



DEPARTEMEN SISTEM INFORMASI
Fak. Teknologi Elektro & Informatika Cerdas



SE235103

MANAJEMEN DATA, INFORMASI, DAN KONTEN

Chapter 06

Data Warehousing

Prof. Ir. Arif Djunaidy, M.Sc., Ph.D.
arif.djunaidy@its.ac.id
adjunaidy@gmail.com

Learning Objectives & Book Reading

Learning Objectives - To Understand:

- Definition of related key terms
- Major reasons of the need for data warehousing
- Data warehouse architecture and multi-dimensional OLAP

Book reading:

- Data Management Body of Knowledge, Chapter 11



Outline

- History of Data Warehouses and Merging Information Resources
- What is a Data Warehouse?
- Data Warehouse and OLAP
- Data Warehouse Architectures
- Data Warehousing Problems and Issues
- Data Model and Design

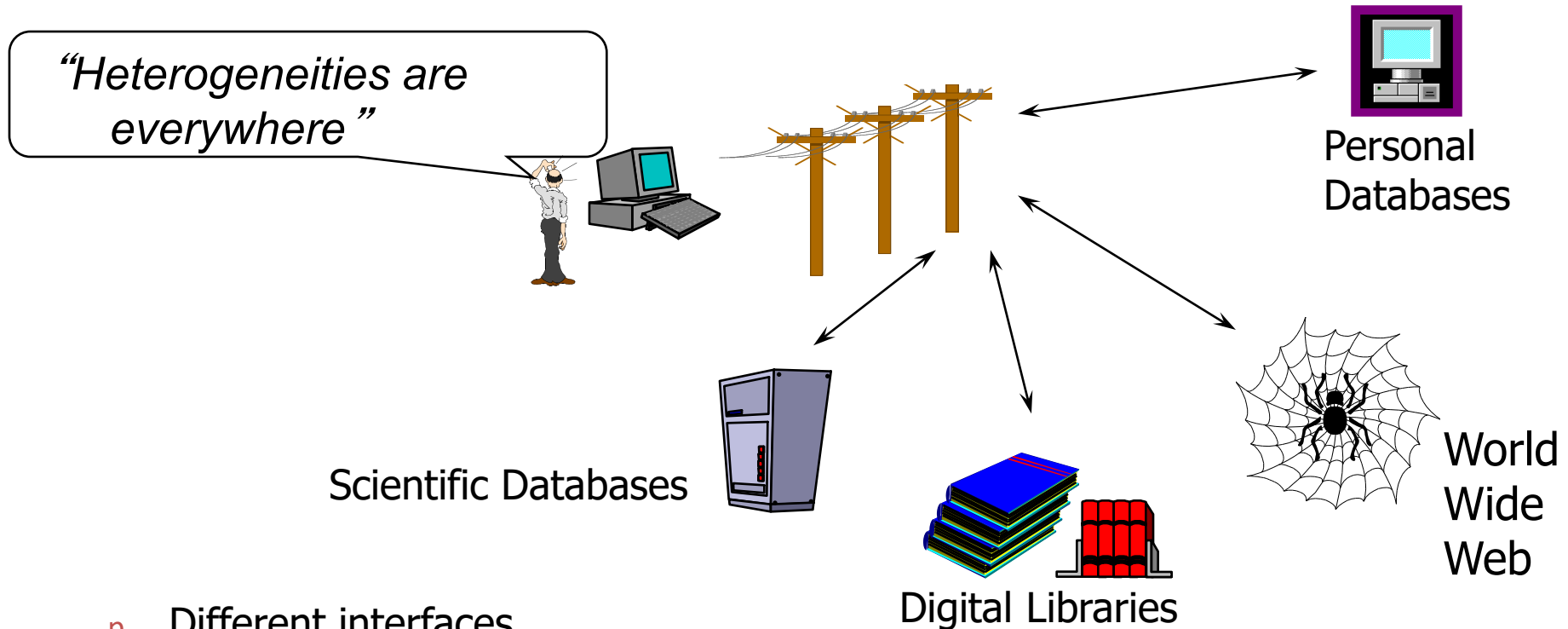


Motivation

- Increasingly, organizations are analyzing current and historical data to identify useful patterns and support business strategies.
- Emphasis is on complex, interactive, exploratory analysis of very large datasets created by integrating data from across all parts of an enterprise; data is fairly static.
 - Contrast such **On-Line Analytic Processing (OLAP)** with traditional **On-line Transaction Processing (OLTP)**: mostly long queries, instead of short update Xacts.



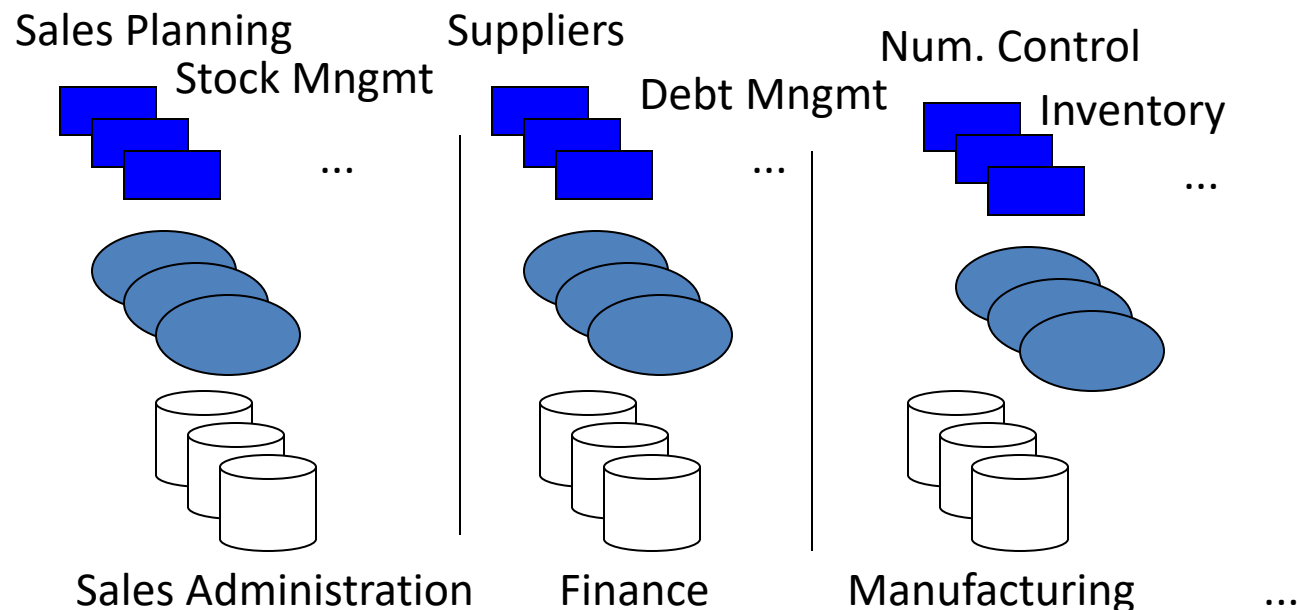
Problem: Heterogeneous Information Sources



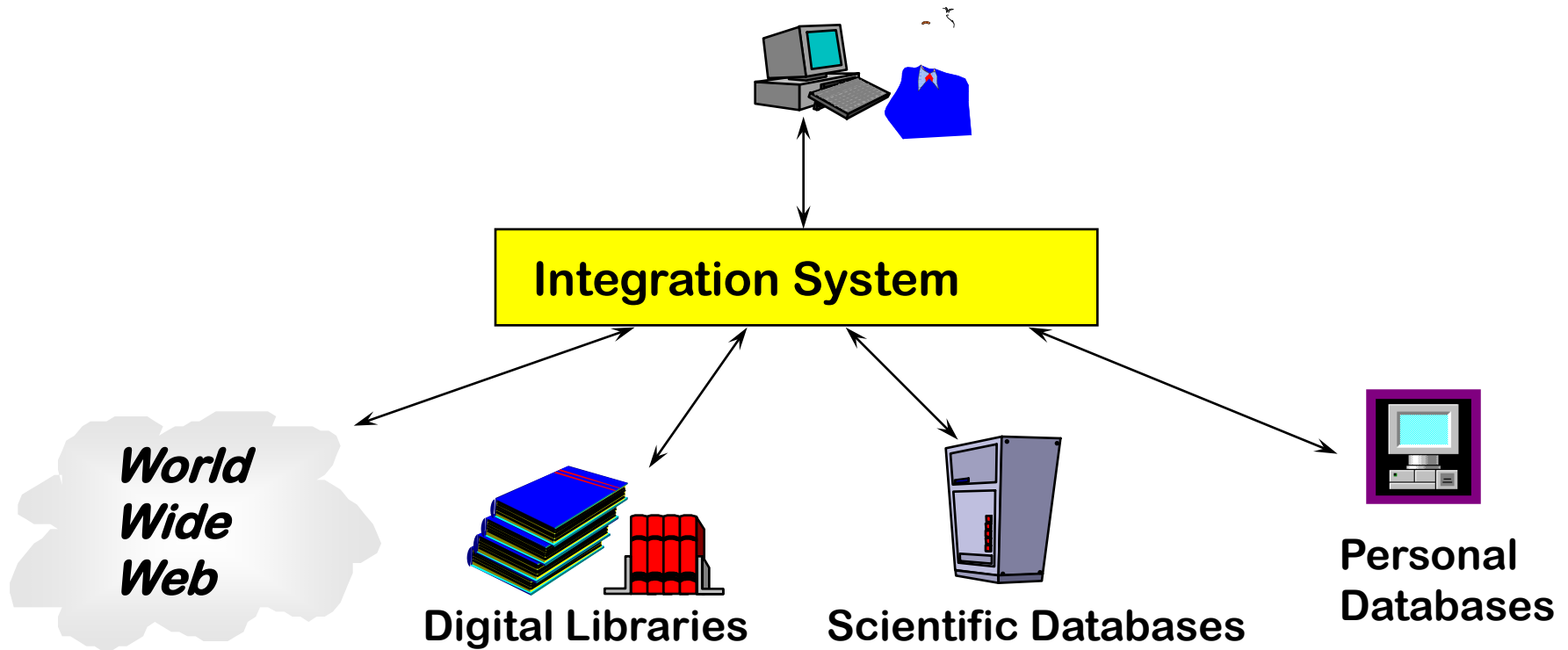
- p Different interfaces
- p Different data representations
- p Duplicate and inconsistent information

Problem: Data Management in Large Enterprises

- Vertical fragmentation of informational systems (vertical stove pipes)
- Result of application (user)-driven development of operational systems



