An Introduction to the Database Management Systems

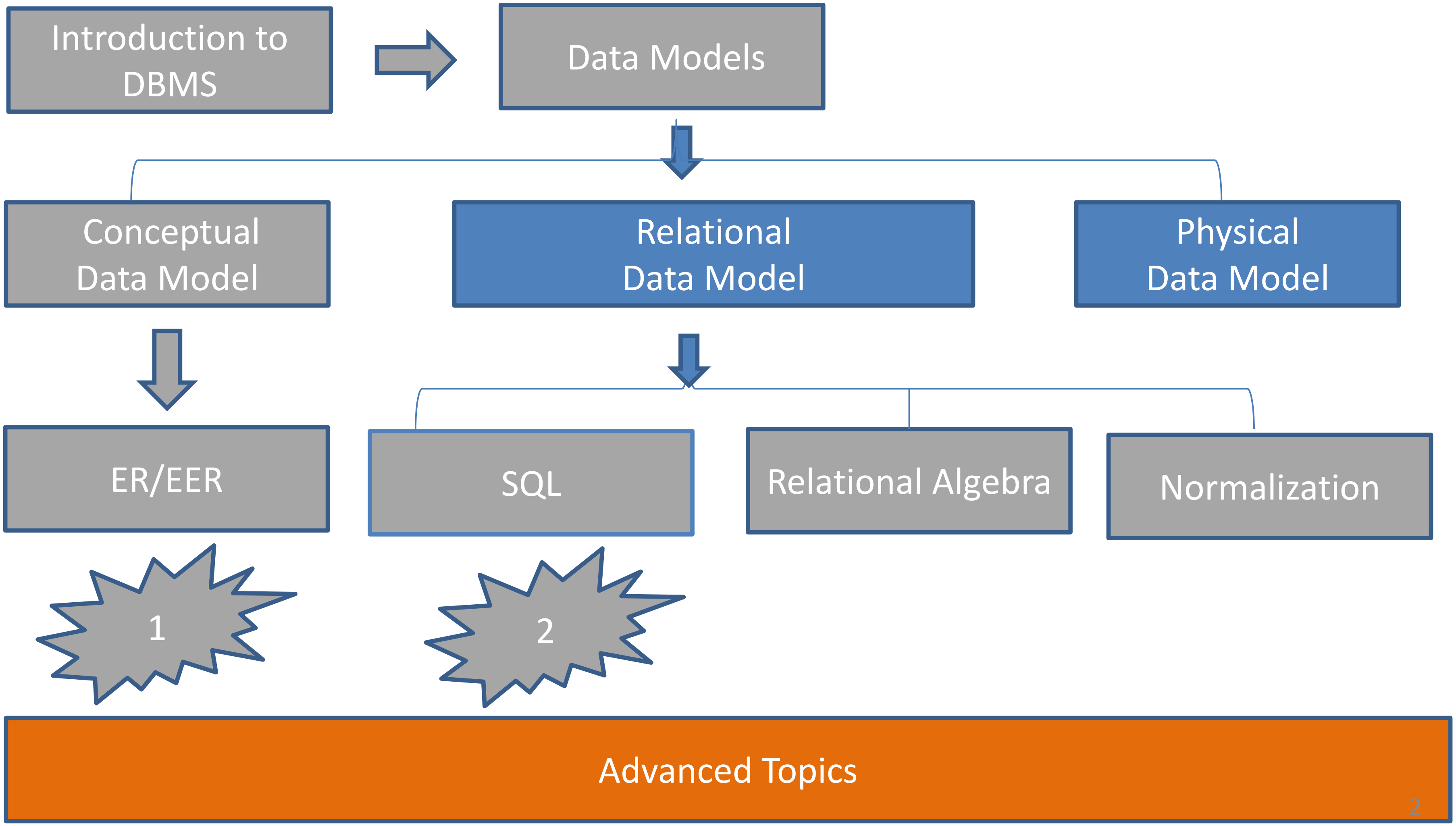
By

Hossein Rahmani

Slides originally by Book(s) Resources

Road Map

(Might change!)



Data Warehousing and On-line Analytical Processing

- Data Warehouse: Basic Concepts
- Data Warehouse Modeling: Data Cube and OLAP
- Data Warehouse Design and Usage
- Data Warehouse Implementation
- Data Generalization by Attribute-Oriented Induction
- Data Cube Computation
- Summary

What is a 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

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 <u>time horizon</u> for the data warehouse is significantly <u>longer</u> than that of operational systems
  - <u>Operational</u> database: <u>current</u> value data
  - <u>Data warehouse</u> data: provide information from a <u>historical</u> perspective (e.g., past 5-10 years)
- Every key structure in the data warehouse
  - Contains an <u>element of time</u>, 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*

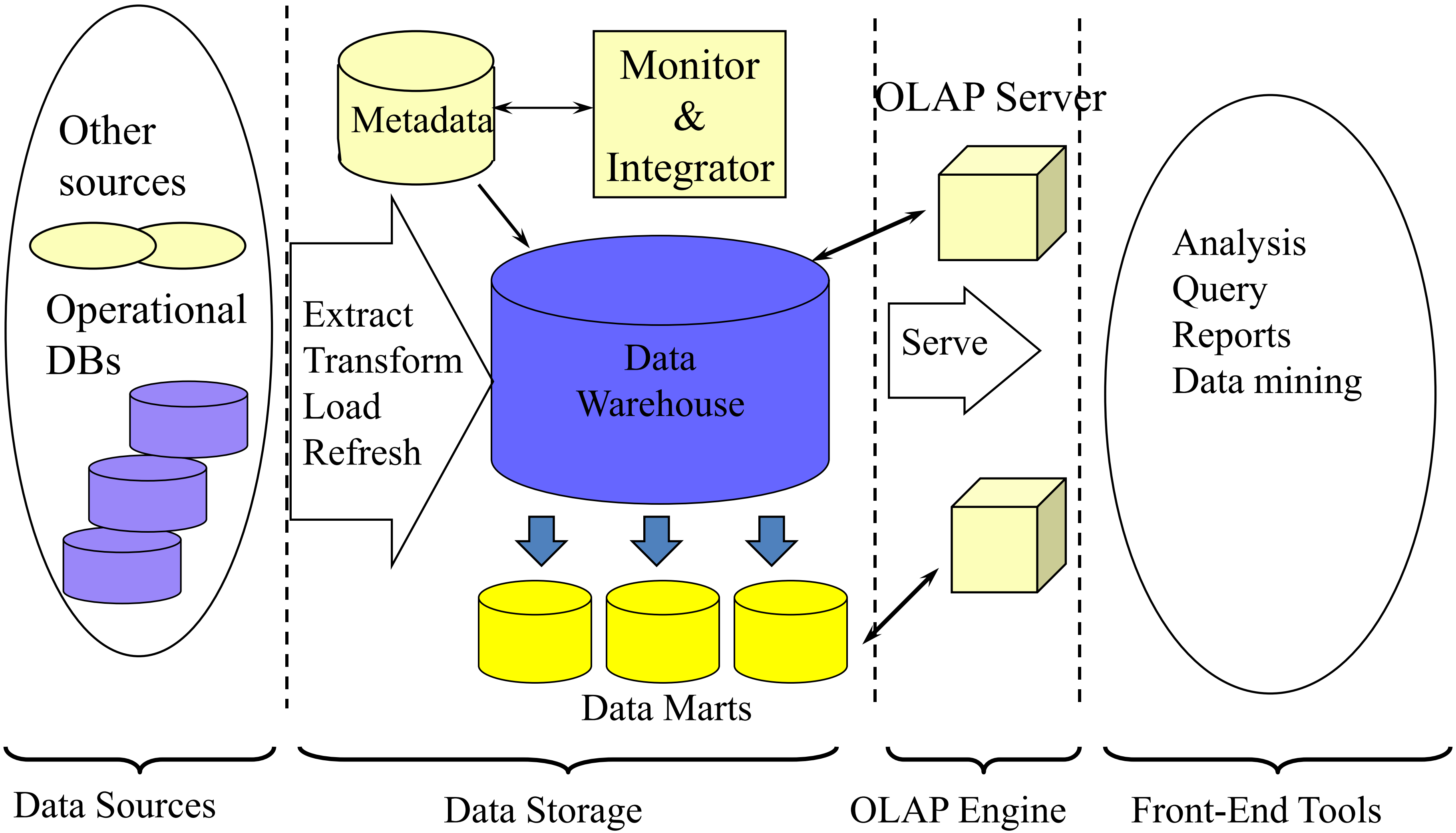
OLTP vs. OLAP

	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

Why a 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 (DS) 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 Warehouse: A Multi-Tiered Architecture



Three Data Warehouse Models

- Enterprise warehouse
  - collects all of the information about subjects spanning the entire organization
- Data Mart
  - a subset of corporate-wide data that is of value to a specific groups of users. Its scope is confined to specific, selected groups, such as marketing data mart
    - Independent vs. dependent (directly from warehouse) data mart
- Virtual warehouse
  - A set of views over operational databases
  - Only some of the possible summary views may be materialized

Extraction, Transformation, and Loading (ETL)

- **Data extraction**
  - get data from multiple, heterogeneous, and external sources
- **Data cleaning**
  - detect errors in the data and rectify them when possible
- **Data transformation**
  - convert data from legacy or host format to warehouse format
- **Load**
  - sort, summarize, consolidate, compute views, check integrity, and build indicies and partitions
- **Refresh**
  - propagate the updates from the data sources to the warehouse

Metadata Repository

- **Meta data** is the data defining warehouse objects. It stores:
- Description of the structure of the data warehouse
  - schema, view, dimensions, hierarchies, derived data defn, data mart locations and contents
- Operational meta-data
  - data lineage (history of migrated data and transformation path), currency of data (active, archived), monitoring information (warehouse usage statistics, error reports)
- The algorithms used for summarization
- The mapping from operational environment to the data warehouse
- Data related to system performance
  - warehouse schema, view and derived data definitions
- Business data
  - business terms and definitions, ownership of data

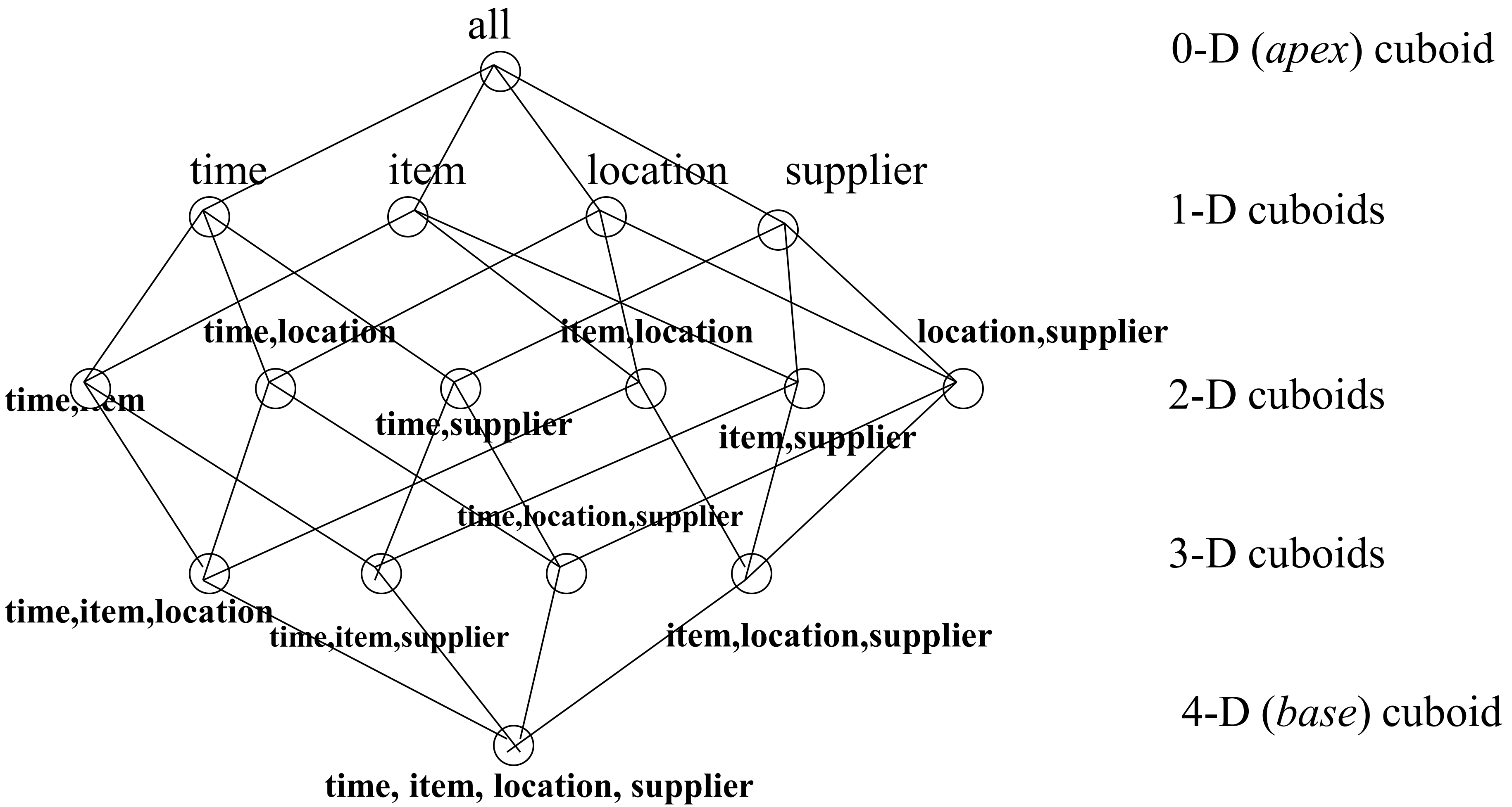
Data Warehousing and On-line Analytical Processing

- Data Warehouse: Basic Concepts
- Data Warehouse Modeling: Data Cube and OLAP
- Data Warehouse Design and Usage
- Data Warehouse Implementation
- Data Generalization by Attribute-Oriented Induction
- Data Cube Computation
- Summary

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.

