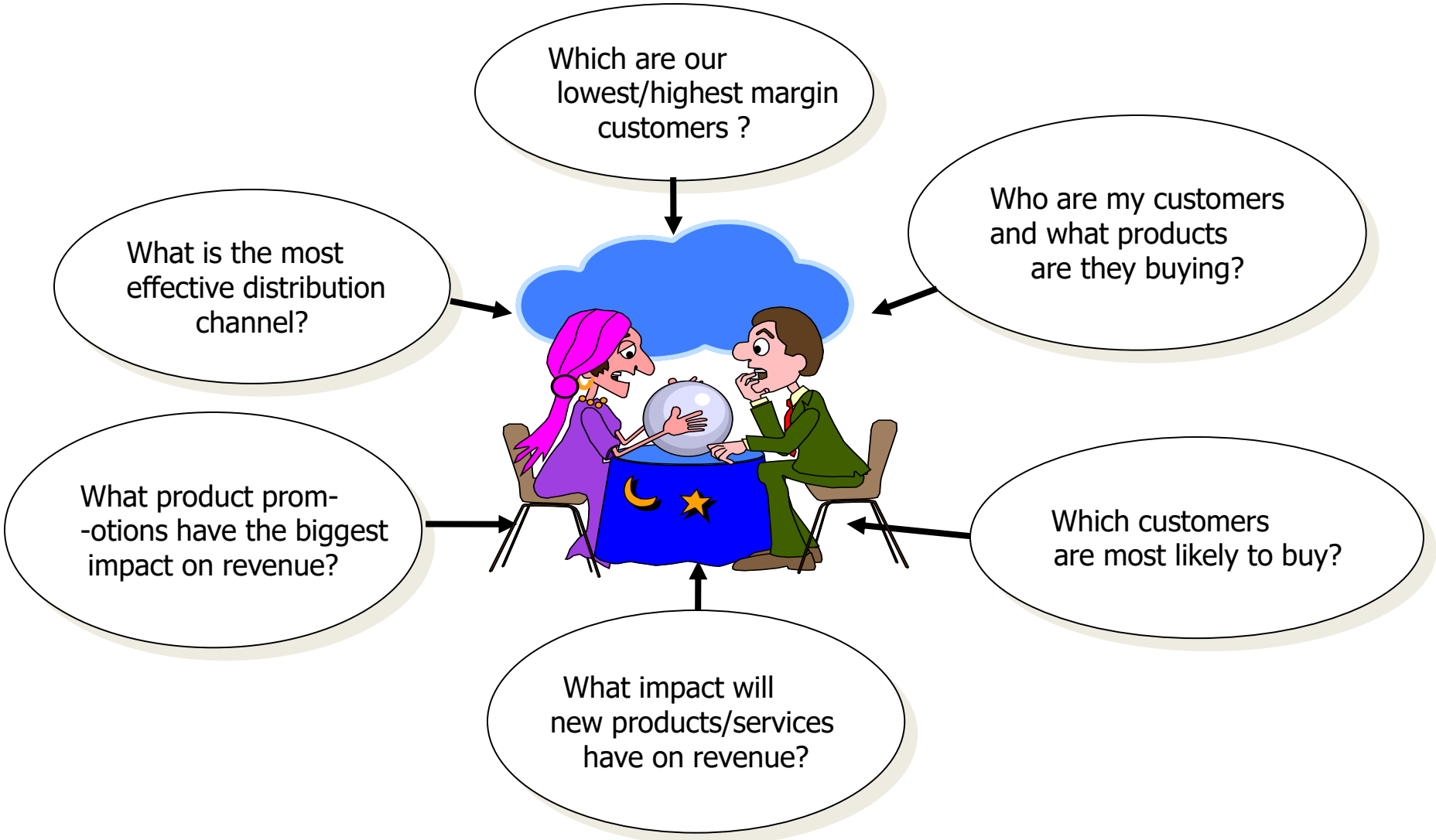
MIT-WPU
॥ विश्वशान्तिर्ध्रुवं ध्रुवा ॥
Dr. Vishwanath Karad
MIT WORLD PEACE UNIVERSITY | PUNE
TECHNOLOGY, RESEARCH, SOCIAL INNOVATION & PARTNERSHIPS

(Computer Engineering and Technology) (TYB.Tech)

UNIT II- Data Warehouse

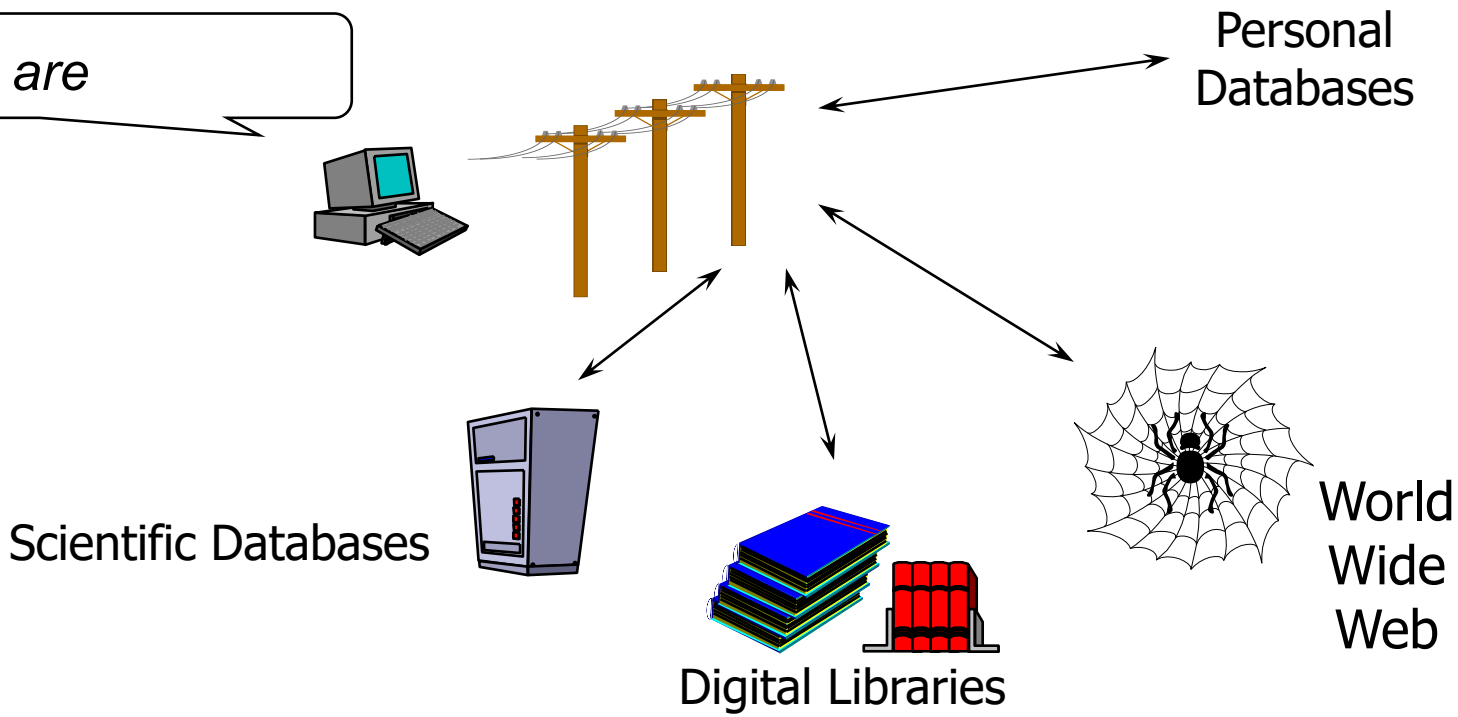
- **DATA WAREHOUSE**
- Introduction to Data Warehouse
- OLTP and OLAP
- Data Warehouse architecture
- Data Warehouse Modeling: A Multidimensional Data Model
- Data Warehouse Design: Stars, Snowflakes, and Fact Constellations Schemas
- Dimensions: The Role of Concept Hierarchies
- Measures: Their Categorization and Computation
- Typical OLAP Operations, ROLAP, MOLAP,
- Materialized views
- Integration of Data Warehouse with other technologies

Scenario:A manager wants to know....



