP and data warehouse environments.

- **OLAP**
  - Enables analysts, executives, and managers to gain useful insights from the presentation of data.
  - Can reorganize metrics along several dimensions and allow data to be viewed from different perspectives.
  - Supports multidimensional analysis.
  - Is able to drill down or roll up within each dimension.
  - Is capable of applying mathematical formulas and calculations to measures.
  - Provides fast response, facilitating speed-of-thought analysis.
  - Complements the use of other information delivery techniques such as data mining.

- **OLAP**
  - Improves the comprehension of result sets through visual presentations using graphs and charts.
  - Can be implemented on the Web.
  - Designed for highly interactive analysis.

- **OLAP Definition**
  - **On-Line Analytical Processing (OLAP)** is a category of software technology that enables analysts, managers and executives to gain insight into data through fast, consistent, interactive access in a wide variety of possible views of information that has been transformed from raw data to reflect the real dimensionality of the enterprise as understood by the user.

- First, let us consider the initial **12 guidelines for an OLAP system:**
  - **Multidimensional Conceptual View.** Provide a multidimensional data model that is intuitively analytical and easy to use. Business users' view of an enterprise is multidimensional in nature. Therefore, a multidimensional data model conforms to how the users perceive business problems.
  - **Transparency.** Make the technology, underlying data repository, computing architecture, and the diverse nature of source data totally transparent to users. Such transparency, supporting a true open system approach, helps to enhance the efficiency and productivity of the users through front-end tools that are familiar to them.
  - **Accessibility.** Provide access only to the data that is actually needed to perform the specific analysis, presenting a single, coherent, and consistent view to the users.

- First, let us consider the initial **12 guidelines for an OLAP system:**
  - **Consistent Reporting Performance**. Ensure that the users do not experience any significant degradation in reporting performance as the number of dimensions or the size of the database increases. Users must perceive consistent run time, response time, or machine utilization every time a given query is run.
  - **Client/Server Architecture.** Conform the system to the principles of client/server architecture for optimum performance, flexibility, adaptability, and interoperability. Make the server component sufficiently intelligent to enable various clients to be attached with a minimum of effort and integration programming.
  - **Generic Dimensionality.** Ensure that every data dimension is equivalent in both structure and operational capabilities. Have one logical structure for all dimensions.

- First, let us consider the initial **12 guidelines for an OLAP system:**
  - **Dynamic Sparse Matrix Handling**. Adapt the physical schema to the specific analytical model being created and loaded that optimizes sparse matrix handling. When encountering a sparse matrix, the system must be able to dynamically deduce the distribution of the data and adjust the storage and access to achieve and maintain consistent level of performance.
  - **Multiuser Support**. Provide support for end users to work concurrently with either the same analytical model or to create different models from the same data. In short, provide concurrent data access, data integrity, and access security.

- First, let us consider the initial **12 guidelines for an OLAP system:**
  - **Unrestricted Cross-Dimensional Operations**. Provide ability for the system to recognize dimensional hierarchies and automatically perform roll-up and drill-down operations within a dimension or across dimensions. Have the interface language allow calculations and data manipulations across any number of data dimensions, without restricting any relations between data cells, regardless of the number of common data attributes each cell contains.
  - **Intuitive Data Manipulation.** Enable consolidation path reorientation (pivoting), drill down and roll up, and other manipulations to be accomplished intuitively and directly via point-and-click and drag-and-drop actions on the cells of the analytical model. Avoid the use of a menu or multiple trips to a user interface.

- First, let us consider the initial **12 guidelines for an OLAP system:**
  - **Flexible Reporting**. Provide capabilities to the business user to arrange columns, rows,and cells in a manner that facilitates easy manipulation, analysis, and synthesis of information. Every dimension, including any subsets, must be able to be displayed with equal ease.
  - **Unlimited Dimensions and Aggregation Levels.** Accommodate at least 15, preferably 20, data dimensions within a common analytical model. Each of these generic dimensions must allow a practically unlimited number of user-defined aggregation levels within any given consolidation path.

- **OLAP Characteristics**
  - Let business users have a multidimensional and logical view of the data in the data warehouse.
  - Facilitate interactive query and complex analysis for the users.
  - Allow users to drill down for greater details or roll up for aggregations of metrics along a single business dimension or across multiple dimensions.
  - Provide the ability to perform intricate calculations and comparisons.
  - Present results in a number of meaningful ways, including charts and graphs.

- **Dimensional Analysis**
  - Let us begin with a simple STAR schema.
  - This STAR schema has three business dimensions, namely, product, time, and store. The fact table contains sales.
  - Figure 15-5 shows the STAR schema and a three-dimensional representation of the model as a cube, with products on the X-axis, time on the Y-axis, and stores on the Z-axis.
  - What are the values represented along each axis? For example, in the STAR schema, time is one of the dimensions and month is one of the attributes of the time dimension.
  - Values of this attribute month are represented on the Y-axis.
  - Similarly, values of the attributes product name and store name are represented on the other two axes.

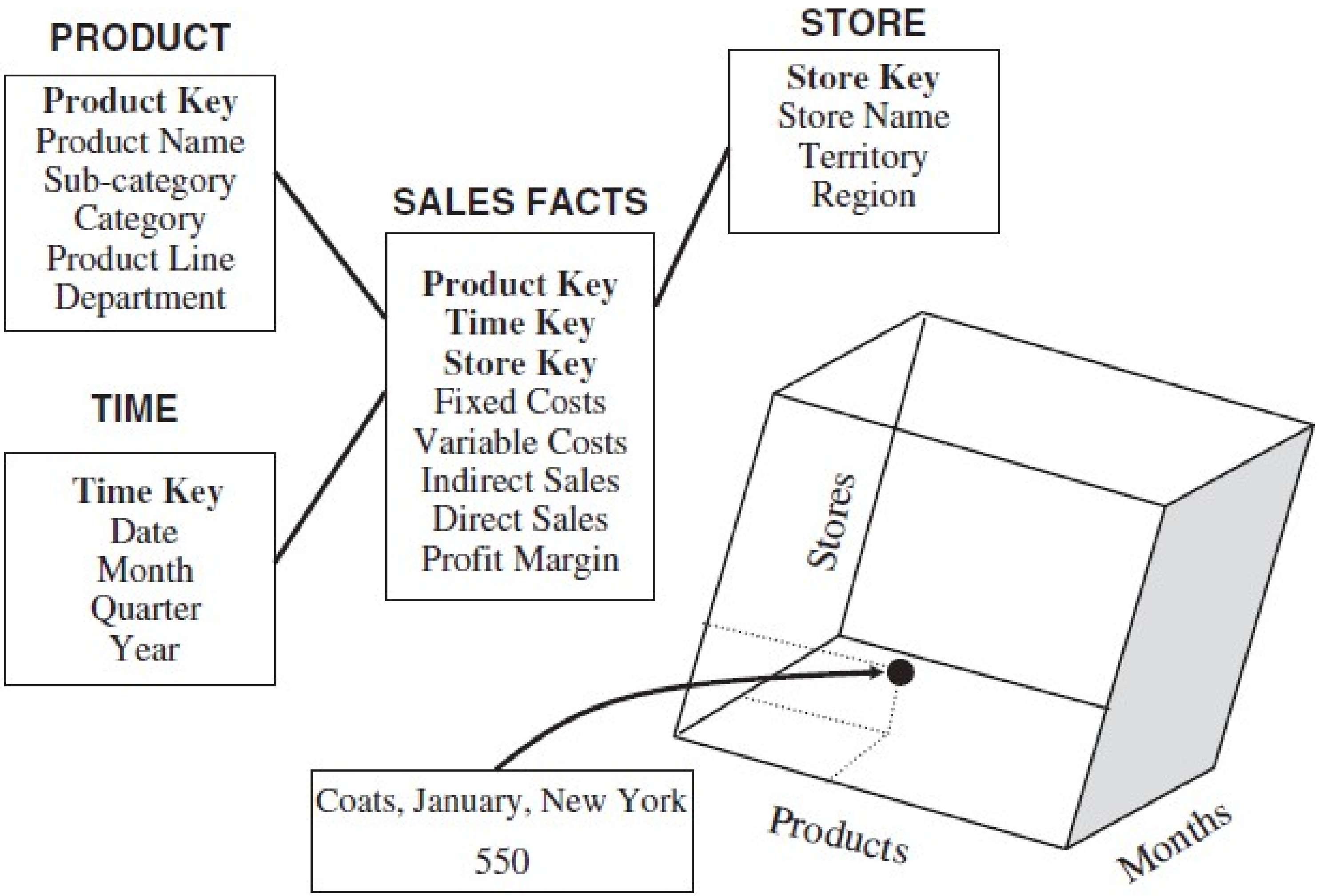


Figure 15-5 Simple STAR schema.