Cube: A Lattice of Cuboids



An Example of Cuboids (Dimension) (add vs. delete dimensions)

0-D (apex) cuboid — Sales(all) →

Sales
24000

Add dim. ↓ ↑ Delete dim.

1-D cuboid — Sales(Q1), Sales(Q2), Sales(Q3), Sales(Q4) →

Time	Sales
Q1	5000
Q2	4000
Q3	8000
Q4	7000

Add dim. ↓ ↑ Delete dim.

2-D cuboid — Sales(Q1,Thailand), Sales(Q1,Japan) Sales(Q2,Thailand), Sales(Q2,Japan) Sales(Q3,Thailand), Sales(Q3,Japan) Sales(Q4,Thailand), Sales(Q4,Japan) →

Time	Thailand	Japan
Q1	2000	3000
Q2	1500	2500
Q3	2000	6000
Q4	3000	4000

Add dim. ↓ ↑ Delete dim.

3-D cuboid

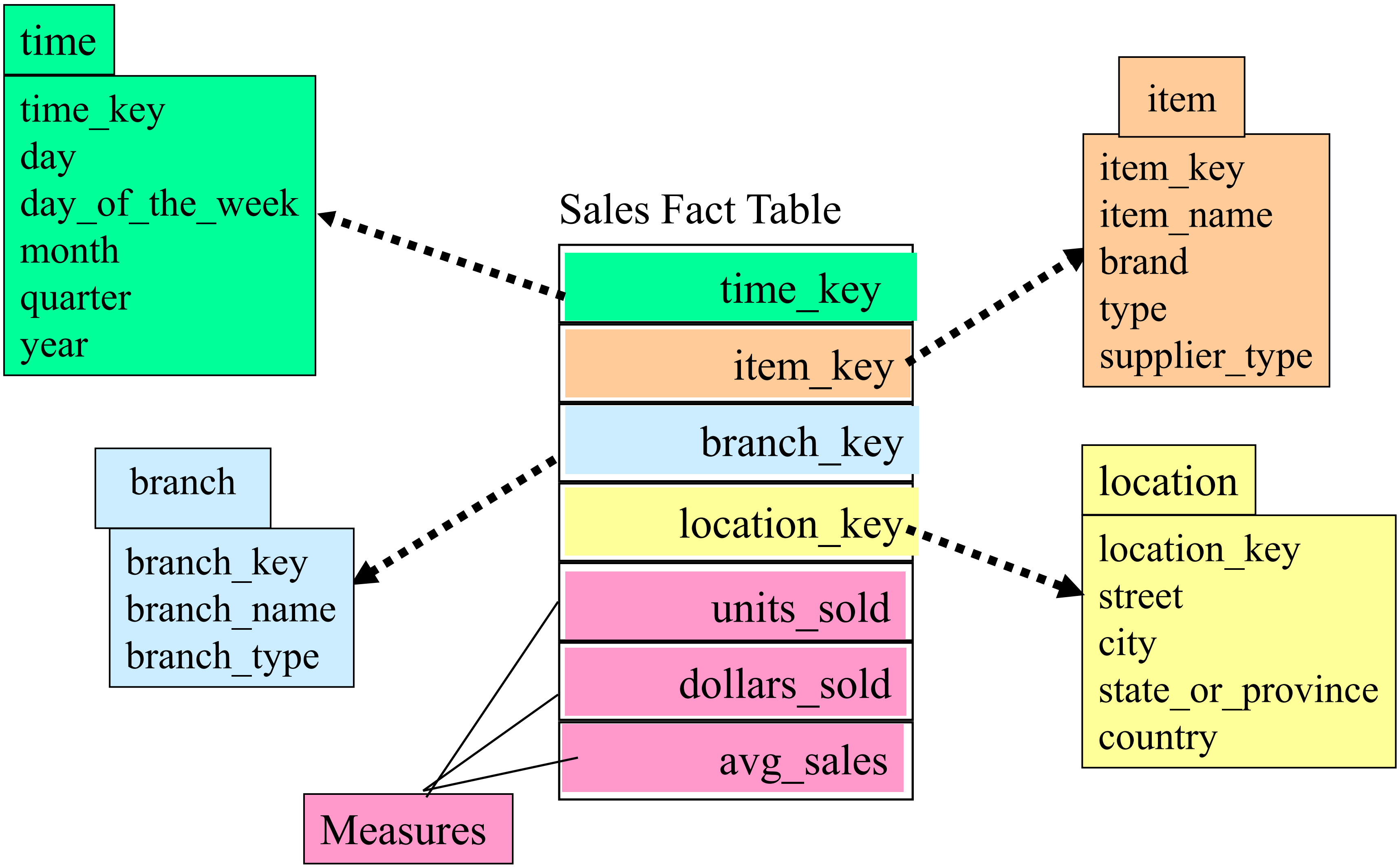
Sales(Q1,Thailand, Food), Sales(Q1,Thailand, NonFood)
Sales(Q2,Thailand, Food), Sales(Q2,Thailand, NonFood)
Sales(Q3,Thailand, Food), Sales(Q3,Thailand, NonFood)
Sales(Q4,Thailand, Food), Sales(Q4,Thailand, NonFood)
Sales(Q1, Japan, Food), Sales(Q1, Japan, NonFood)
Sales(Q2, Japan, Food), Sales(Q2, Japan, NonFood)
Sales(Q3, Japan, Food), Sales(Q3, Japan, NonFood)
Sales(Q4, Japan, Food), Sales(Q4, Japan, NonFood) →

Time	Thailand		Japan	
	Food	NonFood	Food	NonFood
Q1	1500	500	2000	1000
Q2	900	600	1500	1000
Q3	1200	800	4000	2000
Q4	2000	1000	2500	1500

