Data Warehousing

Unit 8

MOTIVATION

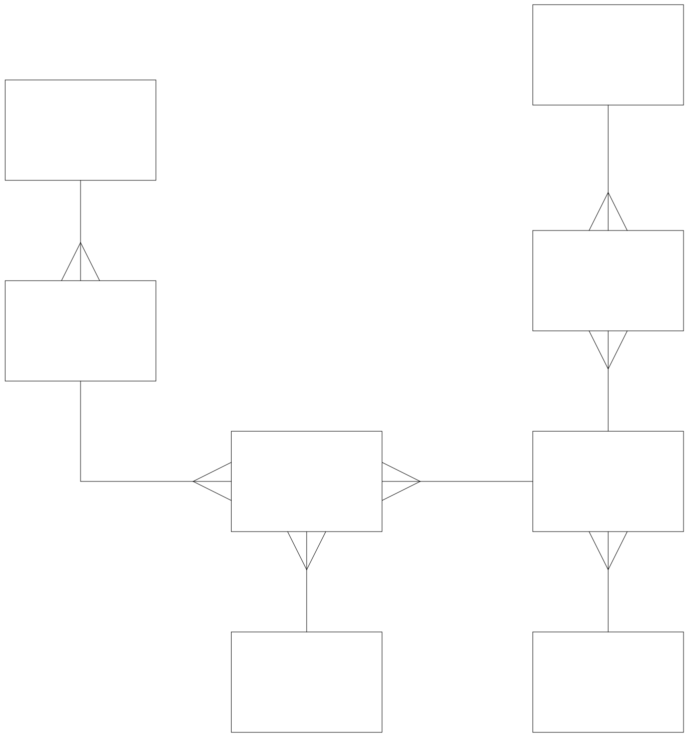
- "Modern organization is drowning in data but starving for information".
- *Operational processing* (transaction processing) captures, stores and manipulates data to support daily operations.
- *Information processing* is the analysis of data or other forms of information to support decision making.
- *Data warehouse* can consolidate and integrate information from many internal and external sources and arrange it in a meaningful format for making business decisions.

Comparison Chart of Database Types

Data warehouse	Operational system
Subject oriented	Transaction oriented
Large (hundreds of GB up to several TB)	Small (MB up to several GB)
Historic data	Current data
De-normalized table structure (few tables, many columns per table)	Normalized table structure (many tables, few columns per table)
Batch updates	Continuous updates
Usually very complex queries	Simple to complex queries
Access type : Read	Access Type: Read, update, delete
Response time in seconds to even minutes	Response time in milliseconds
Small number of users	Large number of users
Heuristic , adhoc, random usage	Predictive usage

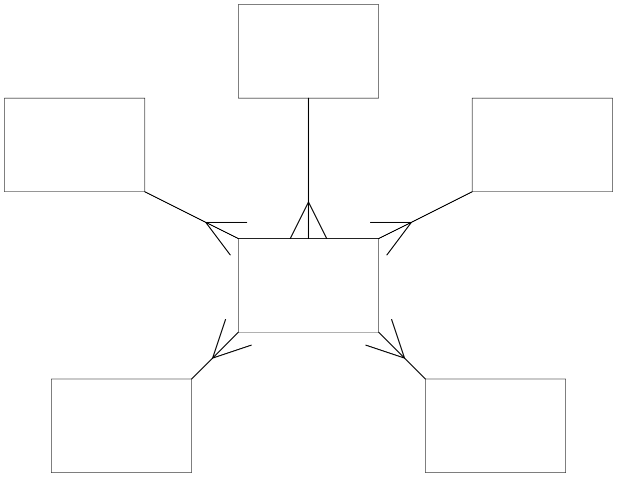
DESIGN DIFFERENCES

Operational System



ER Diagram

Data Warehouse



Star Schema

DECISION SUPPORT AND OLAP

- Information technology to help the knowledge worker (executive, manager, analyst) make faster and better decisions
  - *What were the sales volumes by region and product category for the last year?*
  - *How did the share price of computer manufacturers correlate with quarterly profits over the past 10 years?*
  - *Which orders should we fill to maximize revenues?*
  - *Will a 10% discount increase sales volume sufficiently?*
  - *Which of two new medications will result in the best outcome: higher recovery rate & shorter hospital stay?*
- On-Line Analytical Processing (OLAP) is an element of decision support systems (DSS)

Data Warehouse vs. Operational DBMS

- OLTP (on-line transaction processing)
  - Major task of traditional relational DBMS
  - Day-to-day operations: purchasing, inventory, banking, manufacturing, payroll, registration, accounting, etc.
- OLAP (on-line analytical processing)
  - Major task of data warehouse system
  - Data analysis and decision making
- Distinct features (OLTP vs. OLAP):
  - User and system orientation: customer vs. market
  - Data contents: current, detailed vs. historical, consolidated
  - Database design: ER + application vs. star + subject
  - View: current, local vs. evolutionary, integrated
  - Access patterns: update vs. read-only but complex queries

OLTP VS. OLAP

	OLTP	**OLAP**
users	clerk, IT professional	knowledge worker
function	day to day operations	decision support
DB design	application-oriented	subject-oriented
data	current, up-to-date detailed, flat relational isolated	historical, summarized, multidimensional integrated, consolidated
usage	repetitive	ad-hoc
access	read/write index/hash on prim. key	lots of scans
unit of work	short, simple transaction	complex query
# records accessed	tens	millions
#users	thousands	hundreds
DB size	100MB-GB	100GB-TB
metric	transaction throughput	query throughput, response

What is Data Warehouse?

- Defined in many different ways, but not rigorously.
  - A decision support database that is maintained **separately** from the organization's operational database
  - Support information processing by providing a solid platform of consolidated, historical data for analysis.
- "A data warehouse is a subject-oriented, integrated, time-variant, and nonvolatile collection of data in support of management's decision-making process."—**W. H. Inmon**
- Data warehousing:
  - The process of constructing and using data warehouses

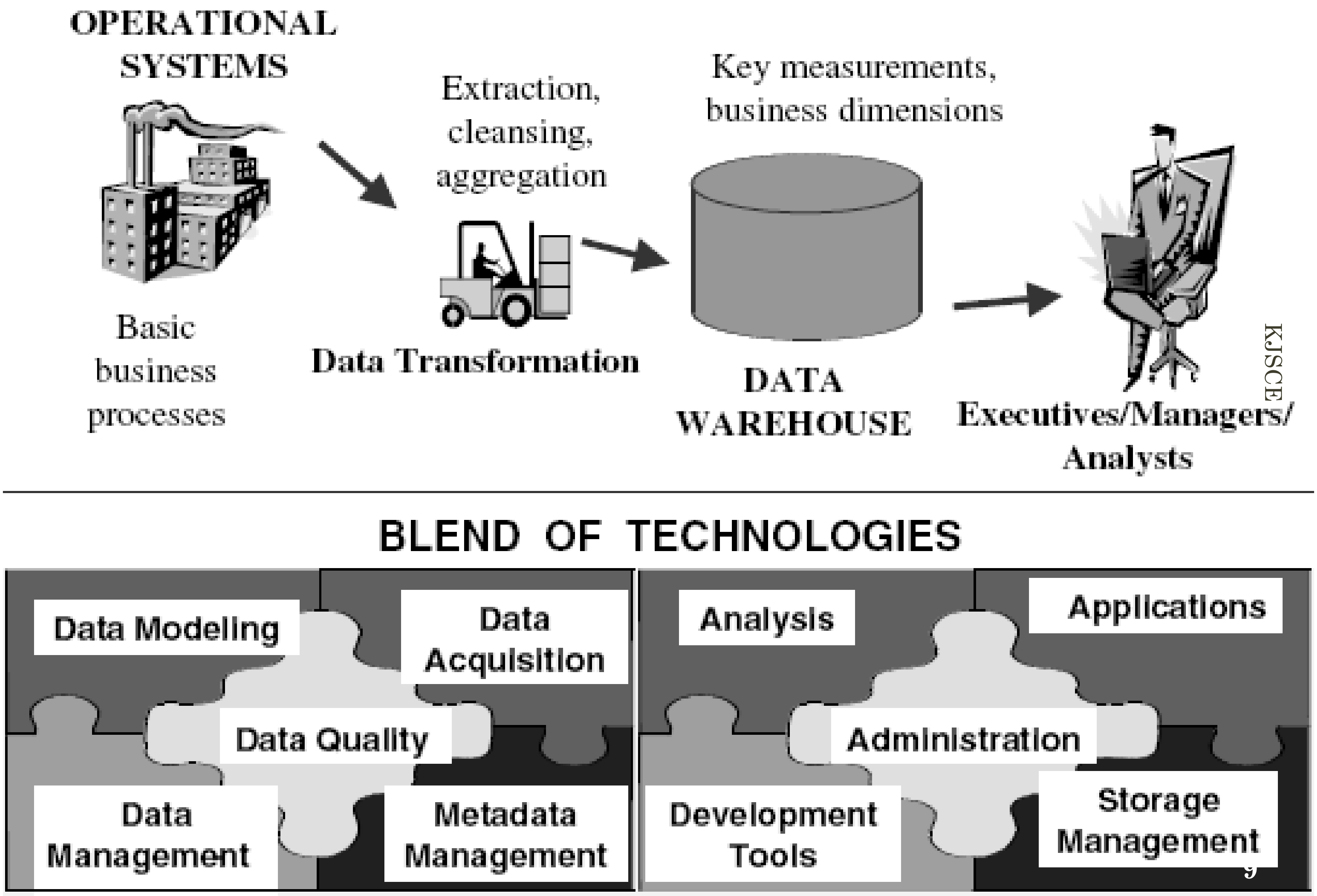


Figure 1-9 The data warehouse: a blend of technologies.

Data Warehouse—Subject-Oriented

- Organized around major subjects, such as customer, product, sales
- Focusing on the modeling and analysis of data for decision makers, not on daily operations or transaction processing
- Provide a simple and concise view around particular subject issues by excluding data that are not useful in the decision support process

Subject-oriented

In the data warehouse, data is not stored by operational applications, but by business subjects.

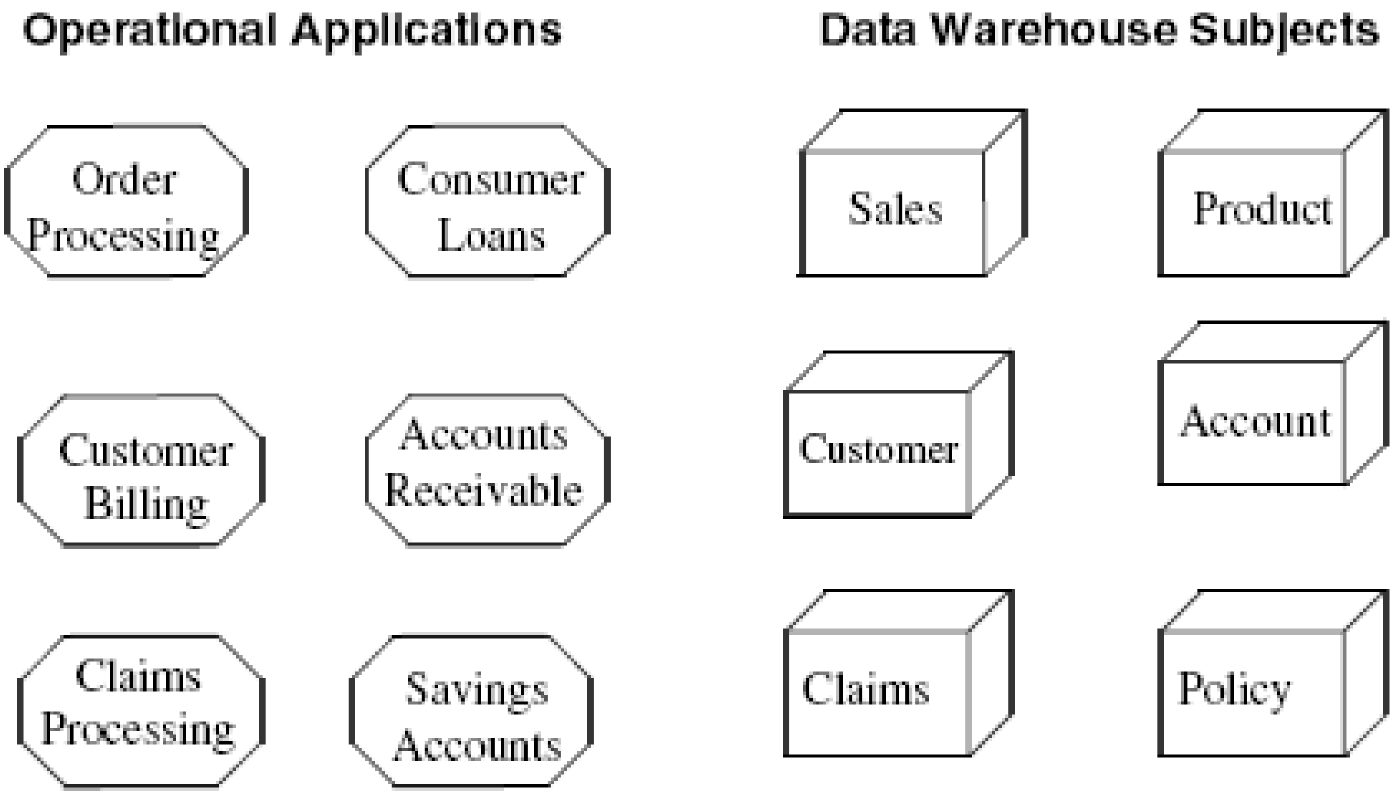


Figure 2-1 The data warehouse is subject oriented.

Data Warehouse—Integrated

- Constructed by integrating multiple, heterogeneous data sources
  - relational databases, flat files, on-line transaction records
- Data cleaning and data integration techniques are applied.
  - Ensure consistency in naming conventions, encoding structures, attribute measures, etc. among different data sources
    - E.g., Hotel price: currency, tax, breakfast covered, etc.
  - When data is moved to the warehouse, it is converted.

Integrated Data

Data inconsistencies are removed; data from diverse operational applications is integrated.

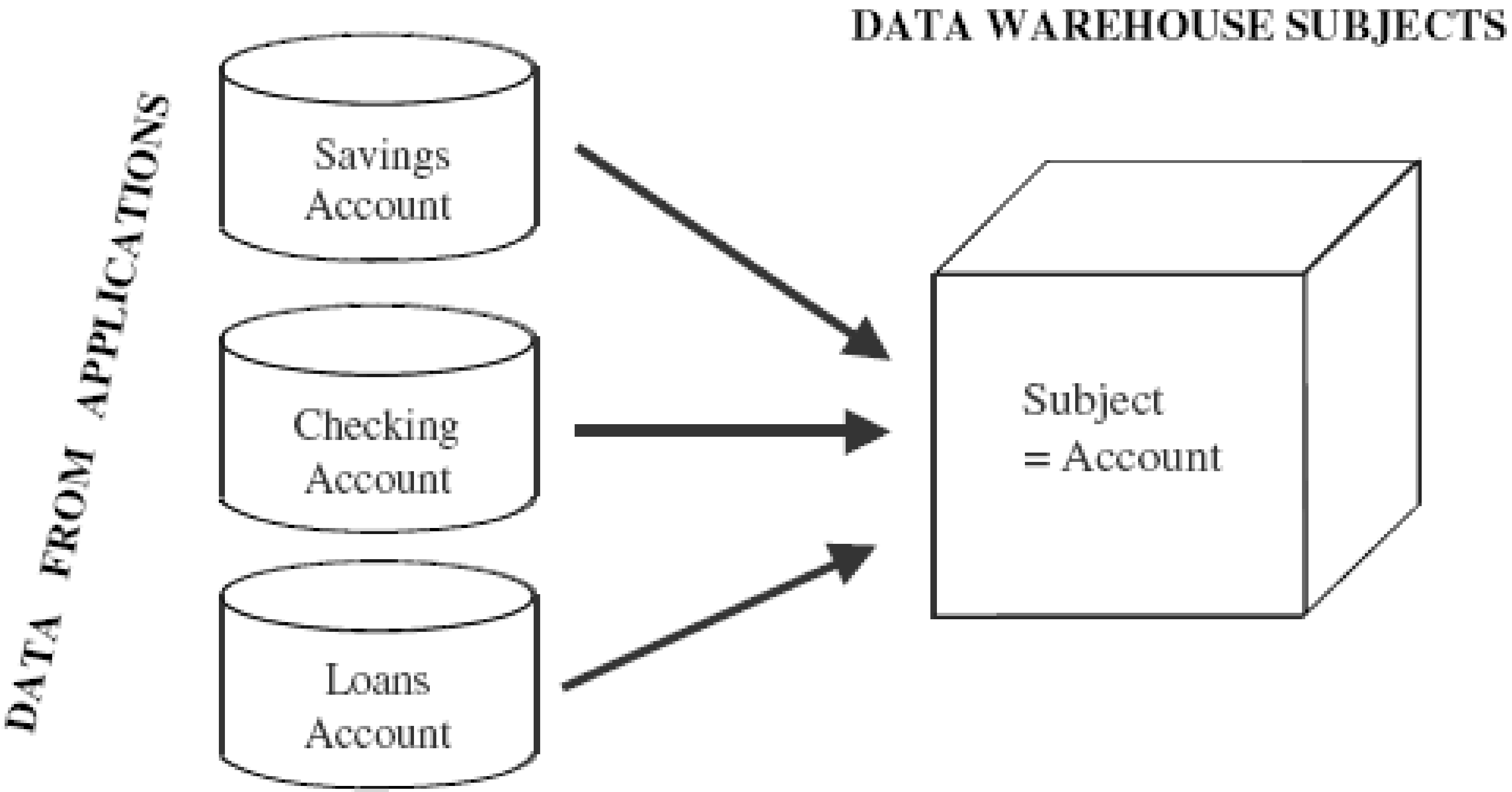


Figure 2-2 The data warehouse is integrated.

