

- What is a data warehouse?
- A multi-dimensional data model
- Data warehouse architecture
- Data warehouse implementation
- From data warehousing to data mining

### 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**

- “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 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
  - When data is moved to the warehouse, it is converted.

## Data Warehouse—Time Variant

- The time horizon for data warehouses 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”

Lecture 7

5

## 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

Lecture 7

7

## 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

Lecture 7

6

## OLTP vs. OLAP

	<b>OLTP</b>
<b>users</b>	clerk, IT professional
<b>function</b>	day to day operations
<b>DB design</b>	application-oriented
<b>data</b>	current, up-to-date detailed, flat relational isolated
<b>usage</b>	repetitive
<b>access</b>	read/write index/hash on prim key

February 12, 2008

Data Mining: Concepts and Techniq

8

# 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

Lecture 7

9

# Chapter 3: Data Warehousing and OLAP Technology: An Overview

- What is a data warehouse?
- A multi-dimensional data model**
- Data warehouse architecture
- Data warehouse implementation
- From data warehousing to data mining

Lecture 7

10

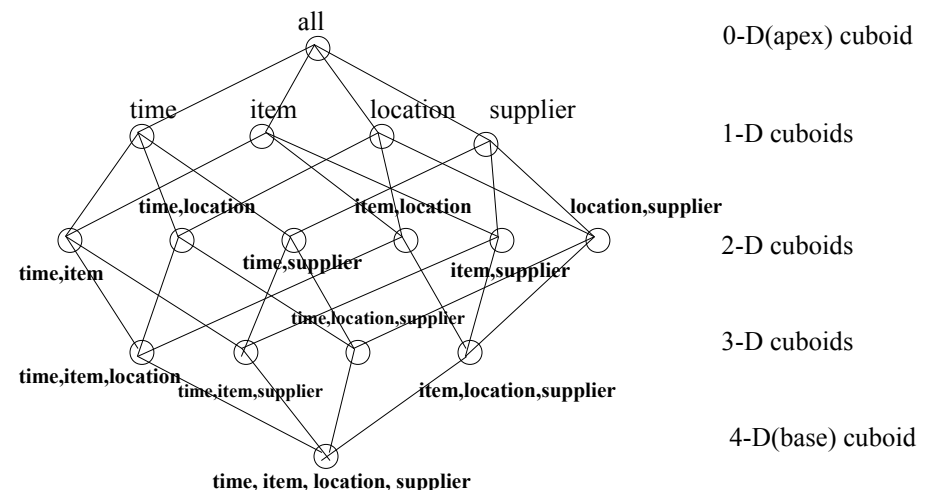
# From Tables and Spreadsheets to Data Cubes

- 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 top most 0-D cuboid, which holds the highest-level of summarization, is called the **apex cuboid**. The lattice of cuboids forms a **data cube**.