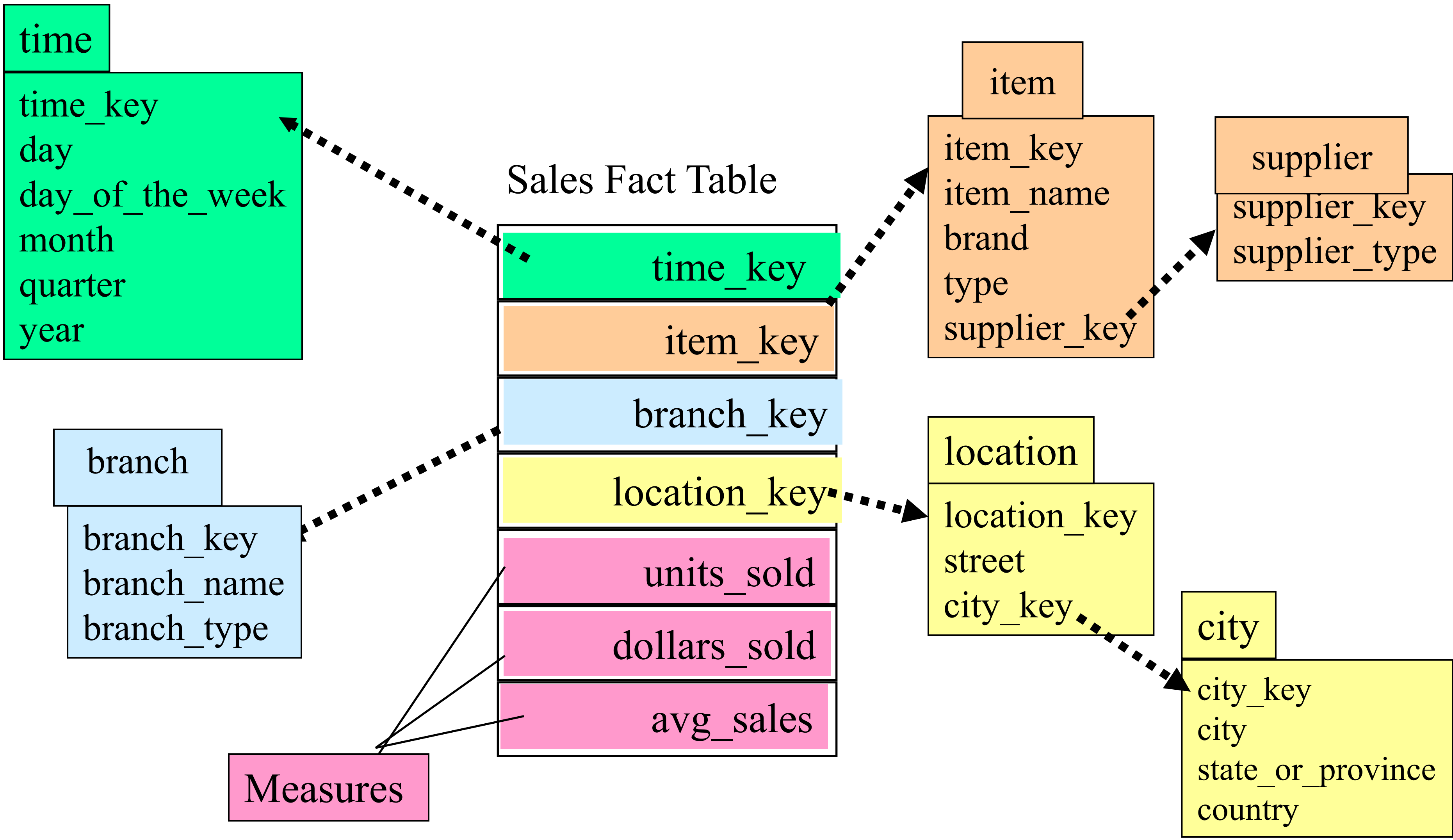
Conceptual Modeling of Data Warehouses

- 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

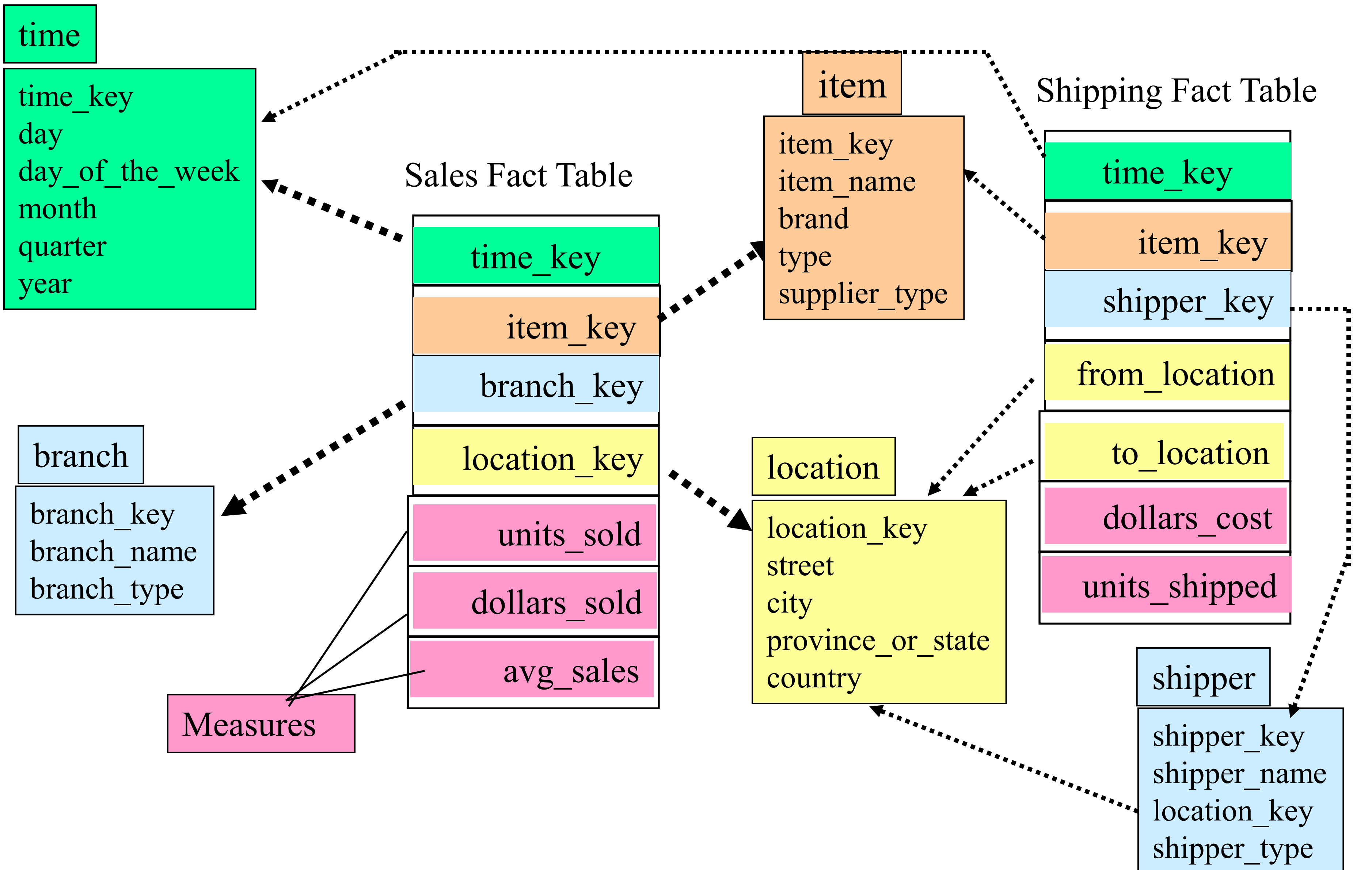
Example of **Star Schema**



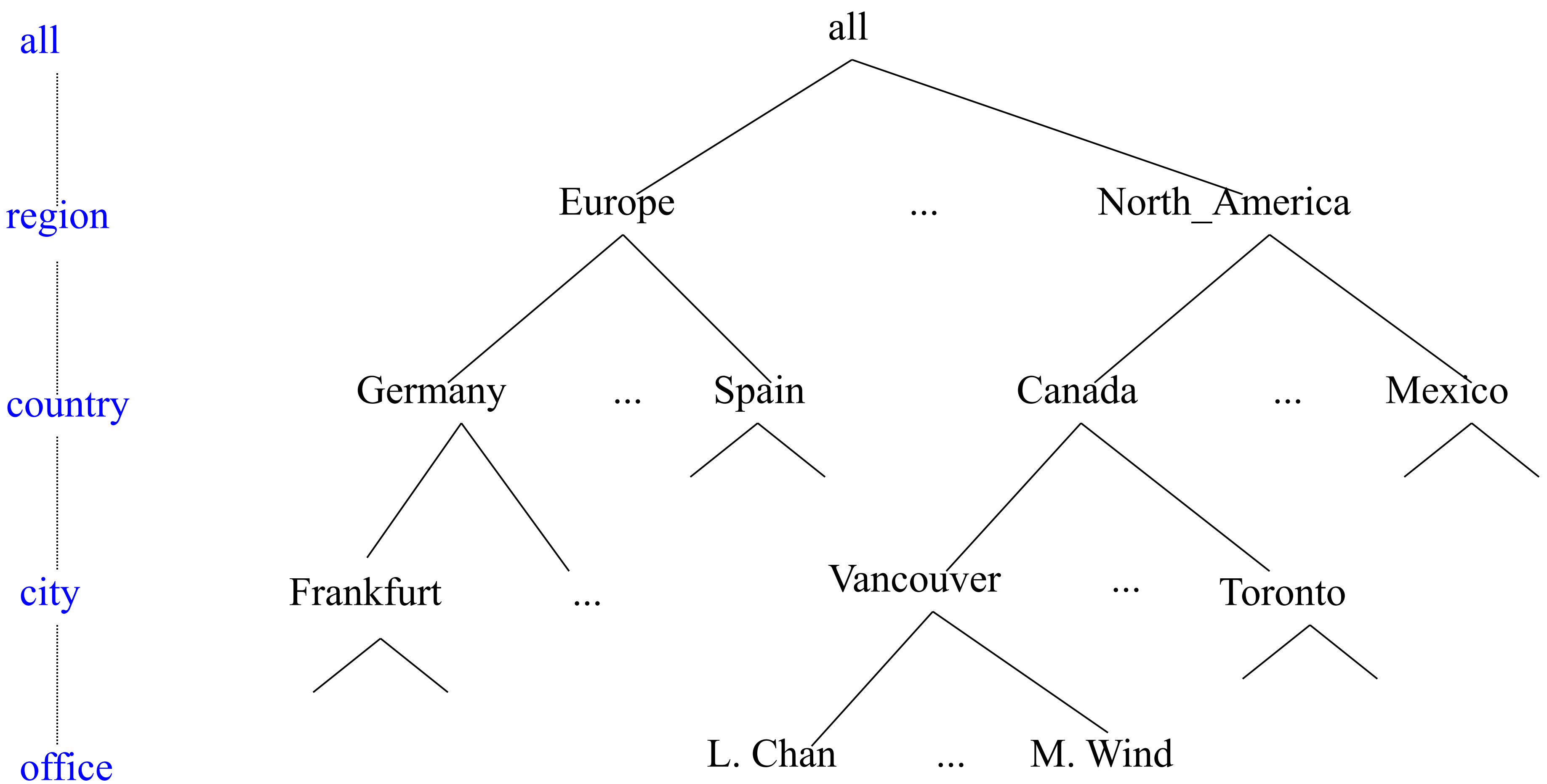
Example of **Snowflake Schema**



Example of **Fact Constellation**



A Concept Hierarchy: **Dimension** (location)



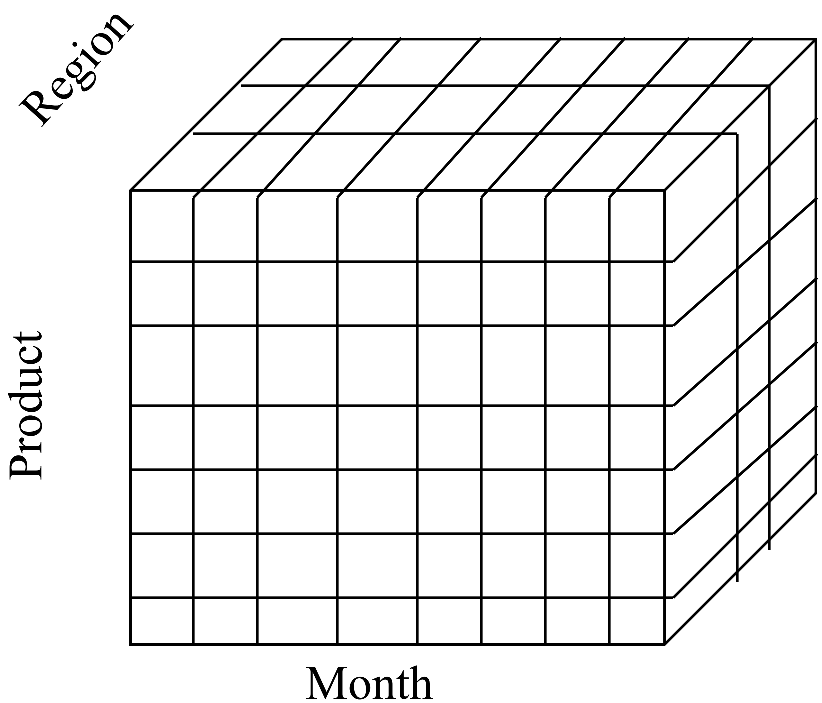
Data Cube Measures: Three Categories

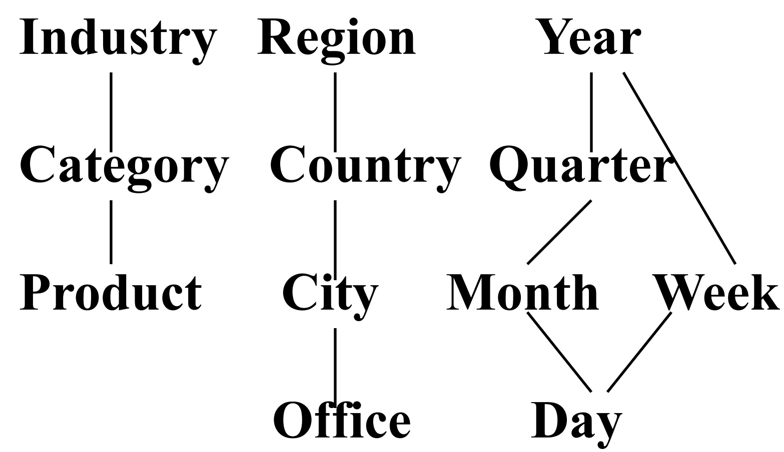
- Distributive: if the result derived by applying the function to *n* aggregate values is the same as that derived by applying the function on all the data without partitioning
  - E.g., count(), sum(), min(), max()
- Algebraic: if it can be computed by an algebraic function with *M* arguments (where *M* is a bounded integer), each of which is obtained by applying a distributive aggregate function
  - E.g., avg(), min_N(), standard_deviation()
- Holistic: There does not exist an algebraic function with M arguments (where M is a constant) that characterizes the computation. E.g., median()

Multidimensional Data

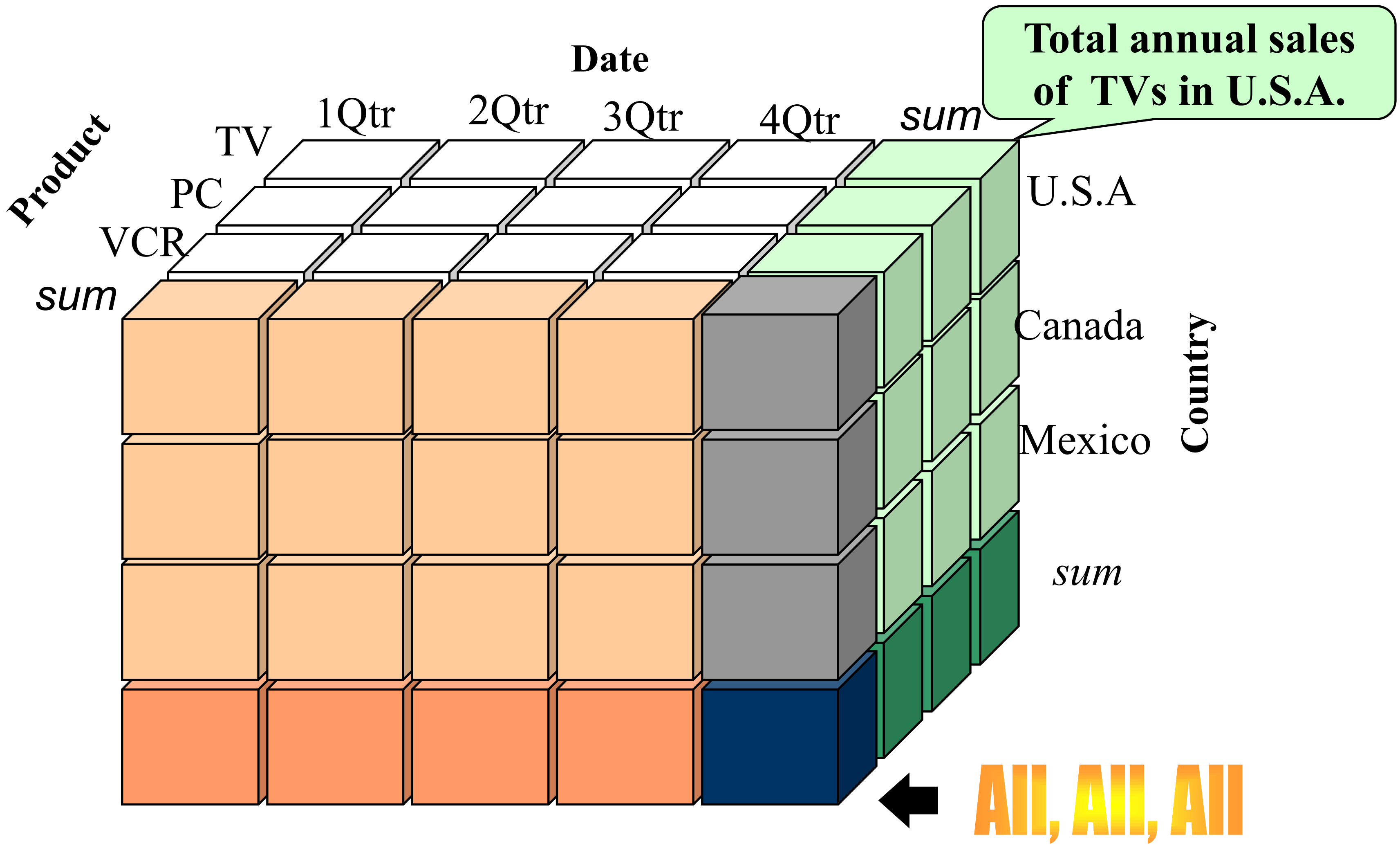
- Sales volume as a function of product, month, and region