Data Warehouse—Time Variant

- The time horizon for the data warehouse is significantly longer than that of operational systems
  - Operational database: current value data
  - Data warehouse data: provide information from a historical perspective (e.g., past 5-10 years)
- Every key structure in the data warehouse
  - Contains an element of time, explicitly or implicitly
  - But the key of operational data may or may not contain “time element”

Data Warehouse—Nonvolatile

- A physically separate store of data transformed from the operational environment
- Operational update of data does not occur in the data warehouse environment
  - **Does not require** transaction processing, recovery, and concurrency control mechanisms
  - Requires only two operations in data accessing:
    - *initial loading of data* and *access of data*

NEED TO SEPARATE OPERATIONAL AND INFORMATION SYSTEMS

Three primary factors:

- A data warehouse centralizes data that are scattered throughout disparate operational systems and makes them available for DS.
- A well-designed data warehouse adds value to data by improving their quality and consistency.
- A separate data warehouse eliminates much of the contention for resources that results when information applications are mixed with operational processing.

Why Separate Data Warehouse?

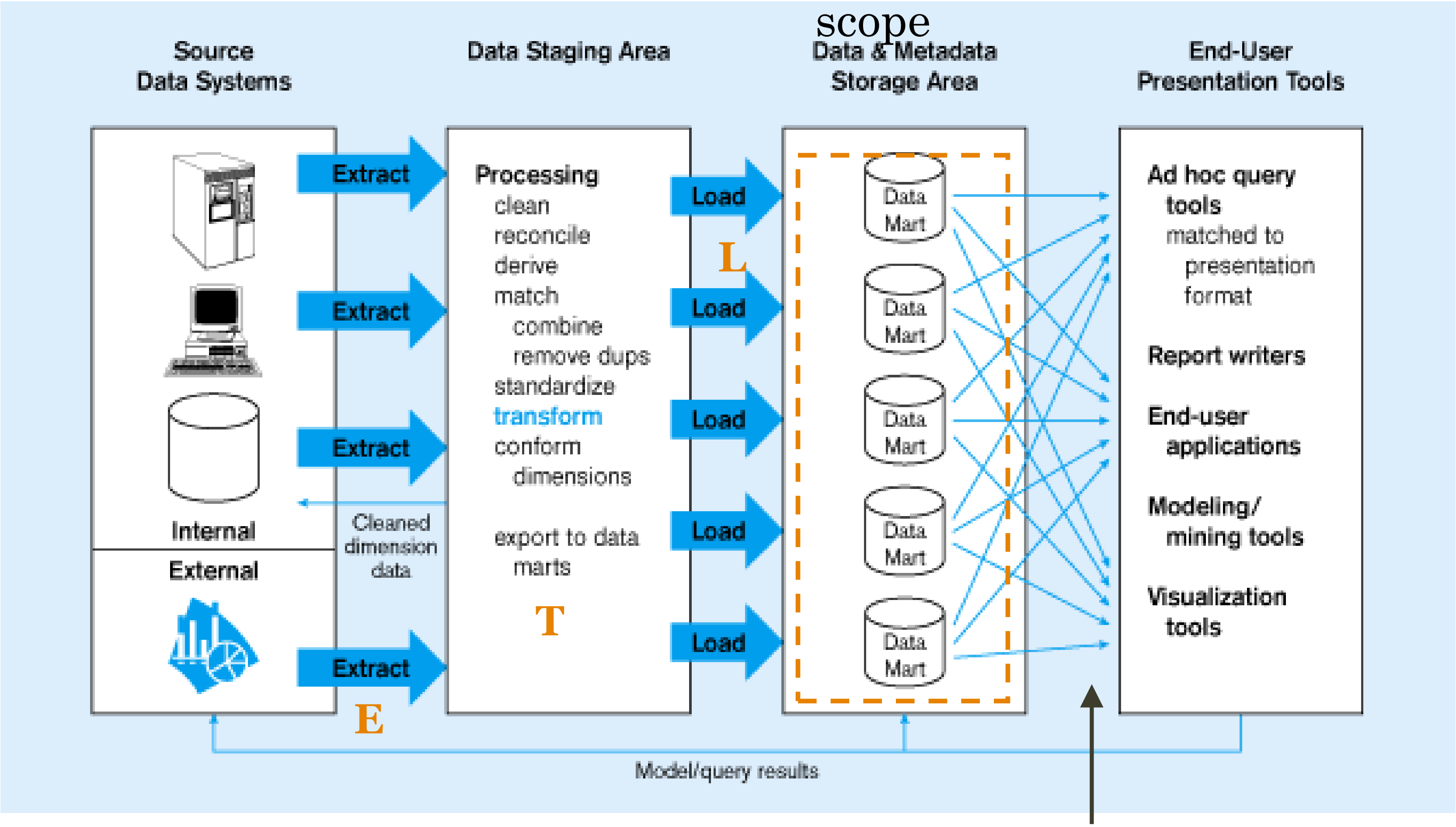
- High performance for both systems
  - DBMS— tuned for OLTP: access methods, indexing, concurrency control, recovery
  - Warehouse—tuned for OLAP: complex OLAP queries, multidimensional view, consolidation
- Different functions and different data:
  - missing data: Decision support requires historical data which operational DBs do not typically maintain
  - data consolidation: DS requires consolidation (aggregation, summarization) of data from heterogeneous sources
  - data quality: different sources typically use inconsistent data representations, codes and formats which have to be reconciled
- Note: There are more and more systems which perform OLAP analysis directly on relational databases

Data mart

- Data mart is a logical subset of a complete data warehouse
- Data marts are analytical stores designed to focus on specific business functions for a specific department with in an organization. E.g, sales, marketing, research, Finance, Dev.
- All the data marts in a particular area form an integrated data warehouse.
- Types of Data marts:
  - Independent data mart: a data mart filled with data extracted from the operational environment without benefits of a data warehouse. This is **Bottom up approach** of designing data ware houses
    - Advantages: Faster and easier to implement manageable data marts
    - Disadvantages: Possibility of redundant data, inconsistent data
  - Dependent data marts : sourced directly from enterprise data warehouses. Top down approach of building data mats out of data warehouses
    - Advantages: Single central storage , having centralized rules.
    - Disadvantages: More time consuming

Figure 11-3: Independent Data Mart

Data marts:
Mini-warehouses, limited in scope

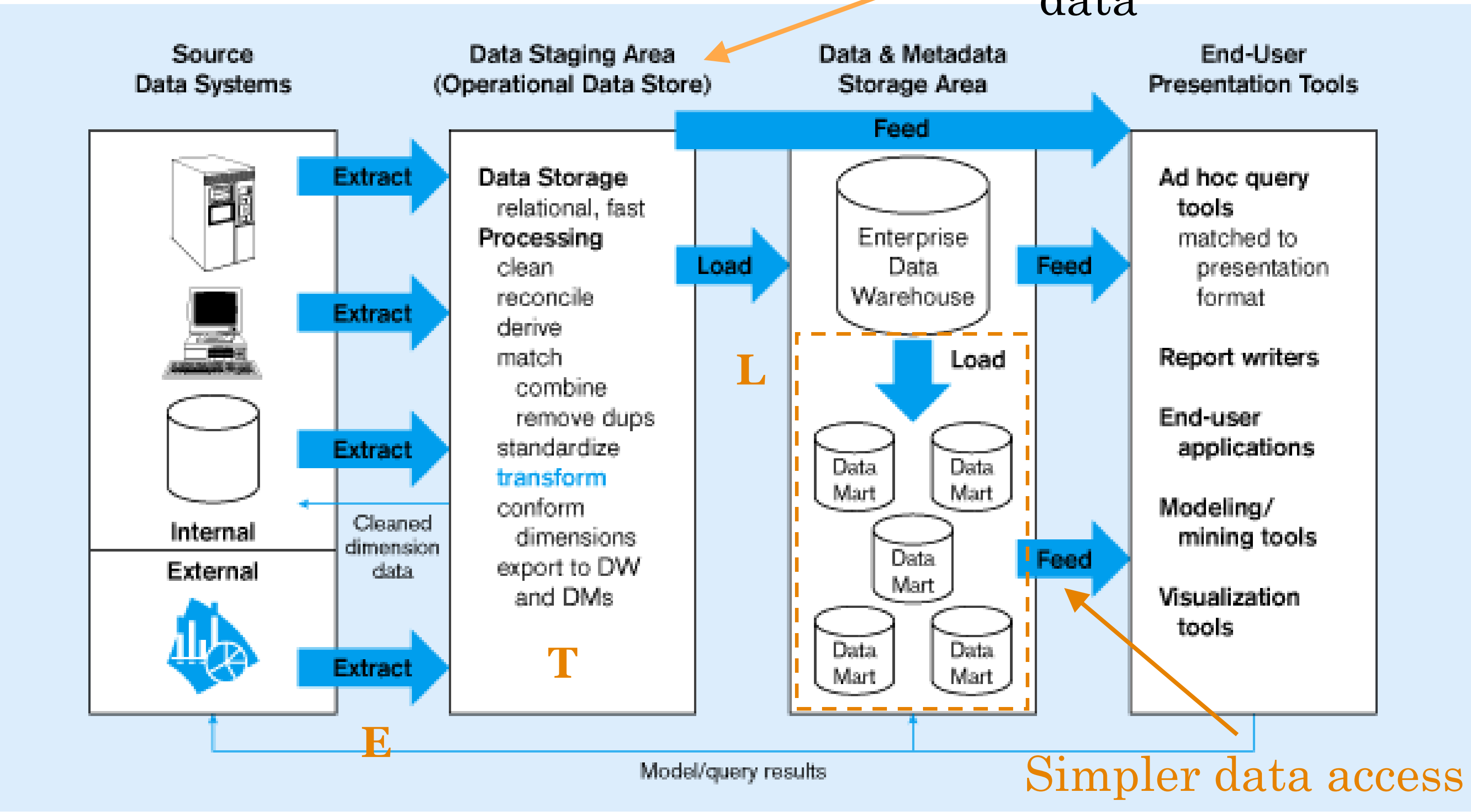


Separate ETL for each ***independent*** data mart

Data access complexity due to ***multiple*** data marts

Figure 11-4:
Dependent data mart with *operational data store*

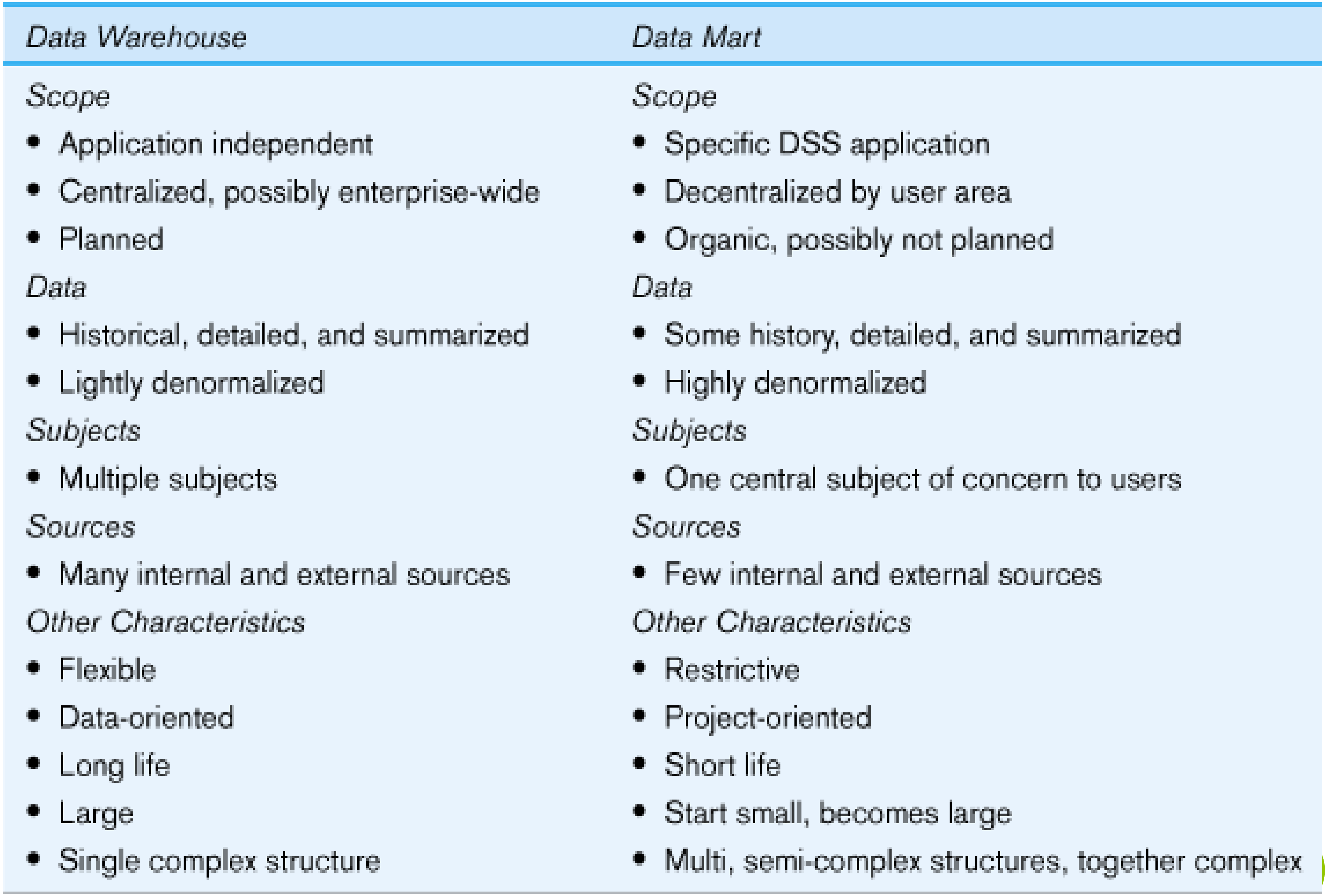
ODS provides option for obtaining ***current*** data