Lecture 7

11

# Cube: A Lattice of Cuboids



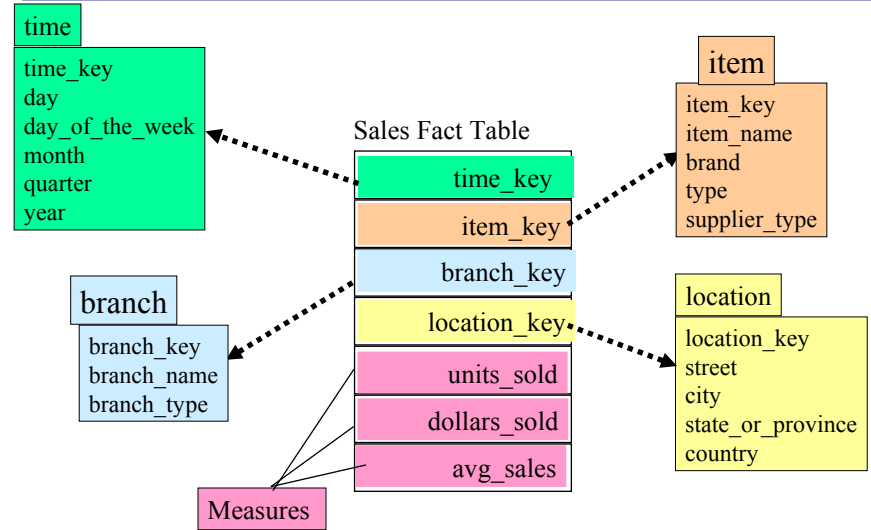
Lecture 7

12

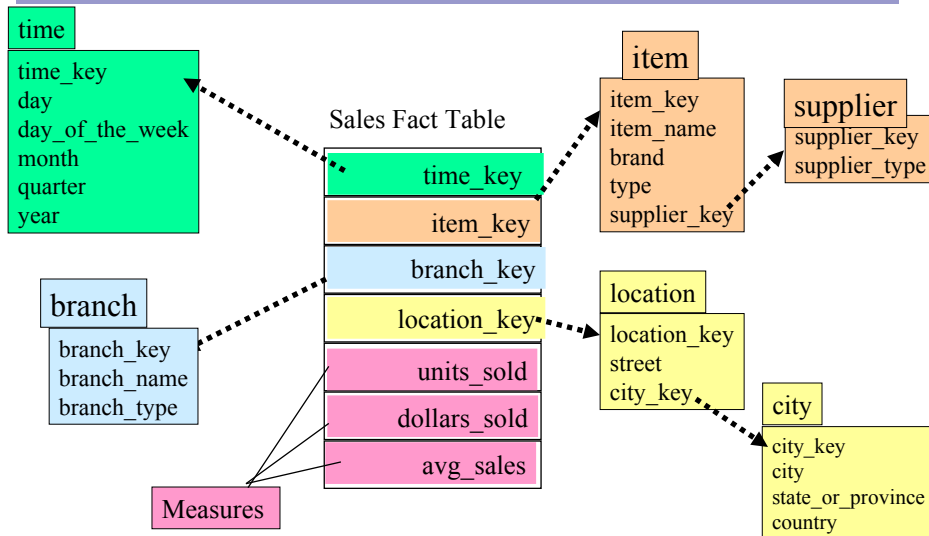
# Conceptual Modeling of Data Warehouses

- Modeling data warehouses: dimensions & measures (facts)
  - 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

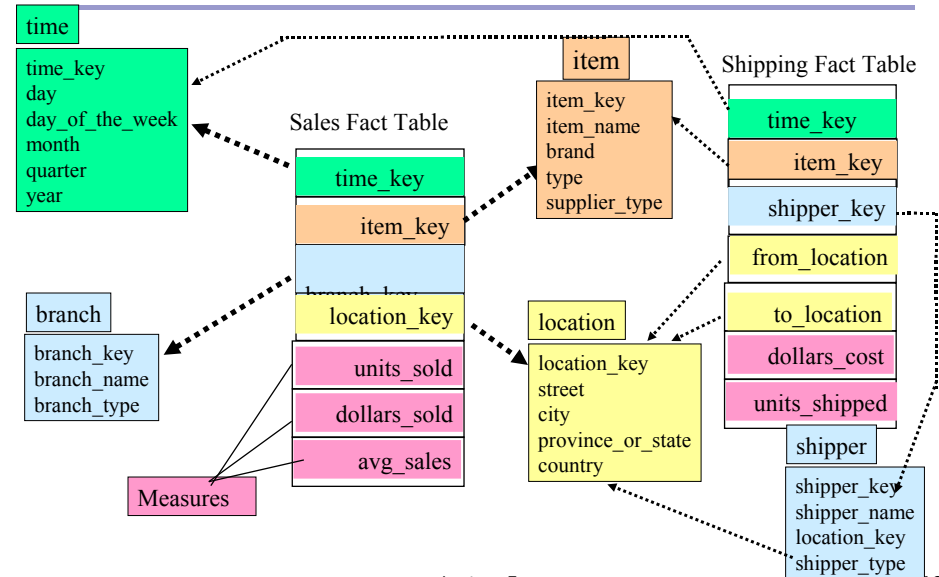
# Example of Star Schema



# Example of Snowflake Schema



# Example of Fact Constellation



# Cube Definition Syntax (BNF) in DMQL

- Cube Definition (Fact Table)  
`define cube <cube_name> [<dimension_list>]:  
 <measure_list>`
- Dimension Definition (Dimension Table)  
`define dimension <dimension_name> as  
 (<attribute_or_subdimension_list>)`
- Special Case (Shared Dimension Tables)
  - First time as “cube definition”
  - `define dimension <dimension_name> as  
 <dimension_name_first_time> in cube  
 <cube_name_first_time>`