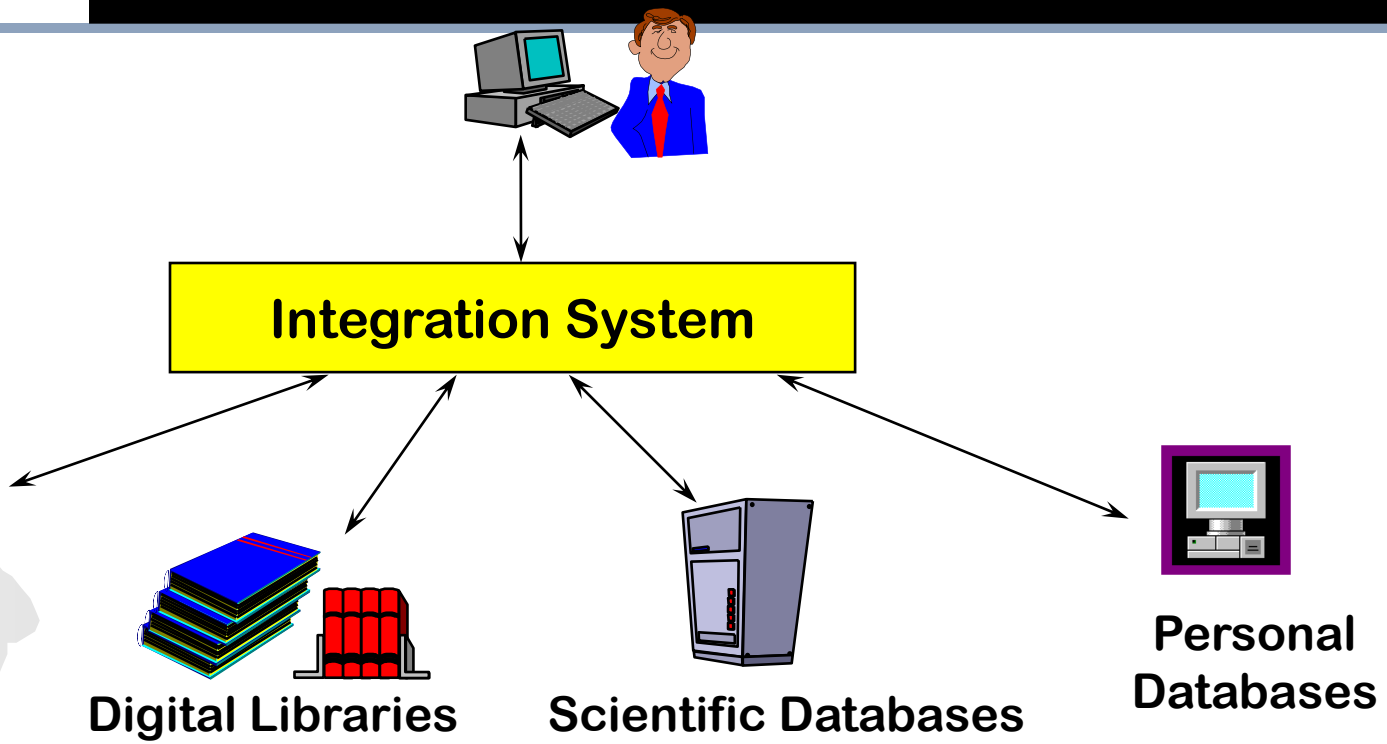
Problem: Heterogeneous Information Sources

"Heterogeneities are everywhere"



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

Unified Access to Data



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

Data, Data everywhere yet ...

- Can't find the data needed
  - data is scattered over the network
  - many versions, subtle differences
- Can't get the data needed
  - need an interface to get the data
- Can't understand the data found
  - available data poorly documented
- Can't use the data found
  - data needs to be transformed from one form to other

What is Data Warehouse?

- A single, complete and consistent store of data obtained from a variety of different sources, made available to end users in a way they can understand and use in a business context. [Barry Devlin]
- A decision support database that is maintained separately from the organization's operational databases.
- 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

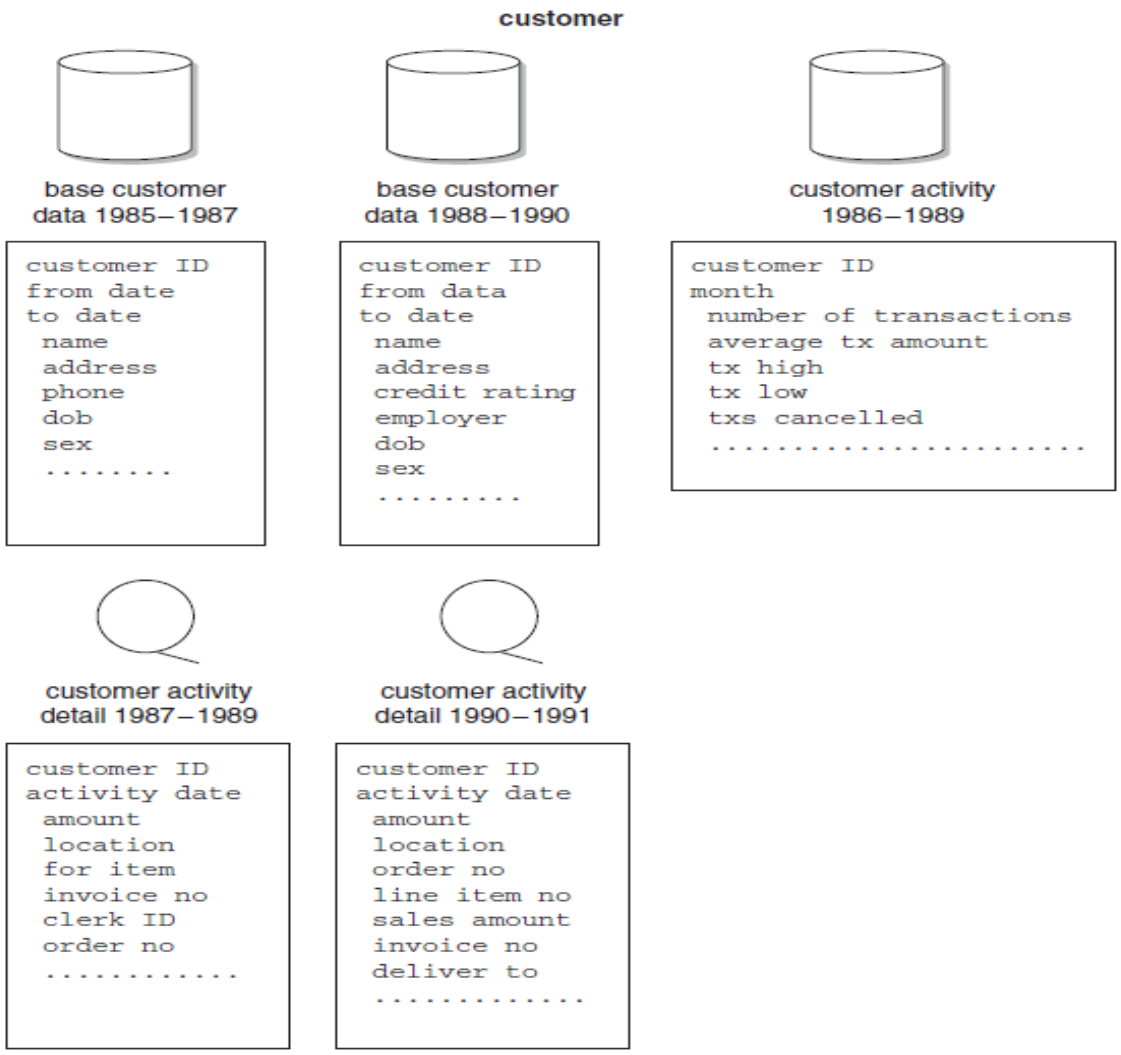
Data Warehouse -Subject-Oriented

- Organized around major subjects, such as customers, products, sales, revenue, etc
- 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