Simpler data access

Single ETL for ***enterprise data warehouse (EDW)***

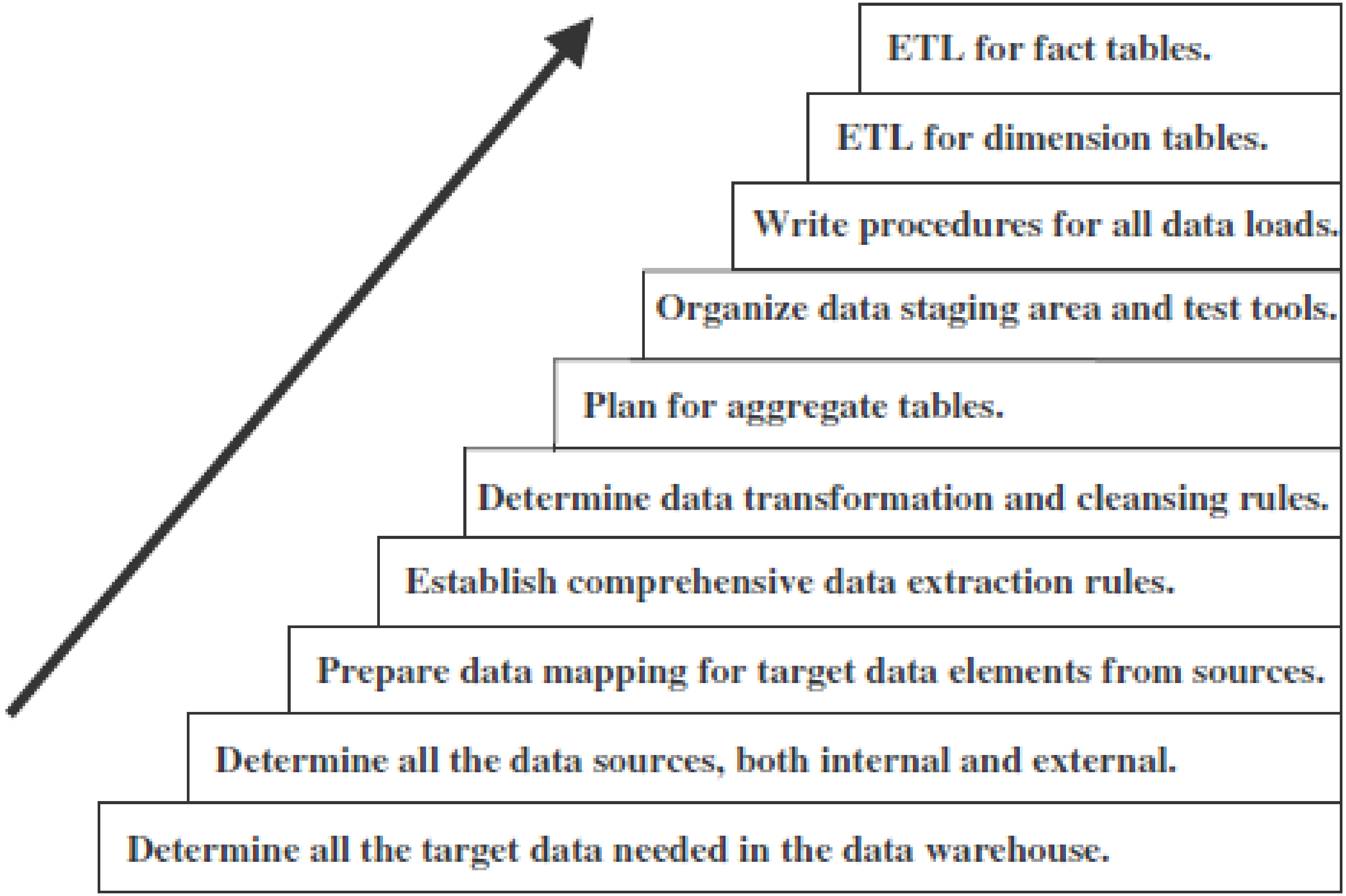
Dependent data marts loaded from EDW



Data Warehouse	*Data Mart*
Scope	*Scope*
• Application independent	• Specific DSS application
• Centralized, possibly enterprise-wide	• Decentralized by user area
• Planned	• Organic, possibly not planned
Data	*Data*
• Historical, detailed, and summarized	• Some history, detailed, and summarized
• Lightly denormalized	• Highly denormalized
Subjects	*Subjects*
• Multiple subjects	• One central subject of concern to users
Sources	*Sources*
• Many internal and external sources	• Few internal and external sources
Other Characteristics	*Other Characteristics*
• Flexible	• Restrictive
• Data-oriented	• Project-oriented
• Long life	• Short life
• Large	• Start small, becomes large
• Single complex structure	• Multi, semi-complex structures, together complex

ETL in Data Staging

Steps in Data Extraction



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.

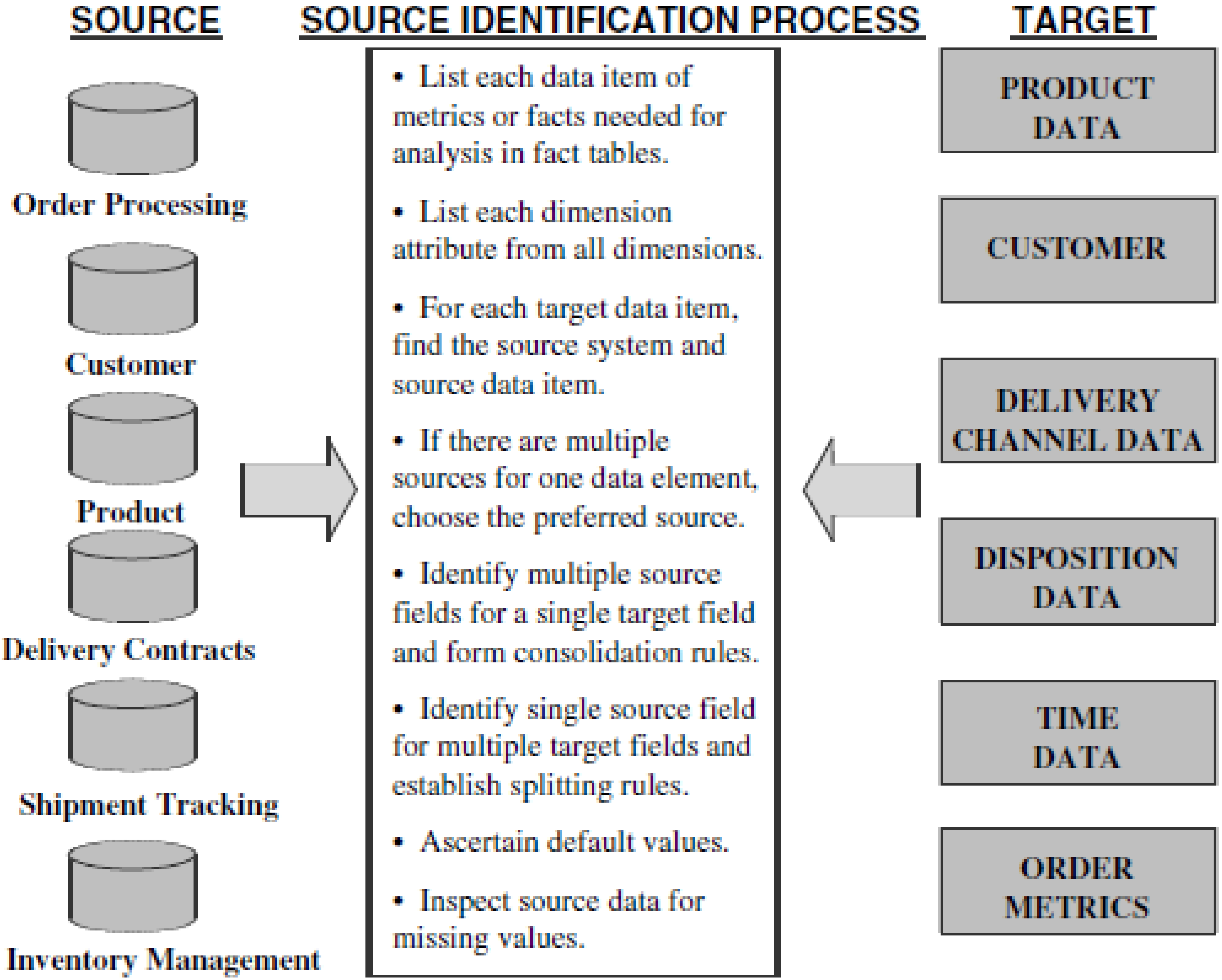
ETL PROCESS

- Combining data from several sources into a SINGLE row in the DW.
- Split the data structure into several data structures for several rows of target database
- Read data:
  - Data dictionaries
  - Flat files, indexed files
  - Legacy file systems
- Load all data for populating FACT tables
- Use Aggregate functions to populate aggregate or Summary fact tables.
- Data transformation from on data format to the target database format
- Change cryptic values to meaningful information used by user.

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.
- *Extraction frequency*—for each data source, establish how frequently the data extraction must by 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.

Source Identification



SOURCE

- Order Processing
- Customer
- Product
- Delivery Contracts
- Shipment Tracking
- Inventory Management

SOURCE IDENTIFICATION PROCESS

- 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.

TARGET

- PRODUCT DATA
- CUSTOMER
- DELIVERY CHANNEL DATA
- DISPOSITION DATA
- TIME DATA
- ORDER METRICS

METHOD OF EXTRACTION

- Data extraction is manual or using a Tool.
- *Immediate Data Extraction*
  - *Occurs as the transaction occurs at the source database*
- *Deferred Data Extraction*
  - *Do not capture the change in the real time.*

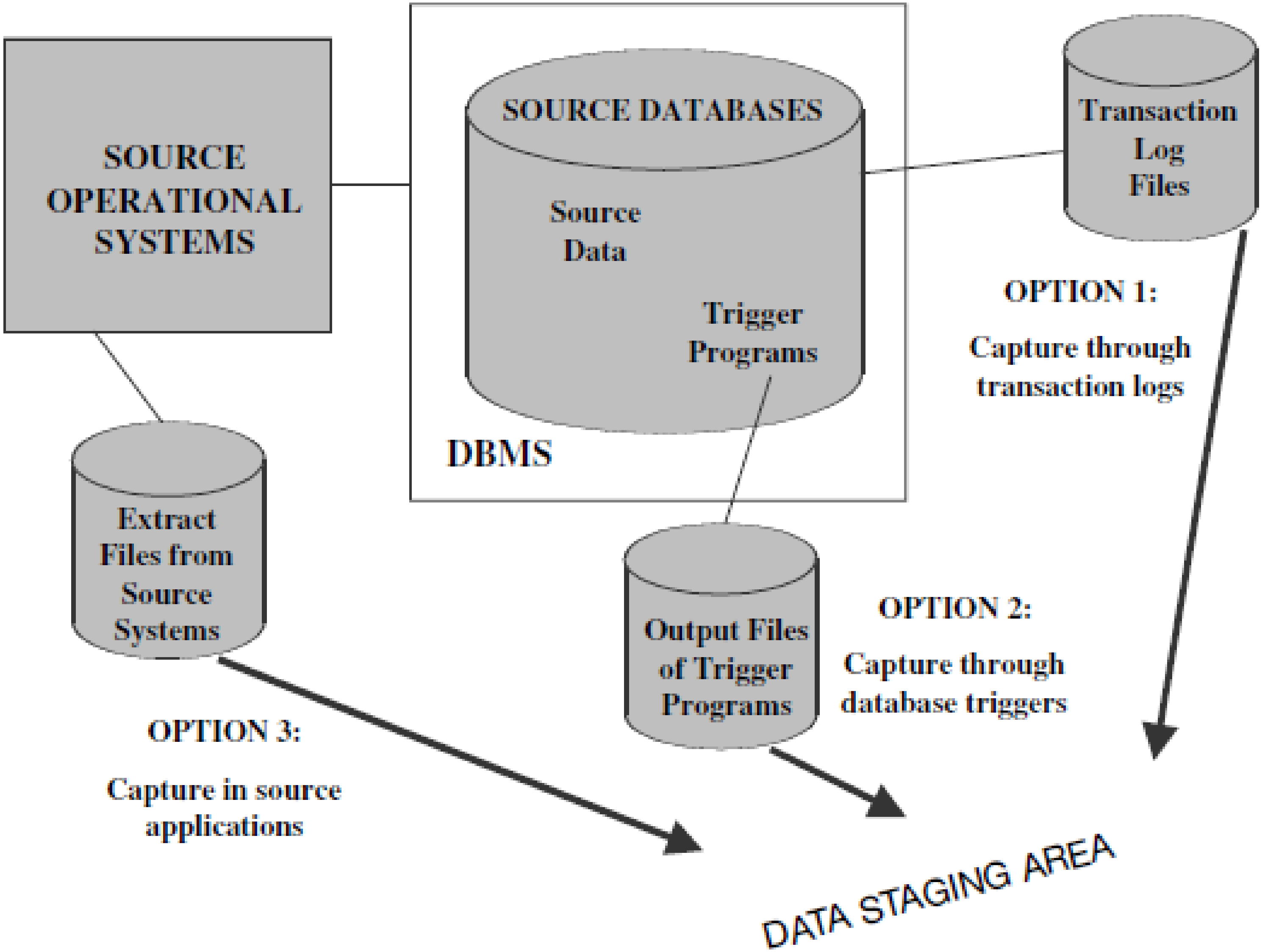
IMMEDIATE DATA EXTRACTION

- Types:
  - Capture through Transaction Logs
  - Capture through Database Triggers
  - Capture in Source Applications

- 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.
  - There is no extra overhead in the operational systems because logging is already part of the transaction processing.

- Capture through Database Triggers
  - 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. 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
- Capture in Source Applications
  - 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. Then other extract programs can use the separate file containing the changes to the source data.

Immediate Data Extraction


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

TYPES:

- Capture Based on Date and Time Stamp
- Capture by Comparing Files

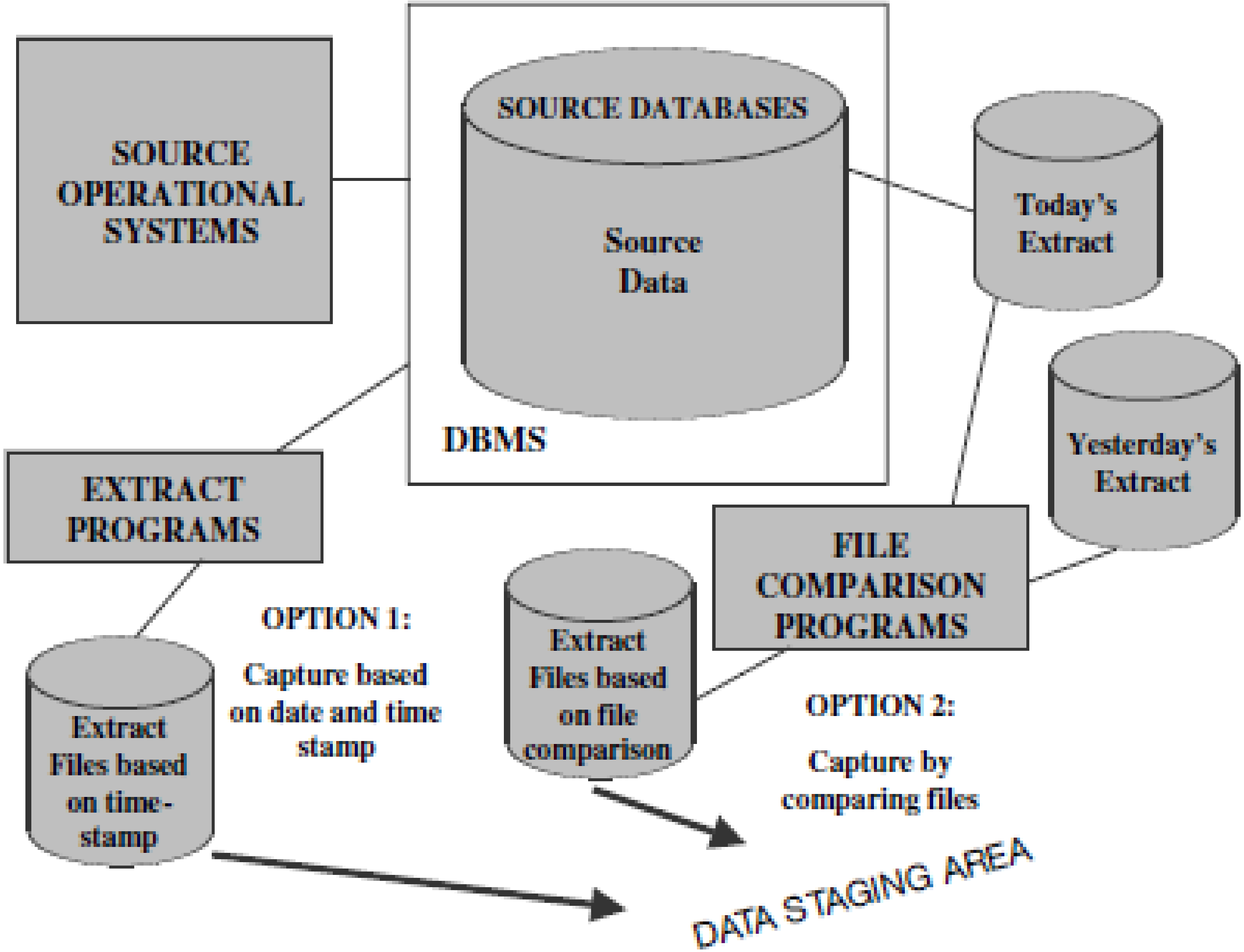
Capture Based on Date and Time Stamp

- 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.
- This technique works well if the number of revised records is small.
- This technique captures the latest state of the source data.
- Any intermediary states between two data extraction runs are lost.
- Deletion of source records presents a special problem.
  - If a source record gets deleted in between two extract runs, the information about the delete is not detected.
  - Workaround:
    - You can get around this by marking the source record for delete first, do the extraction run, and then go ahead and physically delete the record. This means you have to add more logic to the source applications.