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

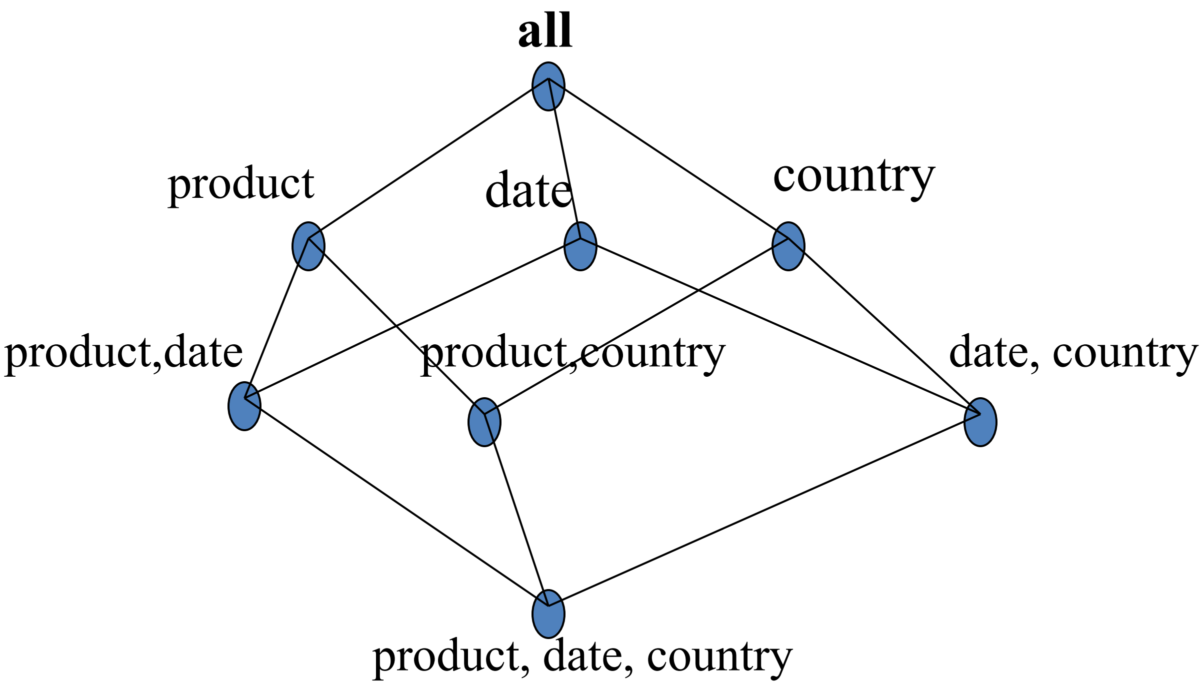




A Sample Data Cube



Cuboids Corresponding to the Cube



0-D (*apex*) cuboid

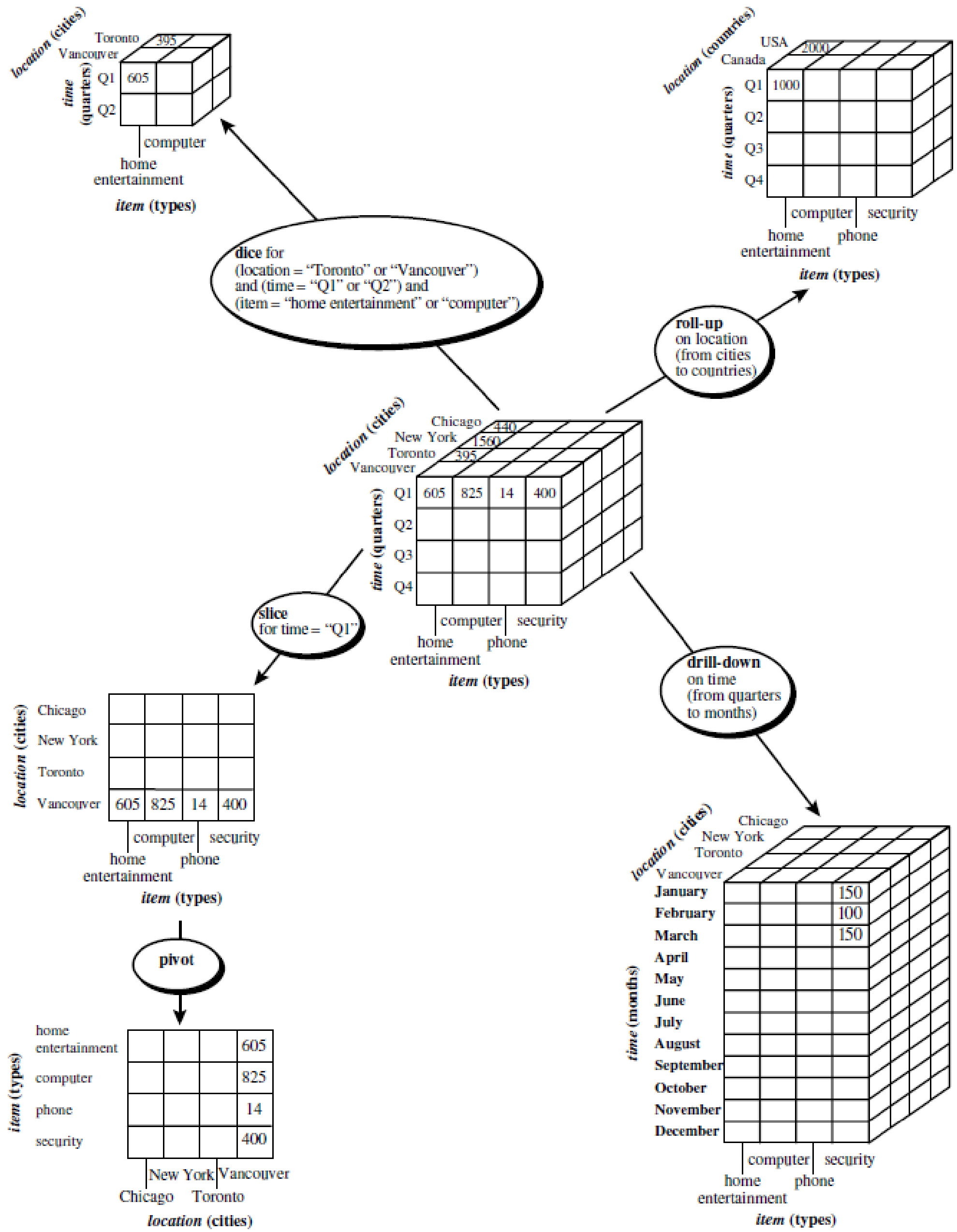
1-D cuboids

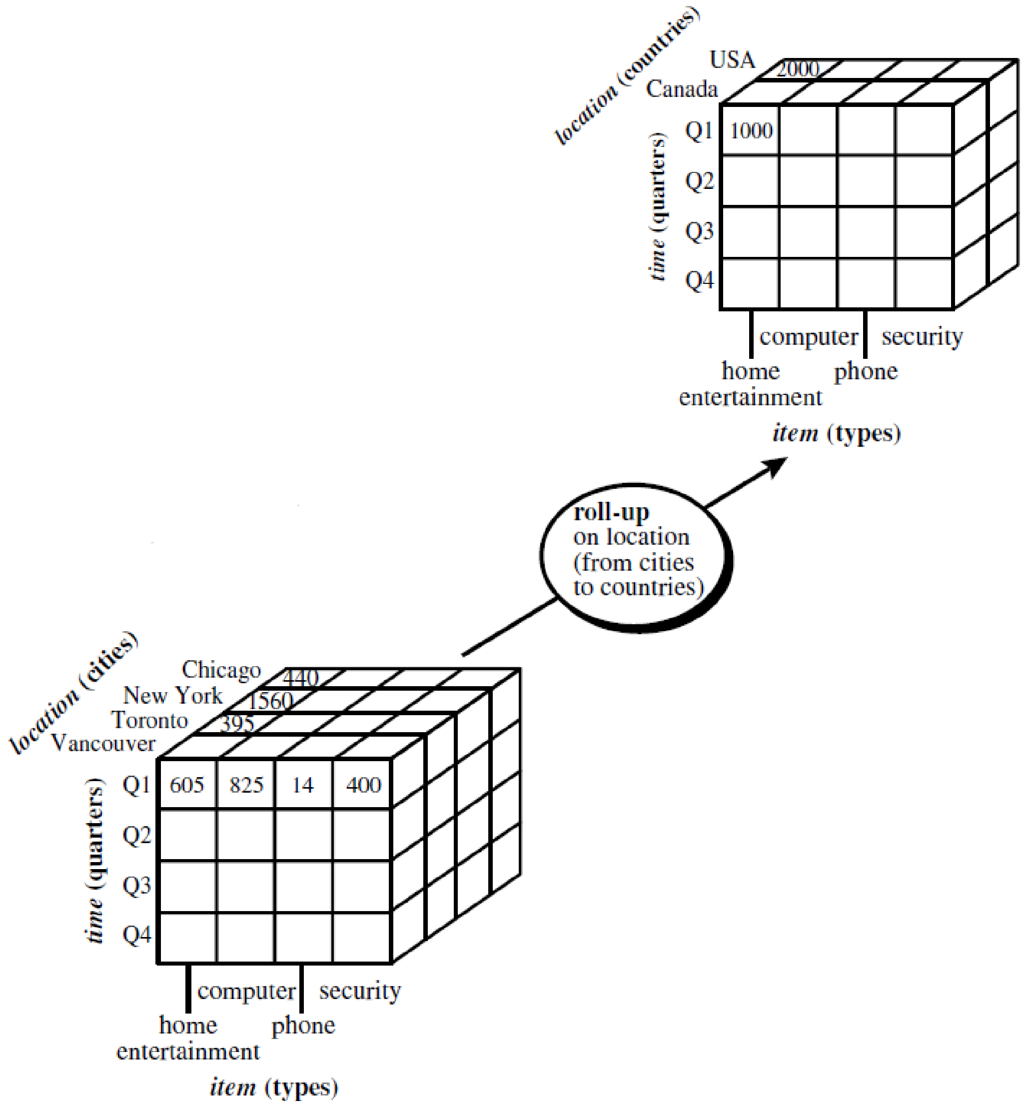
2-D cuboids

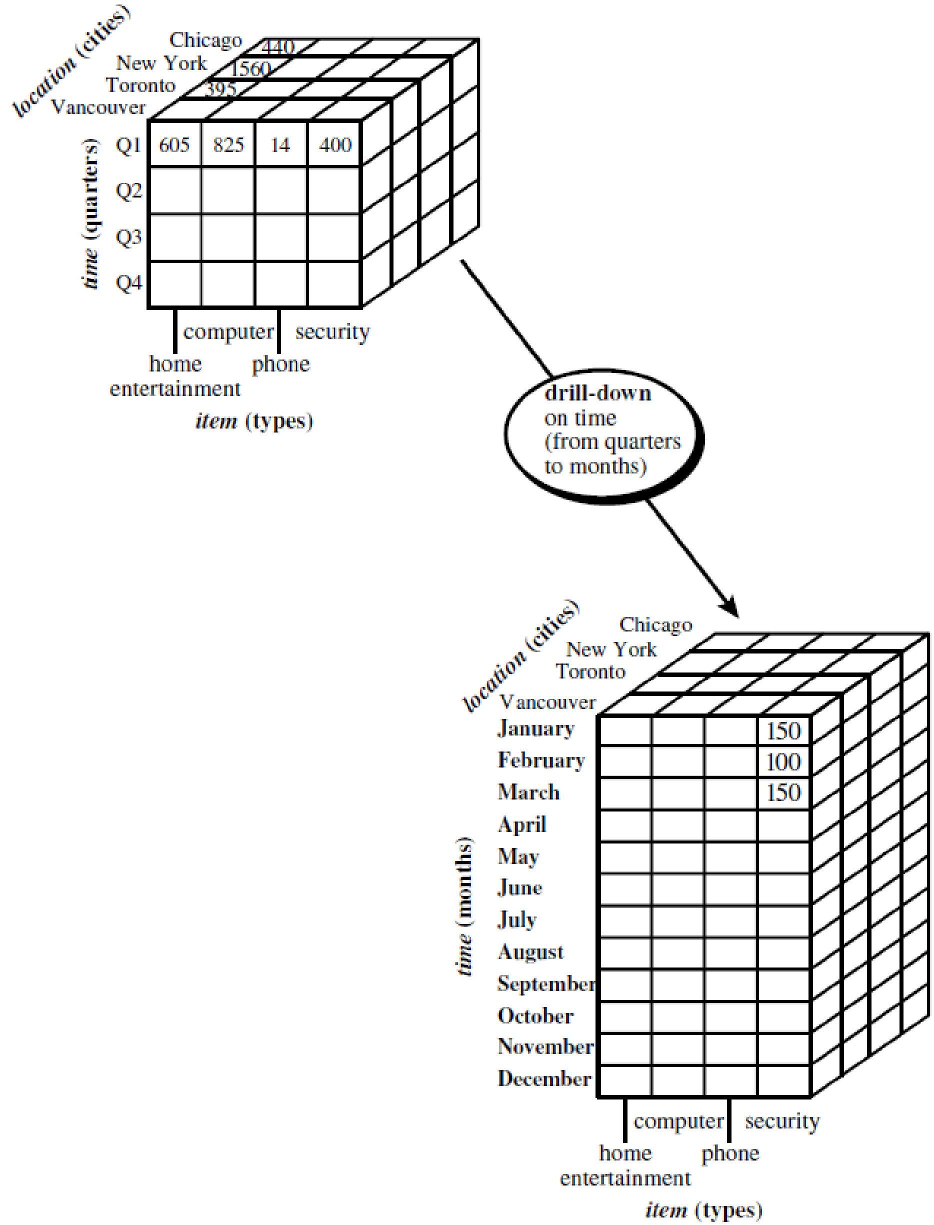
3-D (*base*) cuboid

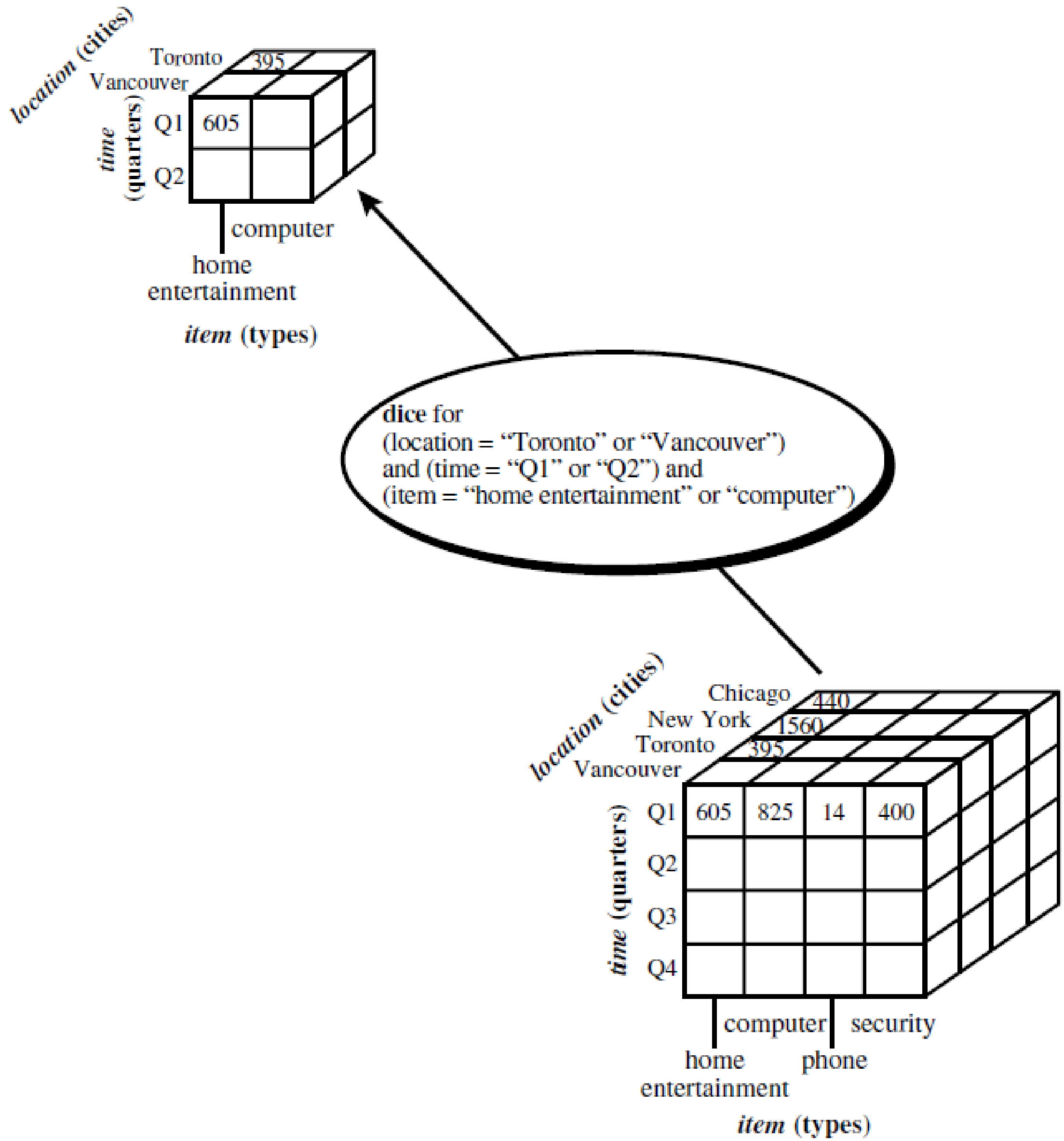
Typical OLAP Operations

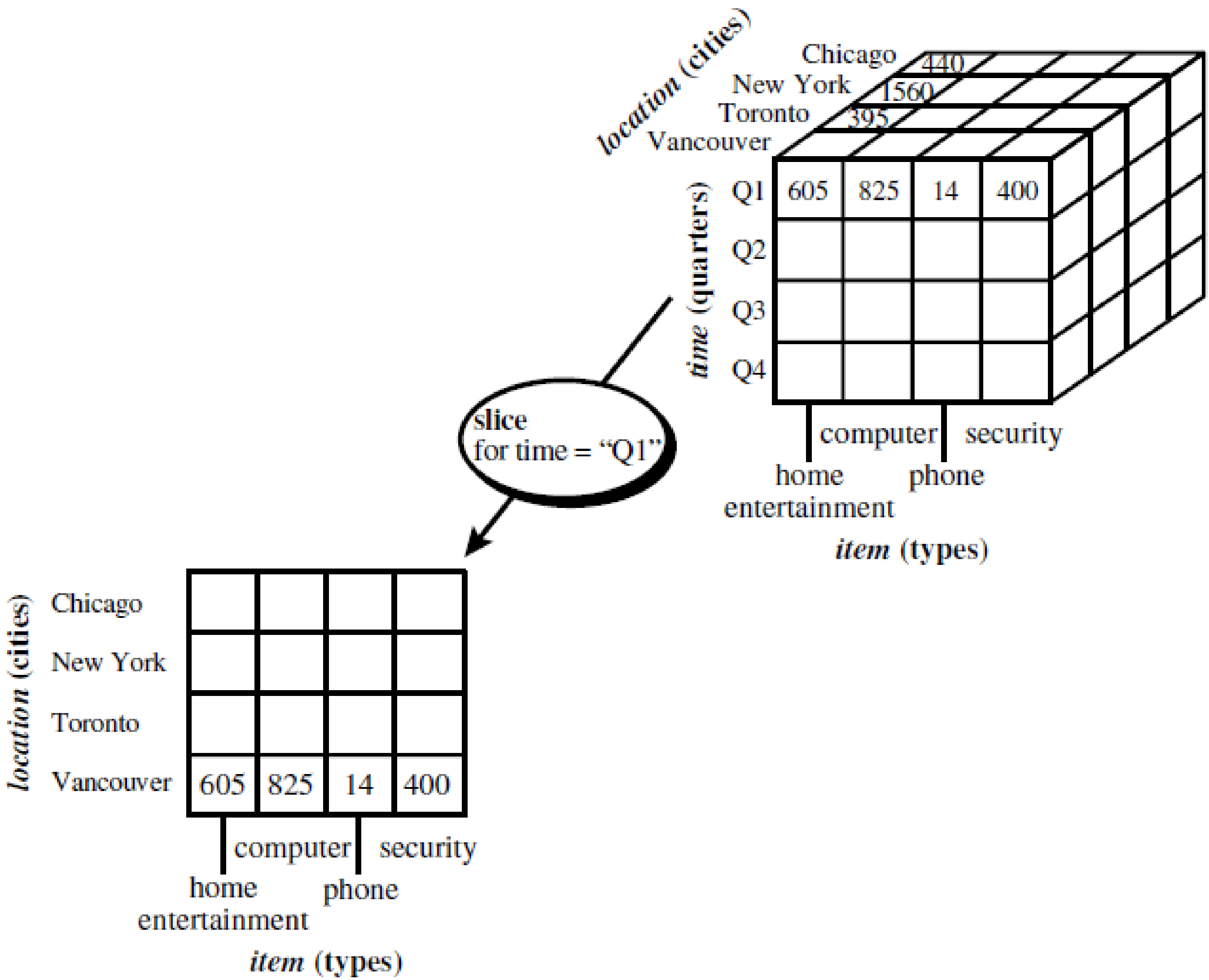
- Roll up (drill-up): summarize data
  - *by climbing up hierarchy or by dimension reduction*
- Drill down (roll down): reverse of roll-up
  - *from higher level summary to lower level summary or detailed data, or introducing new dimensions*
- Slice and dice: *project and select*
- Pivot (rotate):
  - *reorient the cube, visualization, 3D to series of 2D planes*


location (cities)
Toronto
Vancouver
395
time (quarters)
Q1
Q2
605
computer
home entertainment
item (types)
dice for (location = "Toronto" or "Vancouver") and (time = "Q1" or "Q2") and (item = "home entertainment" or "computer")
location (countries)
USA
Canada
2000
1000
time (quarters)
Q1
Q2
Q3
Q4
computer
security
home entertainment
phone
item (types)
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"
drill-down on time (from quarters to months)