Lecture 7

17

## Defining Snowflake Schema in DMQL

```
define cube sales_snowflake [time, item, branch, location]:  
  dollars_sold = sum(sales_in_dollars), avg_sales =  
  avg(sales_in_dollars), units_sold = count(*)  
define dimension time as (time_key, day, day_of_week, month,  
  quarter, year)  
define dimension item as (item_key, item_name, brand, type,  
  supplier(supplier_key, supplier_type))  
define dimension branch as (branch_key, branch_name,  
  branch_type)  
define dimension location as (location_key, street,  
  city(city_key, province_or_state, country))
```

Lecture 7

19

# Defining Star Schema in DMQL

```
define cube sales_star [time, item, branch,  
  location]:  
  dollars_sold = sum(sales_in_dollars), avg_sales =  
  avg(sales_in_dollars), units_sold = count(*)  
define dimension time as (time_key, day,  
  day_of_week, month, quarter, year)  
define dimension item as (item_key, item_name,  
  brand, type, supplier_type)  
define dimension branch as (branch_key,  
  branch_name, branch_type)  
define dimension location as (location_key, street,  
  city, province_or_state, country)
```

Lecture 7

18

## Defining Fact Constellation in DMQL

```
define cube sales [time, item, branch, location]:  
  dollars_sold = sum(sales_in_dollars), avg_sales =  
  avg(sales_in_dollars), units_sold = count(*)  
define dimension time as (time_key, day, day_of_week, month,  
  quarter, year)  
define dimension item as (item_key, item_name, brand, type,  
  supplier_type)  
define dimension branch as (branch_key, branch_name, branch_type)  
define dimension location as (location_key, street, city,  
  province_or_state, country)  
define cube shipping [time, item, shipper, from_location, to_location]:  
  dollar_cost = sum(cost_in_dollars), unit_shipped = count(*)  
define dimension time as time in cube sales  
define dimension item as item in cube sales  
define dimension shipper as (shipper_key, shipper_name, location as  
  location in cube sales, shipper_type)  
define dimension from_location as location in cube sales  
define dimension to_location as location in cube sales
```

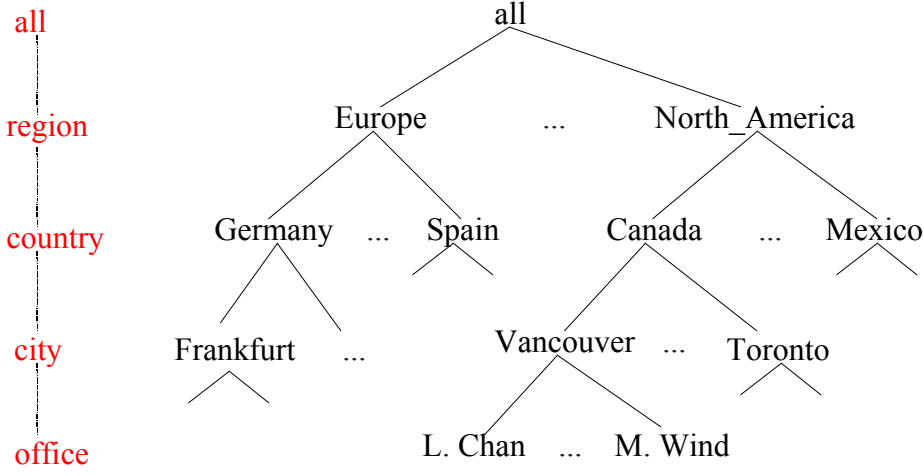
Lecture 7

20

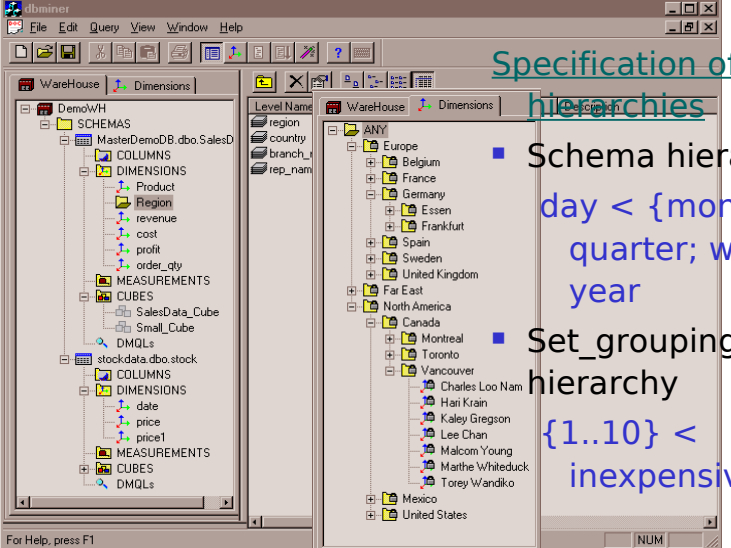
# Measures of Data Cube: Three Categories