Capture by Comparing Files

- Also known as snapshot differential technique because it compares two snapshots of the source 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.
- 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.

Deferred Data Extraction


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

DATA TRANSFORMATION

- Map input data to data for data warehouse repository
- Clean data, de-duplicate, and merge/purge
- Denormalize extracted data structures as required by the dimensional model of the data warehouse
- Convert data types
- Calculate and derive attribute values
- Check for referential integrity
- Aggregate data as needed
- Resolve missing values
- Consolidate and integrate data

STEPS OF TRANSFORMATION:

- Cleaning Data
  - Correction of spellings
  - Default values for missing data elements
  - Elimination of duplicates when the data comes from multiple sources.
  - Cleaning Data involves:
    - Data conditioning
    - Scrubbing
    - Merging

- Standardize Data
  - Standardize data types and field lengths for same data elements retrieved from different data sources

TASKS IN TRANSFORMATION:

- Selection
  - Select either whole records or parts of several records from the source systems.
- 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.

- 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 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.
  - For example, for a grocery chain, sales data at the lowest level of detail for every transaction at the checkout may not be needed. *Storing sales by product by store by day in the data warehouse may be quite adequate*. So, in this case, the data transformation function includes summarization of daily sales by product and by store.

- Enrichment.
  - This task is the rearrangement and simplification of individual fields to make them more useful for the data warehouse environment.
  - One or more fields can be used 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.

Types of Transformations:

- Most common transformation types:
  - Format Revisions
  - Decoding of Fields
  - Calculated and Derived Values
  - Splitting of Single Field
  - Merging of Information
  - Character Set Conversion
  - Conversion of Units of Measurement
  - Date/Time Conversions
  - Summarizations
  - Key Restructuring
  - De-duplication

- Format Revisions
  - 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.

- Decoding the Fields:
  - 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. What do the codes AC, IN, RE, and SU mean in a customer file? You need to decode all such cryptic codes and change these into values that make sense to the users. Change the codes to Active, Inactive, Regular, and Suspended.

- 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 data warehouse. Average daily balance and operating ratios are examples of derived fields.

- 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.

- 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*.

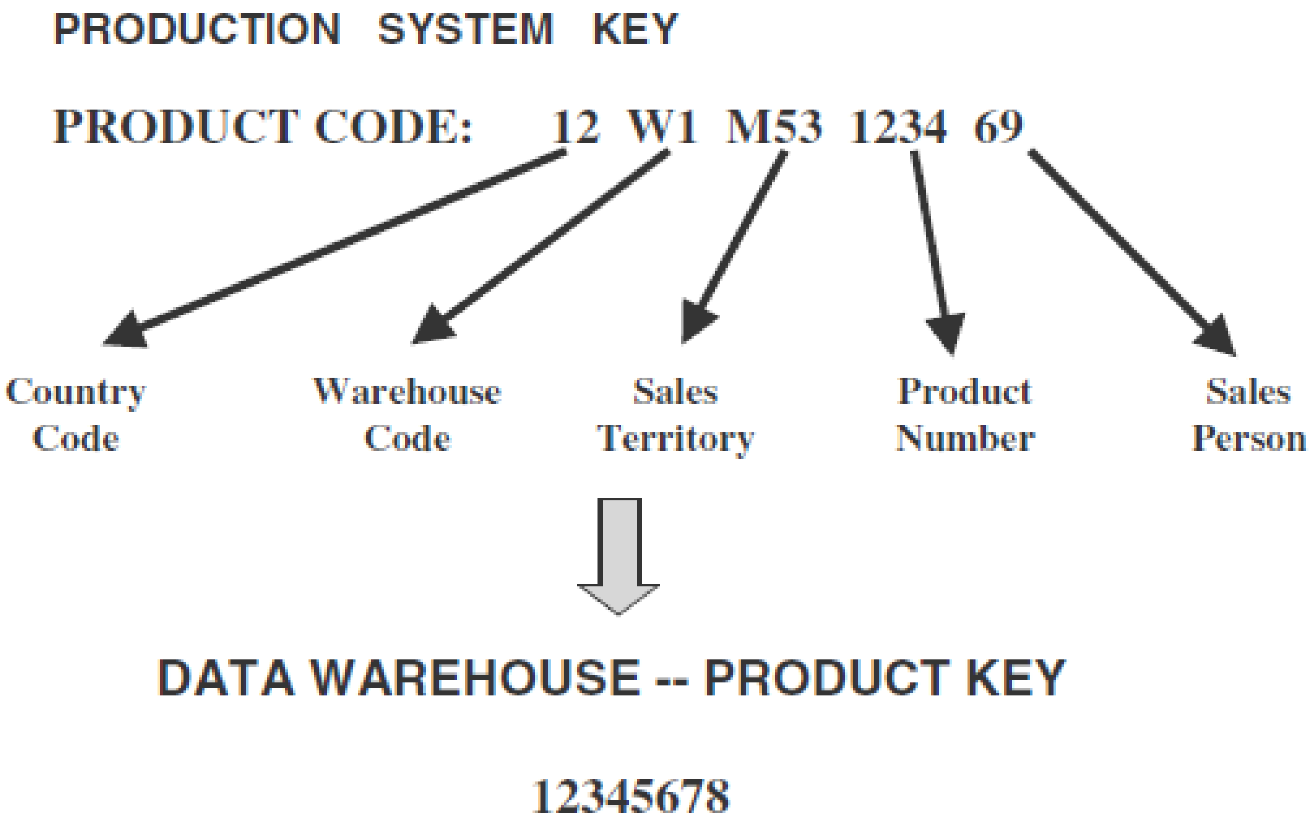
- Character Set Conversion.
  - This type of data transformation relates to the conversion of character sets to an agreed standard character set for textual data in the data warehouse.
  - If you have mainframe legacy systems as source systems, the source data from these systems will be in EBCDIC characters. 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.

- 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 may all be 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, 2000 is written as
- 10/11/2000 in the U.S. format and
- as 11/10/2000 in the British format.
- This date may be standardized to be written as 11 OCT 2000.

- **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.

- **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.
  - The product code in this organization is structured to have inherent meaning. If you use this product code as the primary key, there would 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.**



PRODUCTION SYSTEM KEY

PRODUCT CODE: 12 W1 M53 1234 69

Country Code

Warehouse Code

Sales Territory

Product Number

Sales Person

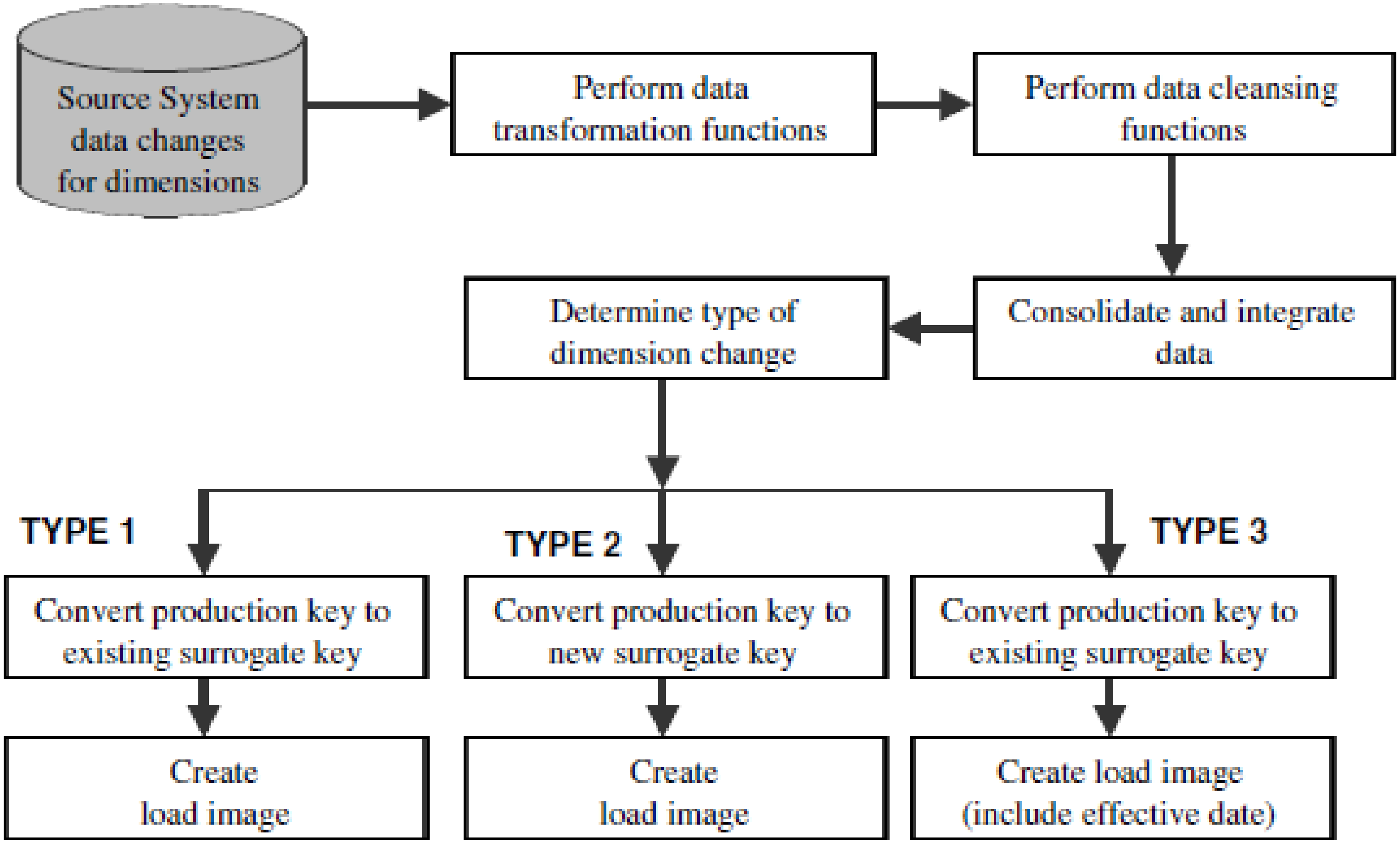
DATA WAREHOUSE -- PRODUCT KEY

12345678

Data transformation: key restructuring.

- 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.

Implementing Transformation



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)

Transformed for dimension changes.

[DBT] KJSCE

Transformation Tools:

- Use of automated tools certainly improves efficiency and accuracy.
- As a data transformation specialist, you just have to specify the parameters, the data definitions, and the rules to the transformation tool. If your input into the tool is accurate, then the rest of the work is performed efficiently by the tool.
- You gain a major advantage from using a transformation tool because of the recording of metadata by the tool. When you specify the transformation parameters and rules, these are stored as metadata by the tool.
- This metadata then becomes part of the overall metadata component of the data warehouse. It may be shared by other components. When changes occur to transformation functions because of changes in business rules or data definitions, you just have to enter the changes into the tool. The metadata for the transformations get automatically adjusted by the tool.

MANUAL TECHNIQUES

- This was the predominant method until recently when transformation tools began to appear in the market. Manual techniques may still be adequate for smaller data warehouses.
- Here manually coded programs and scripts perform every data transformation.
- Mostly, these programs are executed in the data staging area.
- The analysts and programmers who already possess the knowledge and the expertise are able to produce the programs and scripts.
- Of course, this method involves elaborate coding and testing. Although the initial cost may be reasonable, ongoing maintenance may escalate the cost. Unlike automated tools, the manual method is more likely to be prone to errors. It may also turn out that several individual programs are required in your environment.
- A major disadvantage relates to metadata. Automated tools record their own metadata, but in-house programs have to be designed differently if you need to store and use metadata.
- Even if the in-house programs record the data transformation metadata initially, every time changes occur to transformation rules, the metadata has to be maintained.
- This puts an additional burden on the maintenance of the manually coded transformation programs.

CONDITIONING, CLEANSING, SCRUBBING, MERGING

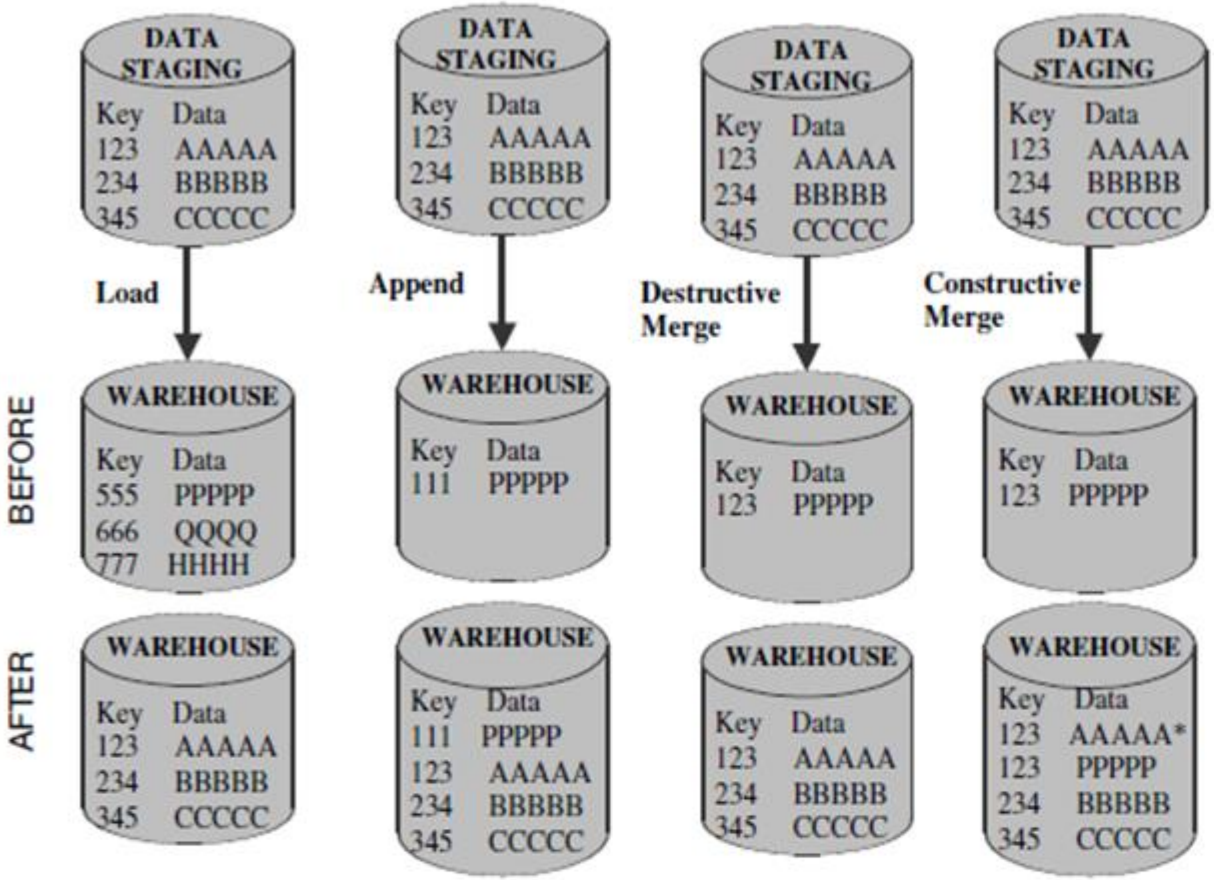
- Conditioning: The conversion of data types from the source to the target data store (warehouse) -- always a relational database
- Cleansing:
  - Looking for and handling data errors,
  - Missing values
  - Controlling duplicity
  - Outlier identification: Outliner is a low frequency value for the variable that is far from the mean value.
  - Involves: Scrubbing and Enrichment of data

- Scrubbing:Sophisticated transformation tools.
  - Used for cleaning the quality of data
  - Clean data is vital for the success of the warehouse
  - Mainly deals with human errors
  - Example
    - Seshadri, Sheshadri, Sesadri, Seshadri S., Srinivasan Seshadri, etc. are the same person
- Data Merging:
  - Eg, information of a Employee comes from different data sources
    - Data Source1: Employee Id and name
    - Data source2: Employee dependents data
    - Data Source3: Department Id for Employee
    - Data Source4 :Department Details

Data Loading

- 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.