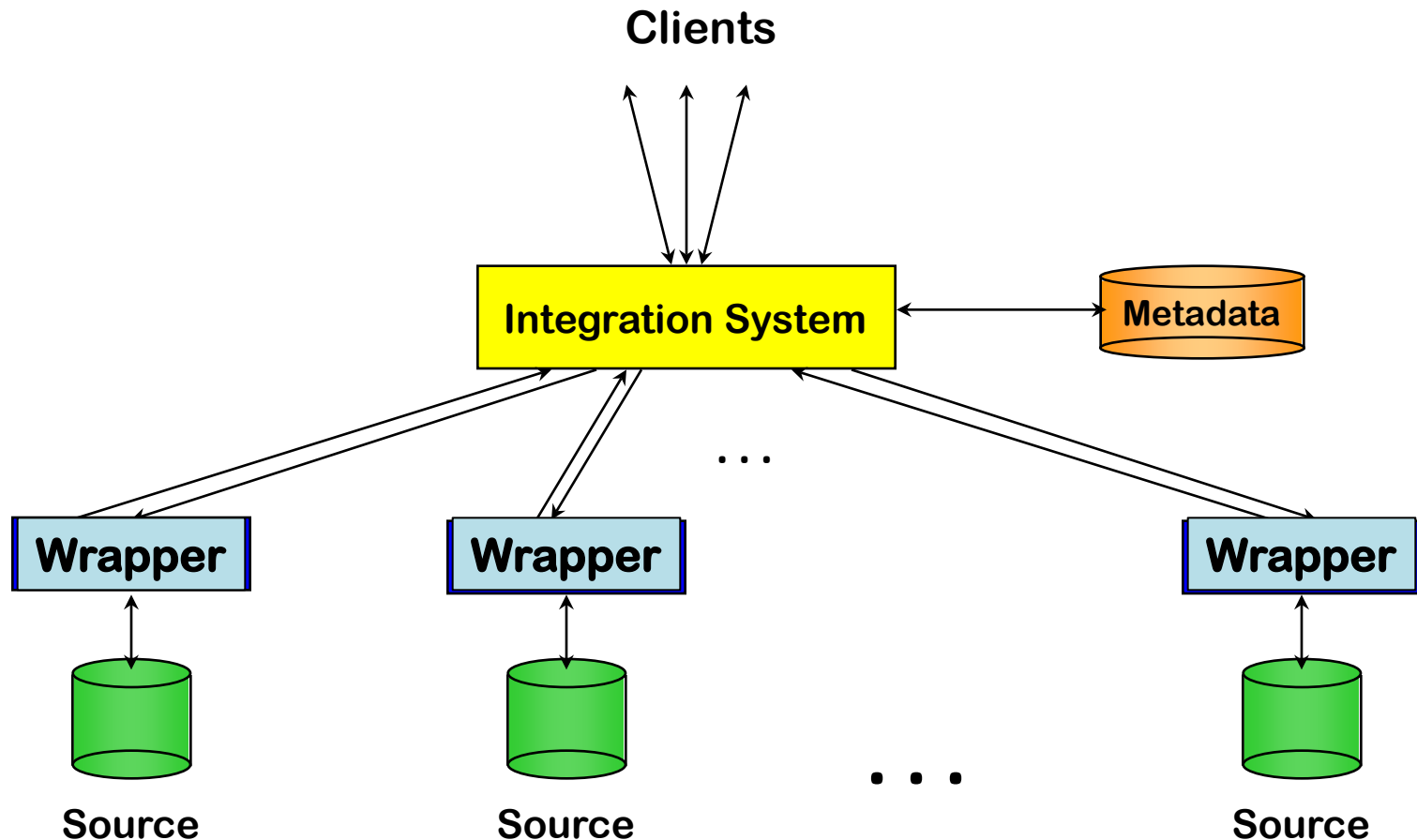
Goal: Unified Access to Data



- Collects and combines information
- Provides integrated view, uniform user interface
- Supports sharing

The Traditional Research Approach

- Query-driven (lazy, on-demand)



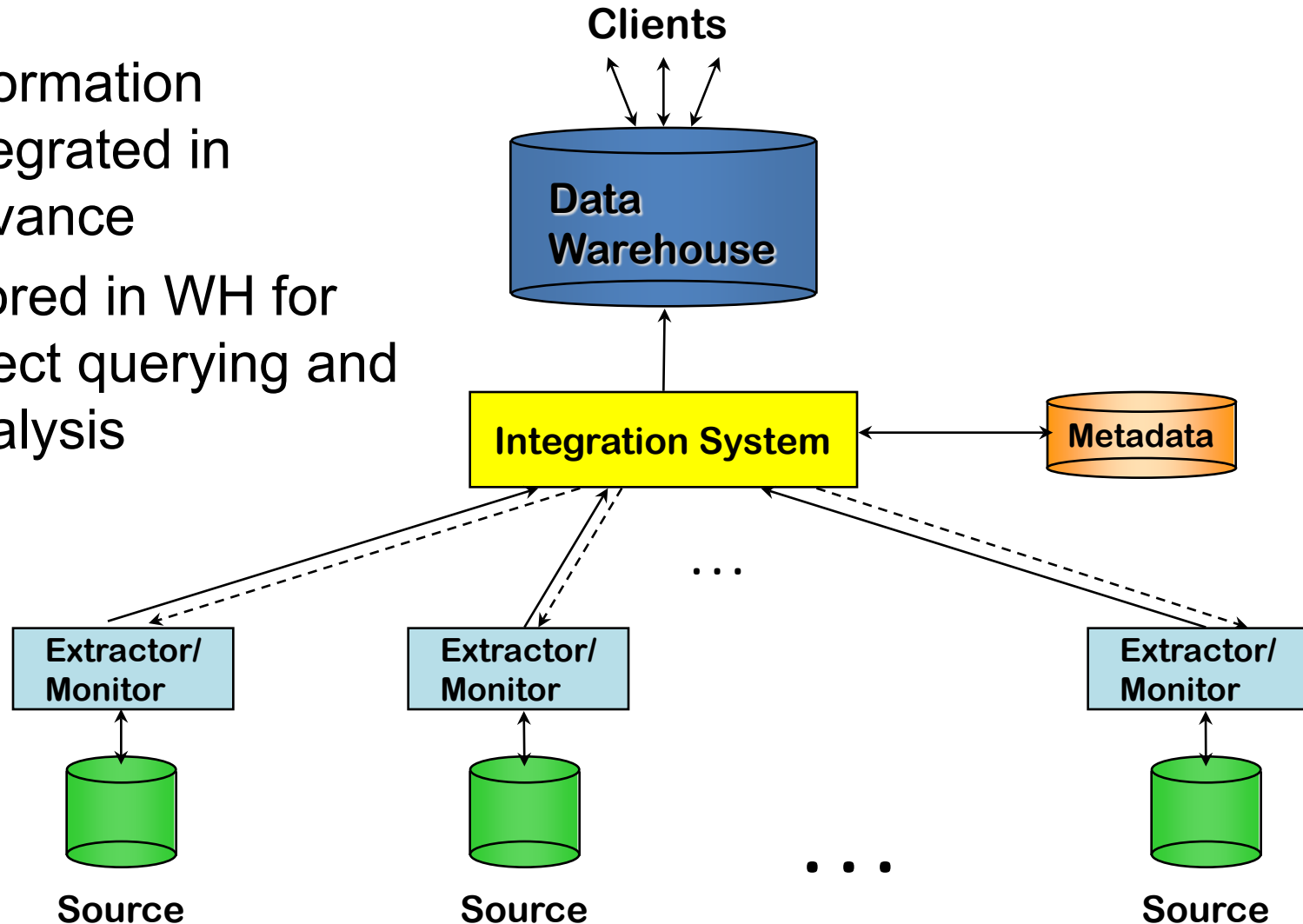
Disadvantages of Query-Driven Approach

- Delay in query processing
 - Slow or unavailable information sources
 - Complex filtering and integration
- Inefficient and potentially expensive for frequent queries
- Competes with local processing at sources
- Hasn't caught on in industry



The Warehousing Approach

- Information integrated in advance
- Stored in WH for direct querying and analysis



Advantages of Warehousing Approach

- High query performance
 - But not necessarily most current information
- Doesn't interfere with local processing at sources
 - Complex queries at warehouse
 - OLTP at information sources
- Information copied at warehouse
 - Can modify, annotate, summarize, restructure, etc.
 - Can store historical information
 - Security, no auditing
- **Has** caught on in industry

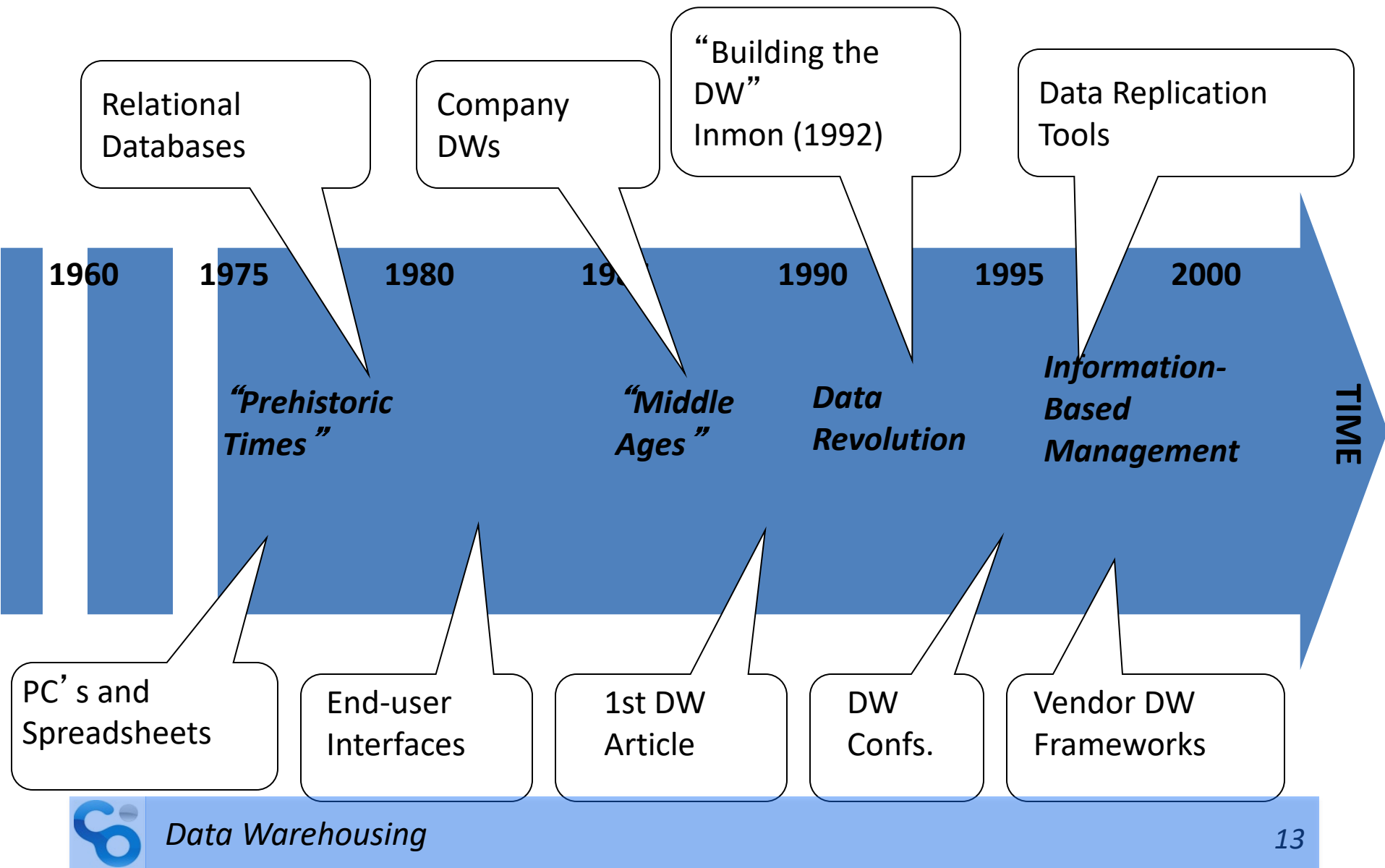


Not Either-Or Decision

- Query-driven approach still better for
 - Rapidly changing information
 - Rapidly changing information sources
 - Truly vast amounts of data from large numbers of sources
 - Clients with unpredictable needs