- **Distributive**: if the result derived by applying the function to  $n$  aggregate values is the same as that derived by applying the function on all the data without partitioning. E.g., count(), sum(), min(), max()
- **Algebraic**: if it can be computed by an algebraic function with  $M$  arguments (where  $M$  is a bounded integer), each of which is obtained by applying a distributive aggregate function
  - E.g., avg(), min\_N(), standard\_deviation()
- **Holistic**: if there is no constant bound on the storage size needed to describe a subaggregate.
  - E.g., median(), mode(), rank()

# A Concept Hierarchy: Dimension (location)



# View of Warehouses and Hierarchies

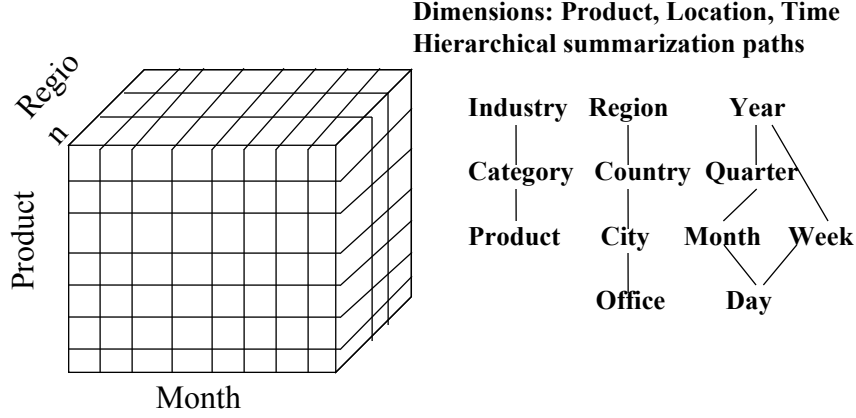


Specification of hierarchies

- Schema hierarchy  
 $day < \{month < quarter; week\} < year$
- Set\_grouping hierarchy  
 $\{1..10\} < inexpensive$