- 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)



Load

DATA STAGING

Key	Data
123	AAAAA
234	BBBBB
345	CCCCC

WAREHOUSE (BEFORE)

Key	Data
555	PPPPP
666	QQQQ
777	HHHH

WAREHOUSE (AFTER)

Key	Data
123	AAAAA
234	BBBBB
345	CCCCC

Append

DATA STAGING

Key	Data
123	AAAAA
234	BBBBB
345	CCCCC

WAREHOUSE (BEFORE)

Key	Data
111	PPPPP

WAREHOUSE (AFTER)

Key	Data
111	PPPPP
123	AAAAA
234	BBBBB
345	CCCCC

Destructive Merge

DATA STAGING

Key	Data
123	AAAAA
234	BBBBB
345	CCCCC

WAREHOUSE (BEFORE)

Key	Data
123	PPPPP

WAREHOUSE (AFTER)

Key	Data
123	AAAAA
234	BBBBB
345	CCCCC

Constructive Merge

DATA STAGING

Key	Data
123	AAAAA
234	BBBBB
345	CCCCC

WAREHOUSE (BEFORE)

Key	Data
123	PPPPP

WAREHOUSE (AFTER)

Key	Data
123	AAAAA*
123	PPPPP
234	BBBBB
345	CCCCC

Modes of applying data.

MODES OF APPLYING DATA:

- ***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.

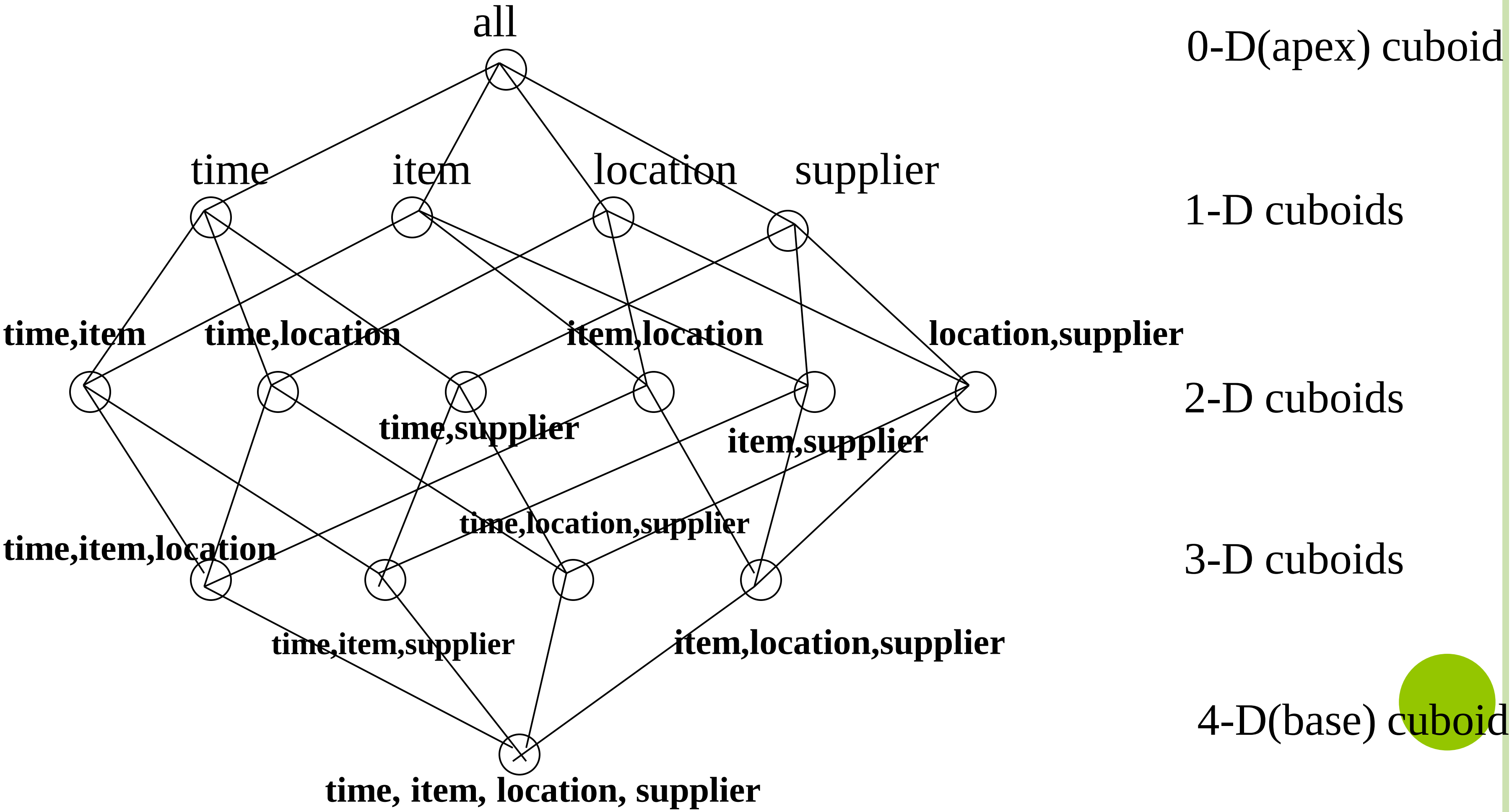
- ***Destructive Merge.*** In this mode, you apply the incoming data to the target data. If the primary key of an incoming record matches with the key of an existing record, update the matching target record. If the incoming record is a new record without a match with any existing record, add the incoming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

Multi-dimensional Data Modelling

From Tables and Spreadsheets to Data Cubes

- A data warehouse is based on a multidimensional data model which views data in the form of a data cube
- A data cube, such as sales, allows data to be modeled and viewed in multiple dimensions
  - Dimension tables, such as item (item_name, brand, type), or time(day, week, month, quarter, year)
  - Fact table contains measures (such as dollars_sold) and keys to each of the related dimension tables
- In data warehousing literature, an n-D base cube is called a base cuboid. The topmost 0-D cuboid, which holds the highest-level of summarization, is called the apex cuboid. The lattice of cuboids forms a data cube.

Cube: A Lattice of Cuboids



all

0-D(apex) cuboid

time

item

location

supplier

1-D cuboids

time,item

time,location

item,location

location,supplier

time,supplier

item,supplier

2-D cuboids

time,location,supplier

time,item,location

3-D cuboids

time,item,supplier

item,location,supplier

4-D(base) cuboid

time, item, location, supplier

Fact Tables

- Contains two or more foreign keys
- Tend to have huge numbers of records
- Useful facts tend to be numeric and additive that can be easily manipulated, particularly by summing together many thousands of rows.
- Grain of Fact Table:
  - When designing a fact table, developers must pay careful attention to the grain of the table -- the level of detail contained within the table. The developer designing the purchase fact table described above would need to decide, for example, whether the grain of the table is a customer transaction or an individual item purchase.
  - In the case of an individual item purchase grain, each customer transaction would generate **multiple** fact table entries, corresponding to each item purchased. The choice of grain is a fundamental decision made during the design process that can have significant impact on the business intelligence effort down the road.

DIMENSION TABLES

- Dimensions describe the objects involved in a business intelligence effort.
- While facts correspond to events, dimensions correspond to people, items, or other objects.
- In a retail scenario : Sales is a Fact.
- On the other hand, items, branch, time and location are dimensions and should be contained in dimension tables.
- Dimension tables contain details about each instance of an object.
- For example, the items dimension table would contain a record for each item sold in the store. It might include information such as the cost of the item, the supplier, color, sizes, and similar data.

- Fact tables and dimension tables are related to each other.
- Primary key in one dimension table is a Foreign key in the fact table.

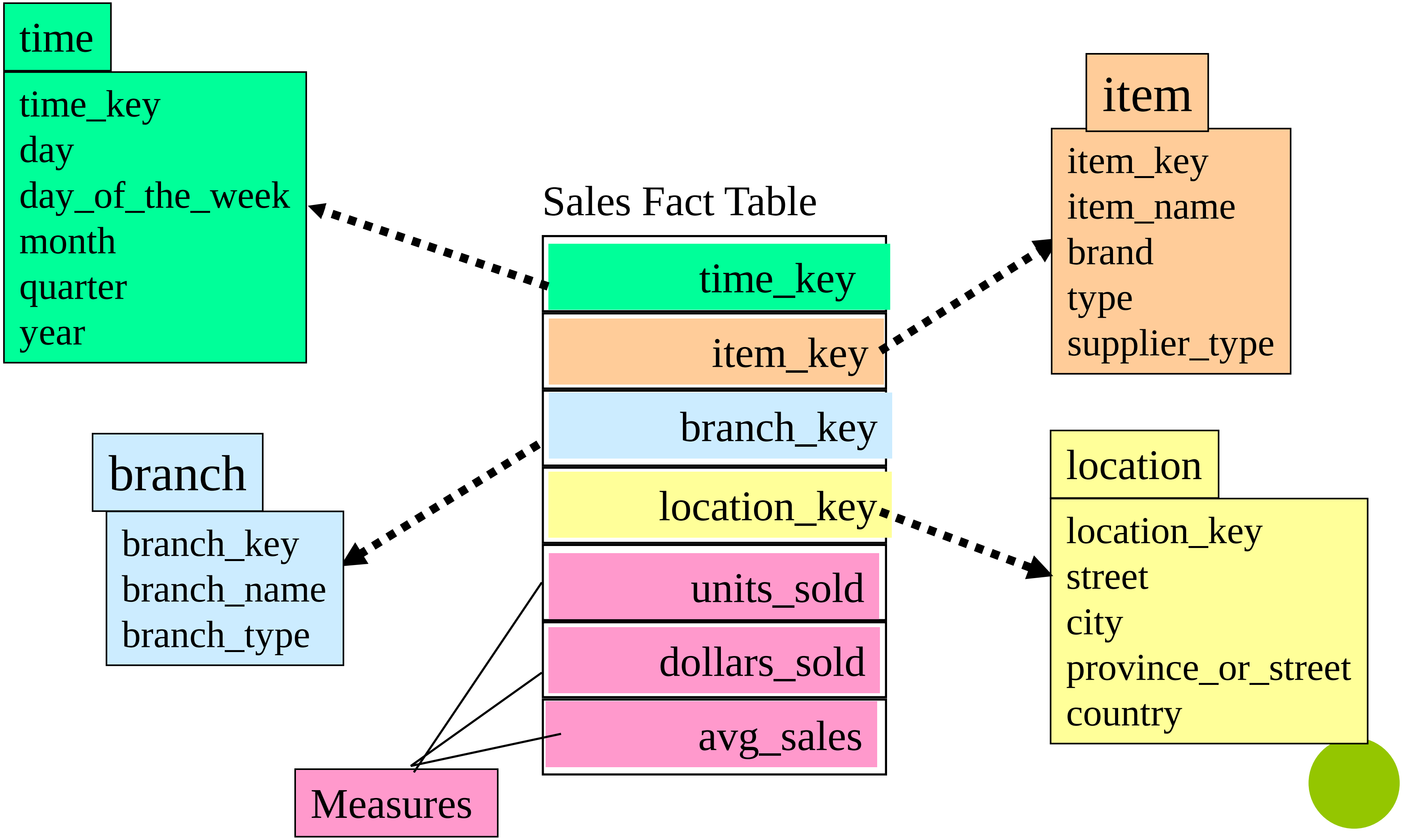
Conceptual Modeling of Data Warehouses

- Modeling data warehouses: dimensions & measures
  - Star schema: A fact table in the middle connected to a set of dimension tables
  - Snowflake schema: A refinement of star schema where some dimensional hierarchy is normalized into a set of smaller dimension tables, forming a shape similar to snowflake
  - Fact constellations: Multiple fact tables share dimension tables, viewed as a collection of stars, therefore called galaxy schema or fact constellation

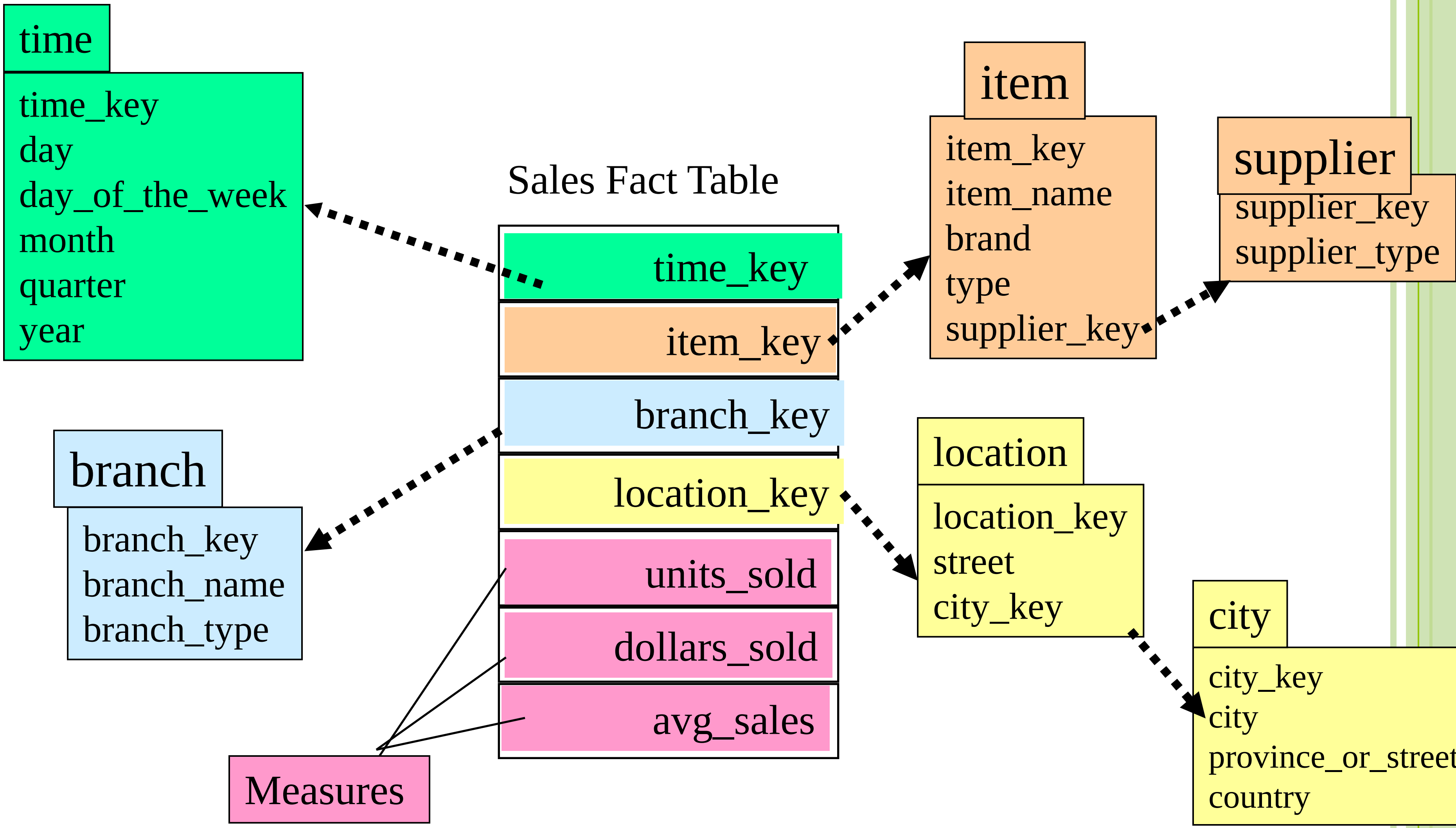
Schema

- Create a Star schema and a Snow flake schema:
- Sales are considered along four dimensions, namely, time, item, branch, and location.
- The schema contains a fact table for sales that contains keys to each of the four dimensions, along with two measures: dollars sold and units sold.
- To minimize the size of the fact table, dimension identifiers (such as time key and item key) are system-generated identifiers.