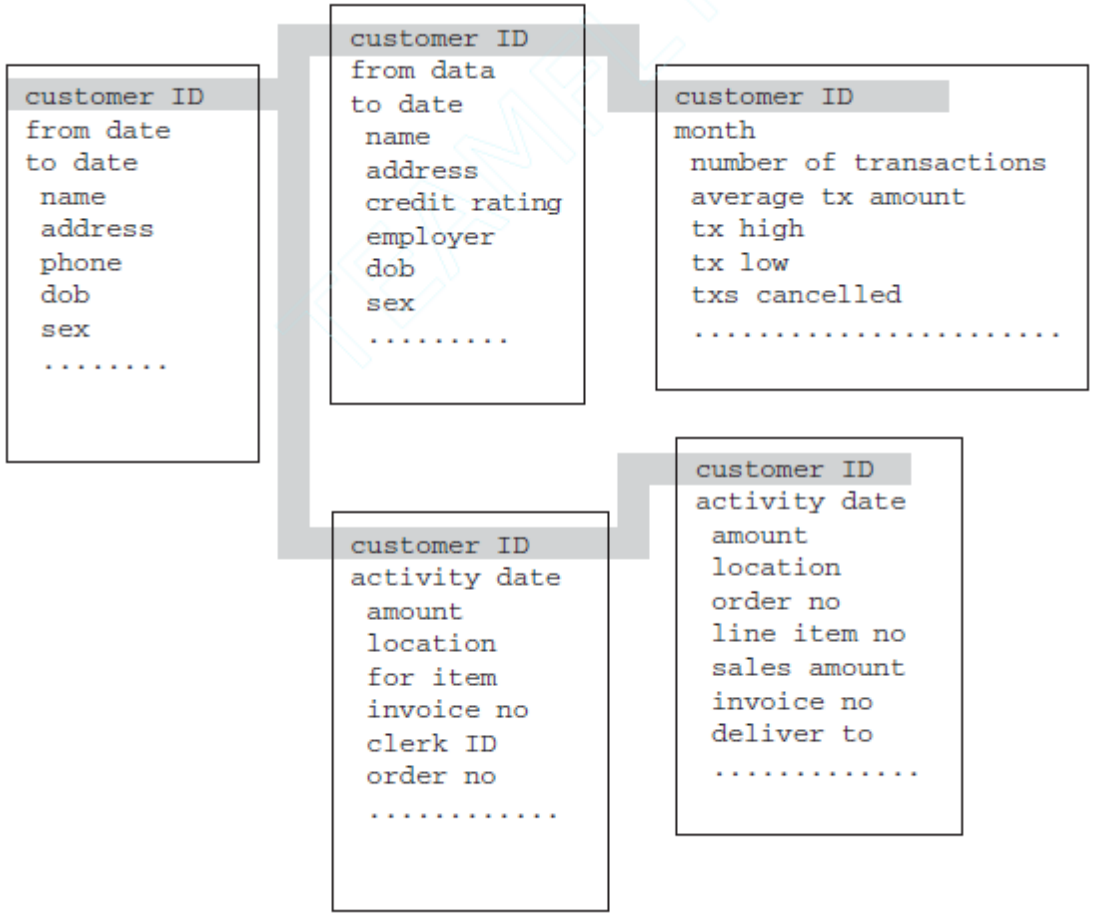
Example : Subject area—customer

- There is a base table for customer information as defined from 1985 to 1987.
- There is another for the definition of customer data between 1988 and 1990.
- There is a cumulative customer activity table for activities between 1986 and 1989.
- Each month a summary record is written for each customer record based on customer activity for the month.
- There are detailed activity files by customer for 1987 through 1989 and another one for 1990 through 1991.
- Definition of data in the files is different, based on the year.
- All of the physical tables and files for the customer subject area are related by a common key.
- Figure shows that the key—customer ID—connects all of the data

Example : Subject area—customer



Example :
Subject area—customer



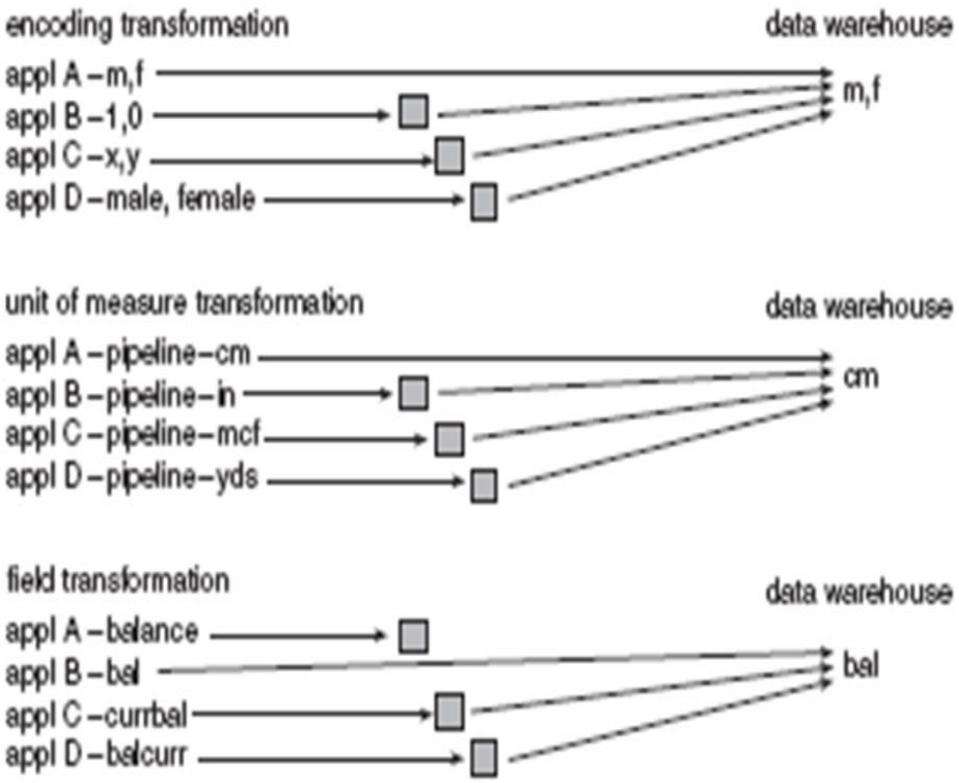
Data Warehouse—Integrated

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

Data Warehouse—Integrated

- Across multiple applications there is no application consistency in encoding, naming conventions, physical attributes, measurement of attributes, and so forth.
- Each application designer has had free rein to make his or her own design decisions.