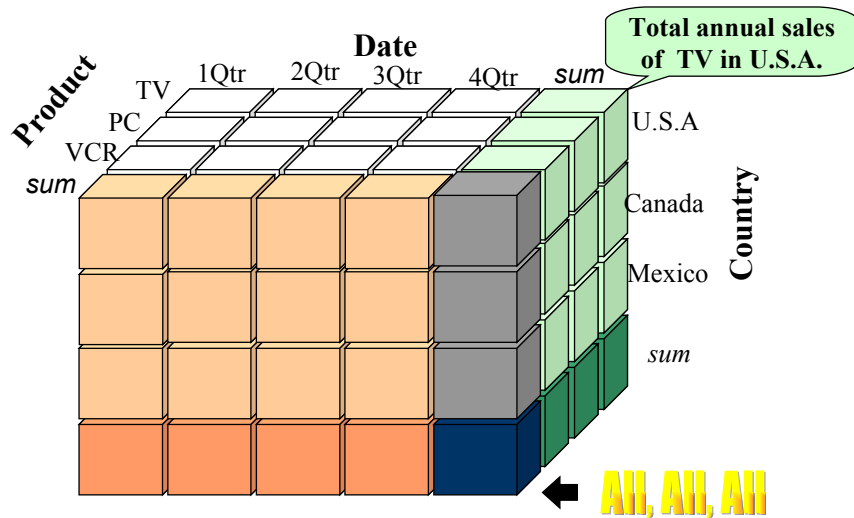
# Multidimensional Data

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



Dimensions: Product, Location, Time  
 Hierarchical summarization paths

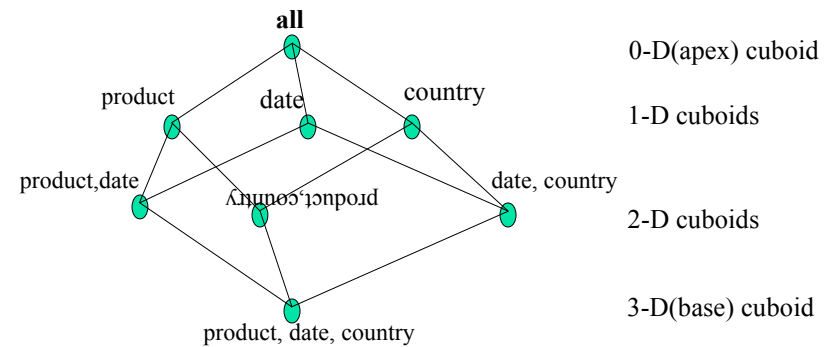
# A Sample Data Cube



Lecture 7

25

# Cuboids Corresponding to the Cube



0-D(apex) cuboid

1-D cuboids

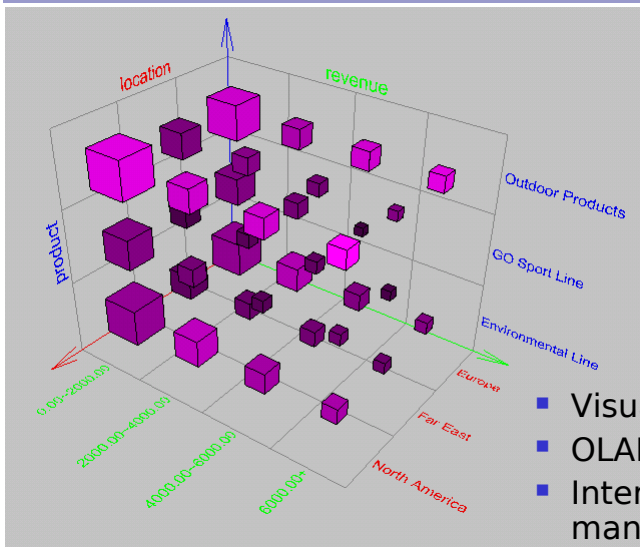
2-D cuboids

3-D(base) cuboid

Lecture 7

26

# Browsing a Data Cube



- Visualization
- OLAP capabilities
- Interactive manipulation

Lecture 7

27

# 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*

Lecture 7

28

# Typical OLAP Operations

- 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)

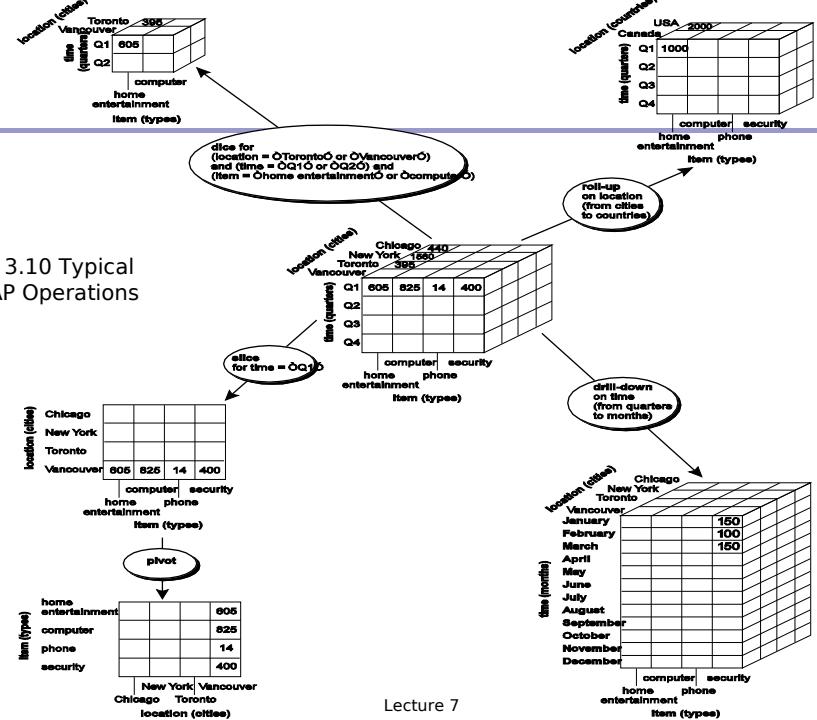


Fig. 3.10 Typical OLAP Operations