Example of Star Schema



Example of Snowflake Schema



time

time_key
day
day_of_the_week
month
quarter
year

Sales Fact Table

time_key
item_key
branch_key
location_key
units_sold
dollars_sold
avg_sales

item

item_key
item_name
brand
type
supplier_key

supplier

supplier_key
supplier_type

branch

branch_key
branch_name
branch_type

location

location_key
street
city_key

city

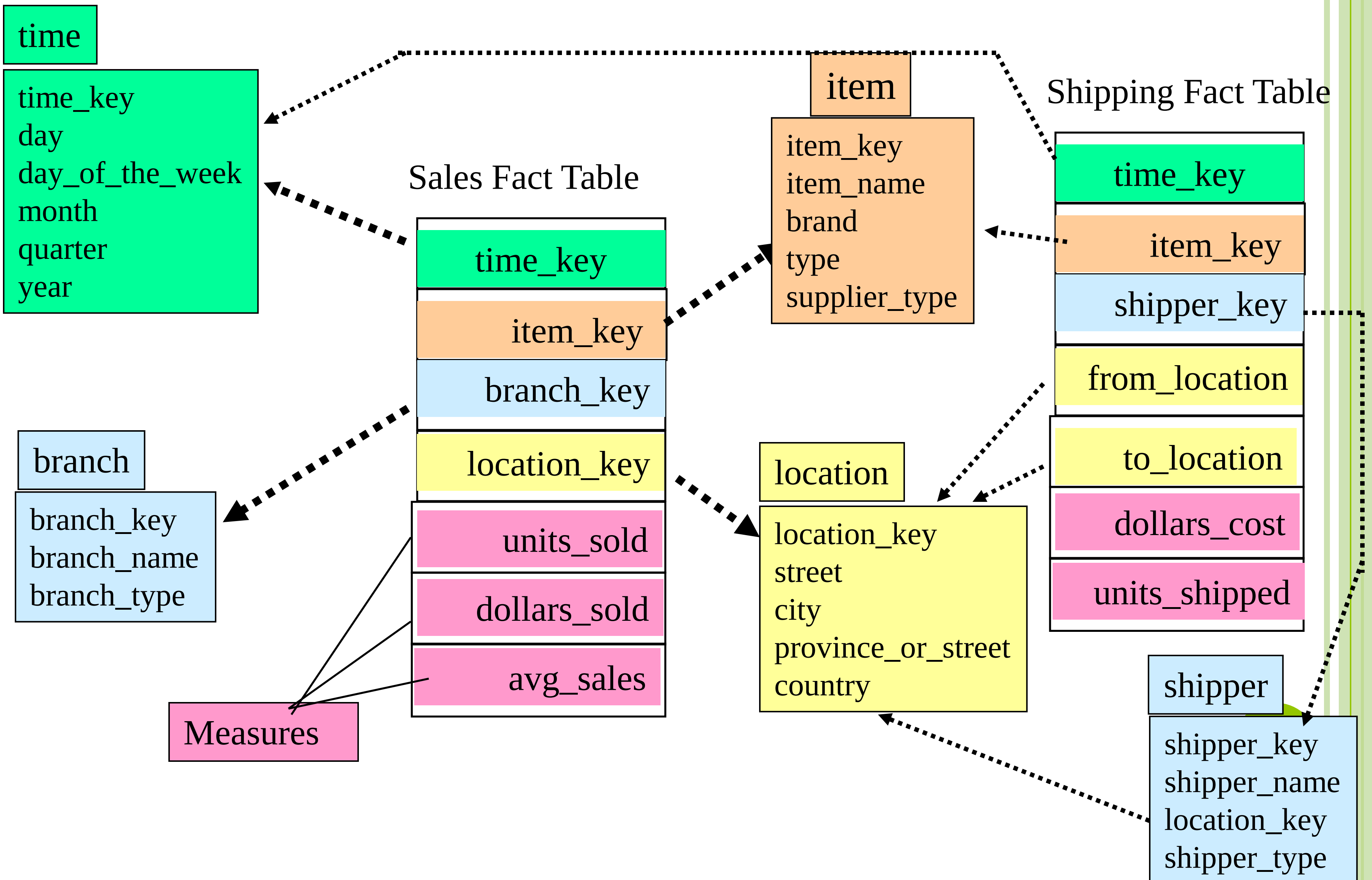
city_key
city
province_or_street
country

Measures

Fact Constellation

- The shipping table has five dimensions, or keys: item key, time key, shipper key, from location, and to location, and two measures: dollars cost and units shipped.
- A fact constellation schema allows dimension tables to be shared between fact tables. For example, the dimensions tables for time, item, and location are shared between both the sales and shipping fact tables.

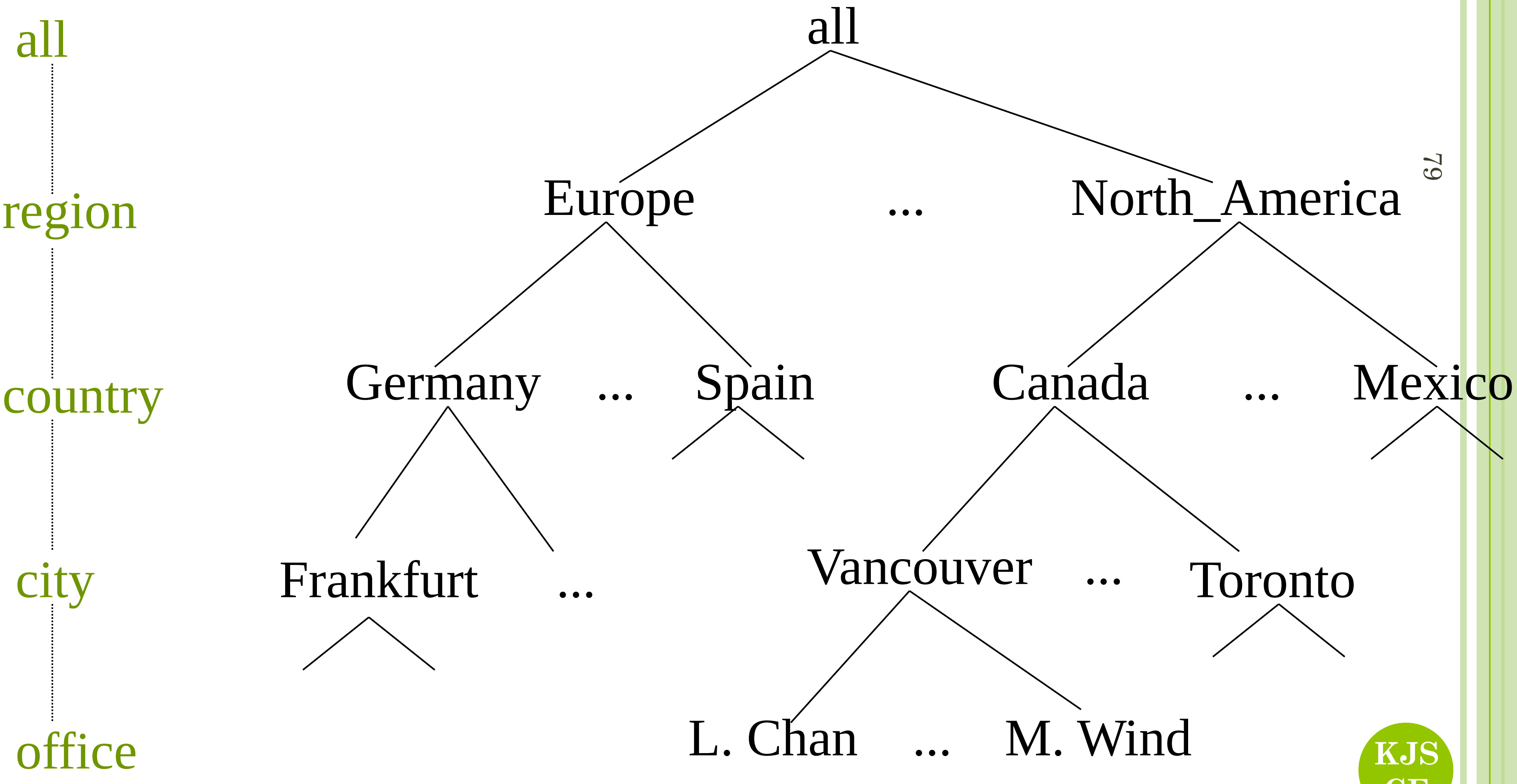
Example of Fact Constellation



Concept Hierarchy

- A concept hierarchy is an order relation between a set of attributes of a concept or dimension.
- It can be manually (users or experts) or automatically generated (statistical analysis).
- Multidimensional data is usually organized into dimension and each dimension is further defined into a lower level of abstractions defined by concept hierarchies.
- Example: Dimension (location)

A Concept Hierarchy: Dimension (location)



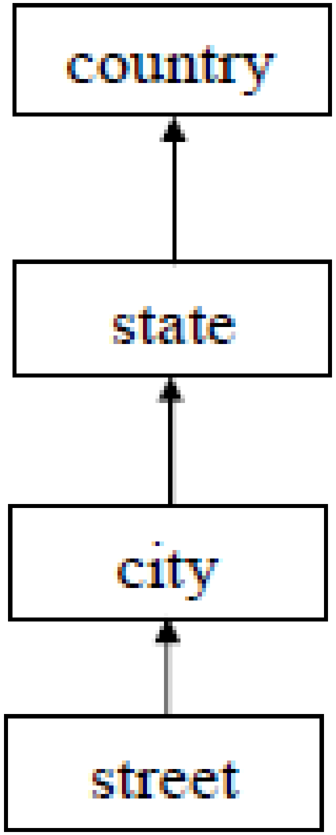
- The order can be either partial or total:
- Location dimension:

  Street <city<state<country

- Time dimension:

Day < {month<quarter ; week} < year

- Set-grouping hierarchy:
  - It is a concept hierarchy among groups of values.
  - Example: {1..10} < inexpensive



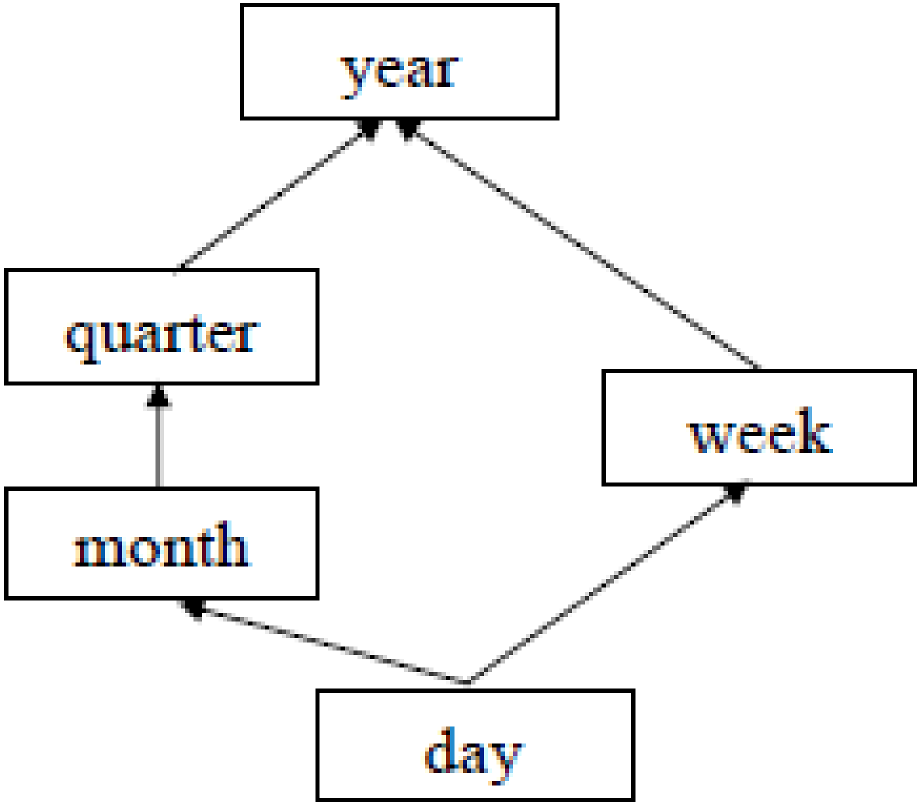
country

state

city

street

Total order hierarchy



year

quarter

week

month

day

Partial order hierarchy

ENTITY-RELATIONSHIP VS. DIMENSIONAL MODELS

Entity Relationship

- One table per entity
- Minimize data redundancy
- Optimize update
- The Transaction Processing Model

Dimensional Model

- One fact table for data organization
- Maximize understandability
- Optimized for retrieval
- The data warehousing model

OLAP

WHAT AND WHY OLAP?

- OLAP is the dynamic synthesis, analysis, and consolidation of large volumes of multi-dimensional data.
- OLAP is the term that describes a technology that uses multi-dimensional view of aggregate data to provide quick access to strategic information for the purposes of advanced analysis.
- OLAP enables users to gain a deeper understanding and knowledge about various aspects of their corporate data through fast, consistent, interactive access to a variety of possible views of data.
- While OLAP systems can easily answer 'who?' and 'what?' questions, it is easier ability to answer 'what if?' and 'why?' type questions that distinguishes them from general-purpose query tools.
- The types of analysis available from OLAP range from basic navigation and browsing (referred to as 'slicing' and dicing') , to calculations, to more complex analysis such as time series and complex modeling.

OLAP Applications

- Finance: Budgeting, activity-based costing, financial performance analysis, and financial modeling.
- Sales: Sales analysis and sales forecasting.
- Marketing: Market research analysis, sales forecasting, promotions analysis, customer analysis, and market/customer segmentation.
- Manufacturing: Production planning and defect analysis.

OLAP Key Features

- Multi-dimensional views of data.
- Support for complex calculations.
- Time Intelligence.