Data Warehouse—Integrated



Data Warehouse—Time Variant

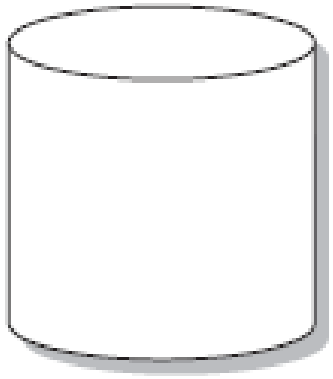
- The time horizon for the data warehouse is significantly longer than that of operational systems
  - Operational database: current value data
  - Data warehouse : provide information from a historical perspective (e.g., past 5-10 years)
- Every key structure in the data warehouse
  - Contains an element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

Data Warehouse—Time Variant

- Different environments have different time horizons
- Collective time horizon for the data found inside a data warehouse is significantly longer than that of operational systems.
- 60-to-90-day time horizon is normal for operational systems
- 5-to-10-year time horizon is normal for the data warehouse
- As a result of this difference in time horizons, the data warehouse contains *much* more history than any other environment

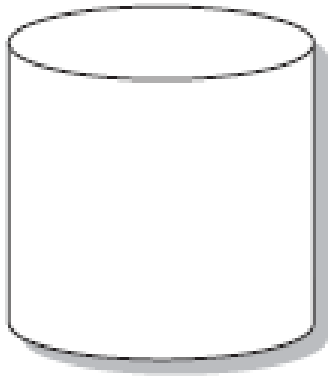
Data Warehouse—Time Variant

operational



- time horizon—current to 60–90 days
- update of records
- key structure may/may not contain an element of time

data warehouse



- time horizon—5–10 years
- sophisticated snapshots of data
- key structure contains an element of time

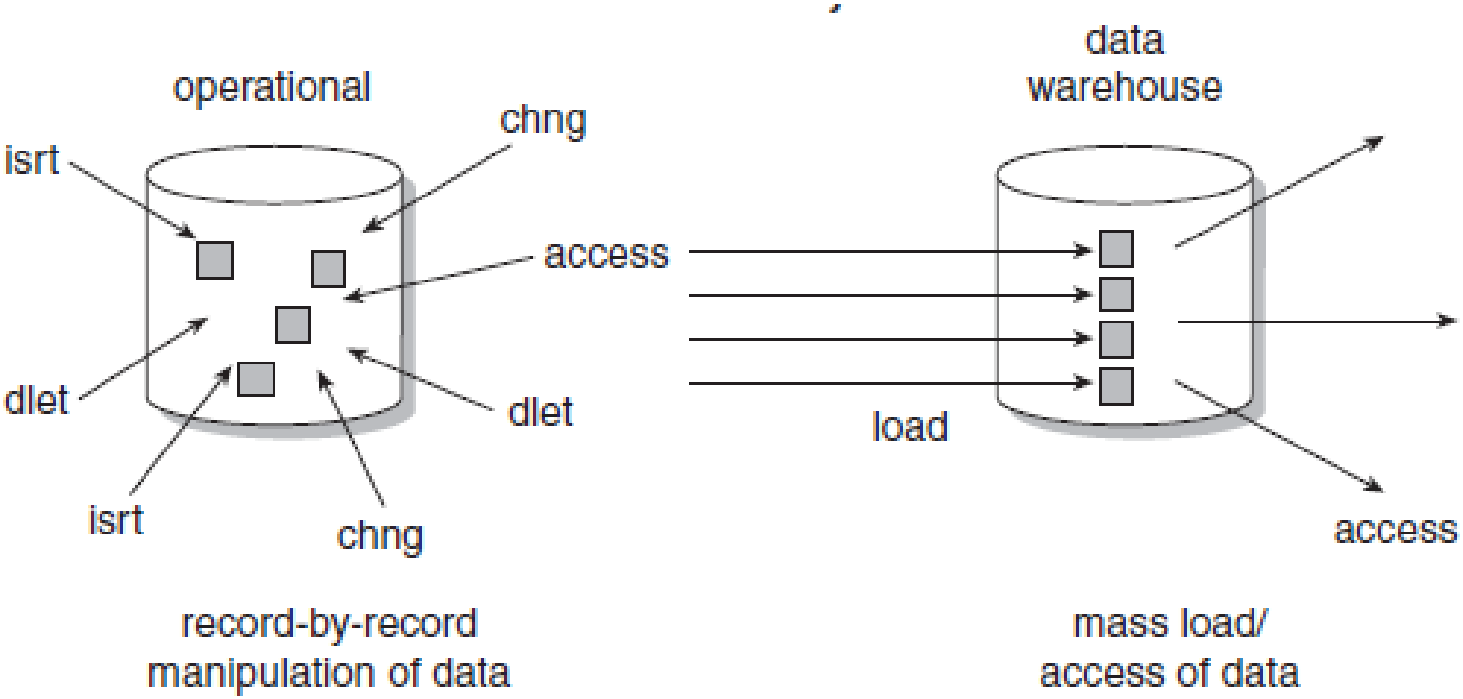
Data Warehouse—Nonvolatile

- 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*

Data Warehouse—Nonvolatile

- Operational data is regularly accessed and manipulated one record at a time.
- Data is updated in the operational environment as a regular matter of course, but data warehouse data exhibits a very different set of characteristics.
- Data warehouse data is loaded and accessed, but it is not modified
- Instead, when data in the data warehouse is loaded, it is loaded in a snapshot, static format
- When subsequent changes occur, a new snapshot record is written
- In doing so a history of data is kept in the data warehouse.

Data Warehouse—Nonvolatile



Data Warehouse: Summary

- Stores the information an enterprise needs to make strategic decisions
- Viewed as an architecture, constructed by integrating data from multiple heterogeneous sources
- Supports complex queries, analytical reporting and decision making
- Allows “knowledge workers” (e.g., managers, analysts and executives) to use the warehouse

Data Warehouse: Summary

How are organizations using the information from data warehouses?

To support business decision-making activities, including

(1) increasing **customer focus**, which includes the analysis of customer buying patterns (such as buying preference, buying time, budget cycles, and appetites for spending)

(2) **repositioning products** and managing product portfolios by comparing the performance of sales by quarter, by year, and by geographic regions in order to fine-tune production strategies

(3) analyzing operations and looking for **sources of profit**

(4) managing customer relationships, making environmental corrections, and managing the cost of corporate assets

Data Warehouse

Applications

Industry	Application
Finance	Credit card Analysis
Insurance	Claims, Fraud Analysis
Telecommunication	Call record Analysis
Transport	Logistics management
Consumer goods	Promotion Analysis

Data Warehouse vs. Traditional DBMS

- Traditional heterogeneous DB integration: A query driven approach
  - Build wrappers/mediators on top of heterogeneous databases
  - When a query is posed, a meta-dictionary is used to translate the query into queries appropriate for individual heterogeneous sites involved, and the results are integrated into a global answer set
  - Complex information filtering, integration processes & competes with local sites for processing resources
  - Inefficient and potentially expensive for frequent queries, especially queries requiring aggregations (complex queries)

Data Warehouse vs. Traditional DBMS

- Data warehouse: An update-driven approach in which
  - Information from multiple, heterogeneous sources is **integrated in advance** and stored in warehouse for direct querying and analysis
  - Data warehouse don't contain **most current information**
  - **Brings high performance** to the integrated heterogeneous database system because data are copied, preprocessed, integrated, annotated, summarized, and restructured into single data store.
  - Query processing in data warehouses **does not interfere** with the processing at local sources
  - Moreover, data warehouses can store and integrate **historic information** and support complex multidimensional queries

Data Warehouse vs. Operational DBMS

- OLTP (on-line transaction processing)
  - Major task of online operational database systems is to perform online transaction and query processing
  - Such systems are called **OLTP** systems
  - Covers day-to-day operations: purchasing, inventory, banking, manufacturing, payroll, registration, accounting, etc.