Data Warehouse Evolution



Outline

- History of Data Warehouses and Merging Information Resources
- **What is a Data Warehouse?**
- Data Warehouse and OLAP
- Data Warehouse Architectures
- Data Warehousing Problems and Issues
- Data Model and Design Issues



What is Data Warehouse?

- Defined in many different ways.
 - 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”
- Data warehousing:
 - The process of constructing and using data warehouses



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**.



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.



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—Non-Volatile

- 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*.



Outline

- History of Data Warehouses and Merging Information Resources
- What is a Data Warehouse?
- **Data Warehouse dan OLAP**
- Data Warehouse Architectures
- Data Warehousing Problems and Issues
- Data Model and Design Issues



Why OLAP & OLTP don't mix (1)

Different performance requirements

- Transaction processing (OLTP):
 - Fast response time important (< 1 second)
 - Data must be up-to-date, consistent at all times
- Data analysis (OLAP):
 - Queries can consume lots of resources
 - Can saturate CPUs and disk bandwidth
 - Operating on static “snapshot” of data usually OK
- OLAP can “crowd out” OLTP transactions
 - Transactions are slow → unhappy users
- Example:
 - Analysis query asks for sum of all sales
 - Acquires lock on sales table for consistency
 - New sales transaction is blocked



Why OLAP & OLTP don't mix (2)

Different data modeling requirements

- Transaction processing (OLTP):
 - **Normalized** schema for consistency
 - Complex data models, many tables
 - Limited number of **standardized queries and updates**
- Data analysis (OLAP):
 - **Simplicity** of data model is important
 - Allow semi-technical users to formulate **ad hoc queries**
 - **De-normalized** schemas are common
 - Fewer joins → improved query performance
 - Fewer tables → schema is easier to understand

Why OLAP & OLTP don't mix (3)

Analysis requires data from many sources

- An OLTP system targets one specific process
 - For example: ordering from an online store
- OLAP integrates data from different processes
 - Combine sales, inventory, and purchasing data
 - Analyze experiments conducted by different labs
- OLAP often makes use of historical data
 - Identify long-term patterns
 - Notice changes in behavior over time
- Terminology, schemas vary across data sources
 - Integrating data from disparate sources is a major challenge

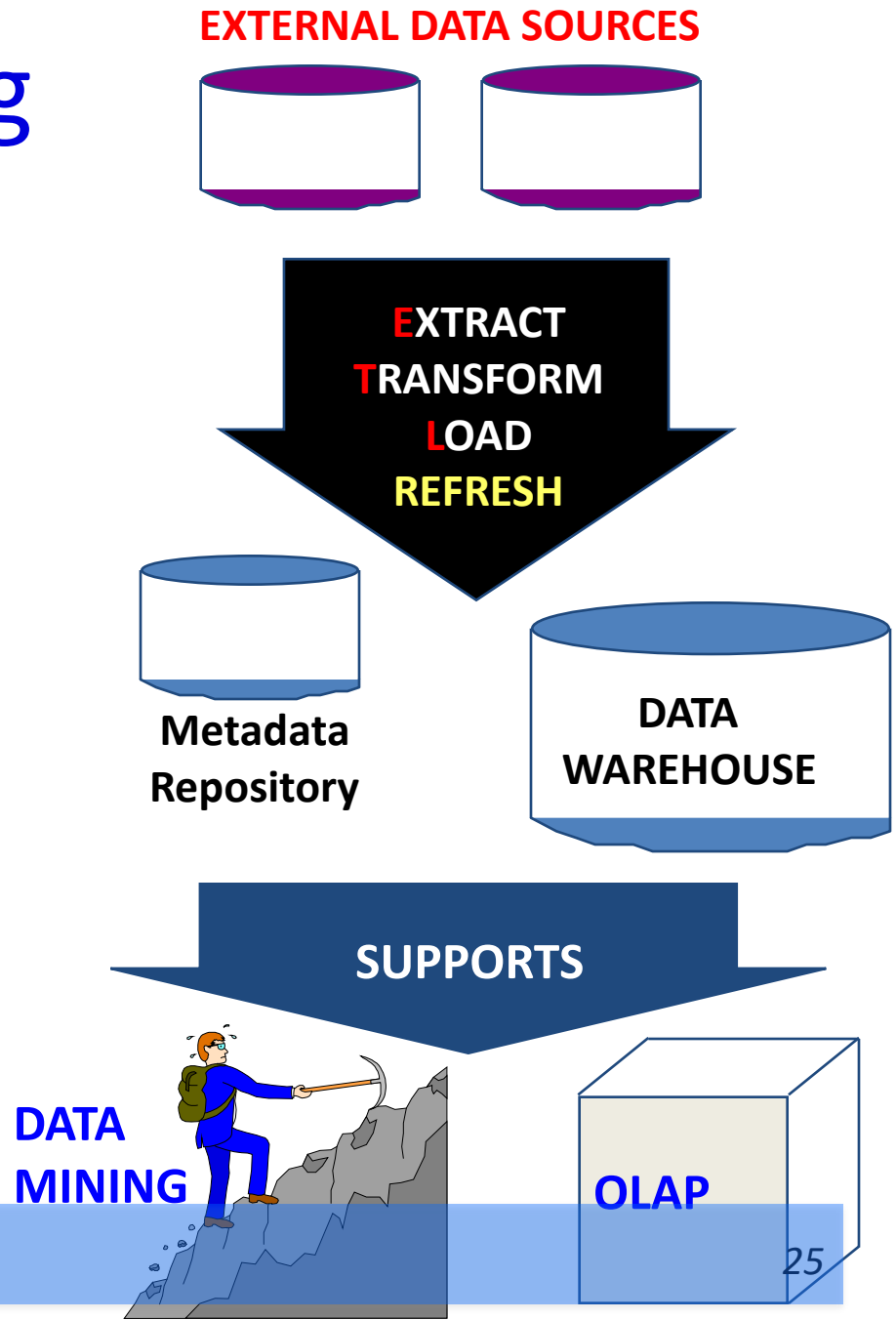
Three Complementary Trends

- **Data Warehousing:** Consolidate data from many sources in one large repository.
 - Loading, periodic synchronization of replicas.
 - Semantic integration.
- **OLAP:**
 - Complex SQL queries and views.
 - Queries based on spreadsheet-style operations and “multidimensional” view of data.
 - Interactive and “online” queries.
- **Data Mining:** Exploratory search for interesting trends and anomalies.



Data Warehousing

- Integrated data spanning long time periods, often augmented with summary information.
- Several gigabytes to terabytes common.
- Interactive response times expected for complex queries; ad-hoc updates uncommon.