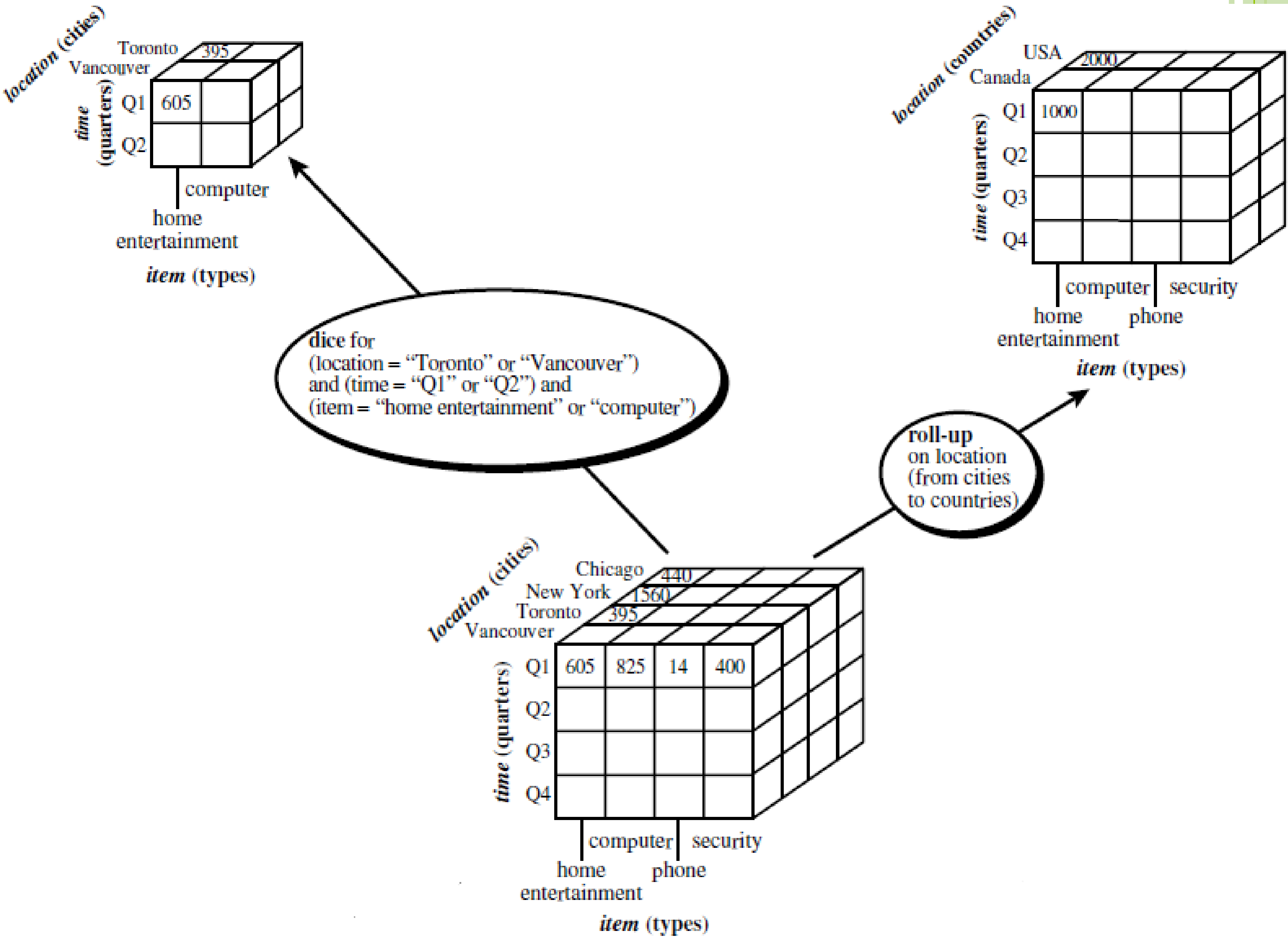
THE DATA CUBE

Multidimensional databases *(sometimes called OLAP)* are specialized data stores that organize facts by dimensions, such as geographical region, product line, salesperson, time. The data in these databases are usually preprocessed and stored in *data cubes.*

- One intersection might be the quantities of a product sold by specific retail locations during certain time periods.
- Another matrix might be Sales volume by department, by day, by month, by year for a specific region
- Cubes provide faster:
  - Queries
  - Slices and Dices of the information
  - Rollups
  - Drill Downs

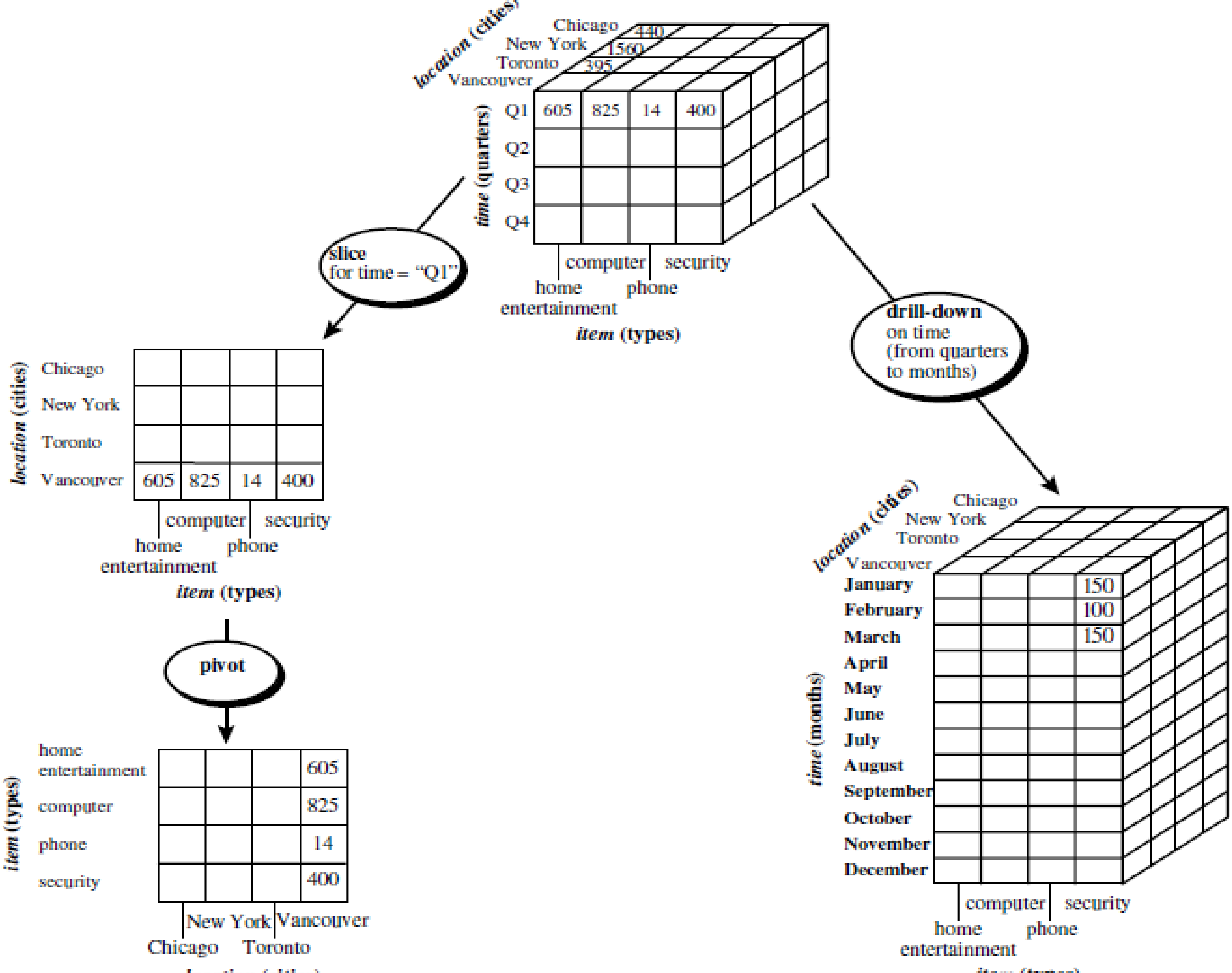
Typical OLAP Operations

- Roll up (drill-up): summarize data
  - *by climbing up hierarchy or by dimension reduction*
- Drill down (roll down): reverse of roll-up
  - *from higher level summary to lower level summary or detailed data, or introducing new dimensions*
- Slice and dice: *project and select*
- Pivot (rotate):
  - *reorient the cube, visualization, 3D to series of 2D planes*
- Other operations
  - *drill across: involving (across) more than one fact table*
  - *drill through: through the bottom level of the cube to its back-end relational tables (using SQL)*

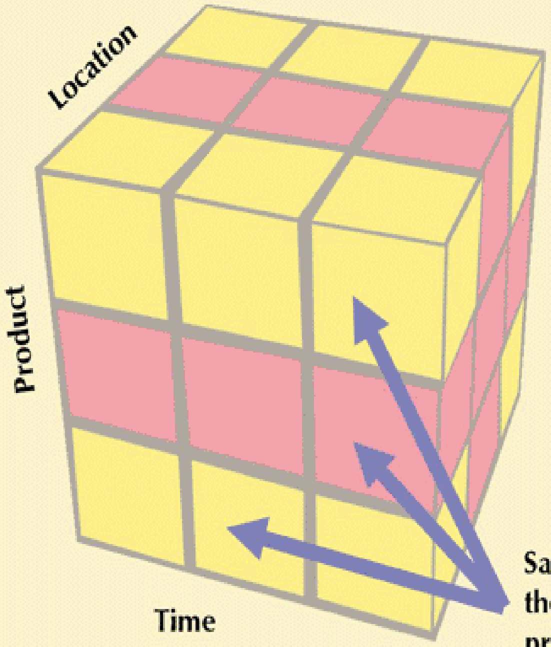


dice for
(location = "Toronto" or "Vancouver")
and (time = "Q1" or "Q2") and
(item = "home entertainment" or "computer")

roll-up
on location
(from cities
to countries)


location (cities)
Chicago
New York
Toronto
Vancouver
440
1560
395
time (quarters)
Q1
Q2
Q3
Q4
605
825
14
400
computer
security
home entertainment
phone
item (types)
slice for time = "Q1"
location (cities)
Chicago
New York
Toronto
Vancouver
605
825
14
400
computer
security
home entertainment
phone
item (types)
pivot
item (types)
home entertainment
computer
phone
security
605
825
14
400
New York
Vancouver
Chicago
Toronto
drill-down on time (from quarters to months)
location (cities)
Chicago
New York
Toronto
Vancouver
time (months)
January
February
March
April
May
June
July
August
September
October
November
December
150
100
150
computer
security
home entertainment
phone

Three-Dimensional View of Sales

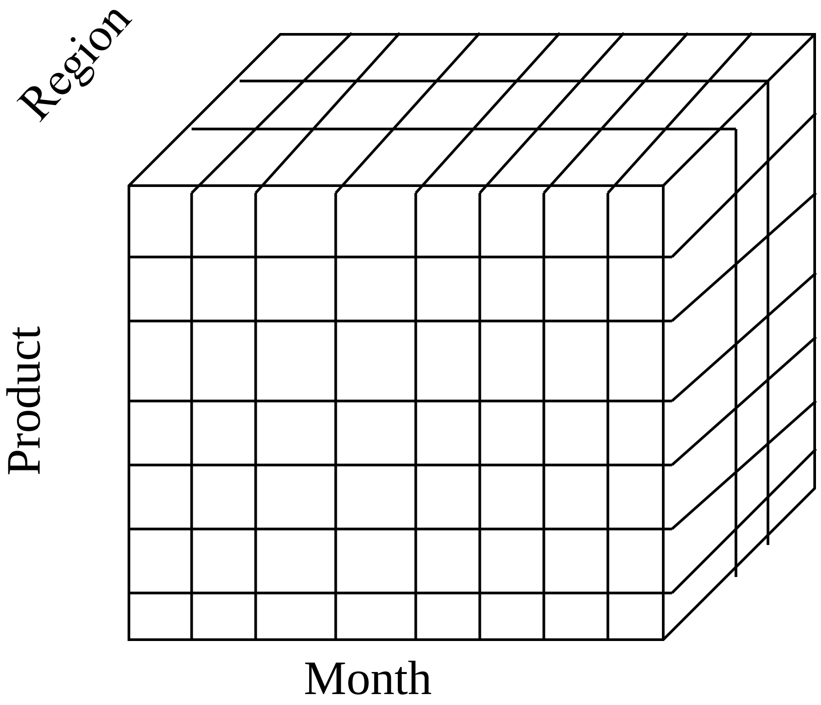


Conceptual three-dimensional cube of sales by product, location, and time

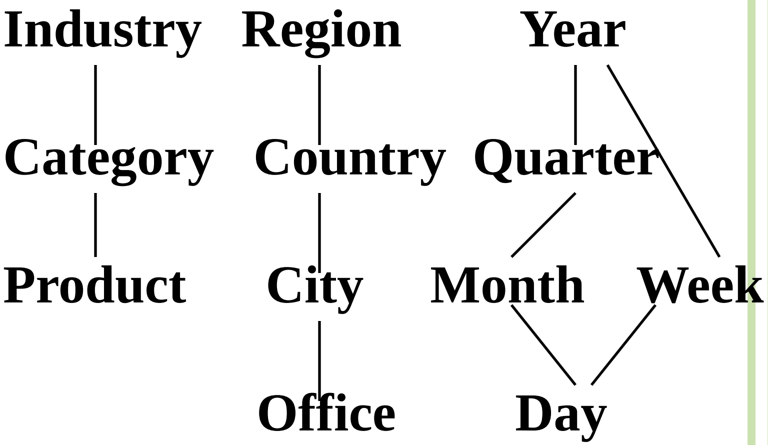
Sales facts are stored in the intersection of each product, time, and location dimension

Multidimensional Data

- Sales volume as a function of product, month, and region



Dimensions: *Product, Location, Time*
Hierarchical summarization paths

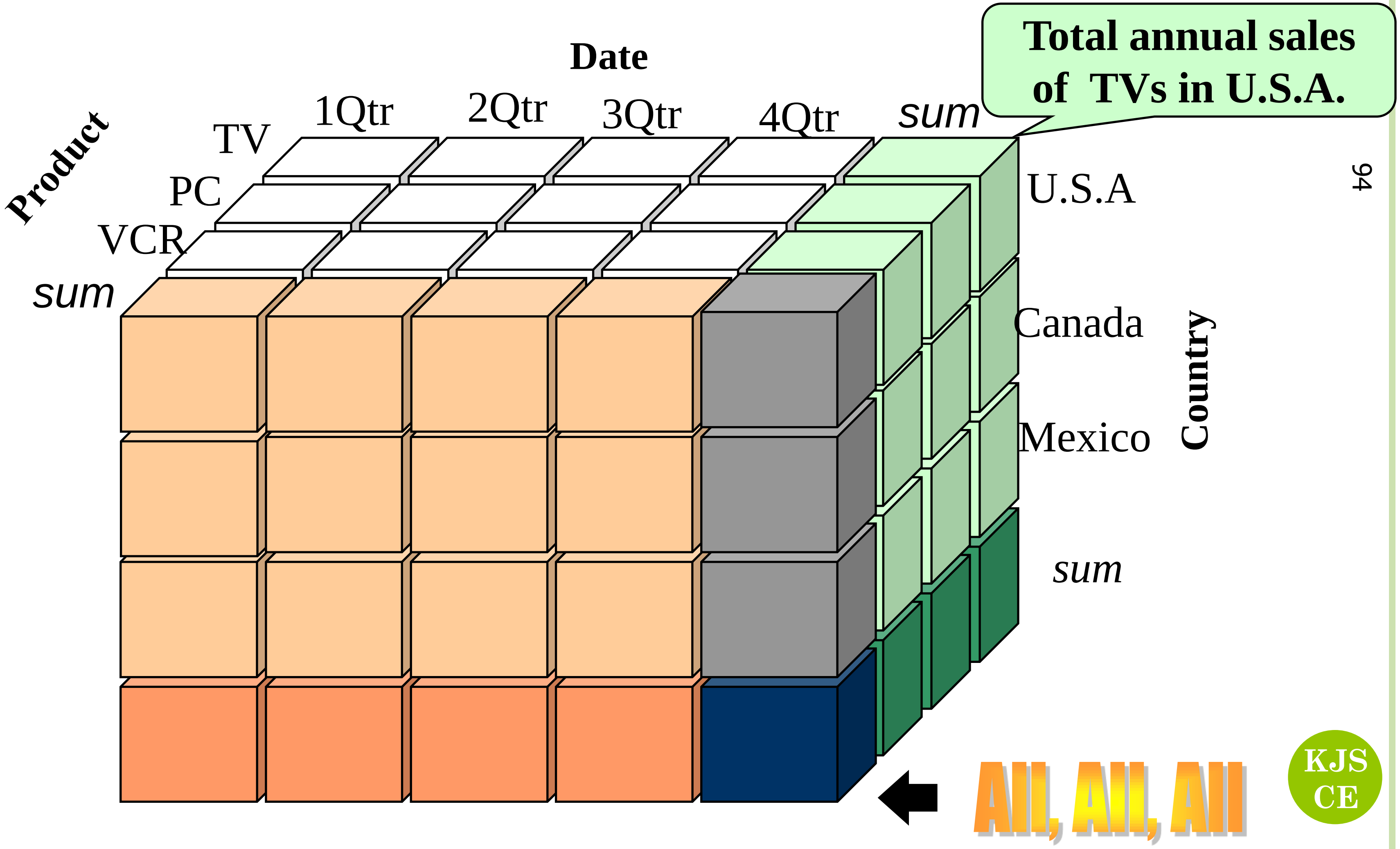


Industry
Category
Product

Region
Country
City
Office

Year
Quarter
Month
Week
Day

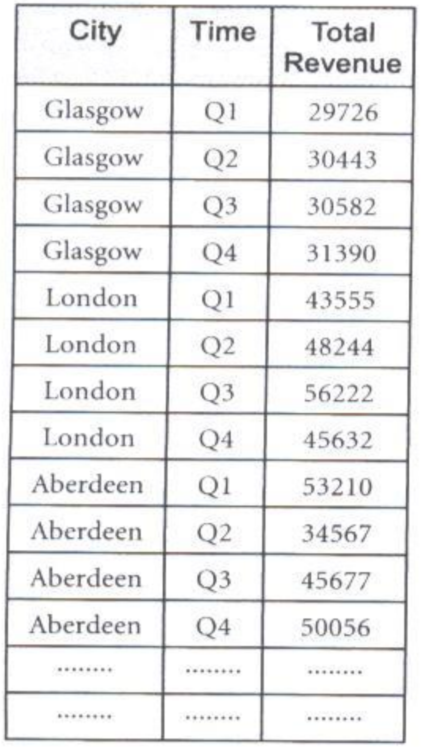
A Sample Data Cube



Total annual sales of TVs in U.S.A.