- OLAP (on-line analytical processing)
  - Serves knowledge workers in the role of data analysis and decision making
  - Such systems are known as **OLAP** systems

OLTP vs OLAP

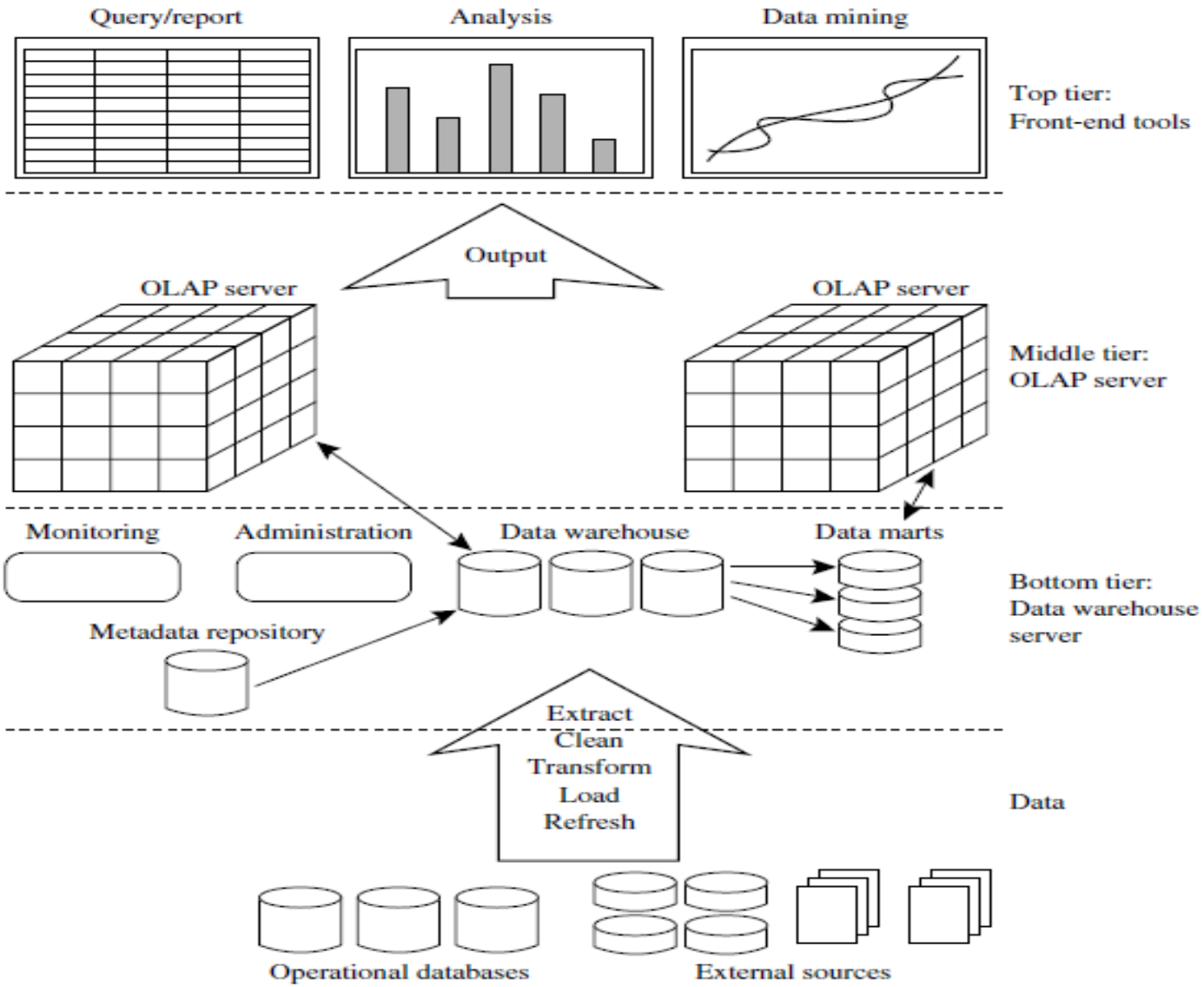
	OLTP	OLAP
users	clerk, IT professional	knowledge worker
function	day to day operations	decision support
DB design	application-oriented	subject-oriented
data	current, up-to-date detailed, relational isolated	historical, summarized, multidimensional integrated, consolidated
usage	repetitive	ad-hoc
access	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 Separate Data Warehouse?

- To promote **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
  - Processing OLAP queries in operational databases would substantially degrade the performance of operational tasks
- Different functions and different data:
  - missing data: Decision support requires historical data which operational DBs do not typically maintain
  - data consolidation: Decision support 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

- Performance: Data warehouses are designed for analytical workloads, which are typically very different from the transactional workloads that operational systems support. By separating the data warehouse from operational systems, we can ensure that both systems can perform optimally.
- Scalability: Data warehouses are often very large and complex systems. By separating the data warehouse from operational systems, we can make it easier to scale the data warehouse to meet the needs of the business.
- Security: Data warehouses often contain sensitive data. By separating the data warehouse from operational systems, we can make it more difficult for unauthorized users to access the data.
- Data quality: Data warehouses typically contain data from a variety of sources, some of which may be of poor quality. By separating the data warehouse from operational systems, we have a chance to clean and transform the data before it is loaded into the data warehouse.
- Flexibility: Data warehouses are designed to be flexible and adaptable to changing business needs. By separating the data warehouse from operational systems, we can make it easier to make changes to the data warehouse without impacting operational systems.

DataWarehousing: A Three-tier Architecture



The three-tier architecture of a data warehouse is a logical model that divides the data warehouse into three layers:

Bottom tier: The bottom tier is the data storage layer. It is responsible for storing the raw data that is extracted from operational systems and external data sources. The bottom tier is typically implemented using a relational database management system (RDBMS).

Middle tier: The middle tier is the data processing layer. It is responsible for transforming the raw data into a format that is suitable for analysis. This includes tasks such as cleaning the data, removing duplicates, and aggregating the data. The middle tier is typically implemented using an online analytical processing (OLAP) server.

- Top tier: The top tier is the presentation layer. It is responsible for providing users with access to the data and for displaying the results of data analysis. The top tier is typically implemented using a variety of tools, such as reporting tools, data mining tools, and analytical dashboards.
- Data flow: The data flows through the three tiers as follows:
-
- Data is extracted from operational systems and external data sources.
- The data is loaded into the bottom tier, where it is stored in a raw format.
- The data is transformed in the middle tier into a format that is suitable for analysis.
- The data is loaded into the top tier, where it is accessible to users for analysis and reporting.
-

- Benefits of three-tier architecture:
- Improved performance: The three-tier architecture separates the data storage, data processing, and data presentation layers. This allows each tier to be optimized for its specific purpose. This can lead to improved performance for data analysis and reporting.
- Scalability: The three-tier architecture is scalable. This means that it can be easily expanded to accommodate more data and more users.
- Security: The three-tier architecture can improve data security. This is because the data is only accessible to users who have authorized access to the top tier.

Data Warehouse Models

- Enterprise warehouse
  - Collects all of the information about subjects spanning the entire organization
  - Has greater size
  - Implemented on traditional mainframes, computer superservers, or parallel architecture platforms

It is a centralized data repository that stores data from all of an organization's operational systems. EDWs are typically used for complex analytics and reporting across 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
  - Implemented on low-cost departmental servers
  - Independent vs. dependent (directly from warehouse) data mart

Depending on the source of data, data marts can be categorized as independent or dependent. Independent data marts are sourced from data captured from one or more operational systems or external information providers, or from data generated locally within a particular department or geographic area. Dependent data marts are sourced directly from enterprise data warehouses.