- **Dimensional Analysis**
  - If you are displaying the data for sales along these three dimensions on a spreadsheet, the columns may display the product names, the rows the months, and pages the data along the third dimension of store names.
  - See Figure 15-6, which shows a screen display of a page of this three-dimensional data.
  - The page displayed on the screen shows a slice of the cube. Now look at the cube and move along a slice or plane passing through the point on the Z-axis representing store: New York.
  - The intersection points on this slice or plane relate to sales along product and time business dimensions for store: New York.
  - Try to relate these sale numbers to the slice on the cube representing store: New York.

Store: New York

Products

PAGES: STORE dimension

COLUMNS: PRODUCT dimension

ROWS: TIME dimension

Months

	Hats	Coats	Jackets	Dresses	Shirts	Slacks
Jan	200	550	350	500	520	490
Feb	210	480	390	510	530	500
Mar	190	480	380	480	500	470
Apr	190	430	350	490	510	480
May	160	530	320	530	550	520
Jun	150	450	310	540	560	330
Jul	130	480	270	550	570	250
Aug	140	570	250	650	670	230
Sep	160	470	240	630	650	210
Oct	170	480	260	610	630	250
Nov	180	520	280	680	700	260
Dec	200	560	320	750	770	310

Figure 15-6 A three-dimensional display.

- **Dimensional Analysis**
  - Now we have a way of depicting three business dimensions and a single fact on a two dimensional page and also on a three-dimensional cube.
  - The numbers in each cell on the page are the sale numbers.
  - What could be the types of multidimensional analysis on this particular set of data? What types of queries could be run during the course of analysis sessions? You could get sale numbers along the hierarchies of a combination of the three business dimensions of product, store, and time.
  - You could perform various types of three-dimensional analysis of sales.
  - The results of queries during analysis sessions will be displayed on the screen with the three dimensions represented in columns, rows, and pages. The following is a sample of simple queries and the result sets during a multidimensional analysis session.

- **Dimensional Analysis**
  - **Query**
  - Display the total sales of all products for past five years in all stores.
  - Display of Results
  - Rows: Year numbers 2009, 2008, 2007, 2006, 2005
  - Columns: Total Sales for all products
  - Page: One store per page analysis session.
  - **Query**
  - Compare total sales for all stores, product by product, between years 2009 and 2008. **Display of Results**
  - Rows: Year numbers 2009, 2008; difference; percentage increase or decrease
  - Columns: One column per product, showing all products
  - Page: All stores

- **What Are Hypercubes?**
  - The data described here may be displayed on a spreadsheet showing metrics as columns, time as rows, and products as pages.
  - Figure 15-7 shows a sample page of the spreadsheet display. In the figure, note the three straight lines, two of which represent the two business dimensions and the third, the metrics.
  - You can independently move up or down along the straight lines.
  - Some experts refer to this representation as a multidimensional domain structure (MDS).

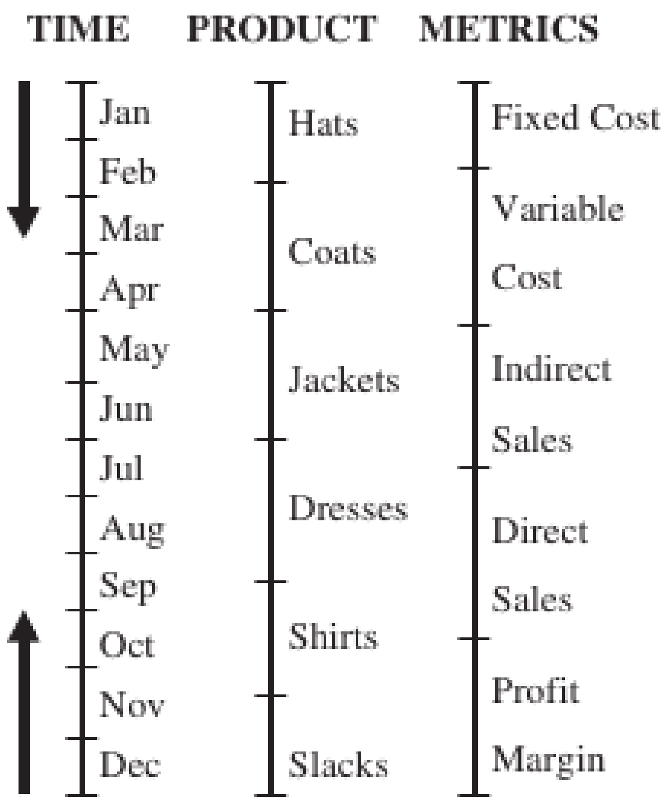
PRODUCT: Coats

PAGES: PRODUCT dimension COLUMNS: Metrics

ROWS: TIME dimension

	Fixed Cost	Variable Cost	Indirect Sales	Direct Sales	Profit Margin
Jan	340	110	230	320	100
Feb	270	90	200	260	100
Mar	310	100	210	270	70
Apr	340	110	210	320	80
May	330	110	230	300	90
Jun	260	90	150	300	100
Jul	310	100	180	300	70
Aug	380	130	210	360	60
Sep	300	100	180	290	70
Oct	310	100	170	310	70
Nov	330	110	210	310	80
Dec	350	120	200	360	90

Multidimensional Domain Structure



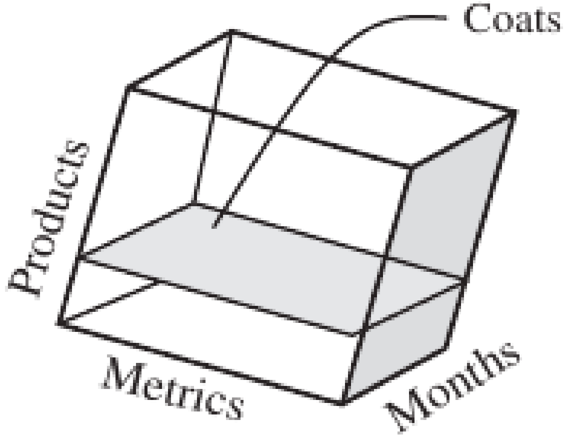


Figure 15-7 Display of columns, rows, and pages.

- **What Are Hypercubes?**
  - The figure also shows **a cube** representing the data points along the edges.
  - Relate the three straight lines to the three edges of the physical cube.
  - Now the page you see in the figure is a slice passing through a single product and the divisions along the other two straight lines shown on the page as columns and rows.
  - With three groups of data—two groups of business dimensions and one group of metrics—we can easily visualize the data as being along the three edges of a cube.