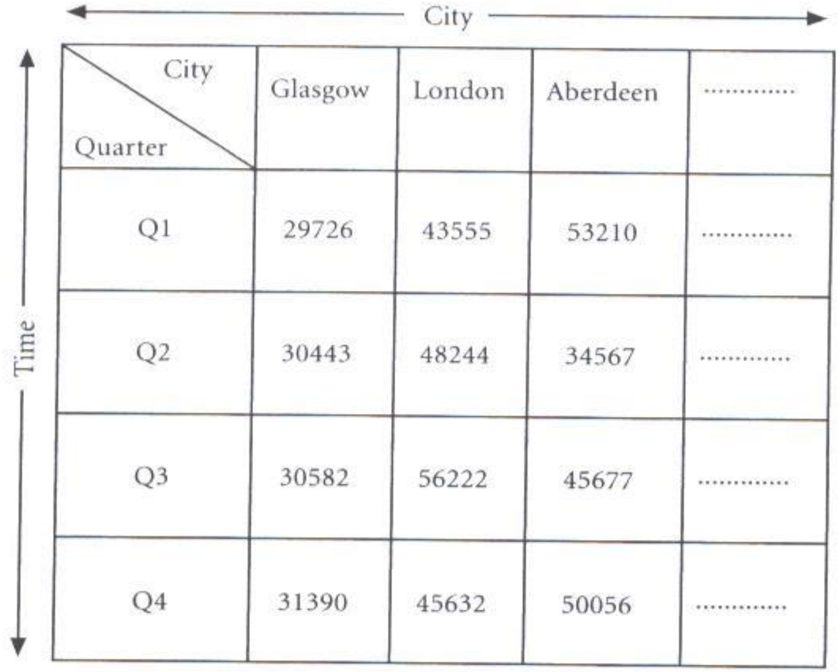
Representation of Multi-Dimensional Data

- OLAP database servers use multi-dimensional structures to store data and relationships between data.
- Multi-dimensional structures are best-visualized as cubes of data, and cubes within cubes of data. Each side of a cube is a dimension.



City	Time	Total Revenue
Glasgow	Q1	29726
Glasgow	Q2	30443
Glasgow	Q3	30582
Glasgow	Q4	31390
London	Q1	43555
London	Q2	48244
London	Q3	56222
London	Q4	45632
Aberdeen	Q1	53210
Aberdeen	Q2	34567
Aberdeen	Q3	45677
Aberdeen	Q4	50056
....	....	....
....	....	....

(a)

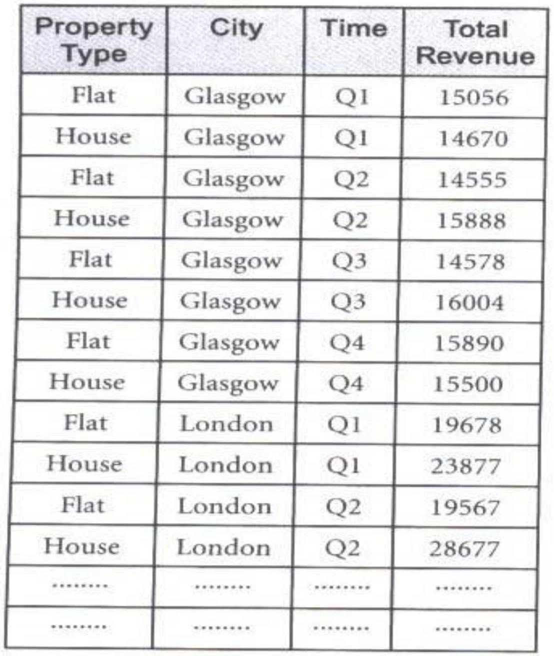


City Quarter	Glasgow	London	Aberdeen	....
Q1	29726	43555	53210	....
Q2	30443	48244	34567	....
Q3	30582	56222	45677	....
Q4	31390	45632	50056	....

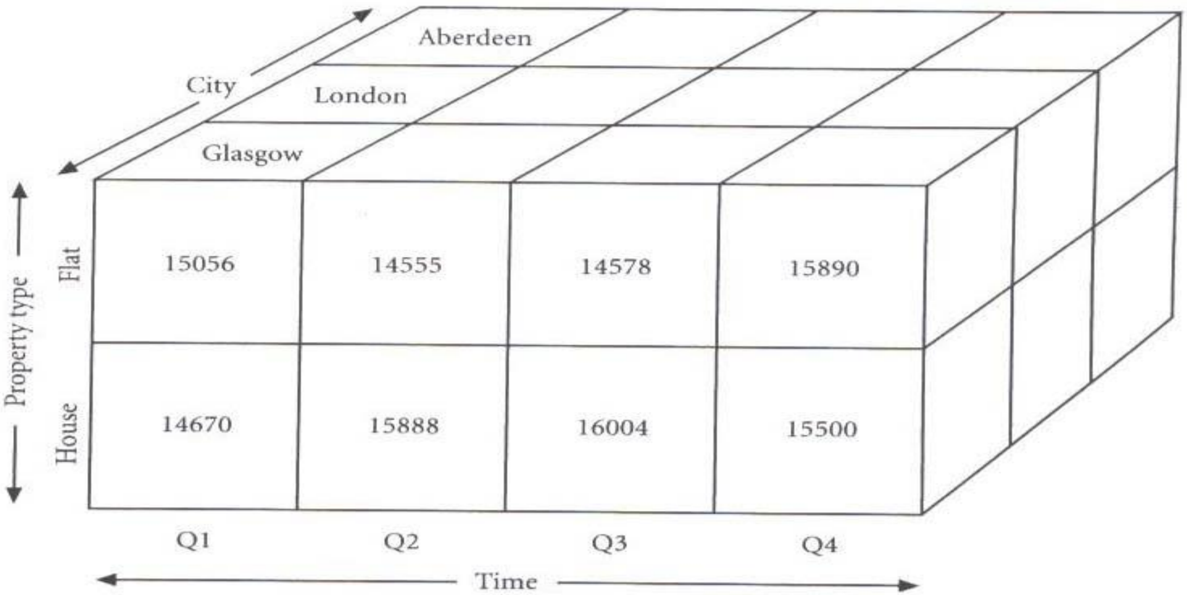
(b)

Representation of Multi-Dimensional Data

- Multi-dimensional databases are a compact and easy-to-understand way of visualizing and manipulating data elements that have many inter-relationships.
- The cube can be expanded to include another dimension, for example, the number of sales staff in each city.
- The response time of a multi-dimensional query depends on how many cells have to be added on-the-fly.
- As the number of dimensions increases, the number of cube's cells increases exponentially.



Property Type	City	Time	Total Revenue
Flat	Glasgow	Q1	15056
House	Glasgow	Q1	14670
Flat	Glasgow	Q2	14555
House	Glasgow	Q2	15888
Flat	Glasgow	Q3	14578
House	Glasgow	Q3	16004
Flat	Glasgow	Q4	15890
House	Glasgow	Q4	15500
Flat	London	Q1	19678
House	London	Q1	23877
Flat	London	Q2	19567
House	London	Q2	28677
....	....	....	....
....	....	....	....

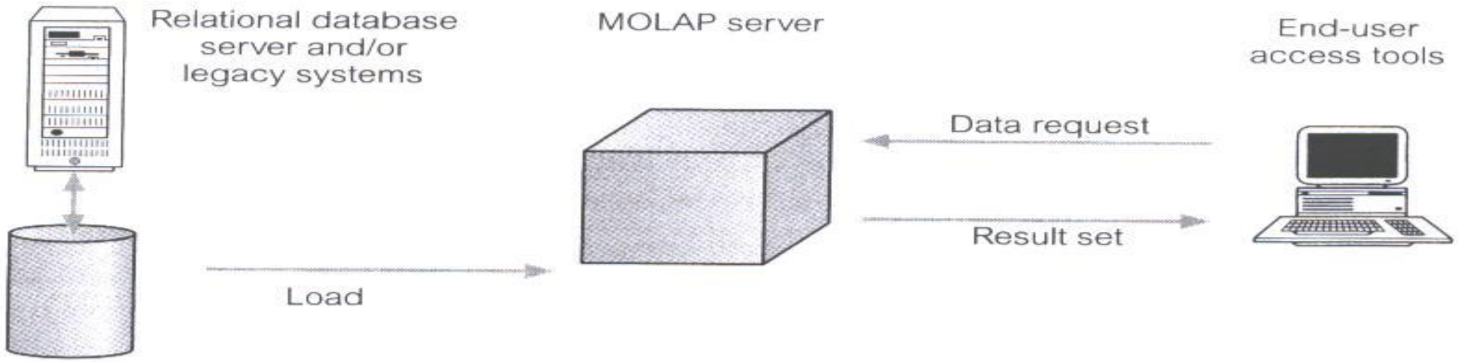


OLAP TOOLS - CATEGORIES

- OLAP tools are categorized according to the architecture used to store and process multi-dimensional data.
- The main categories of OLAP tools :
  - Multi-dimensional OLAP (MOLAP)
  - Relational OLAP (ROLAP)
  - Hybrid OLAP (HOLAP)

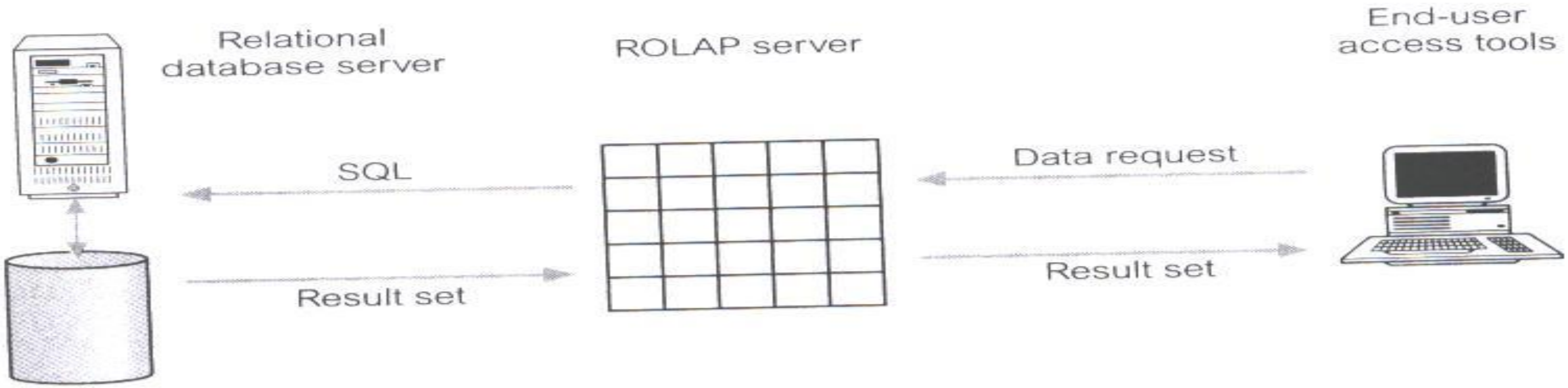
MULTI-DIMENSIONAL OLAP (MOLAP)

- MOLAP tools use specialized data structures and multi-dimensional database management systems (MDDBMS) to organize, navigate, and analyze data.
- To enhance query performance the data is typically aggregated and stored according to predicted usage.
- MOLAP data structures use array technology and efficient storage techniques that minimize the disk space requirements through sparse data management.
- The development issues associated with MOLAP:
  - Only a limited amount of data can be efficiently stored and analyzed.
  - Navigation and analysis of data are limited because the data is designed according to previously determined requirements.



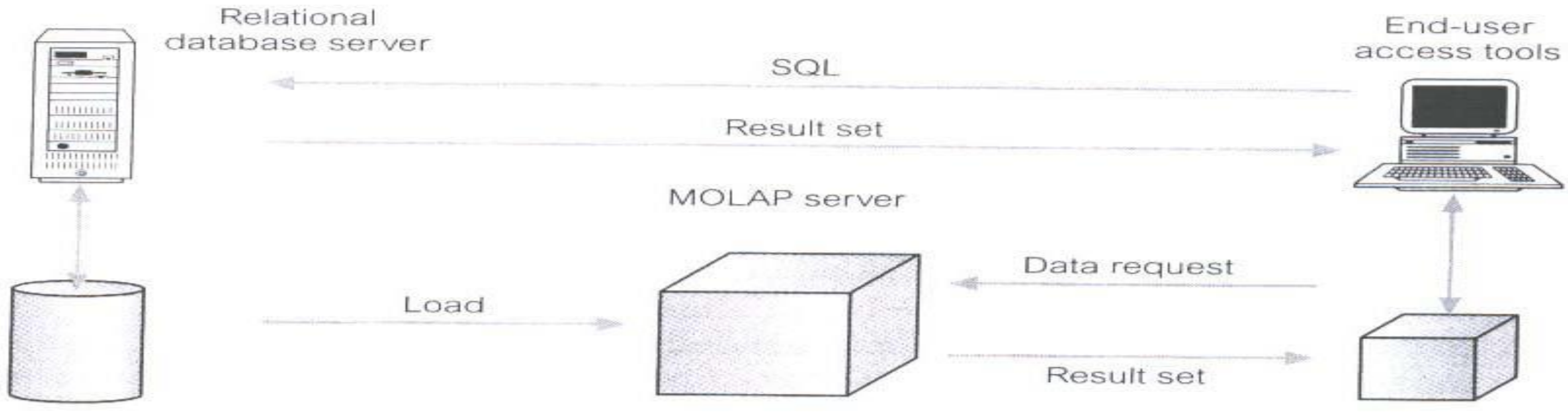
Relational OLAP (ROLAP)

- ROLAP is the fastest-growing type of OLAP tools.
- ROLAP supports RDBMS products through the use of a metadata layer, thus avoiding the requirement to create a static multi-dimensional data structure.
- This facilitates the creation of multiple multi-dimensional views of the two-dimensional relation.
- To improve performance, some ROLAP products have enhanced SQL engines to support the complexity of multi-dimensional analysis, while others recommend, or require, the use of highly denormalized database designs such as the star schema.
- The development issues associated with ROLAP technology:
  - Performance problems associated with the processing of complex queries that require multiple passes through the relational data.
  - Development of middleware to facilitate the development of multi-dimensional applications.



HYBRID OLAP (HOLAP)

- HOLAP tools provide limited analysis capability, either directly against RDBMS products, or by using an intermediate MOLAP server.
- HOLAP tools deliver selected data directly from DBMS or via MOLAP server to the desktop (or local server) in the form of data cube, where it is stored, analyzed, and maintained locally is the fastest-growing type of OLAP tools.
- The issues associated with HOLAP tools:
  - The architecture results in significant data redundancy and may cause problems for networks that support many users.
  - Ability of each user to build a custom data cube may cause a lack


Relational
database server
End-user
access tools
SQL
Result set
MOLAP server
Data request
Load
Result set