A data mart is a subset of an EDW that is focused on a specific business area, such as sales or marketing. Data marts are typically used for more focused analytics and reporting within a specific business unit.

Data Warehouse Models

- Virtual warehouse
  - Set of views over operational databases
  - Only some of the possible summary views may be materialized
  - Easy to build but requires excess capacity on operational database servers

A virtual data warehouse is a logical data warehouse that does not physically store data. Instead, it creates a unified view of data from multiple underlying data sources. Virtual data warehouses are typically used for businesses that need to access data from a variety of sources, such as cloud-based applications and on-premises databases.

- Cloud-based data warehouse
  - They can typically perform complex analytical queries much faster because they are massively parallel processing (MPP).
  - There is no need to purchase physical hardware.
  - It's quicker and cheaper to set up and scale cloud data warehouses

Data Warehouse Back-End Tools and Utilities

- **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

- Metadata is the **data defining warehouse objects**.
- Created for the **data names and definitions** of the given warehouse
- Also for **timestamping** any extracted data, the **source of the extracted data**, and **missing fields** that have been added by data cleaning or integration processes

A metadata repository in a data warehouse is a centralized database that stores information about the data in the data warehouse. This information includes the data's structure, lineage, and business context. The metadata repository is used by a variety of tools and applications to access and manage the data in the data warehouse.

Benefits of Data Repository

- Improved data quality: The metadata repository can help to ensure that the data in the data warehouse is consistent, accurate, and complete.
- Reduced development time: The metadata repository can help to reduce the time it takes to develop new data warehouse applications and reports.
- Improved data governance: The metadata repository can help to improve data governance by providing a central repository for all data-related information.

Metadata Repository

Contains following

- **Description of the structure of the data warehouse**
  - schema, view, dimensions, hierarchies, derived data definition, data mart locations and contents
- **Operational metadata**
  - **data lineage** (history of migrated data and sequence of transformations performed), **currency of data** (active, archived, or purged), **monitoring information** (warehouse usage statistics, error reports, etc)

Algorithms used for summarization(which include measure and dimension,definition algorithms, data on granularity, partitions, subject areas, aggregation,summarization, and predefined queries and reports.)

Mapping from operational environment to the data warehouse(which includes source databases and their contents, gateway descriptions, data partitions, data extraction, cleaning, transformation rules and security (user authorization and access control)

- **Data related to system performance**
  - Improve data access and retrieval performance, in addition to rules for the scheduling of refresh, etc
- **Business data**
  - business terms and definitions, ownership of data, charging policies

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 (n-dimensional)
- A data cube, such as sales, allows data to be modeled and viewed in multiple dimensions
- Defined by dimensions and facts

Dimensions are the perspectives or entities with respect to which an organization wants to keep records.These dimensions allow the store to keep track of things like monthly sales of items and the branches and locations at which the items were sold. Each dimension may have a table associated with it, called a **dimension table**, which further describes the dimension

- Eg. AllElectronics may create a sales data warehouse in order to keep records of the store's sales with respect to the dimensions **time, item, location** and **supplier**
- Eg. dimension table for item may contain the attributes **item_name, brand, and type**

From Tables and Spreadsheets to Data Cubes

- Multidimensional data model is typically organized around a central theme such as *sales*.
- This theme is represented by a **fact table**. **Facts** are numeric measures.
- Facts are the quantities by which we want to analyze relationships between dimensions
- **Fact table** contains the names of the *facts*, or measures, as well as keys to each of the related dimension tables.
  - Eg of facts for a sales data warehouse includes dollars_ sold (sales amount in dollars), units_ sold (number of units sold), etc
  - Eg. of keys to each of the related dimension tables like time_key, item_key, etc

2-D Data Cube

- 2-D data cube is like a table or spreadsheet
- Eg. *AllElectronics* sales data for items sold per quarter in the city of Vancouver
- Sales for Vancouver are shown with respect to the *time* dimension (organized in quarters) and the *item* dimension (organized according to the types of items sold)
- Fact or measure displayed is *dollars_sold* (in thousands)

2-D View of Sales Data for *AllElectronics* According to *time* and *item*

	location = "Vancouver"			
	item (type)			
time (quarter)	home entertainment	computer	phone	security
Q1	605	825	14	400
Q2	680	952	31	512
Q3	812	1023	30	501
Q4	927	1038	38	580

Note: The sales are from branches located in the city of Vancouver. The measure displayed is *dollars_sold* (in thousands).

3-D Data Cube

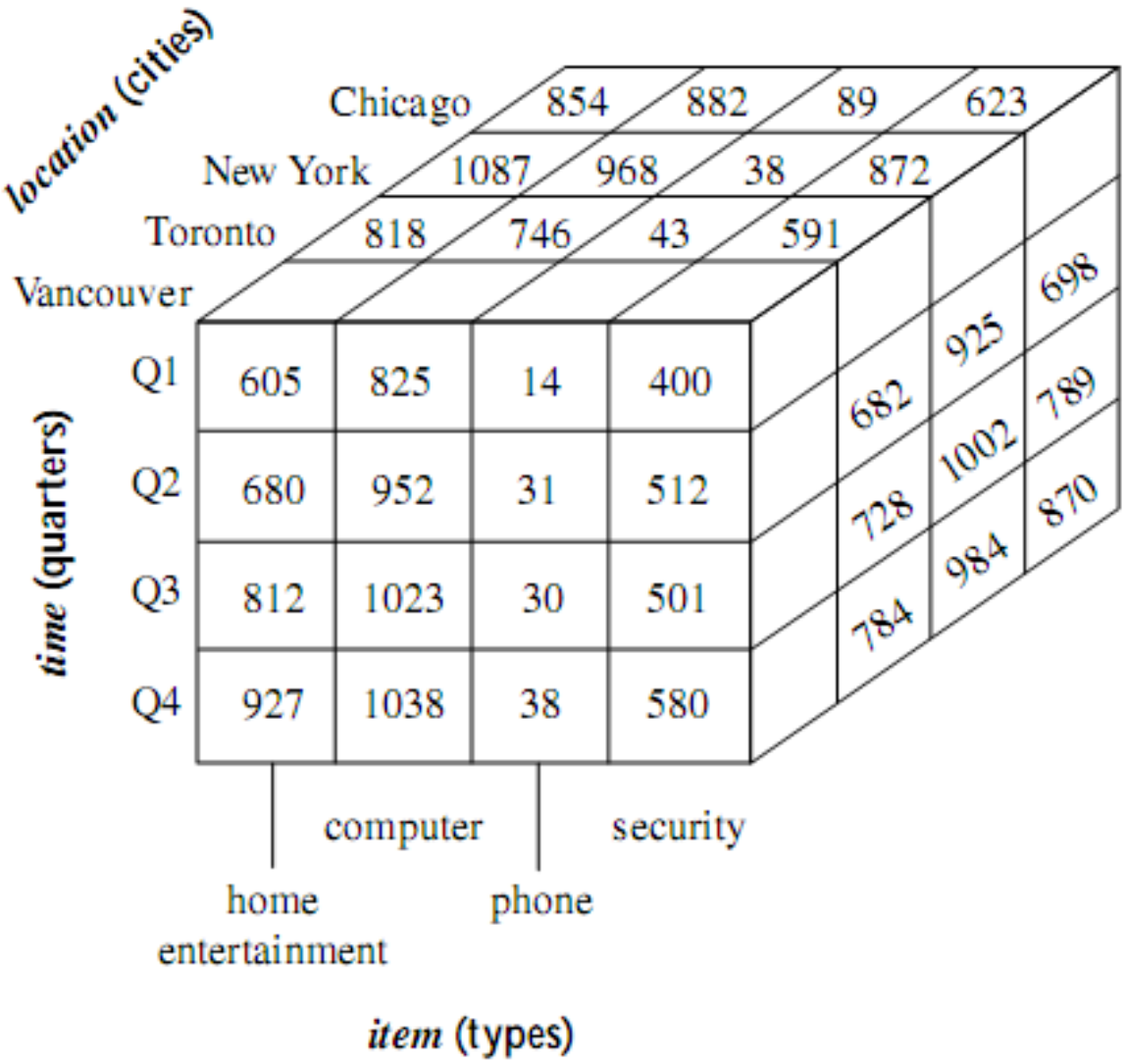
- To view the data according to time and item, as well as location, for the cities Chicago, New York, Toronto, and Vancouver
- 3-D data in the table are represented as a series of 2-D tables.
- Conceptually, we may also represent the same data in the form of a **3-D data cube**