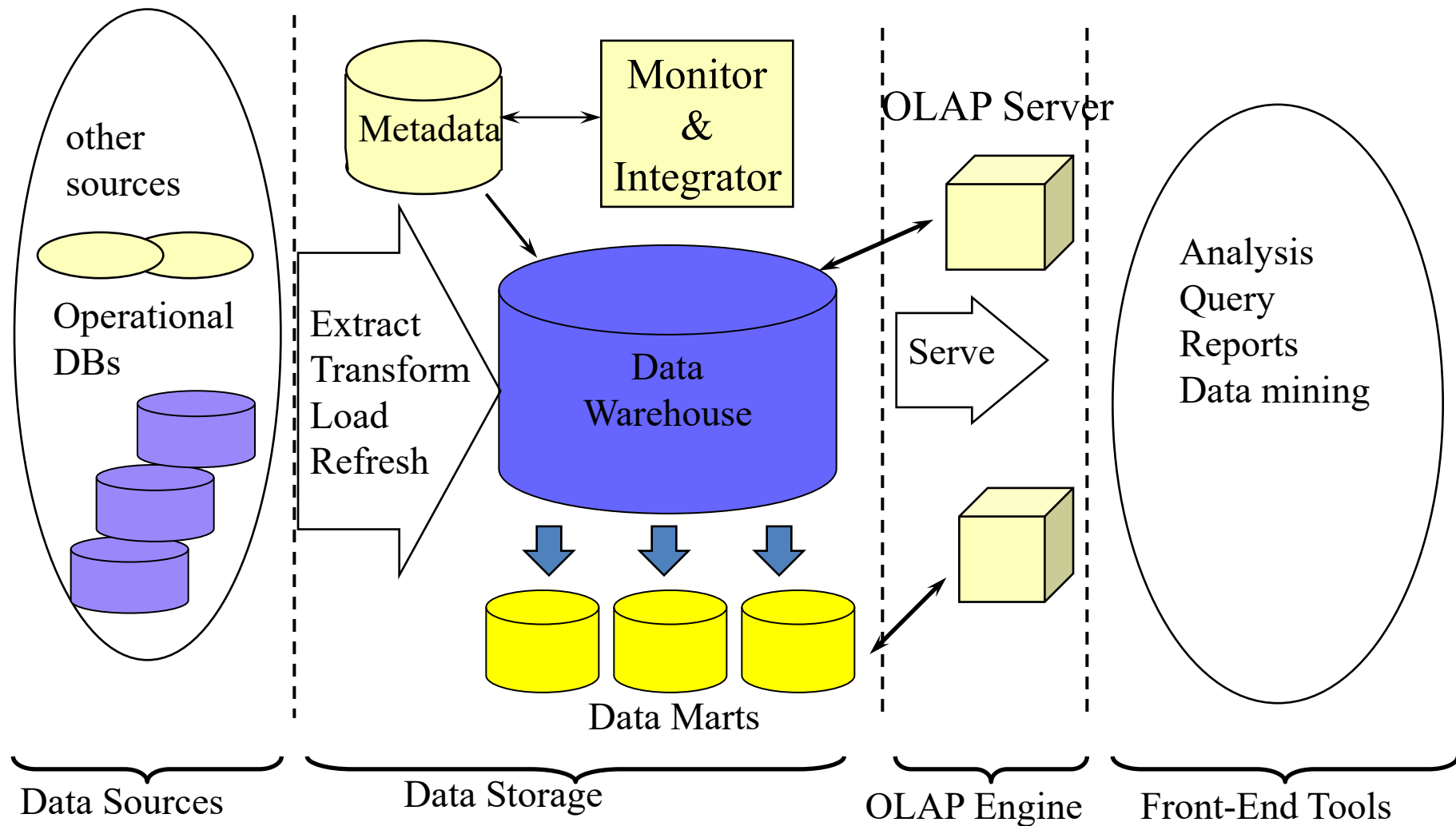


Outline

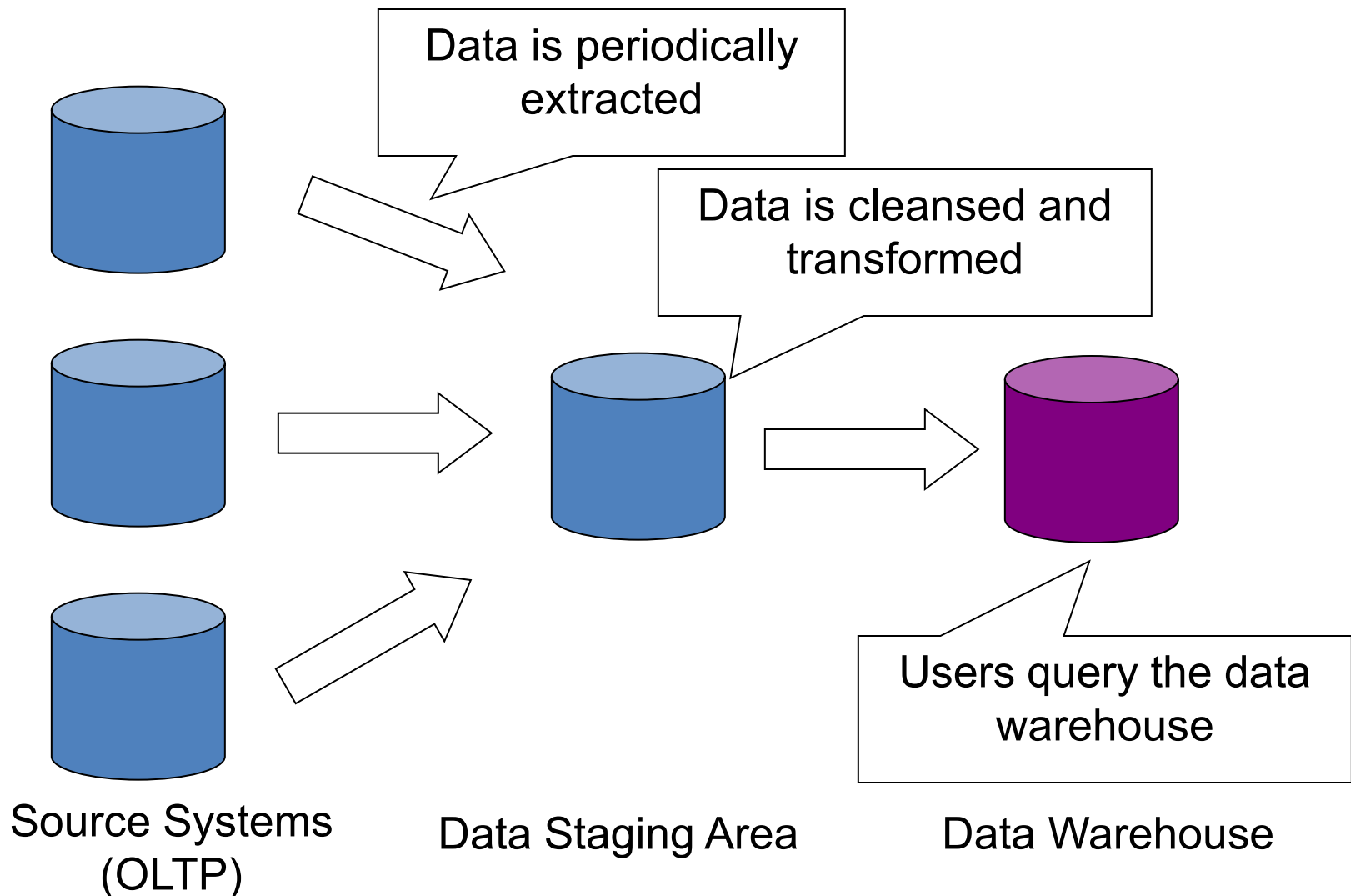
- History of Data Warehouses and Merging Information Resources
- What is a Data Warehouse?
- Data Warehouse dan OLAP
- **Data Warehouse Architectures**
- Data Warehousing Problems and Issues
- Data Model and Design Issues



Data Warehouse Multi-Tiered Architecture



Loading the Data Warehouse

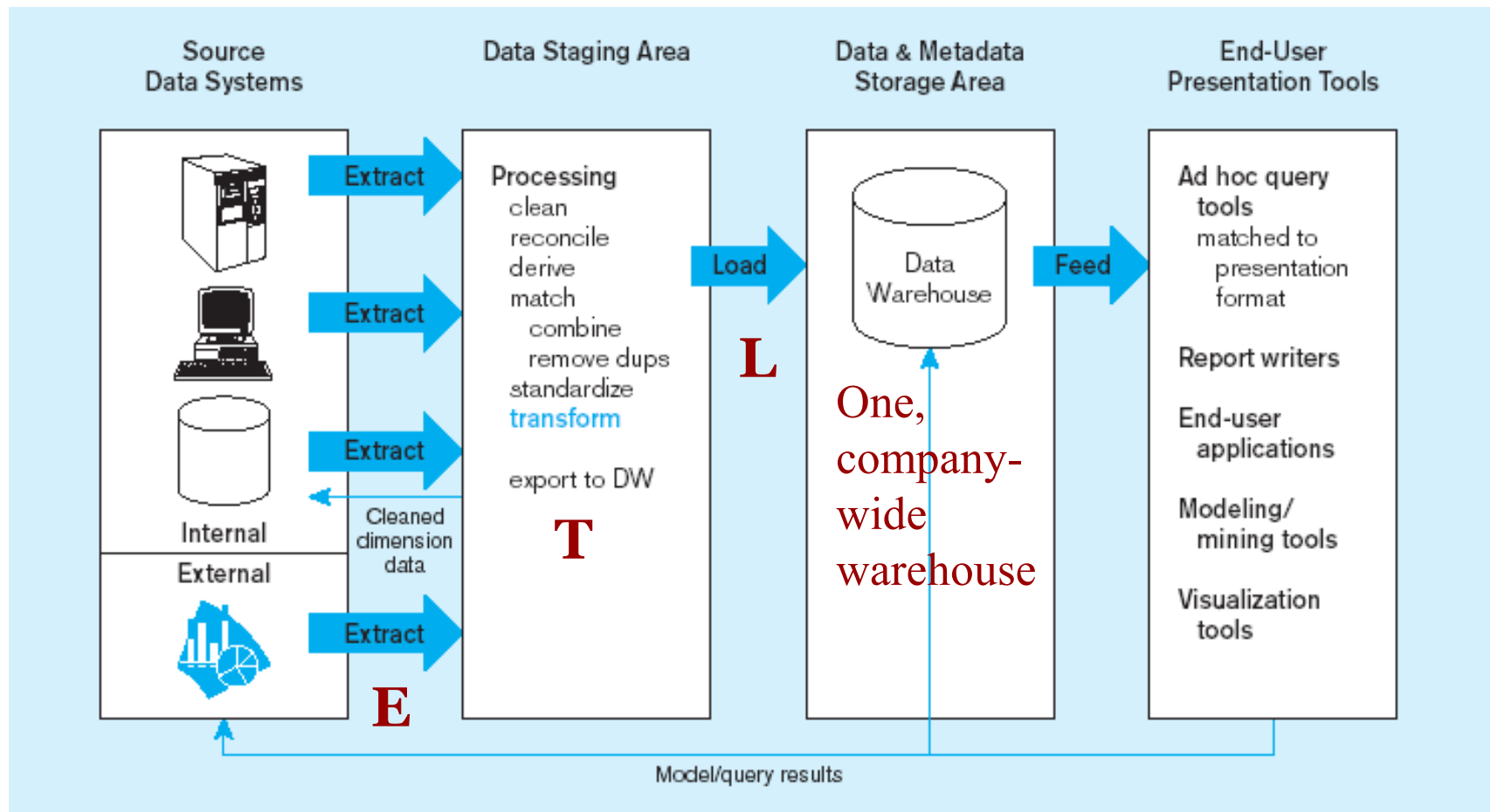


Terminology: ETL

- ETL = **E**xtraction, **T**ransformation, & **L**oad
- **Extraction**: Get the data out of the source systems
- **Transformation**: Convert the data into a useful format for analysis
- **Load**: Get the data into the data warehouse (...and build indexes, materialized views, etc.)