- **What Are Hypercubes?**
  - This is where an MDS diagram comes in handy.
  - Now you need not try to perceive four-dimensional data as along the edges of the three-dimensional cube.
  - All you have to do is draw four straight lines to represent the data as an MDS.
  - These four lines represent the data (see Fig. 15-8).
  - Looking at this figure, you realize that the metaphor of a physical cube to represent data breaks down when you try to represent four dimensions.

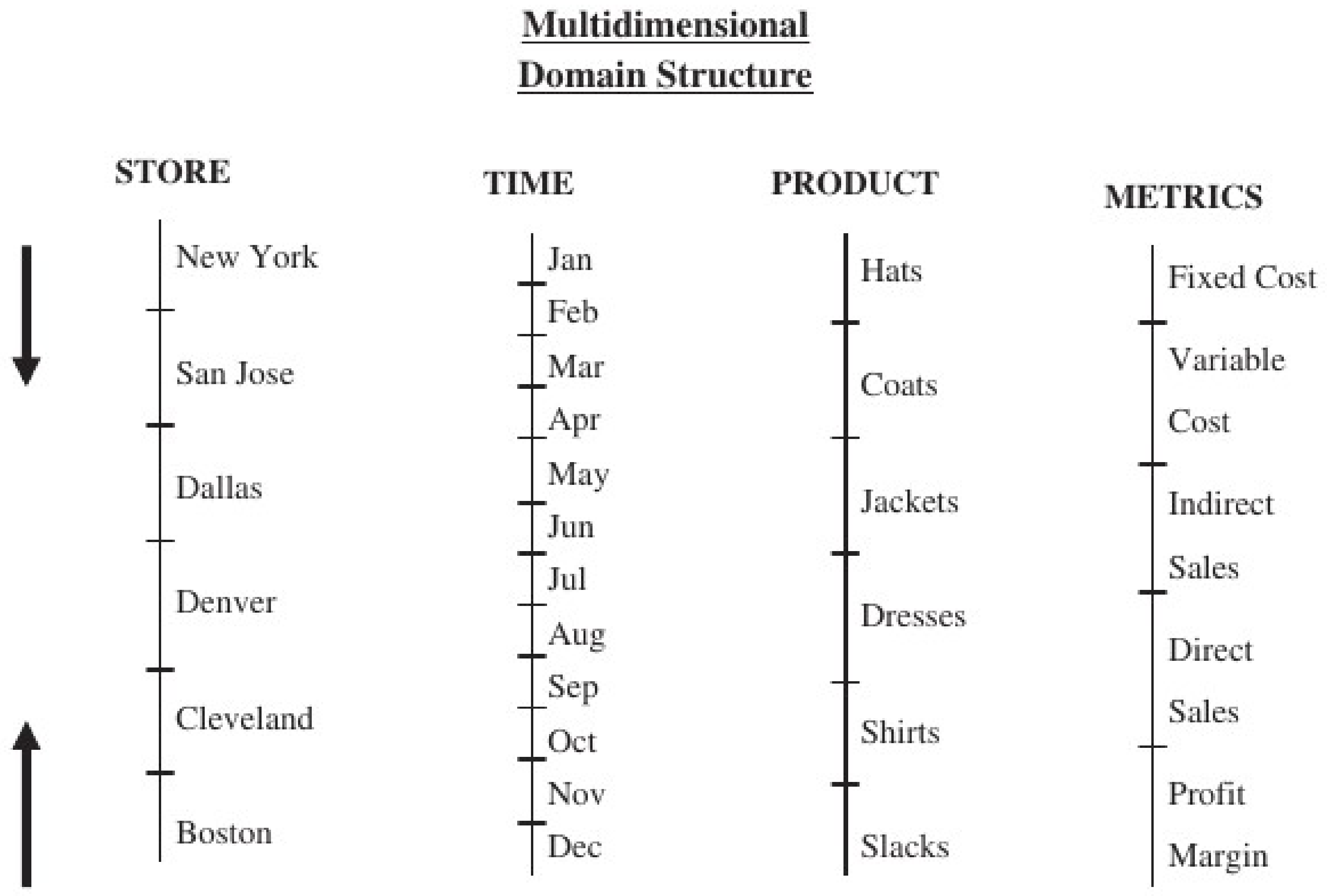
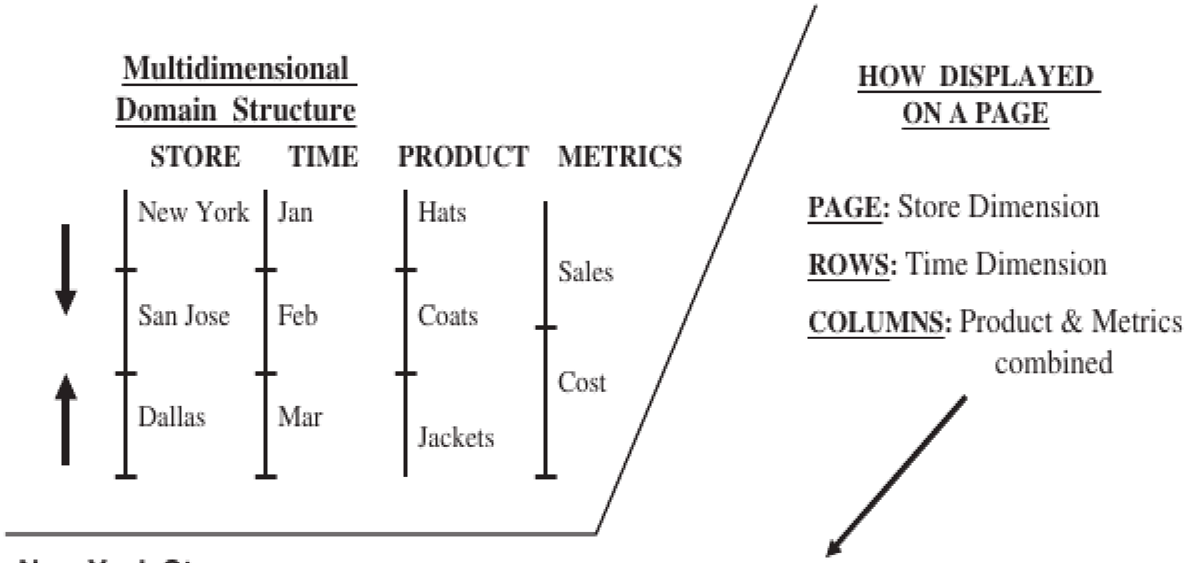


Figure 15-8 MDS for four dimensions.

- **What Are Hypercubes?**
  - The MDS is well suited to represent four dimensions.
  - **Can you think of the four straight lines of the MDS intuitively to represent a “cube” with four primary edges?**
  - **This intuitive representation is a hypercube, a representation that accommodates more than three dimensions.**
  - At a lower level of simplification, a hypercube can very well accommodate three dimensions.
  - A hypercube is a general metaphor for representing multidimensional data.

- **What Are Hypercubes?**
  - The MDS is well suited to represent four dimensions.
  - **Can you think of the four straight lines of the MDS intuitively to represent a "cube" with four primary edges?**
  - **This intuitive representation is a hypercube, a representation that accommodates more than three dimensions.**
  - At a lower level of simplification, a hypercube can very well accommodate three dimensions.
  - A hypercube is a general metaphor for representing multidimensional data.



New York Store

	Hats:Sales	Hats:Cost	Coats:Sales	Costs:Cost	Jackets:Sales	Jackets:Cost
Jan	450	350	550	450	500	400
Feb	380	280	460	360	400	320
Mar	400	310	480	410	450	400

Figure 15-9 Page displays for four-dimensional data.