3-D View of Sales Data for *AllElectronics* According to *time*, *item*, and *location*

	location = "Chicago"				***location*** = "New York"				***location*** = "Toronto"				***location*** = "Vancouver"			
	item				***item***				***item***				***item***			
	home				*home*				*home*				*home*			
time	*ent.*	*comp.*	*phone*	*sec.*	*ent.*	*comp.*	*phone*	*sec.*	*ent.*	*comp.*	*phone*	*sec.*	*ent.*	*comp.*	*phone*	*sec.*
Q1	854	882	89	623	1087	968	38	872	818	746	43	591	605	825	14	400
Q2	943	890	64	698	1130	1024	41	925	894	769	52	682	680	952	31	512
Q3	1032	924	59	789	1034	1048	45	1002	940	795	58	728	812	1023	30	501
Q4	1129	992	63	870	1142	1091	54	984	978	864	59	784	927	1038	38	580

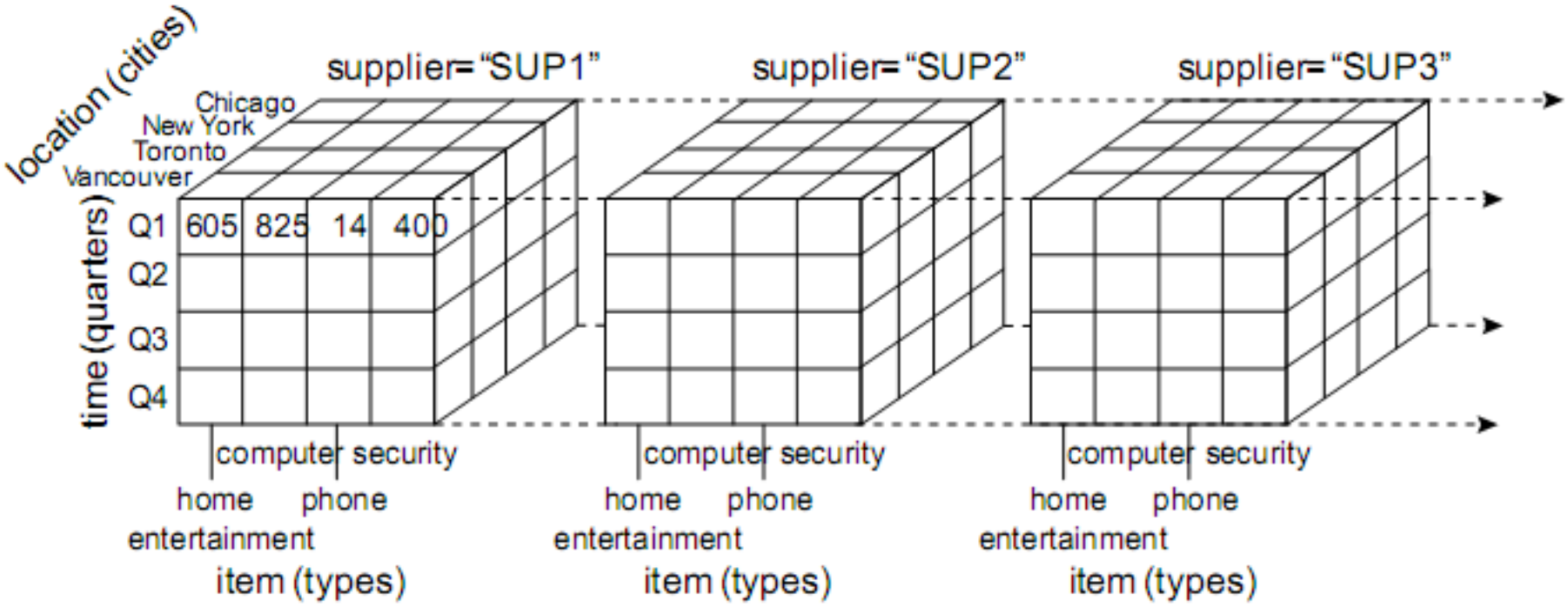
Note: The measure displayed is *dollars_sold* (in thousands).

3-D Data Cube


location (cities)
Chicago 854 882 89 623
New York 1087 968 38 872
Toronto 818 746 43 591
Vancouver
Q1 605 825 14 400
Q2 680 952 31 512
Q3 812 1023 30 501
Q4 927 1038 38 580
682 925 698
728 1002 789
784 984 870
time (quarters)
computer
security
home entertainment
phone
item (types)

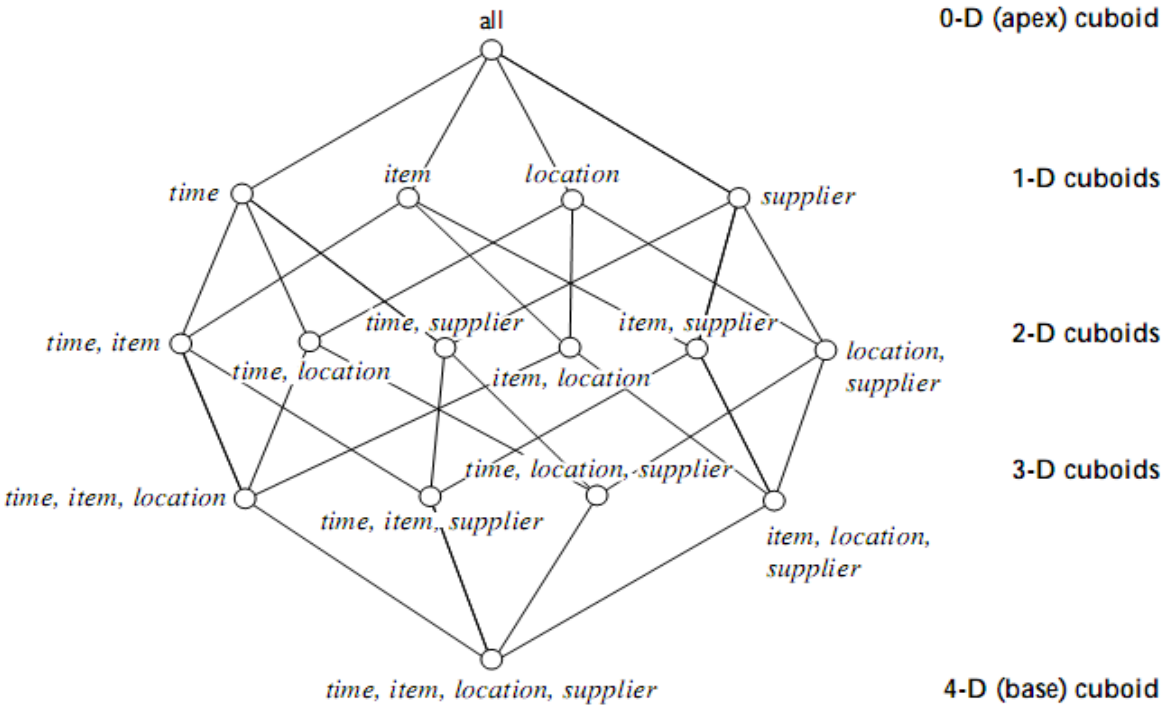
4-D Data Cube

- To view our sales data with an additional fourth dimension such as supplier
- we can think of a 4-D cube as being a series of 3-D cubes
- Any n-dimensional data can be viewed as a series of (n-1)-dimensional cubes
- Its a logical representation & physical storage may differ