Generic two-level data warehousing architecture

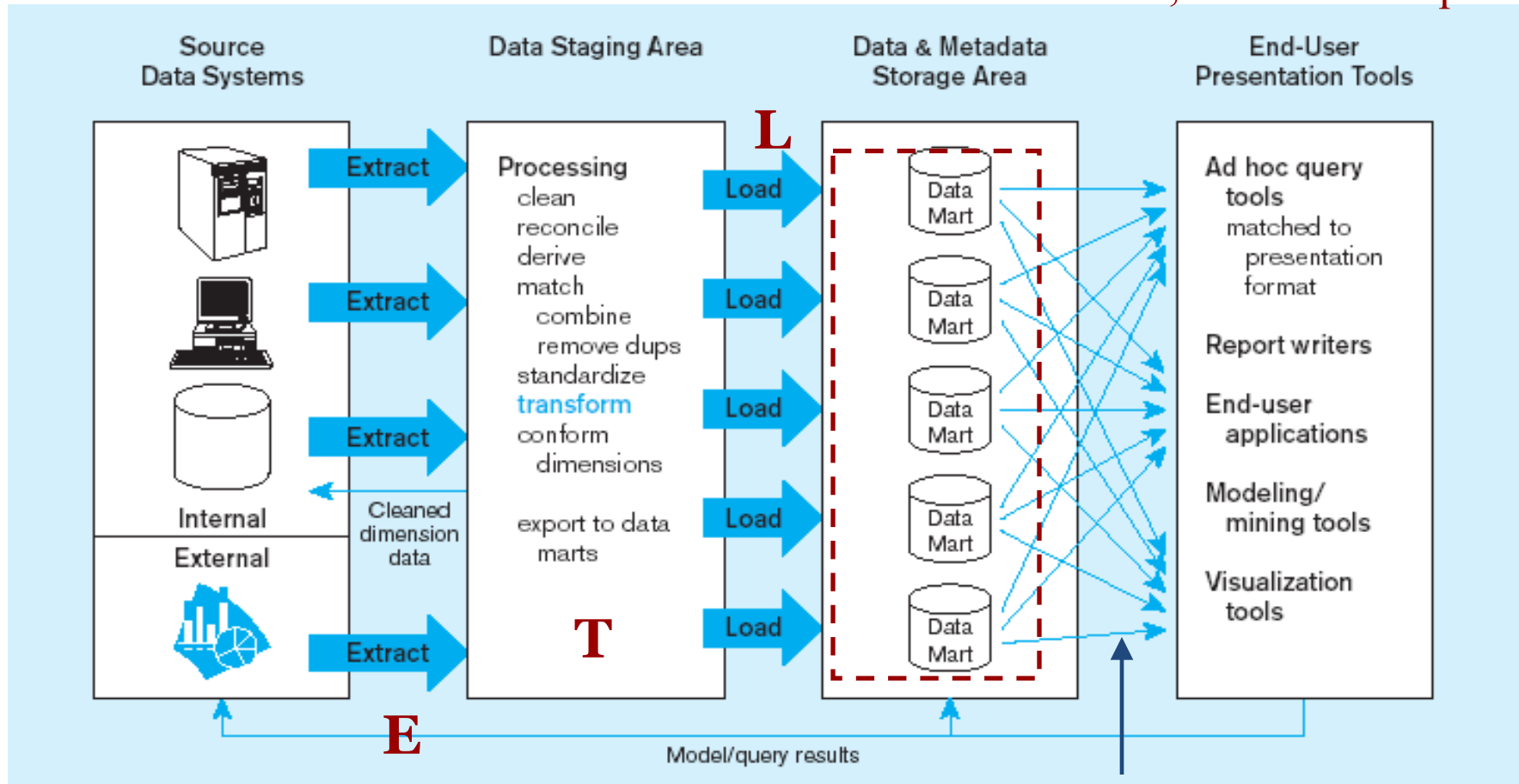


Periodic extraction → data is not completely current in warehouse

Independent data mart data architecture

Data marts:

Mini-warehouses, limited in scope

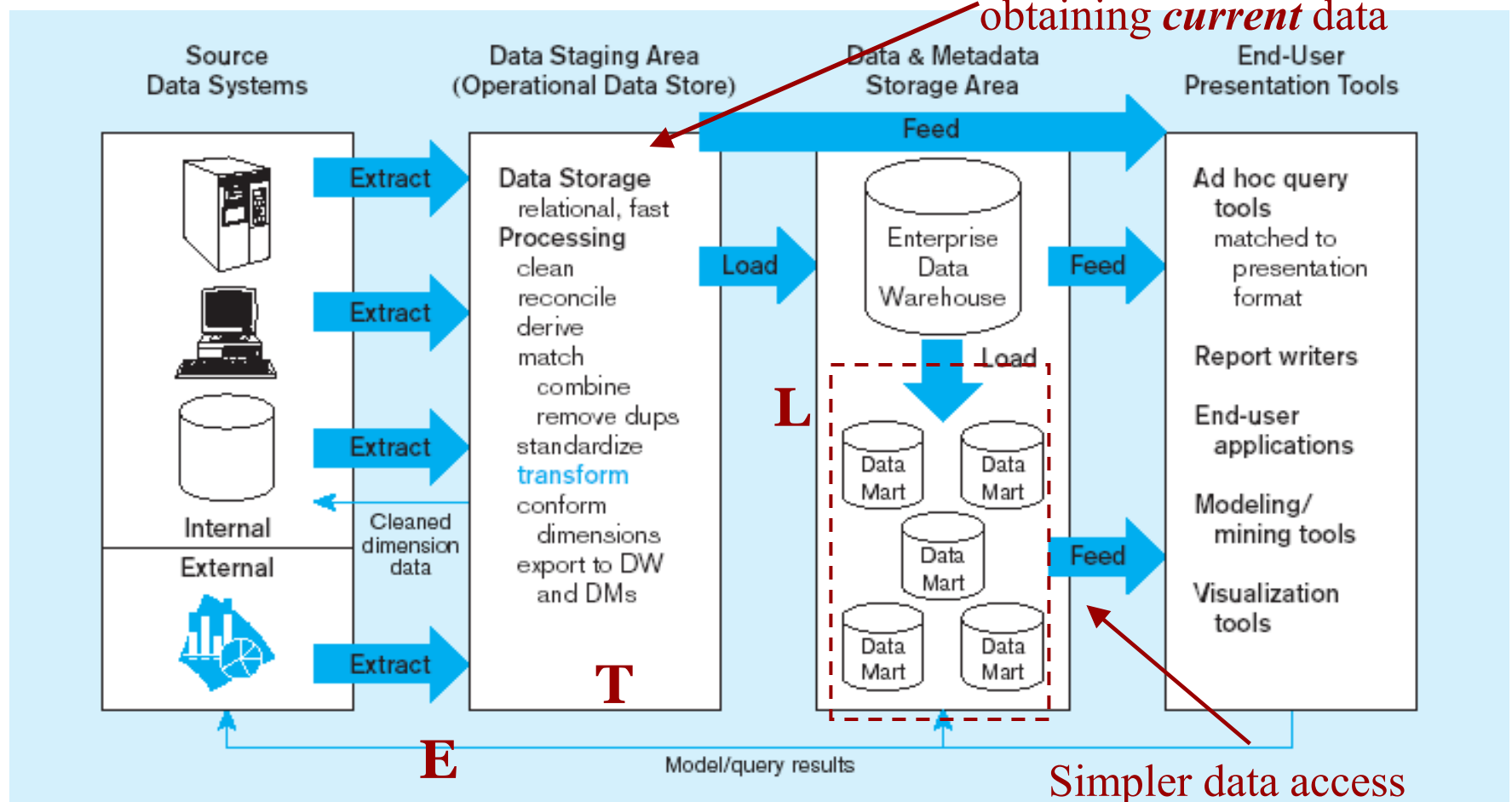


Separate ETL for each
independent data mart

Data access complexity
due to *multiple* data marts



Dependent data mart with operational data store: a three-level architecture



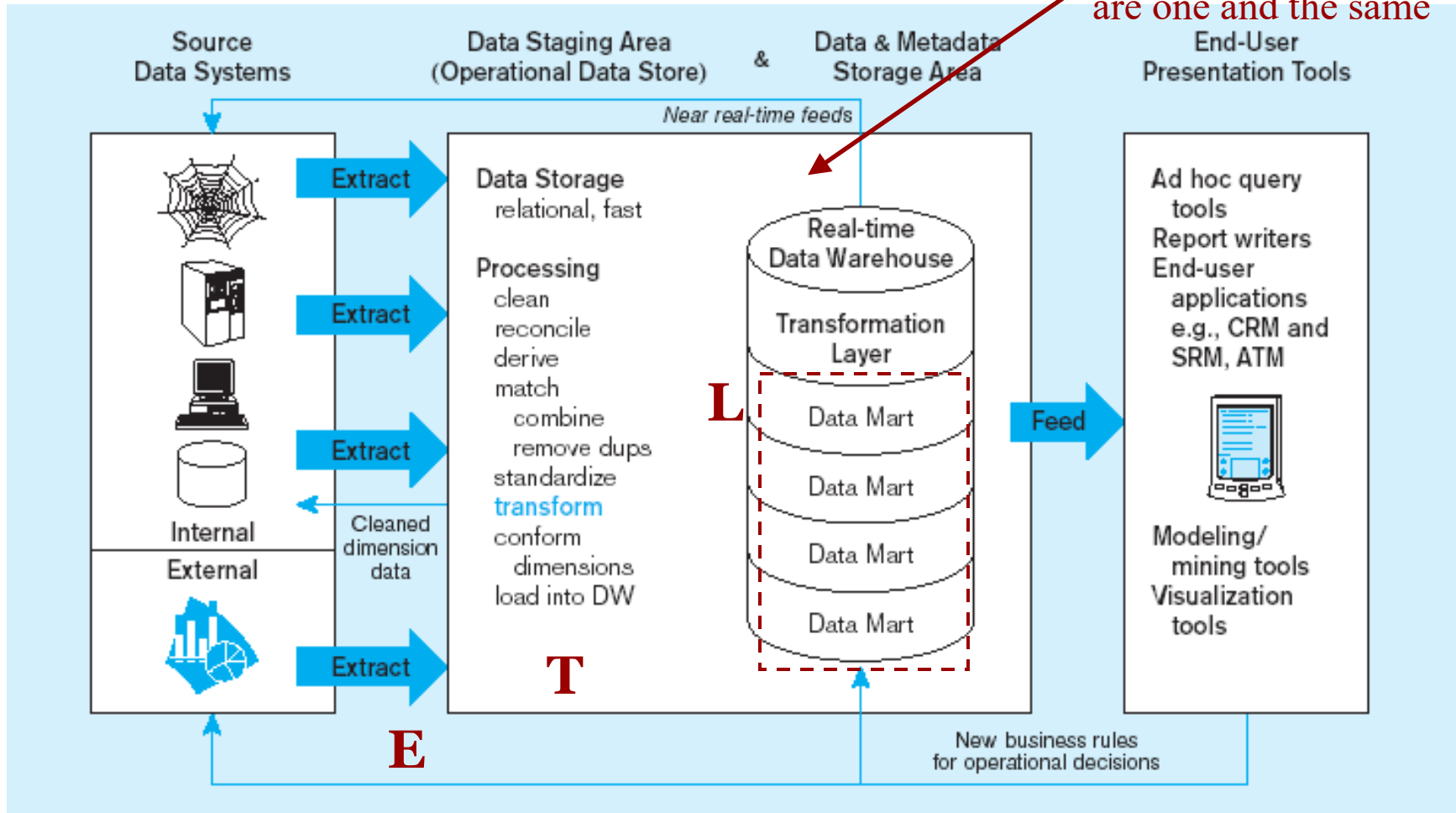
Single ETL for
enterprise data warehouse (EDW)