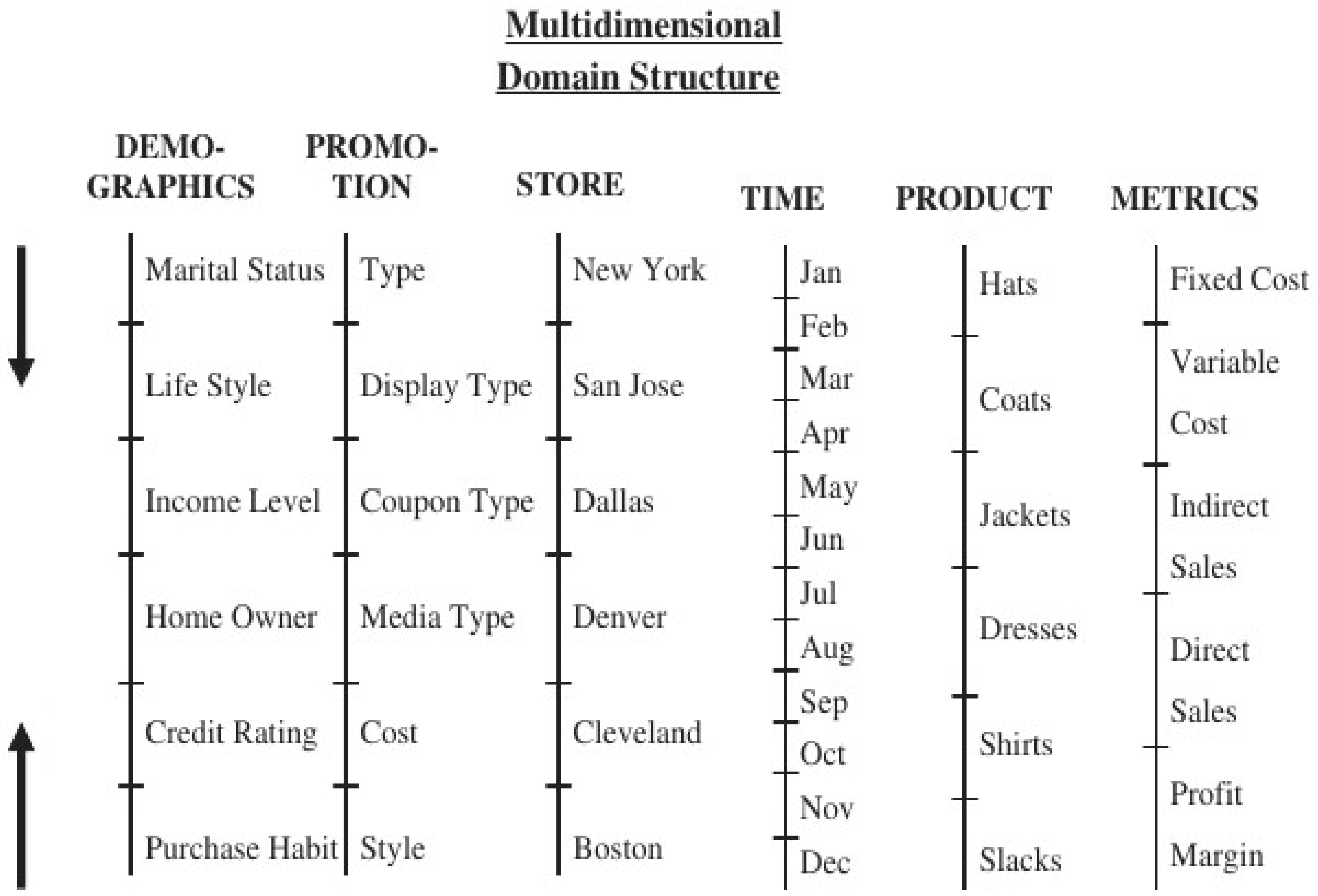
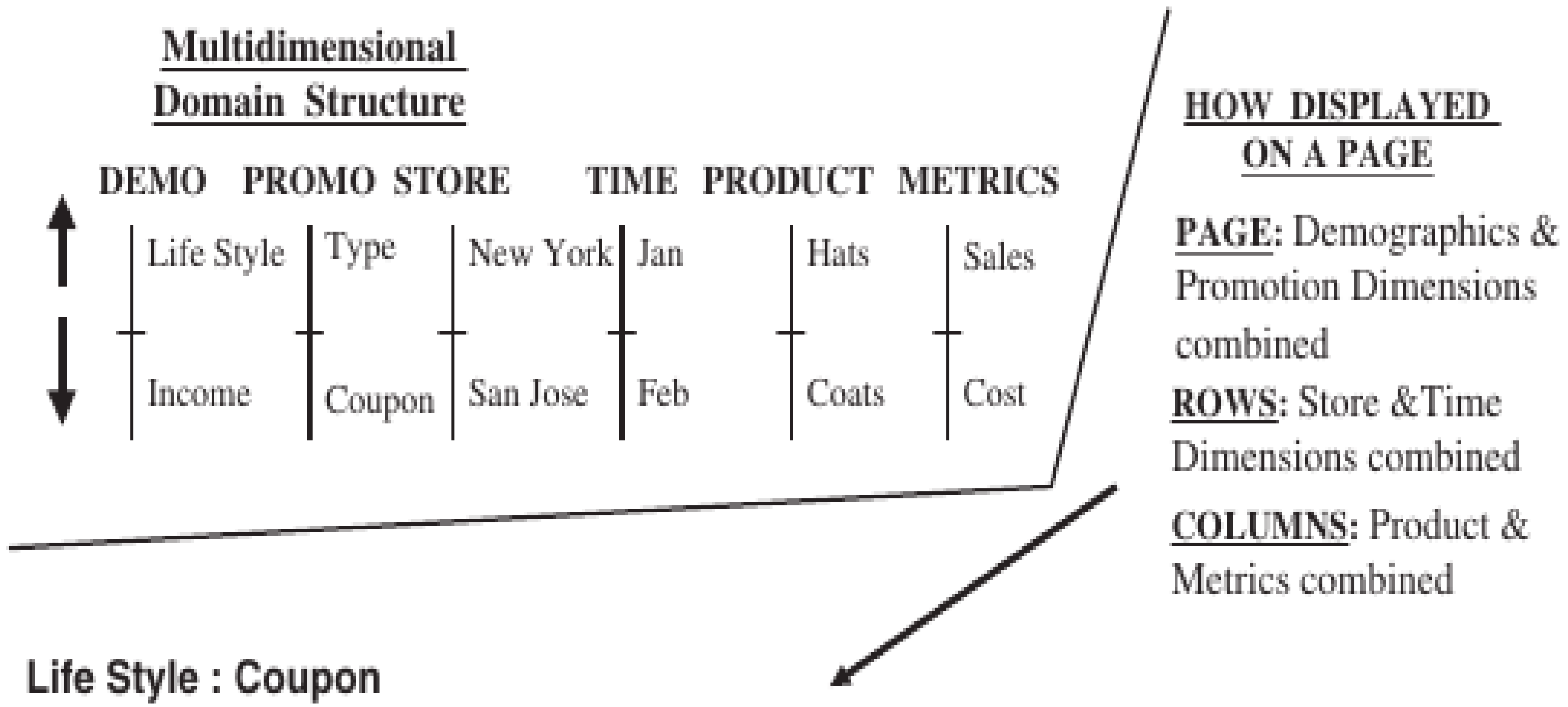


Figure 15-10 Six-dimensional MDS.


Multidimensional Domain Structure
DEMO
PROMO
STORE
TIME
PRODUCT
METRICS
Life Style
Income
Type
Coupon
New York
San Jose
Jan
Feb
Hats
Coats
Sales
Cost
HOW DISPLAYED ON A PAGE
PAGE: Demographics & Promotion Dimensions combined
ROWS: Store &Time Dimensions combined
COLUMNS: Product & Metrics combined

Life Style : Coupon

		Hats	Hats	Coats	Coats
		Sales	Cost	Sales	Cost
New York	Jan	220	170	270	220
	Feb	190	140	230	180
Boston	Jan	200	160	240	200
	Feb	180	130	220	170

Figure 15-11 Page displays for six-dimensional data.