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
location (cities)
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
item (types)


location (countries)
USA
Canada
2000
Q1
Q2
Q3
Q4
1000
time (quarters)
computer
security
home
entertainment
phone
item (types)
roll-up
on location
(from cities
to countries)
location (cities)
Chicago
New York
Toronto
Vancouver
440
1560
395
Q1
Q2
Q3
Q4
605
825
14
400
time (quarters)
computer
security
home
entertainment
phone
item (types)

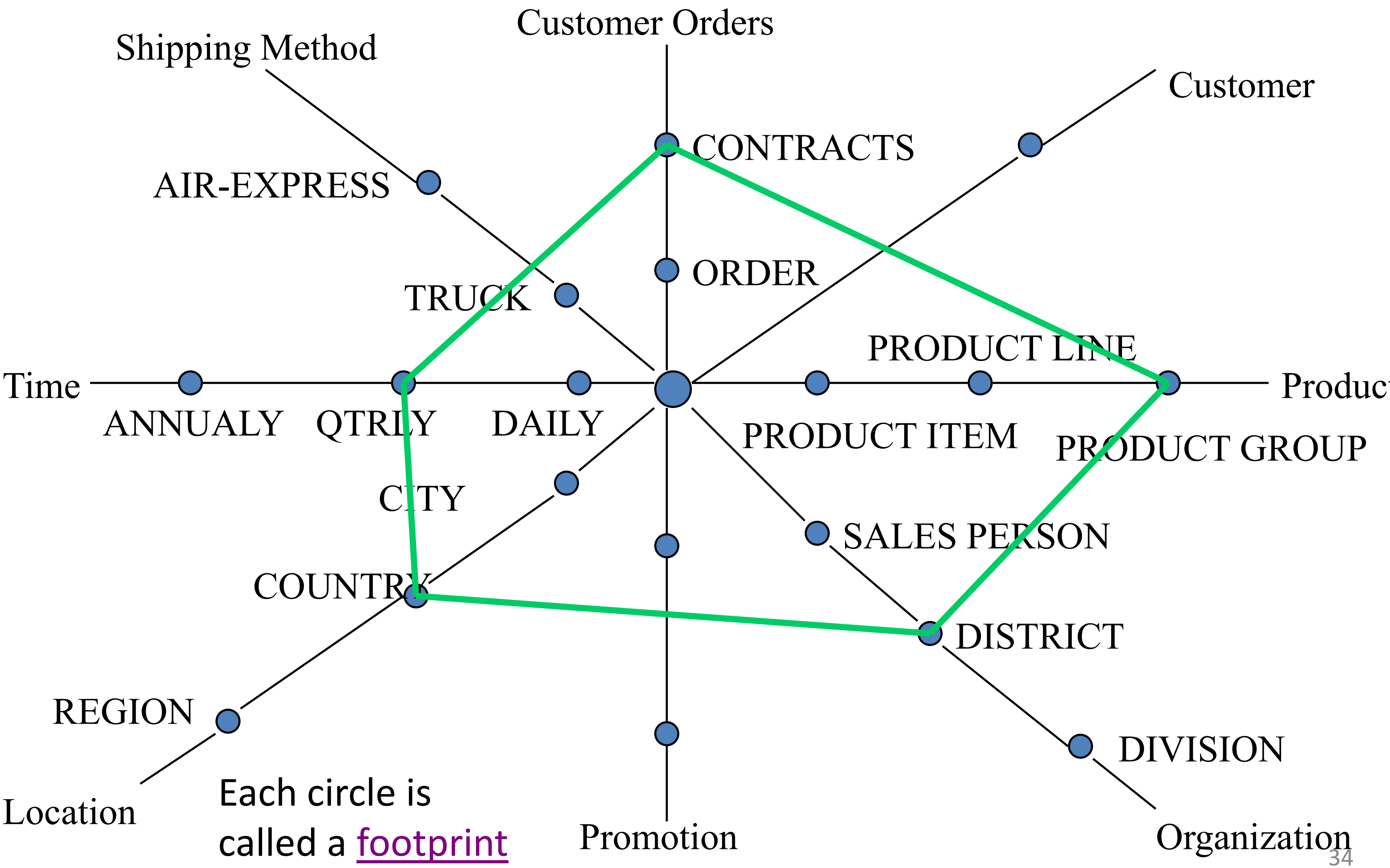

location (cities)
Chicago
New York
Toronto
Vancouver
440
1560
395
time (quarters)
Q1
Q2
Q3
Q4
605
825
14
400
computer
security
home
entertainment
phone
item (types)
drill-down
on time
(from quarters
to months)
location (cities)
Chicago
New York
Toronto
Vancouver
time (months)
January
February
March
April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


location (cities)
Toronto
395
Vancouver
time (quarters)
Q1
605
Q2
computer
home entertainment
item (types)
dice for
(location = “Toronto” or “Vancouver”)
and (time = “Q1” or “Q2”) and
(item = “home entertainment” or “computer”)
location (cities)
Chicago
440
New York
1560
Toronto
395
Vancouver
time (quarters)
Q1
605
825
14
400
Q2
Q3
Q4
computer
security
home entertainment
phone
item (types)