Dependent data marts
loaded from EDW



Logical data mart and real time warehouse architecture

ODS and data warehouse
are one and the same



Near real-time ETL for
Data Warehouse

Data marts are NOT separate databases, but logical *views* of the data warehouse

→ Easier to create new data marts



Outline

- History of Data Warehouses and Merging Information Resources
- What is a Data Warehouse?
- Data Warehouse dan OLAP
- Data Warehouse Architectures
- **Data Warehousing Problems and Issues**
- Data Model and Design Issues

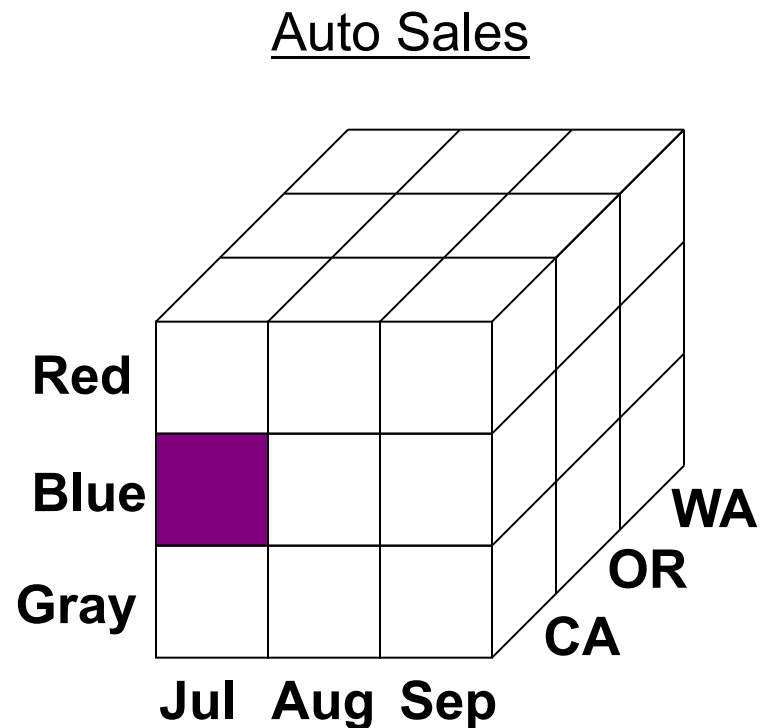


Warehousing Issues

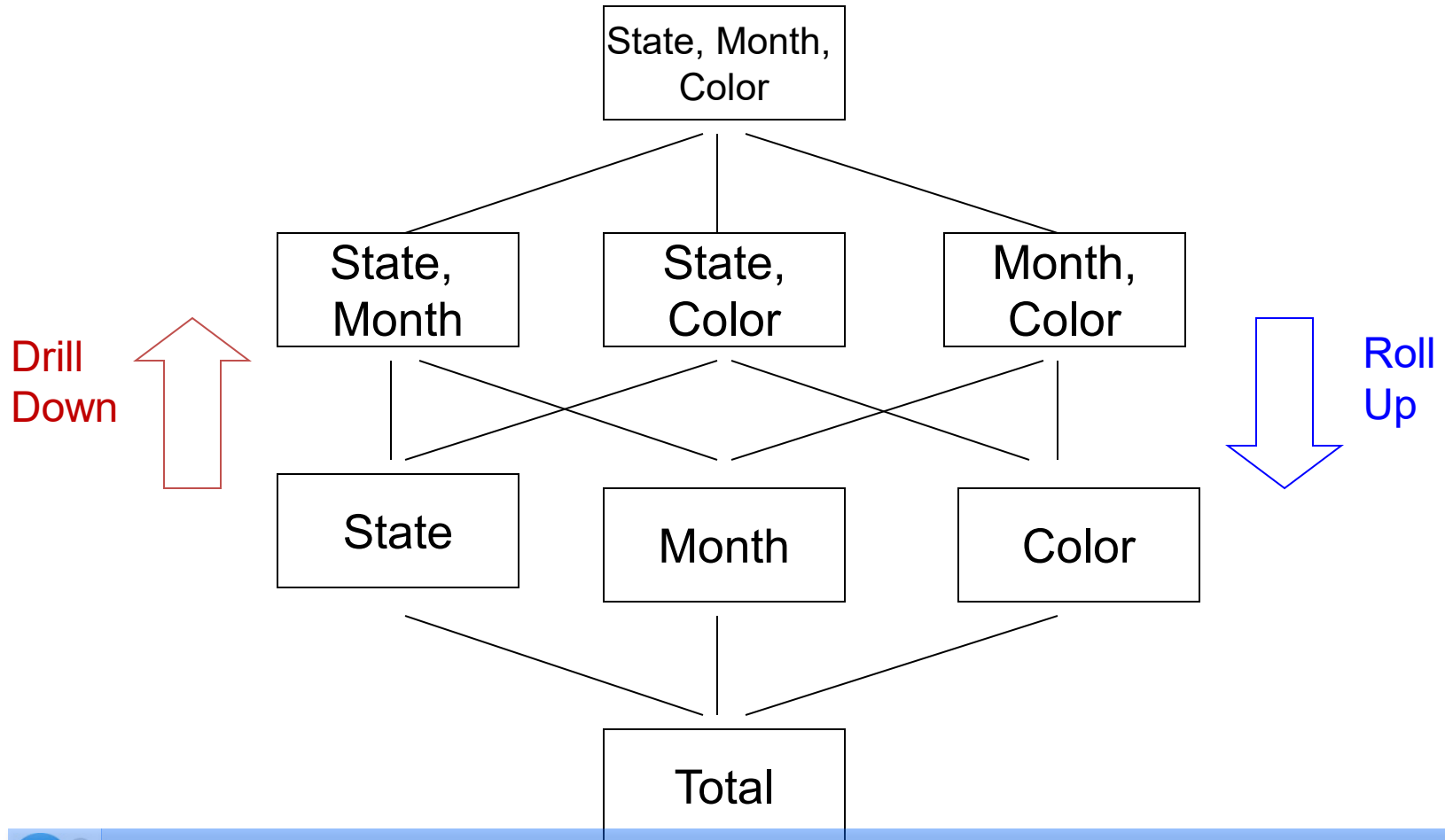
- **Semantic Integration:** When getting data from multiple sources, must eliminate mismatches, e.g., different currencies, schemas.
- **Heterogeneous Sources:** Must access data from a variety of source formats and repositories.
 - Replication capabilities can be exploited here.
- **Load, Refresh, Purge:** Must load data, periodically refresh it, and purge too-old data.
- **Metadata Management:** Must keep track of source, loading time, and other information for all data in the warehouse.

Data Cube

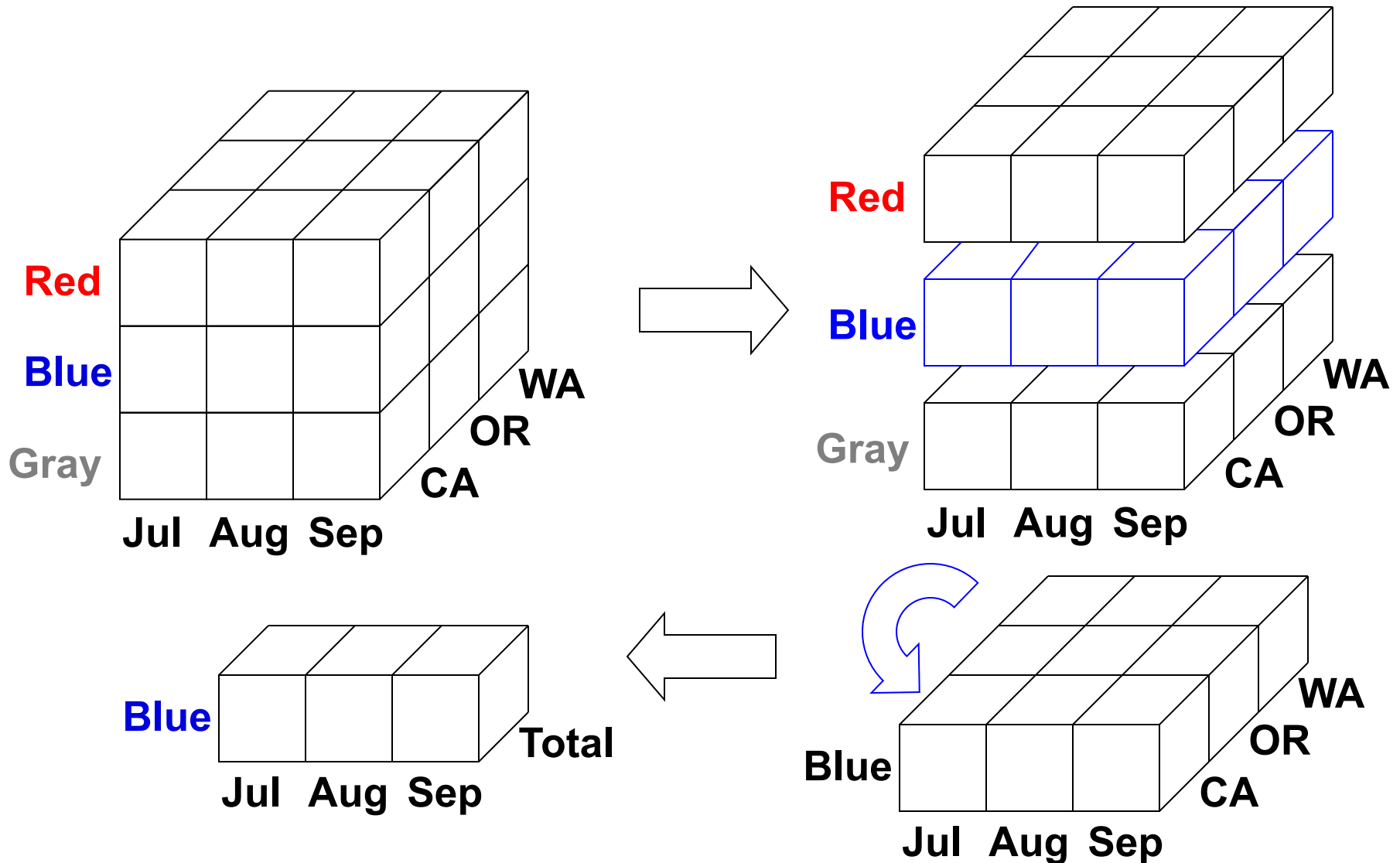
- Axes of the cube represent attributes of the data records
 - Generally discrete-valued / categorical
 - e.g. color, month, state
 - Called **dimensions**
- Cells hold aggregated measurements
 - e.g. total \$ sales, number of autos sold
 - Called **facts**
- Real data cubes have $\gg 3$ dimensions



Data Cube Lattice



Slicing and Dicing



Querying the Data Cube

Number of Autos Sold

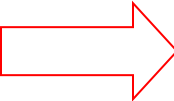
	CA	OR	WA	Total
Jul	45	33	30	108
Aug	50	36	42	128
Sep	38	31	40	109
Total	133	100	112	345

- Cross-tabulation
 - “Cross-tab” for short
 - Report data grouped by 2 dimensions
 - Aggregate across other dimensions
 - Include subtotals
- Operations on a cross-tab
 - Roll up (further aggregation)
 - Drill down (less aggregation)

Roll Up and Drill Down

Number of Autos Sold

	CA	OR	WA	Total
Jul	45	33	30	108
Aug	50	36	42	128
Sep	38	31	40	109
Total	133	100	112	345


Roll up
by Month

Number of Autos Sold

CA	OR	WA	Total
133	100	112	345


Drill down
by Color

Number of Autos Sold

	CA	OR	WA	Total
Red	40	29	40	109
Blue	45	31	37	113
Gray	48	40	35	123
Total	133	100	112	345



MOLAP vs. ROLAP

- MOLAP = Multidimensional OLAP
- Store data cube as multidimensional array
- (Usually) pre-compute all aggregates
- Advantages:
 - Very efficient data access → fast answers
- Disadvantages:
 - Doesn't scale to large numbers of dimensions
 - Requires special-purpose data store

Sparsity

- Imagine a data warehouse for Safeway.
- Suppose dimensions are: Customer, Product, Store, Day
- If there are 100,000 customers, 10,000 products, 1,000 stores, and 1,000 days...
- ...data cube has 1,000,000,000,000,000 cells!
- Fortunately, most cells are empty.
- A given store doesn't sell every product on every day.
- A given customer has never visited most of the stores.
- A given customer has never purchased most products.
- Multi-dimensional arrays are not an efficient way to store sparse data.