- **OLAP Operations:**
  - **Drill Down and Roll Up**
  - OLAP systems provide drill-down and roll-up capabilities.
  - Try to understand what we mean by these capabilities with reference to the above example.
  - Figure 15-12 illustrates these capabilities with reference to the product dimension hierarchies.
  - Note the different types of information given in the figure.
  - It shows the rolling up to higher hierarchical levels of aggregation and the drilling down to lower levels of detail.
  - You drill down to get the lower level breakdown of sales.
  - The figure also shows the drill across to another OLAP summarization using a different set of hierarchies of other dimensions.

- **OLAP Operations:**
  - **Drill Down and Roll Up**
  - Notice also the drill through to the lower levels of granularity, as stored in the source data warehouse repository.
  - Roll up, drill down, drill across, and drill through are extremely useful features of OLAP systems supporting multidimensional analysis.

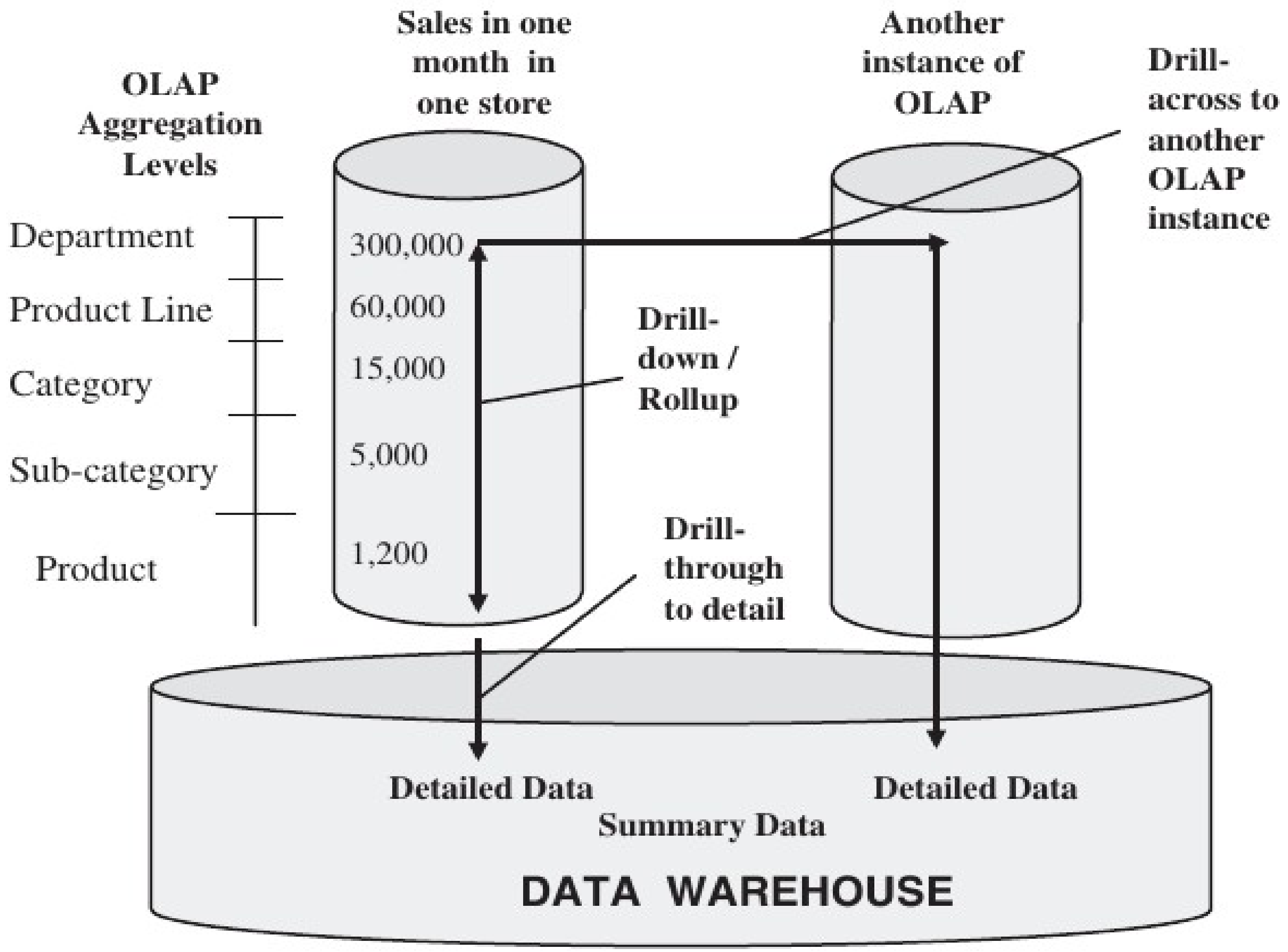
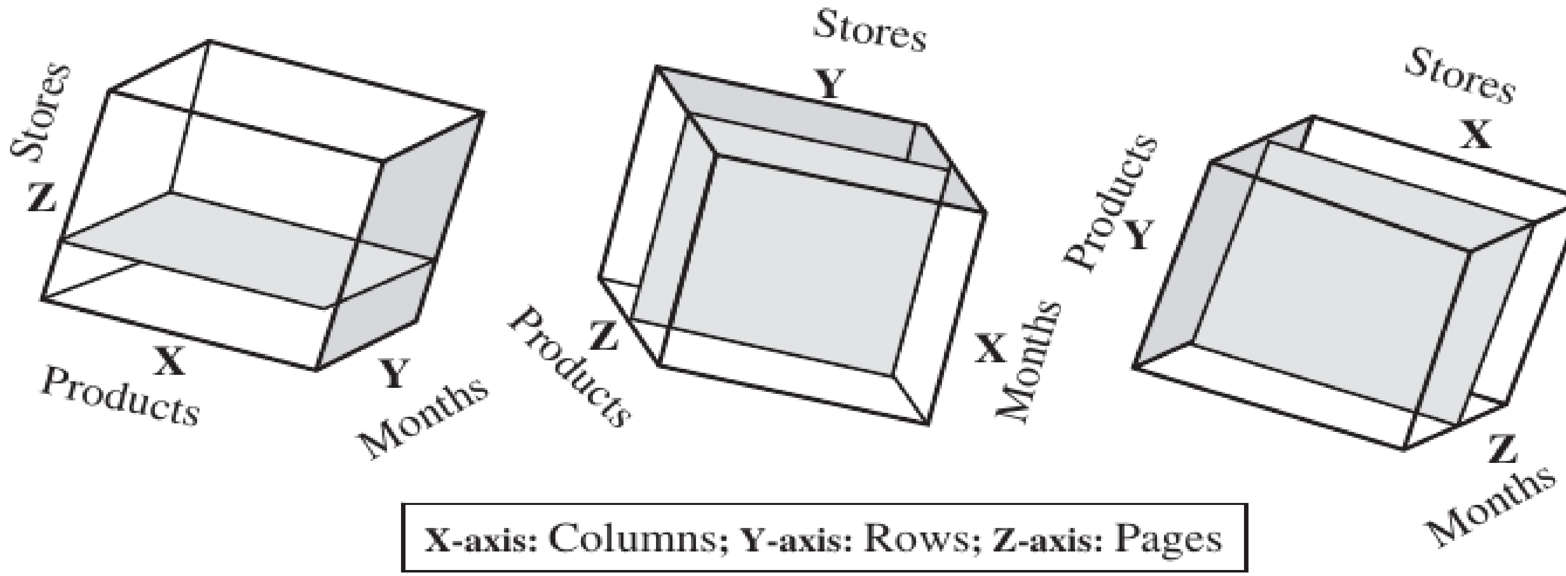


Figure 15-12 Roll-up and drill-down features of OLAP.