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)

A Star-Net Query Model



Each circle is called a footprint

Data Warehousing and On-line Analytical Processing

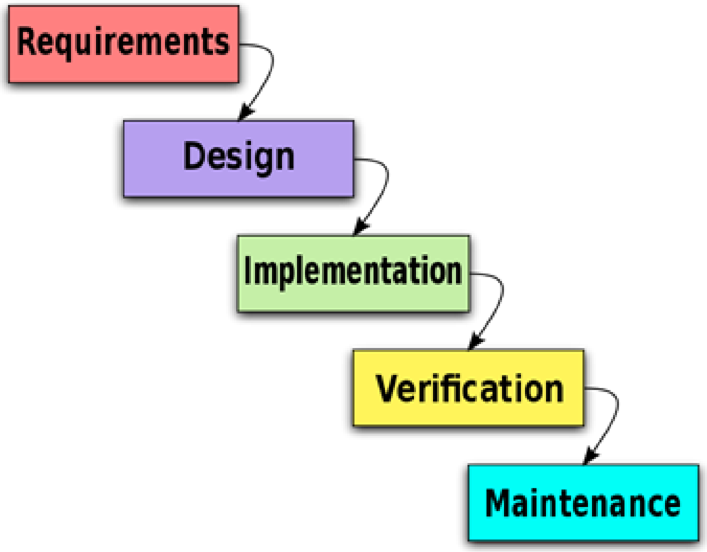
- Data Warehouse: Basic Concepts
- Data Warehouse Modeling: Data Cube and OLAP
- Data Warehouse Design and Usage
- Data Warehouse Implementation
- Data Generalization by Attribute-Oriented Induction
- Data Cube Computation
- Summary

Design of Data Warehouse: A Business Analysis Framework

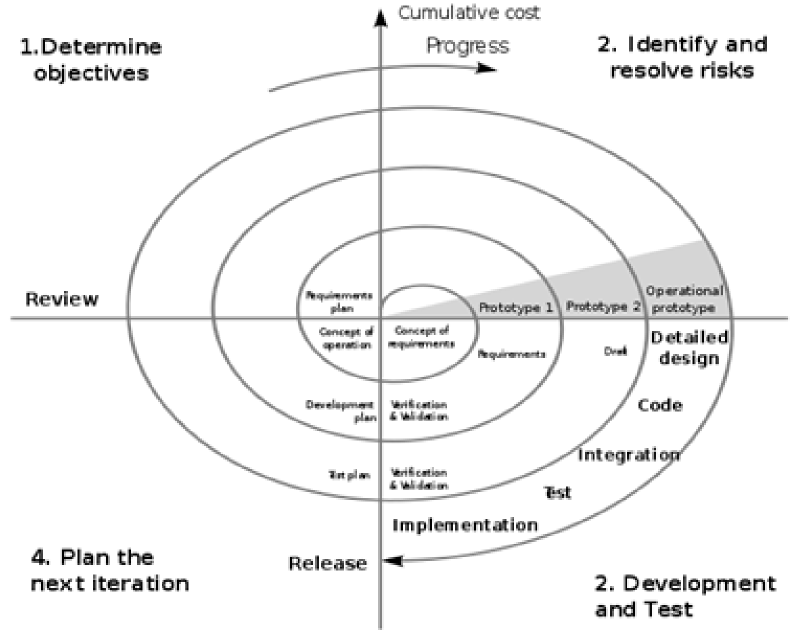
- Four views regarding the design of a data warehouse
  - Top-down view
    - allows selection of the relevant information necessary for the data warehouse
  - Data source view
    - exposes the information being captured, stored, and managed by operational systems (Such as ERD etc)
  - Data warehouse view
    - consists of fact tables and dimension tables
  - Business query view
    - sees the perspectives of data in the warehouse from the view of end-user

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)
- **From software engineering point of view**
  - Waterfall: structured and systematic analysis at each step before proceeding to the next
  - Spiral: rapid generation of increasingly functional systems, short turn around time, quick turn around
- **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



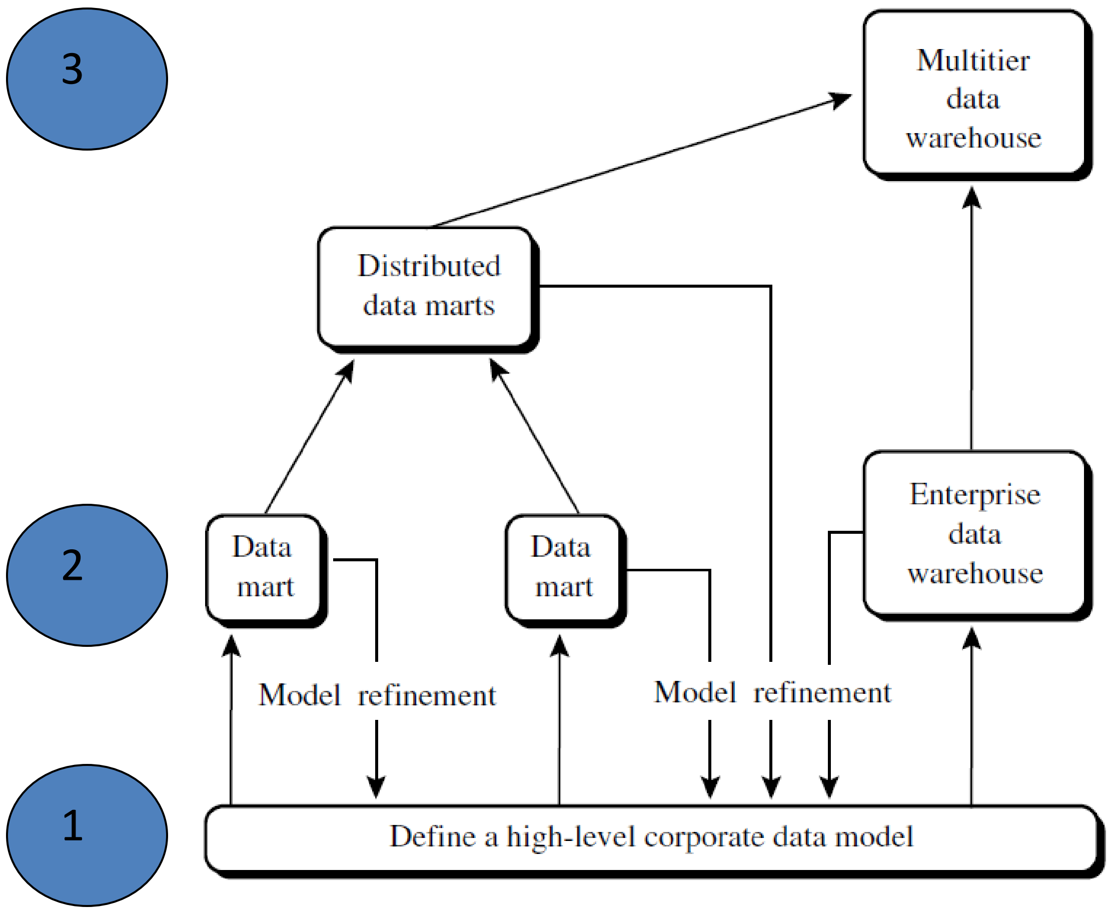
Waterfall Model



Spiral Model

Data Warehouse Development: A Recommended Approach

- Implement the DW in an Incremental and Evolutionary Manner



Data Warehouse Usage

- Three kinds of data warehouse applications
  - Information processing
    - supports querying, basic statistical analysis, and reporting using crosstabs, tables, charts and graphs
  - Analytical processing
    - multidimensional analysis of data warehouse data
    - supports basic OLAP operations, slice-dice, drilling, pivoting
  - Data mining
    - knowledge discovery from hidden patterns
    - supports associations, constructing analytical models, performing classification and prediction, and presenting the mining results using visualization tools

From On-Line Analytical Processing (OLAP) to On Line Analytical Mining (OLAM)

- OLAM integrates OLAP with data mining to uncover knowledge in multidimensional databases.
- Why online analytical mining?
  - High quality of data in data warehouses
    - DW contains integrated, consistent, cleaned data
  - Available information processing structure surrounding data warehouses
    - ODBC, OLEDB, Web accessing, service facilities, reporting and OLAP tools
  - OLAP-based exploratory data analysis
    - Mining with drilling, dicing, pivoting, etc.
  - On-line selection of data mining functions
    - Integration and swapping of multiple mining functions, algorithms, and tasks

Data Warehousing and On-line Analytical Processing

- Data Warehouse: Basic Concepts
- Data Warehouse Modeling: Data Cube and OLAP
- Data Warehouse Design and Usage
- Data Warehouse Implementation
- Data Generalization by Attribute-Oriented Induction
- Data Cube Computation
- Summary

Efficient Data Cube Computation

- Data cube can be viewed as a lattice of cuboids
  - The bottom-most cuboid is the base cuboid
  - The top-most cuboid (apex) contains only one cell
- Materialization of data cube
  - Materialize every (cuboid) (**full materialization**), none (**no materialization**), or some (**partial materialization**)
  - Selection of which cuboids to materialize
    - Based on size, sharing, access frequency, etc.

Indexing OLAP Data: **Bitmap Index**

- Index on a particular column
- Each value in the column has a bit vector: bit-op is fast
- The length of the bit vector: # of records in the base table
- The *i*-th bit is set if the *i*-th row of the base table has the value for the indexed column
- not suitable for high cardinality domains
  - A recent bit compression technique, Word-Aligned Hybrid (WAH), makes it work for high cardinality domain as well [Wu, et al. TODS'06]

Base table

Cust	Region	Type
C1	Asia	Retail
C2	Europe	Dealer
C3	Asia	Dealer
C4	America	Retail
C5	Europe	Dealer

Index on Region

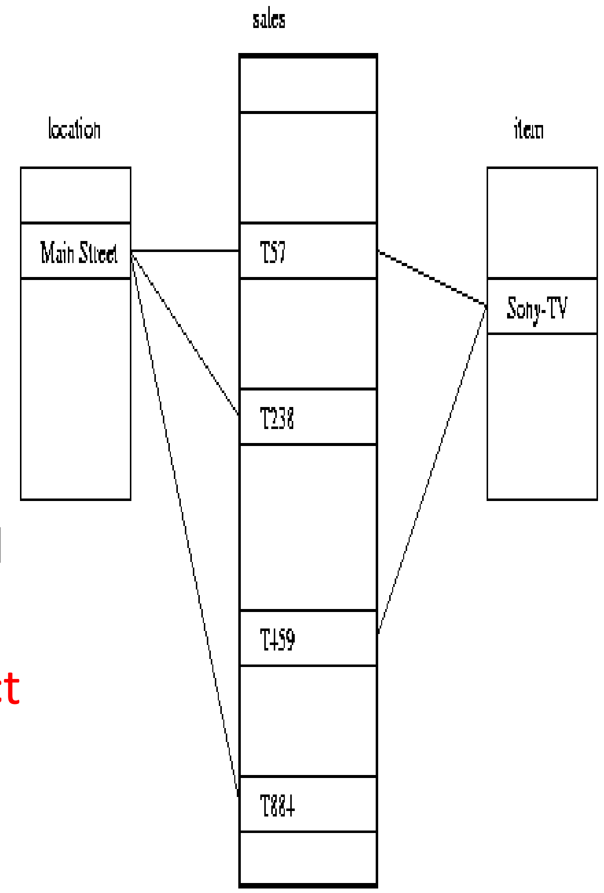
RecID	Asia	Europe	America
1	1	0	0
2	0	1	0
3	1	0	0
4	0	0	1
5	0	1	0