MOLAP vs. ROLAP

- ROLAP = Relational OLAP
- Store data cube in relational database
- Express queries in SQL
- Advantages:
 - Scales well to high dimensionality
 - Scales well to large data sets
 - Sparsity is not a problem
 - Uses well-known, mature technology
- Disadvantages:
 - Query performance is slower than MOLAP
 - Need to construct explicit indexes



Outline

- History of Data Warehouses and Merging Information Resources
- What is a Data Warehouse?
- Data Warehouse dan OLAP
- Data Warehouse Architectures
- Data Warehousing Problems and Issues
- **Data Model and Design Issues**



Multidimensional Data Model

- Data warehouse and OLAP tools are based on a multidimensional data model. This model views data in the form of a data cube.
- Composed of one **fact table** and a set of **dimension tables**.
- **Fact table**: with a composite primary key
- **Dimensional table**: each dimension table has a simple table (non-composite) primary key that corresponds exactly to one of the components of the composite key in the fact table.

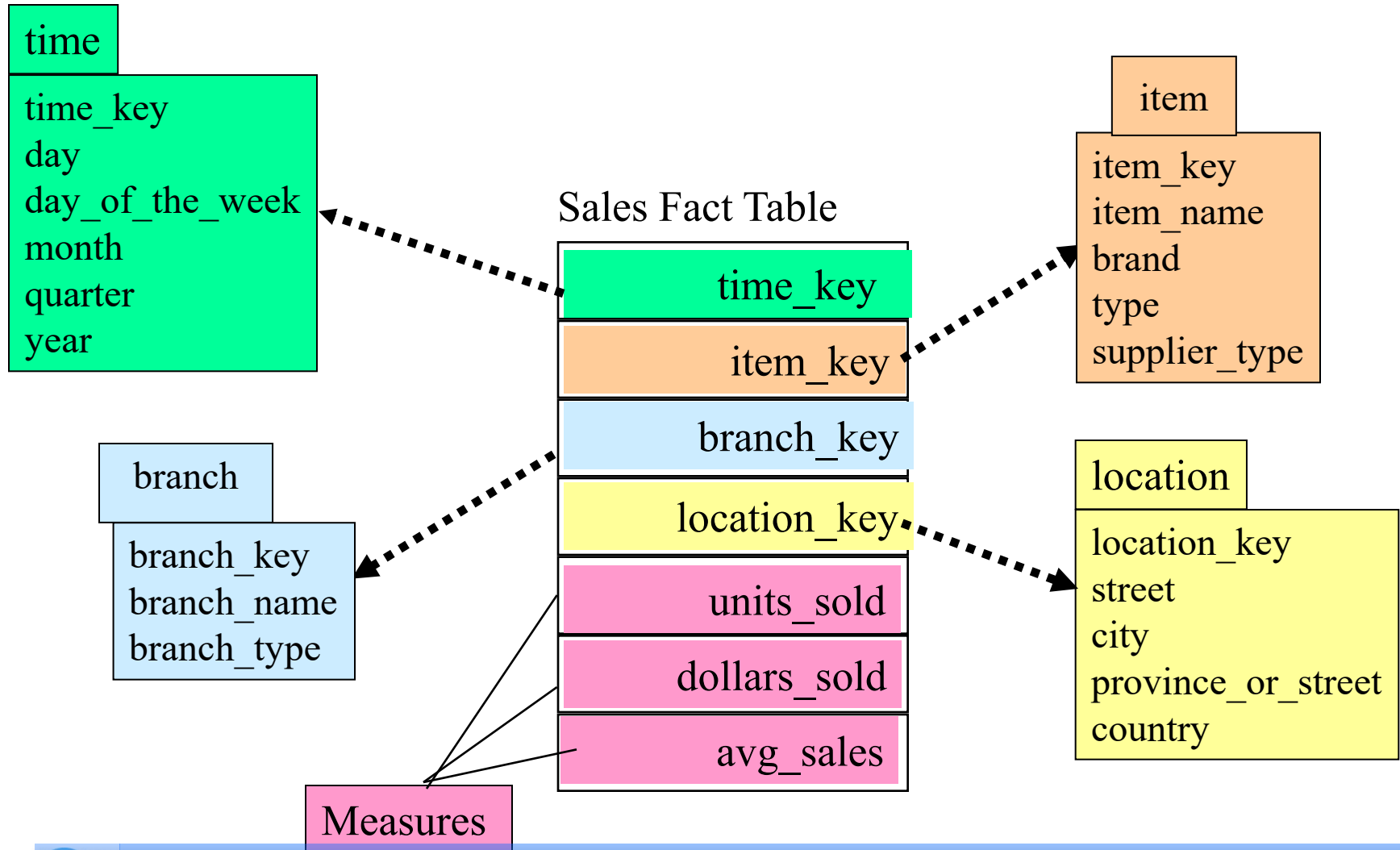
Conceptual Modeling of Data Warehouses

Modeling data warehouses: dimensions & measures

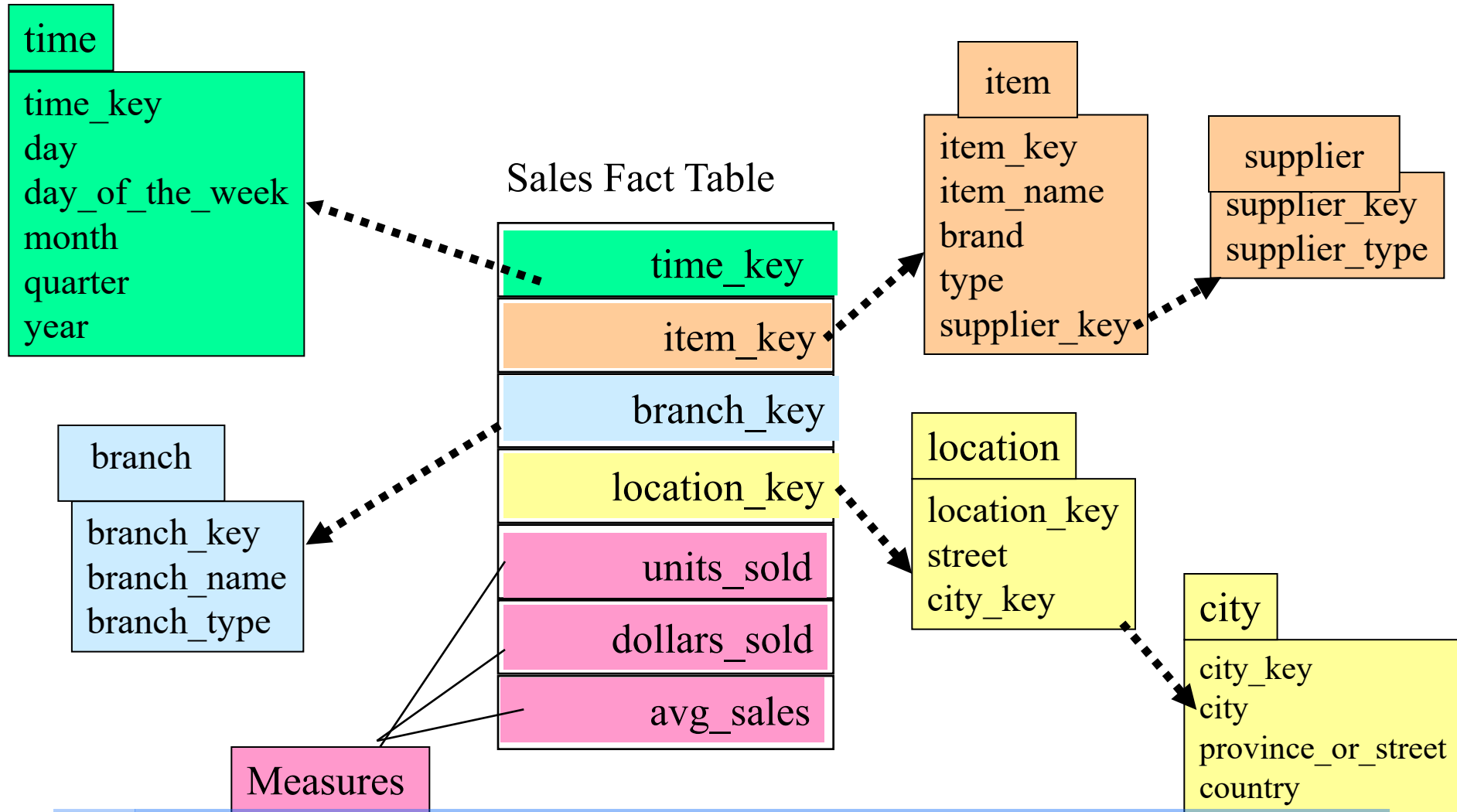
- Star schema: A fact table in the middle connected to a set of dimension tables
- Snowflake schema: A refinement of star schema where some dimensional hierarchy is **normalized** into a set of smaller dimension tables, forming a shape similar to snowflake
- Fact constellations: Multiple fact tables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