- **OLAP Operations:**
  - **Slice and Dice or Rotation**
  - The data model corresponds to a physical cube with these data elements represented by its primary edges.
  - The page displayed is a slice or two-dimensional plane of the cube. In particular, this display page for the New York store is the slice parallel to the product and time axes.
  - Now look at Figure 15-14 carefully.
  - On the left side, the first part of the diagram shows this alignment of the cube.
  - For the sake of simplicity, only three products, three months, and three stores are chosen for illustration.



Store: New York

	Hats	Coats	Jackets
Jan	200	550	350
Feb	210	480	390
Mar	190	480	380

Product: Hats

	Jan	Feb	Mar
New York	200	210	190
Boston	210	250	240
San Jose	130	90	70

Month: January

	New York	Boston	San Jose
Hats	200	210	130
Coats	550	500	200
Jackets	350	400	100

Figure 15-14 Slicing and dicing.

- **OLAP Operations:**
  - **Slice and Dice or Rotation**
  - Now rotate the cube so that products are along the Z-axis, months are along the X-axis, and stores are along the Y-axis.
  - The slice we are considering also rotates.
  - What happens to the display page that represents the slice? Months are now shown as columns and stores as rows.
  - The display page represents the sales of one product, namely product: hats.

- **OLAP Operations:**
  - **Slice and Dice or Rotation**
  - You can go to the next rotation so that months are along the Z-axis, stores are along the X-axis, and products are along the Y-axis.
  - The slice we are considering also rotates. What happens to the display page that represents the slice? Stores are now shown as columns and products as rows.
  - The display page represents the sales of one month, namely month: January.
  - **Advantages:**
  - The users can view the data from many angles, understand the numbers better, and arrive at meaningful conclusions.

- OLAP Operations:
  - **Uses and Benefits**
  - Increased productivity of business managers, executives, and analysts.
  - Inherent flexibility of OLAP systems means that users may be self-sufficient in running their own analysis without IT assistance.
  - Benefit for IT developers because using software specifically designed for the system development results in faster delivery of applications.
  - Self-sufficiency for users, resulting in reduction in backlog.
  - Faster delivery of applications following from the previous benefits.
  - More efficient operations through reducing time on query executions and in network traffic.
  - Ability to model real-world challenges with business metrics and dimensions.

- **OLAP models: MOLAP, ROLAP**
  - Have you heard of the terms ROLAP or MOLAP? How about the variation, DOLAP?
  - A very simple explanation of the variations relates to the way data is stored for OLAP.
  - The processing is still online analytical processing; basically, the storage methodology is different.
  - **ROLAP.** Refers to relational online analytical processing.
  - In this case, the OLAP system is built on top of a relational database.
  - **MOLAP.** Refers to multidimensional online analytical processing.
  - In this case, the OLAP system is implemented through a specialized multidimensional database.

- OLAP models: MOLAP, ROLAP
  - **MOLAP and ROLAP** are the fundamental models.
  - **In the MOLAP model**, online analytical processing is best implemented by storing the data multidimensionally, that is, easily viewed in a multidimensional way.
  - Here the data structure is fixed so that the logic to process multidimensional analysis can be based on well-defined methods of establishing data storage coordinates.
  - Usually, multidimensional databases (MDDBs) are vendors' proprietary systems.
  - **On the other hand, the ROLAP model** relies on the existing relational DBMS of the data warehouse.
  - OLAP features are provided against the relational database.

- **OLAP models: MOLAP, ROLAP**
  - Notice the MOLAP model shown on the left side of the figure. The OLAP engine resides on a special server.
  - Proprietary multidimensional databases (MDDBs) store data in the form of multidimensional hypercubes.
  - You have to run special extraction and aggregation jobs from the relational database of the data warehouse to create these multidimensional data cubes in the MDDBs.
  - The special server presents the data as OLAP cubes for processing by the users.

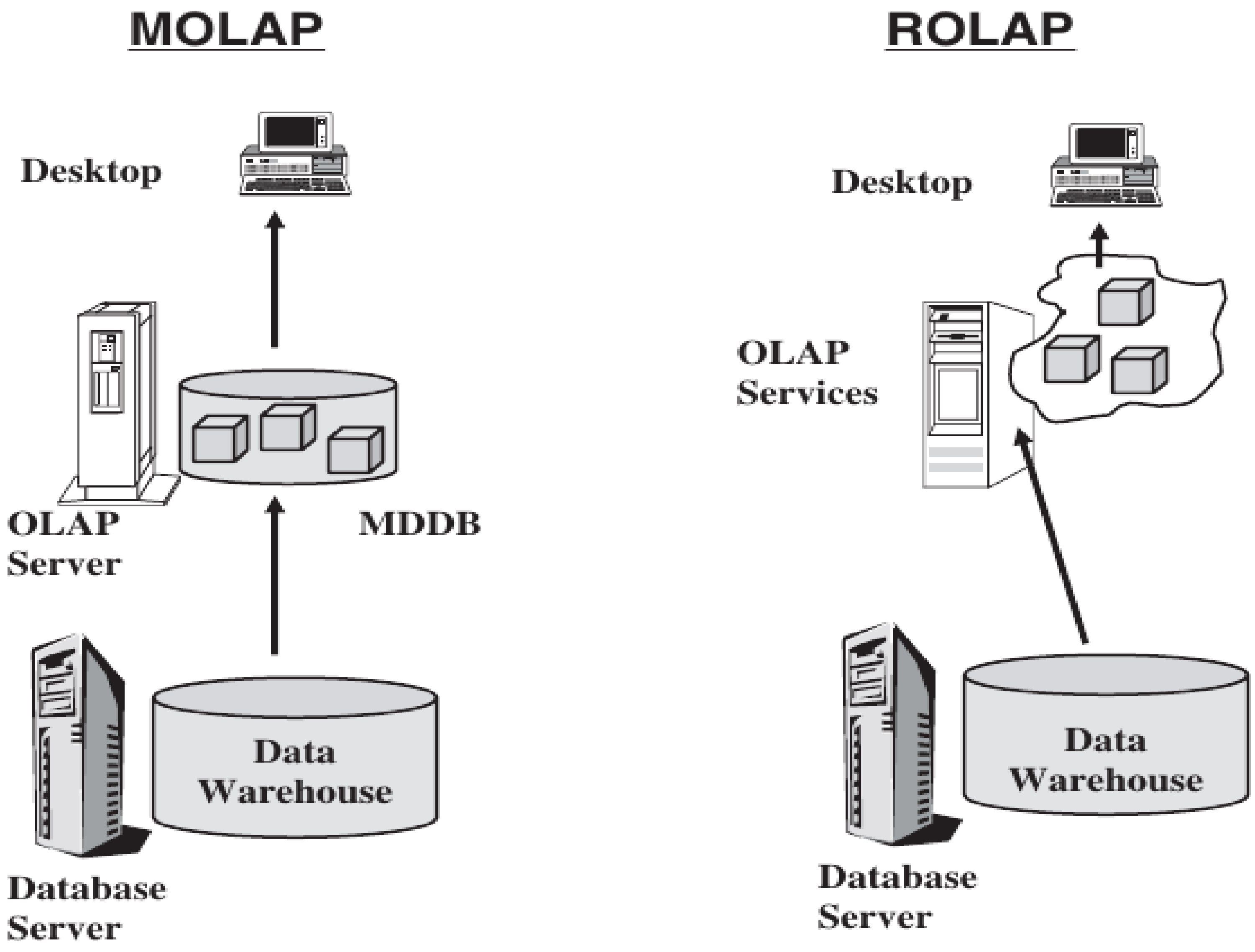


Figure 15-15 OLAP models.