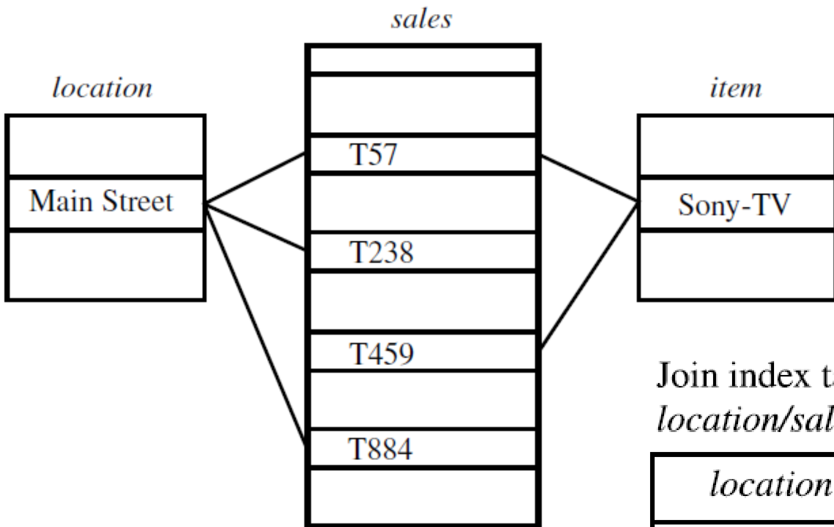
Index on Type

RecID	Retail	Dealer
1	1	0
2	0	1
3	0	1
4	1	0
5	0	1

Indexing OLAP Data: **Join Indices**

- Join index: JI(R-id, S-id) where R (R-id, …) ▷◁ S (S-id, …)
- Traditional indices map the values to a list of record ids
  - It materializes relational join in JI file and speeds up relational join
- In data warehouses, join index relates the values of the dimensions of a start schema to rows in the fact table.
  - E.g. fact table: *Sales* and two dimensions *city* and *product*
    - A join index on *city* maintains for each distinct city a list of R-IDs of the tuples recording the Sales in the city
  - Join indices can span multiple dimensions





Join index table for
location/sales

location	*sales_key*
. . .	. . .
Main Street	T57
Main Street	T238
Main Street	T884
. . .	. . .

Join index table for
item/sales

item	*sales_key*
. . .	. . .
Sony-TV	T57
Sony-TV	T459
. . .	. . .

Join index table linking two dimensions
location/item/sales

location	*item*	*sales_key*
. . .	. . .	. . .
Main Street	Sony-TV	T57
. . .	. . .	. . .

Efficient Processing OLAP Queries

- **Determine which operations** should be performed on the available cuboids
  - Transform drill, roll, etc. into corresponding SQL and/or OLAP operations, e.g., dice = selection + projection

Efficient Processing OLAP Queries (Example)

- Suppose that we define a datacube for ALLElectronics of the form
- sales[time, item, location]: sum(salles_in_euro)
- Dimension hierarchies
  - time: day < month < quater < year
  - Item: item_name < brand < type

- **Determine which materialized cuboid(s)** should be selected for OLAP op.
  - Let the query to be processed be on {*brand, province_or_state*} with the condition "*year = 2004*", and there are 4 materialized cuboids available:

    1) {*year, item_name, city*}

    2) {*year, brand, country*}

    3) {*year, brand, province_or_state*}

    4) {*item_name, province_or_state*} where *year = 2004*

    Which should be selected to process the query?

- Finer granularity data cannot be generated from coarser-granularity data

- Cuboid 2 cannot be used since *country* is more general concept then *province_or_state*
- Cuboids 1, 3, 4 can be used
  - They have the same set or superset of the dimensions of the query
  - The selection clause in the query can imply the selection in the cuboid
  - The abstraction levels for the *item* and location *dimension* in these cuboids are at a finer level than *brand* and *province_or_state*

How would the costs of each cuboid compare?

- Cuboid 1 would cost the most, since both *item_name* and *city* are at a lower level than *brand* and *province_or_state*

- If not many *year* values associated with *items* in the cube, and there are several *item names* for each *brand*, then cuboid 3 will be better than cuboid 4
- Efficient indices available for cuboid 4, cuboid 4 better choice (bitmap indexes)

OLAP Server Architectures

- Relational OLAP (ROLAP)
  - Use relational or extended-relational DBMS to store and manage warehouse data and OLAP middle ware
  - Include optimization of DBMS backend, implementation of aggregation navigation logic, and additional tools and services
  - Greater scalability
- Multidimensional OLAP (MOLAP)
  - Sparse array-based multidimensional storage engine
  - Fast indexing to pre-computed summarized data
- Hybrid OLAP (HOLAP) (e.g., Microsoft SQLServer)
  - Flexibility, e.g., low level: relational, high-level: array

Data Warehousing and On-line Analytical Processing

- Data Warehouse: Basic Concepts
- Data Warehouse Modeling: Data Cube and OLAP
- Data Warehouse Design and Usage
- Data Warehouse Implementation
- Data Generalization by Attribute-Oriented Induction
- Data Cube Computation
- Summary

Attribute-Oriented Induction

- How it is done?
  - Collect the task-relevant data (*initial relation*) using a relational database query
  - Perform generalization by <u>attribute removal</u> or <u>attribute generalization</u>
  - Apply aggregation by merging identical, generalized tuples and accumulating their respective counts
  - Interaction with users for knowledge presentation

Attribute-Oriented Induction: An Example

Example: Describe general characteristics of graduate students in the University database

- Step 1. Fetch relevant set of data using an SQL statement, e.g.,

  Select * (i.e., name, gender, major, birth_place, birth_date, residence, phone#, gpa)

  from student

  where student_status in {“Msc”, “MBA”, “PhD” }
- Step 2. Perform attribute-oriented induction
- Step 3. Present results in generalized relation, cross-tab, or rule forms

Class Characterization: An Example

Initial Relation

Name	Gender	Major	Birth-Place	Birth_date	Residence	Phone #	GPA
Jim Woodman	M	CS	Vancouver,BC, Canada	8-12-76	3511 Main St., Richmond	687-4598	3.67
Scott Lachance	M	CS	Montreal, Que, Canada	28-7-75	345 1st Ave., Richmond	253-9106	3.70
Laura Lee	F	Physics	Seattle, WA, USA	25-8-70	125 Austin Ave., Burnaby	420-5232	3.83
…	…	…	…	…	…	…	…
Removed	Retained	Sci,Eng, Bus	Country	Age range	City	Removed	Excl, VG,..

Prime Generalized Relation

Gender	Major	Birth_region	Age_range	Residence	GPA	Count
M	Science	Canada	20-25	Richmond	Very-good	16
F	Science	Foreign	25-30	Burnaby	Excellent	22
…	…	…	…	…	…	…

Gender \ Birth_Region	Canada	Foreign	Total
M	16	14	30
F	10	22	32
Total	26	36	62

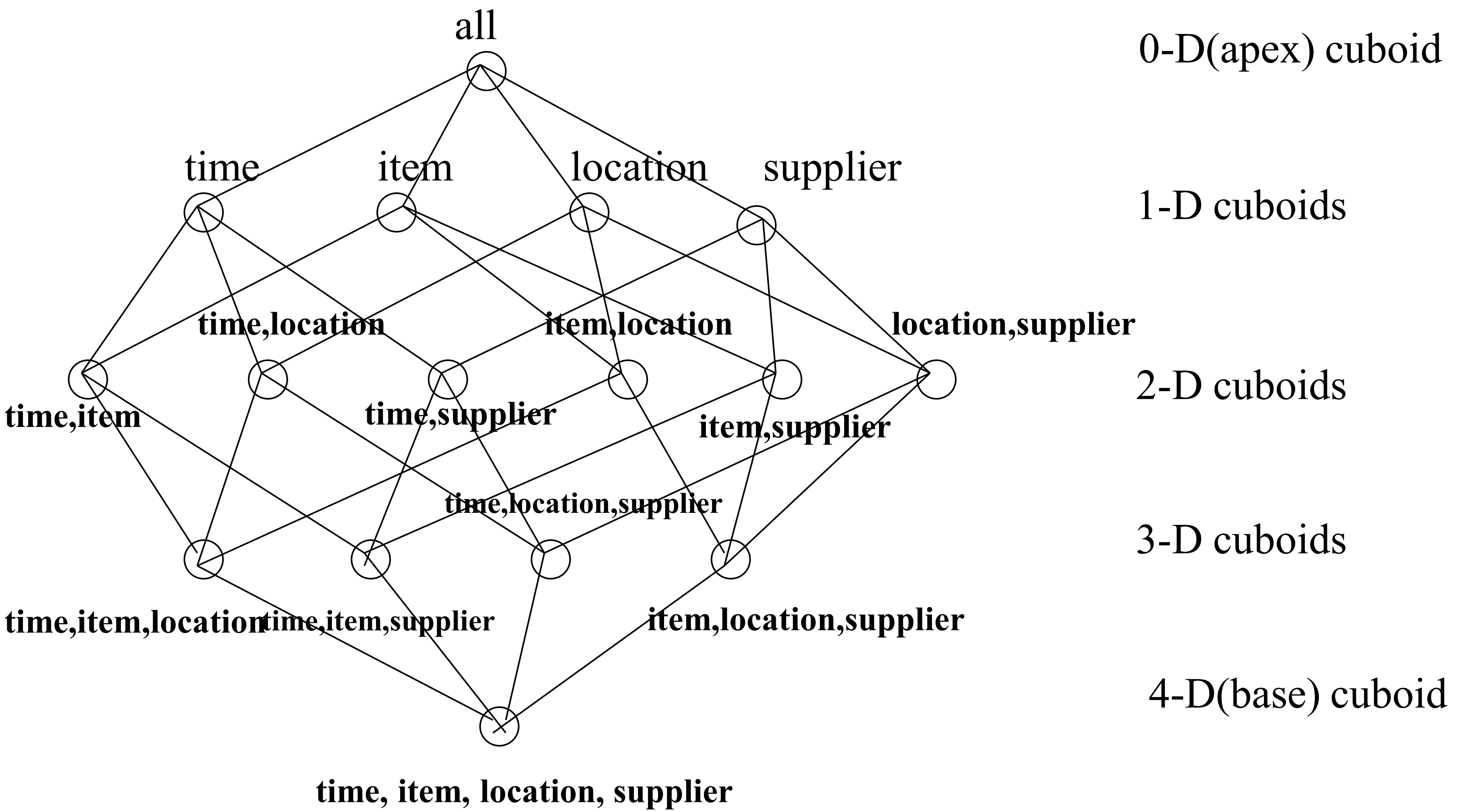
Basic Principles of Attribute-Oriented Induction

- Data focusing: task-relevant data, including dimensions, and the result is the *initial relation*
- Attribute-removal: remove attribute *A* if there is a large set of distinct values for *A* but (1) there is no generalization operator on *A*, or (2) *A*'s higher level concepts are expressed in terms of other attributes
- Attribute-generalization: If there is a large set of distinct values for *A*, and there exists a set of generalization operators on *A*, then select an operator and generalize *A*
- Attribute-threshold control: (number of distinct values)
  - typical 2-8, specified/default
- Generalized relation threshold control: control the final relation/rule size