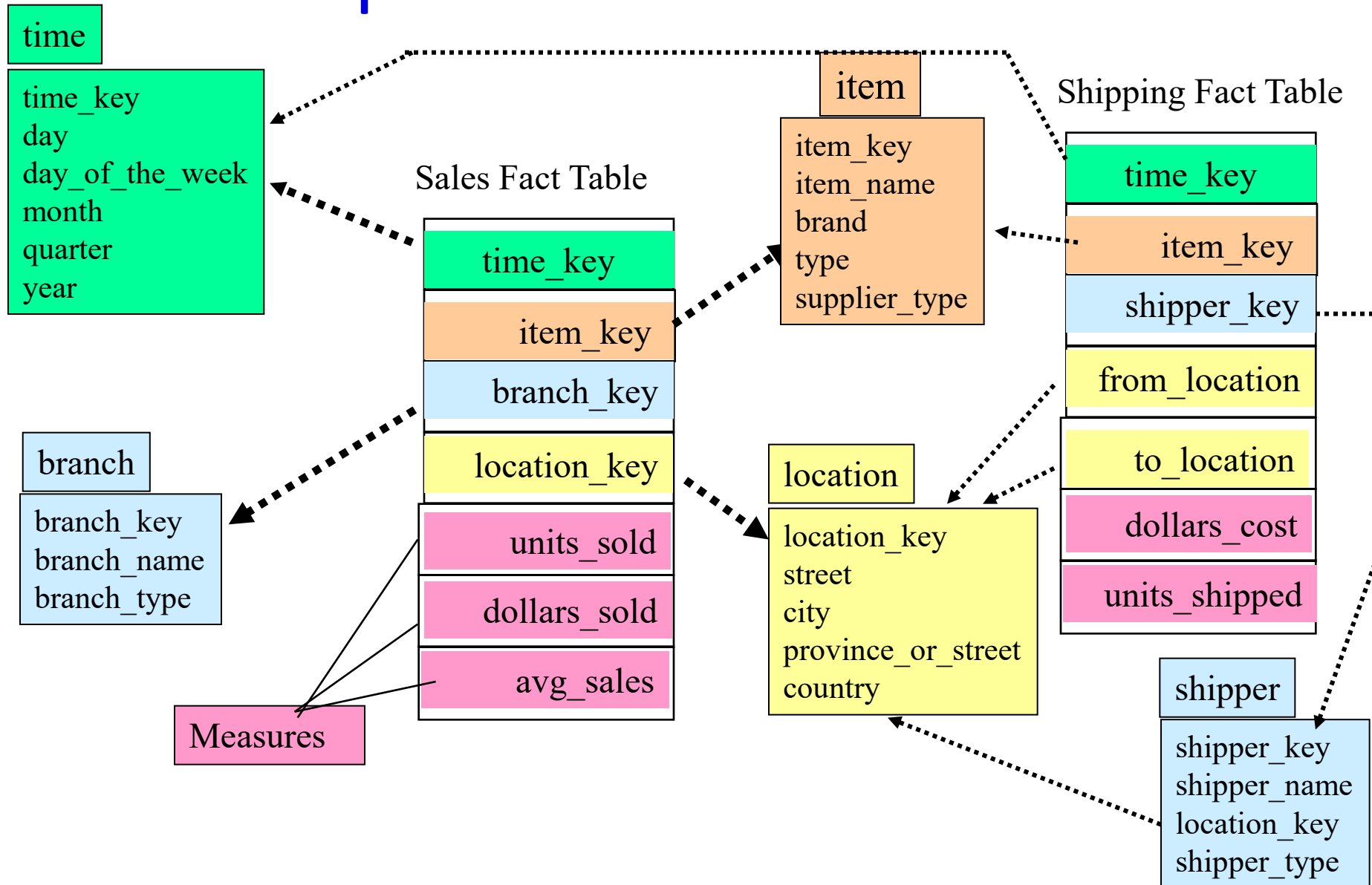
Example of Star Schema



Example of Snowflake Schema



Example of Fact Constellation



Data Warehouse Design Process

- Top-down, bottom-up approaches, or a combination of both
 - Top-down: Starts with overall design and planning (mature)
 - Bottom-up: Starts with experiments and prototypes (rapid)
- Typical data warehouse design process
 - Choose a **business process** to model, e.g., orders, invoices, etc.
 - Choose the ***grain (atomic level of data)*** of the business process
 - Choose the **dimensions** that will apply to each fact table record
 - Choose the **measure** that will populate each fact table record

Two Approaches to Data Warehousing

- **Data mart**: like a data warehouse, but **smaller** and **more focused**
- **Top-down approach**
 - First build single unified data warehouse with all enterprise data
 - Then create data marts containing specialized subsets of the data from the warehouse
- **Bottom-up approach**
 - First build a data mart to solve the most pressing problem
 - Then build another data mart, then another
 - Data warehouse = union of all data marts
- In practice, not much difference between the two



Running example: Retail Sales

- Grocery store chain recording POS retail sales
 - POS = Point of sale
 - Data collected by bar-code scanners at cash register
 - 100 grocery stores in 5 states
 - ~60,000 product SKUs
 - SKU = stock keeping unit
 - Represents an individual product
 - Some have UPCs (Universal Product Codes) assigned by manufacturer
 - Others don't (for example, bakery, meat, floral)
 - **Goal:** understand **impact** of pricing & promotions on sales, profits
 - Promotions = coupons, discounts, advertisements, etc.

How many dimensions?

- Should two concepts be modeled as separate dimensions or two aspects of the same dimension?
- Example: Different types of promotions
 - Ads, discounts, coupons, end-of-aisle displays
 - Option A: 4 dimensions
 - Separate dimension for each type of promotion
 - Option B: 1 dimension
 - Each dimension row captures a **combination** of ad, discount, coupon, and end-of-aisle display
- Factors to consider
 - How do the users think about the data?
 - Are an ad and a coupon separate promotions or two aspects of the same promotion?
 - Fewer tables → good
 - Generally fewer tables → simpler design
 - Performance implications
 - See following slides...



How many dimensions?

Performance Implications

- Most OLAP queries are “I/O bound”
 - Data-intensive not compute-intensive
 - Reading the data from disk is the bottleneck
 - For “typical” queries, on “typical” hardware
- Size of data on disk \approx query performance
 - Keeping storage requirements small is important
- Dimensional modeling impacts storage requirements

Performance Implications

- Let's consider the extremes
- Assumptions:
 - 100 million fact rows
 - 3 four-byte measurement columns in the fact table
 - 100 dimensional attributes, average size = 20 bytes
- Three modeling options
 - One “Everything” dimension
 - Each attribute gets its own dimension table
 - 5 dimensions (Date, Product, Store, Promotion, Transaction ID)

Option A

- Option A: One “Everything” dimension
 - Fact table is very thin (16 bytes per row)
 - 3 four-byte fact columns
 - 1 four-byte foreign key to the Everything dimension
 - Dimension table is very wide (2000 bytes per row)
 - 100 attributes * 20 bytes each
 - Dimension table has as many rows as fact table!
 - Each fact row has a different combination of attributes
 - Total space = 1.6 GB fact + 200 GB dimension
 - 16 bytes * 100 million rows = 1.6 GB
 - 2000 bytes * 100 million rows = 200 GB



Option B

- Option B: Each attribute gets its own dimension table
 - Store Manager First Name dimension, Store Manager Last Name dimension, etc.
 - Fact table is wide (212 bytes per row)
 - Assume 2-byte keys for all dimension tables
 - This is a **generous** assumption!!
 - Dimension tables are very thin, have few rows
 - Space for fact table = **21.2 GB** (212 bytes * 100 million rows)
 - Space for dimension tables → negligible, with assumptions:
 - < 132 MB total for all dimensions (100 * 22 * 60,000)
 - No dimension table has more than 60,000 rows
 - Each dimension row is 22 bytes
 - 100 dimension tables



Option C

- Option C: Five dimensions (Date, Product, Store, Promotion, Transaction ID)
 - Fact table is quite thin (28 bytes)
 - 2-byte keys for Date and Store
 - 4-byte keys for Product, Promotion, Transaction ID
 - 3 * 4-byte fact columns
 - Dimension tables are wide, but have few rows
 - Assume no dimension table has more than 60,000 rows
 - Space for fact table = **2.8 GB**
 - 28 bytes * 100 million rows
 - Space for dimension tables = negligible
 - < 130 MB for all dimensions → $60,000 * (100 * 20 + (4 + 12))$



Why is Option C the best?

- Attributes that pertain to the same logical object have a high degree of correlation
 - Correlated attributes
 - (product name, brand)
 - Number of distinct combinations = number of distinct products
 - Product name and brand are completely correlated
 - Uncorrelated attributes
 - (product name, date)
 - Number of distinct combinations = number of products * number of dates
 - No correlation between date and product
 - Most possible combinations of values will appear in the fact table
 - Combining non-correlated attributes in the same dimension leads to blow-up in size of dimension table
- When attributes are semi-correlated, designer has a choice
 - Frequently, multiple types of promotion occur together
 - E.g. product being promoted has ad, coupon, and in-store display
 - Number of (ad, coupon, discount, display) combinations is small
 - Combining them in a single Promotion dimension is reasonable



In Class Exercise (1)

- Anda diminta membuat *data warehouse* untuk keperluan pelaporan dan analisis nilai semua mata kuliah yang diambil oleh mahasiswa. Untuk ini terdapat **empat tabel dimensi** dengan atribut sebagai berikut:
 - Tabel Dimensi **SeksiMataKuliah**, dengan atribut: NomorSeksi, IDMataKuliah, NamaMataKuliah, SKS, IDRuang, dan KapasitasRuang. Dalam satu semester, universitas menawarkan rata-rata 1000 seksi mata kuliah setiap semester.
 - Tabel Dimensi **Dosen**, dengan atribut: IDDosen, NamaDosen, Gelar, IDDepartemen, dan NamDepartemen. Diasumsikan terdapat 500 dosen yang dimiliki oleh universitas.
 - Tabel dimensi **Mahasiswa**, dengan atribut: IDMahasiswa, NamaMahasiswa, dan Departemen. Diasumsikan secara rata-rata setiap Seksi Mata Kuliah terdiri dari 50 mahasiswa, dan biasanya setiap mahasiswa mengambil 5 mata kuliah yang berbeda pada setiap semester.
 - Tabel dimensi **Semester**, terdiri dari atribut: IDSemester dan Tahun. Diasumsikan bahwa basis data akan berisikan data 20 semester (selama 10 tahun).
 - Satu-satunya nilai numerik yang harus dicatat dalam tabel fakta adalah **NilaiMataKuliah**

In Class Exercise (2)

- **Pertanyaan:**

- a) Desain sebuah *star schema* untuk kasus di atas
- b) Hitung estimasi jumlah baris dalam tabel fakta
- c) Hitung estimasi total ukuran tabel fakta (dalam bytes) dengan asumsi bahwa setiap *field* secara rata-rata memerlukan ruang penyimpan sebesar 10 bytes



END OF CHAPTER 05