- **The MOLAP Model**
  - As discussed, in the MOLAP model, data for analysis is stored in specialized multidimensional databases.
  - Large multidimensional arrays form the storage structures.
  - For example,to store sales number of 500 units for product ProductA, in month number 2009/01, in store StoreS1, under distributing channel Channe105, the sales number of 500 is stored in an array represented by the values (ProductA, 2009/01, StoreS1. Channe105).
  - The array values indicate the location of the cells.
  - These cells are intersections of the values of dimension attributes.
  - If you note how the cells are formed, you will realize that not all cells have values of metrics.
  - If a store is closed on Sundays, then the cells representing Sundays will all be nulls.

- **The MOLAP Model**
    - Let us now consider the **architecture fo**r the MOLAP model.
    - Please go over each part of Figure 15-16 carefully.
    - Note the three layers in the multitier architecture.
    - Pre-calculated and prefabricated multidimensional data cubes are stored in multidimensional databases.
    - The MOLAP engine in the application layer pushes a multidimensional view of the data from the MDDBs to the users.

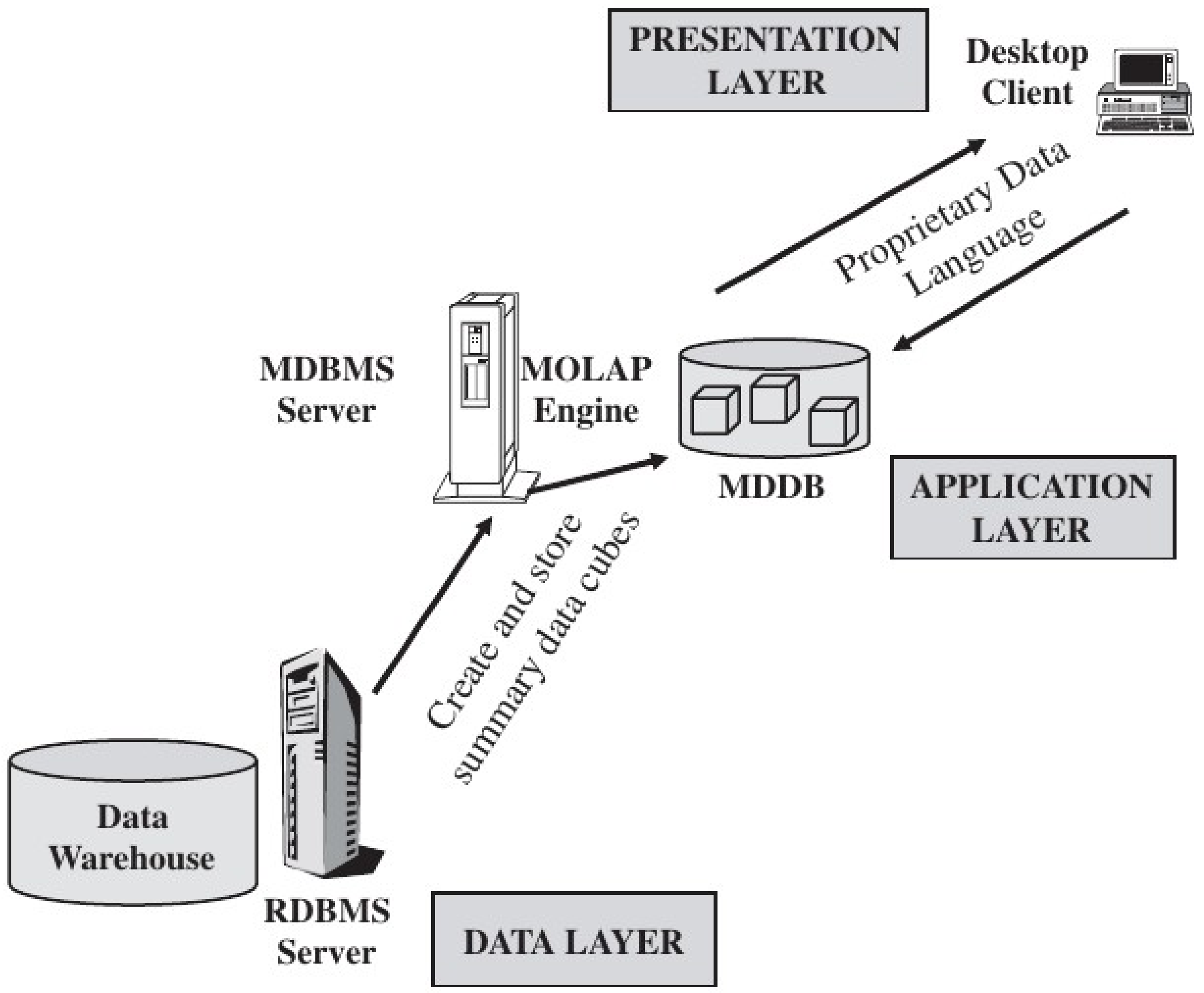


Figure 15-16 The MOLAP model.

- **The MOLAP Model**
  - As mentioned earlier, multidimensional database management systems are proprietary software systems.
  - These systems provide the capability to consolidate and fabricate summarized cubes during the process that loads data into the MDDBs from the main data warehouse.
  - The users who need summarized data enjoy fast response times from the preconsolidated data.

- **The ROLAP Model**
  - In the ROLAP model, data is stored as rows and columns as in a relational data model.
  - This model presents data to the users in the form of business dimensions.
  - In order to hide the storage structure to the user and present data multidimensionally, a semantic layer of meta-data is created.
  - The metadata layer supports the mapping of dimensions to the relational tables.
  - Additional metadata supports summarizations and aggregations.
  - You may store the meta-data in relational databases.

- **The ROLAP Model**
  - Now see Figure 15-17. T**his figure shows the architecture of** the ROLAP model.
  - What you see is a **three-tier architecture.**
  - The analytical server in the middle tier application layer creates multidimensional views on the fly.
  - The multidimensional system at the presentation layer provides a multidimensional view of the data to the users.
  - When the users issue complex queries based on this multidimensional view, the queries are transformed into complex SQL directed to the relational database.
  - Unlike the MOLAP model, static multidimensional structures are not created and stored.