Data Warehousing and On-line Analytical Processing

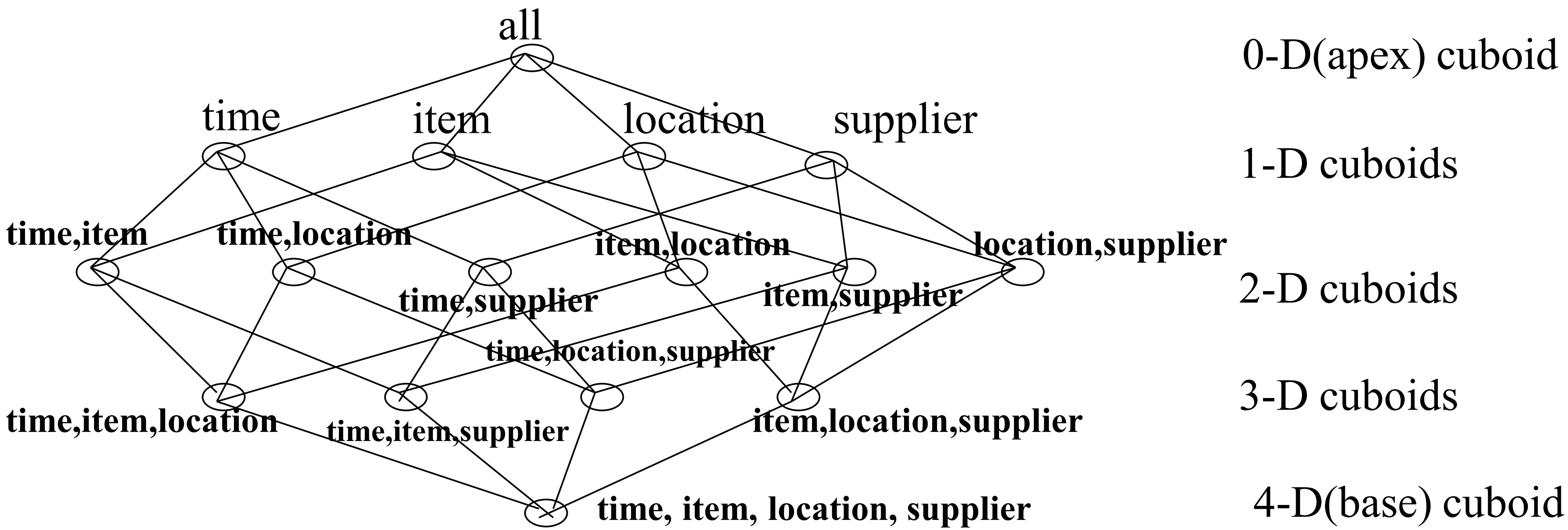
- Data Warehouse: Basic Concepts
- Data Warehouse Modeling: Data Cube and OLAP
- Data Warehouse Design and Usage
- Data Warehouse Implementation
- Data Generalization by Attribute-Oriented Induction
- Data Cube Computation
- Summary

Data Cube: A Lattice of Cuboids



59

Data Cube: A Lattice of Cuboids



- Base vs. aggregate cells; ancestor vs. descendant cells; parent vs. child cells
  1. (9/15, milk, Urbana, Dairy_land)
  2. (9/15, milk, Urbana, *)
  3. (*, milk, Urbana, *)
  4. (*, milk, Urbana, *)
  5. (*, milk, Chicago, *)
  6. (*, milk, *, *)

Cube Materialization

- Compute data cubes in advance
  - Readily available for query processing
- Cube materialization (i.e., pre-computation)
- Precompute the full cube (i.e., all the cells of all of the cuboids for a given data cube).
  - N dimensions -> 2^N
  - Even more cuboids if we consider concept hierarchies for each dimension
  - Size of each cuboid depends on the cardinality of its dimensions.
    - Huge and often excessive amounts of memory.

Partial materialization

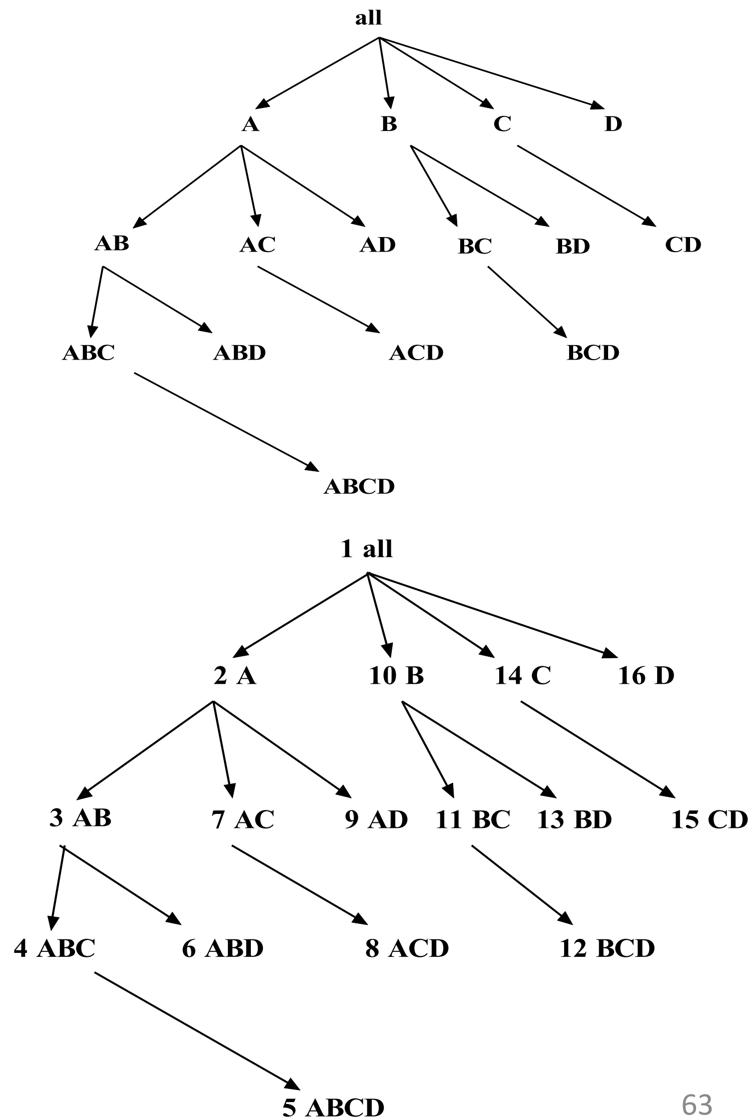
- Instead of computing the full cube, we can compute only
  - A subset of the data cube's cuboids,

  or

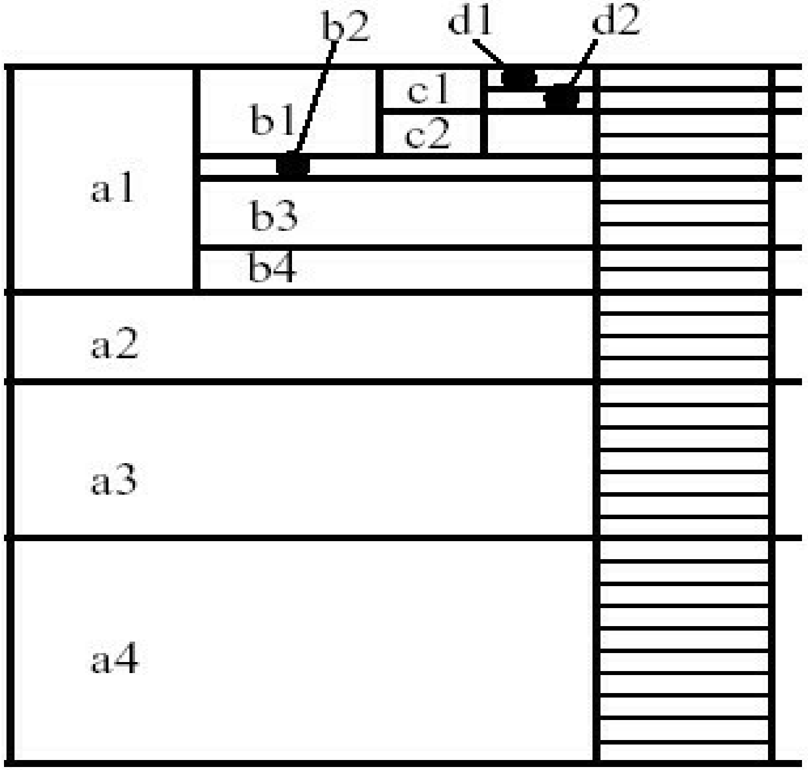
  - Subcubes consisting of subsets of cells from the various cuboids.
- Trade-off between storage space and response time for OLAP.

Bottom-Up Computation (BUC)

- Bottom-up cube computation
  (Note: top-down in our view!)
- Divides dimensions into partitions and facilitates iceberg pruning
  - If a partition does not satisfy *min_sup*, its descendants can be pruned
  - If *minsup* = 1 $\Rightarrow$ compute full CUBE!
- No simultaneous aggregation



BUC: Partitioning



- Usually, entire data set can't fit in main memory
- Sort *distinct* values
  - partition into blocks that fit
- Continue processing

- Performance of BUC is sensitive to the order of the dimensions and to skew in the data.
  - The most discriminating dimensions should be processed first
  - More partitions greater opportunity for pruning

Data Warehousing and On-line Analytical Processing

- Data Warehouse: Basic Concepts
- Data Warehouse Modeling: Data Cube and OLAP
- Data Warehouse Design and Usage
- Data Warehouse Implementation
- Data Generalization by Attribute-Oriented Induction
- Data Cube Computation
- Summary

Summary

- Data warehousing: A multi-dimensional model of a data warehouse
  - A data cube consists of *dimensions* & *measures*
  - Star schema, snowflake schema, fact constellations
  - OLAP operations: drilling, rolling, slicing, dicing and pivoting
- Data Warehouse Architecture, Design, and Usage
  - Multi-tiered architecture
  - Business analysis design framework
  - Information processing, analytical processing, data mining, OLAM (Online Analytical Mining)
- Implementation: Efficient computation of data cubes
  - Partial vs. full vs. no materialization
  - Indexing OALP data: Bitmap index and join index
  - OLAP query processing
  - OLAP servers: ROLAP, MOLAP, HOLAP
- Data generalization: Attribute-oriented induction

Summary

- Data Cube Computation: Preliminary Concepts
- Data Cube Computation Methods
  - MultiWay Array Aggregation
  - BUC

Quiz 1

Suppose that a data warehouse consists of the three dimensions *time, doctor*, and *patient*, and the two measures *count* and *charge*, where *charge* is the fee that a doctor charges a patient for a visit.

(a) Enumerate three classes of schemas that are popularly used for modeling data warehouses.

(b) Draw a schema diagram for the above data warehouse using one of the schema classes listed in (a).

(c) Starting with the base cuboid $[day, doctor, patient]$, what specific *OLAP operations* should be performed in order to list the total fee collected by each doctor in 2004?

(d) To obtain the same list, write an SQL query assuming the data is stored in a relational database with the schema *fee* (*day, month, year, doctor, hospital, patient, count, charge*).

Quiz 2

- Apply *attribute-oriented induction On the following table:*

class	*birth_place*	count
Programmer	USA	180
	others	120
DBA	USA	20
	others	80

References (I)

- S. Agarwal, R. Agrawal, P. M. Deshpande, A. Gupta, J. F. Naughton, R. Ramakrishnan, and S. Sarawagi. On the computation of multidimensional aggregates. VLDB'96
- D. Agrawal, A. E. Abbadi, A. Singh, and T. Yurek. Efficient view maintenance in data warehouses. SIGMOD'97
- R. Agrawal, A. Gupta, and S. Sarawagi. Modeling multidimensional databases. ICDE'97
- S. Chaudhuri and U. Dayal. An overview of data warehousing and OLAP technology. ACM SIGMOD Record, 26:65-74, 1997
- E. F. Codd, S. B. Codd, and C. T. Salley. Beyond decision support. Computer World, 27, July 1993.
- J. Gray, et al. Data cube: A relational aggregation operator generalizing group-by, cross-tab and sub-totals. Data Mining and Knowledge Discovery, 1:29-54, 1997.
- A. Gupta and I. S. Mumick. Materialized Views: Techniques, Implementations, and Applications. MIT Press, 1999.
- J. Han. Towards on-line analytical mining in large databases. *ACM SIGMOD Record*, 27:97-107, 1998.
- V. Harinarayan, A. Rajaraman, and J. D. Ullman. Implementing data cubes efficiently. SIGMOD'96
- J. Hellerstein, P. Haas, and H. Wang. Online aggregation. SIGMOD'97

References (II)

- C. Imhoff, N. Galemmo, and J. G. Geiger. Mastering Data Warehouse Design: Relational and Dimensional Techniques. John Wiley, 2003
- W. H. Inmon. Building the Data Warehouse. John Wiley, 1996
- R. Kimball and M. Ross. The Data Warehouse Toolkit: The Complete Guide to Dimensional Modeling. 2ed. John Wiley, 2002
- P. O'Neil and G. Graefe. Multi-table joins through bitmapped join indices. *SIGMOD Record*, 24:8–11, Sept. 1995.
- P. O'Neil and D. Quass. Improved query performance with variant indexes. SIGMOD'97
- Microsoft. OLEDB for OLAP programmer's reference version 1.0. In http://www.microsoft.com/data/oledb/olap, 1998
- S. Sarawagi and M. Stonebraker. Efficient organization of large multidimensional arrays. ICDE'94
- A. Shoshani. OLAP and statistical databases: Similarities and differences. PODS'00.
- D. Srivastava, S. Dar, H. V. Jagadish, and A. V. Levy. Answering queries with aggregation using views. *VLDB'96*
- P. Valduriez. Join indices. ACM Trans. Database Systems, 12:218-246, 1987.
- J. Widom. Research problems in data warehousing. CIKM'95
- K. Wu, E. Otoo, and A. Shoshani, Optimal Bitmap Indices with Efficient Compression, ACM Trans. on Database Systems (TODS), 31(1): 1-38, 2006