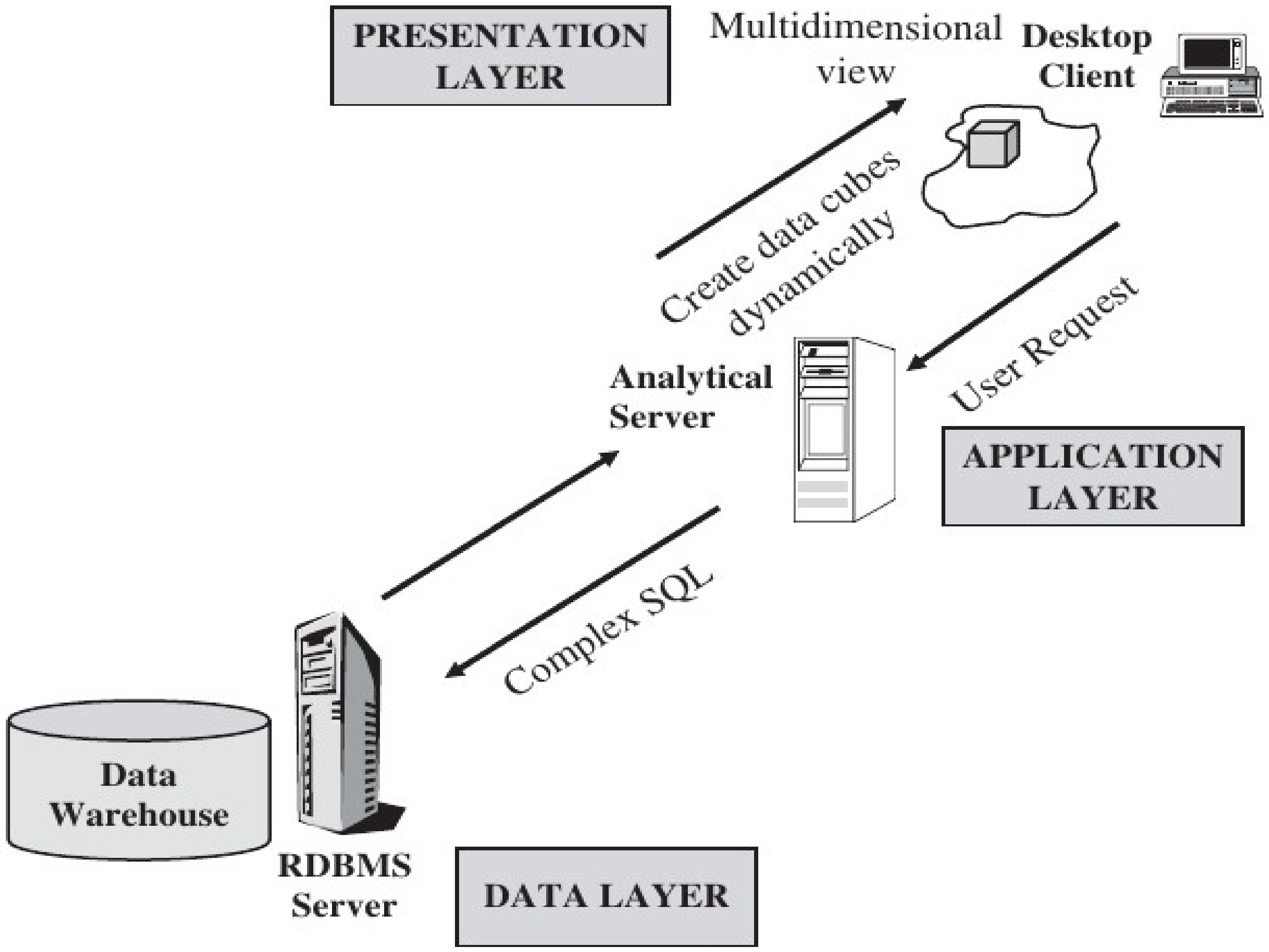
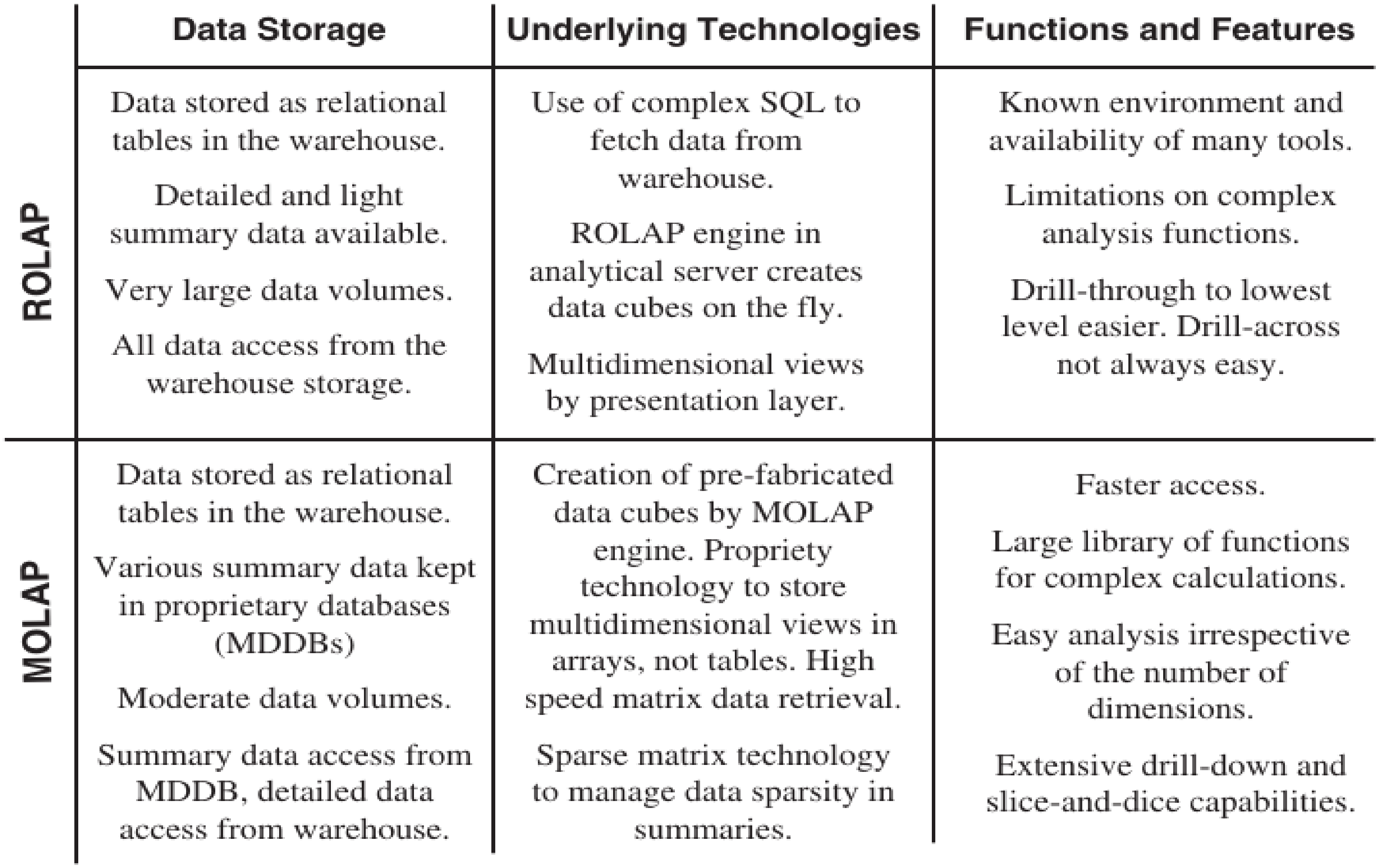


Figure 15-17 The ROLAP model.

- **The ROLAP Model**
  - True ROLAP has three distinct characteristics:
    - Supports all the basic OLAP features and functions discussed earlier
    - Stores data in a relational form
    - Supports some form of aggregation

- ROLAP Versus MOLAP



	Data Storage	Underlying Technologies	Functions and Features
ROLAP	Data stored as relational tables in the warehouse. Detailed and light summary data available. Very large data volumes. All data access from the warehouse storage.	Use of complex SQL to fetch data from warehouse. ROLAP engine in analytical server creates data cubes on the fly. Multidimensional views by presentation layer.	Known environment and availability of many tools. Limitations on complex analysis functions. Drill-through to lowest level easier. Drill-across not always easy.
MOLAP	Data stored as relational tables in the warehouse. Various summary data kept in proprietary databases (MDDBs) Moderate data volumes. Summary data access from MDDB, detailed data access from warehouse.	Creation of pre-fabricated data cubes by MOLAP engine. Propriety technology to store multidimensional views in arrays, not tables. High speed matrix data retrieval. Sparse matrix technology to manage data sparsity in summaries.	Faster access. Large library of functions for complex calculations. Easy analysis irrespective of the number of dimensions. Extensive drill-down and slice-and-dice capabilities.

Figure 15-19 ROLAP versus MOLAP.