Cuboids corresponding to the Cube

- In data warehousing research literature, a data cube is often referred to as a cuboid
- Cuboid can be generated for each possible subsets of the given dimensions
- This would form a lattice of cuboids, each showing the data at a different level of summarization



Cuboids corresponding to the Cube

- Cuboid that holds the lowest level of summarization is called the base cuboid
- 0-D cuboid, which holds the highest level of summarization, is called the apex cuboid
- In our eg., this is the total_sales, or dollars_sold, summarized over **all four dimensions**
- Apex cuboid is typically denoted by all

Dimensional nature of business data

- Managers think of business in terms of business dimensions
- **Marketing vice president** is interested in revenue numbers broken down by
  -month
  -division
  -customer
  -demographic
  -sales office
  -product version

These are the business dimensions

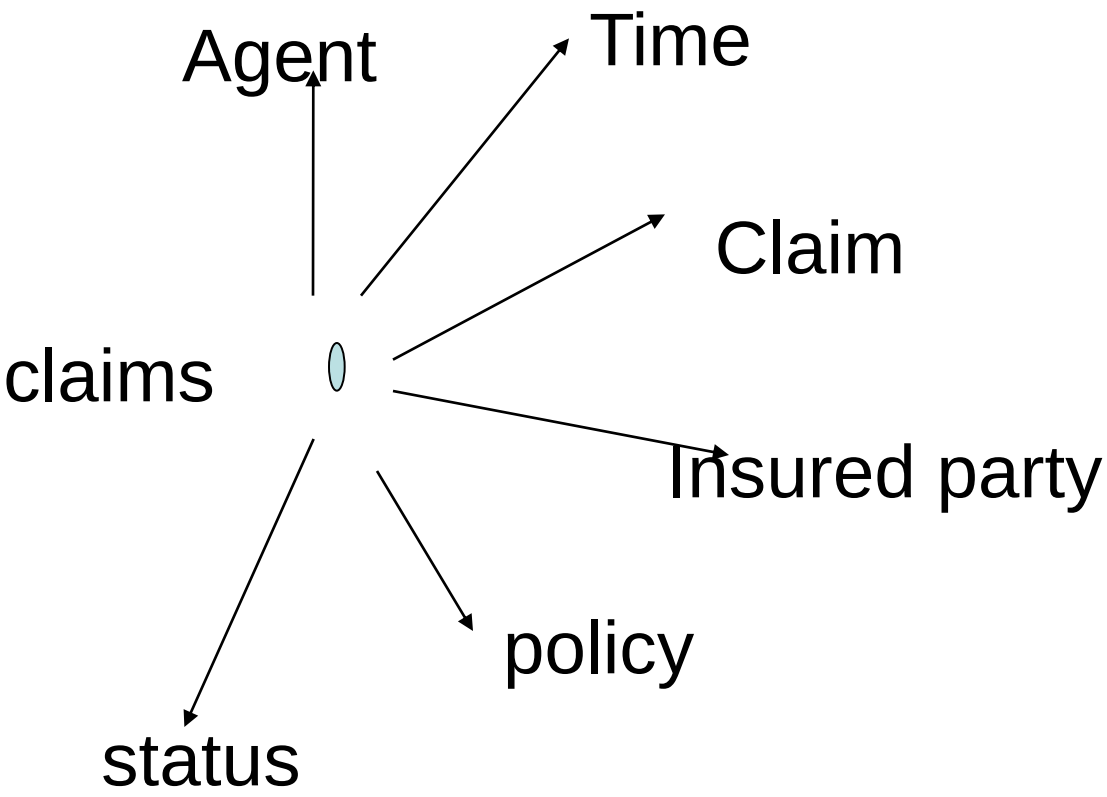
Dimensional nature of business data

- **Marketing Manager** business dimensions are
  -product
  -product category
  -time(day,week,month)
  -sale district
  -distribution channel

Datawarehouse : business dimensions

- Identify dimensions for building a DW for claim analysis of insurance business.

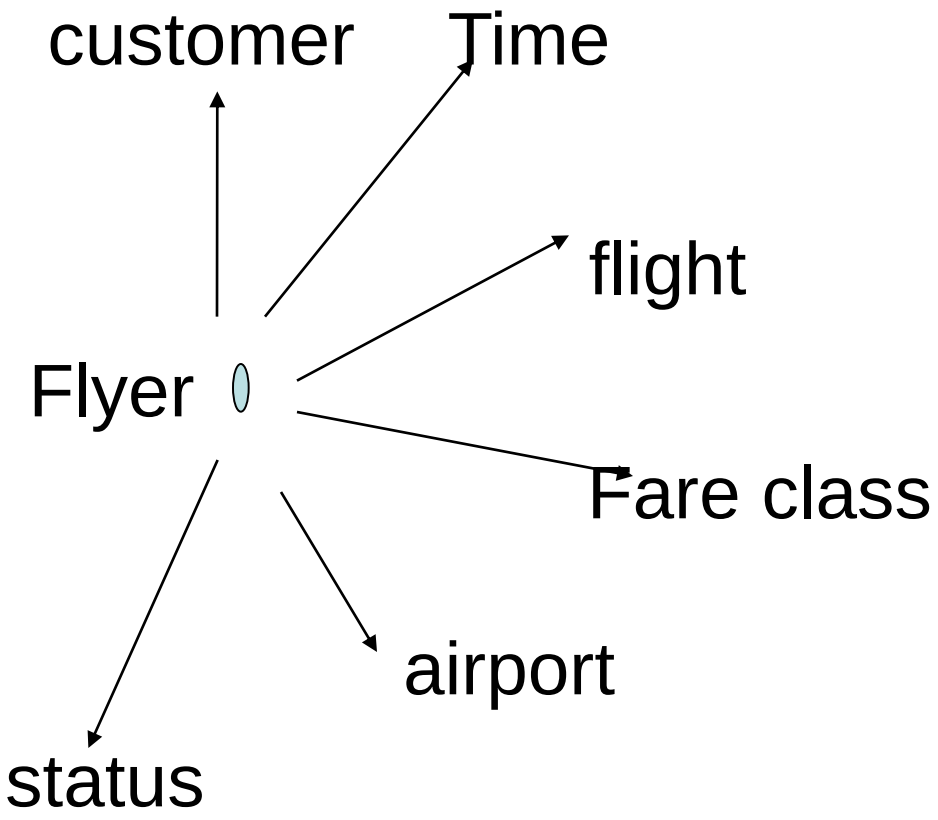
Example of business dimensions



Datawarehouse : business dimensions

- Identify dimensions for building a DW for flyer analysis of Airline Company

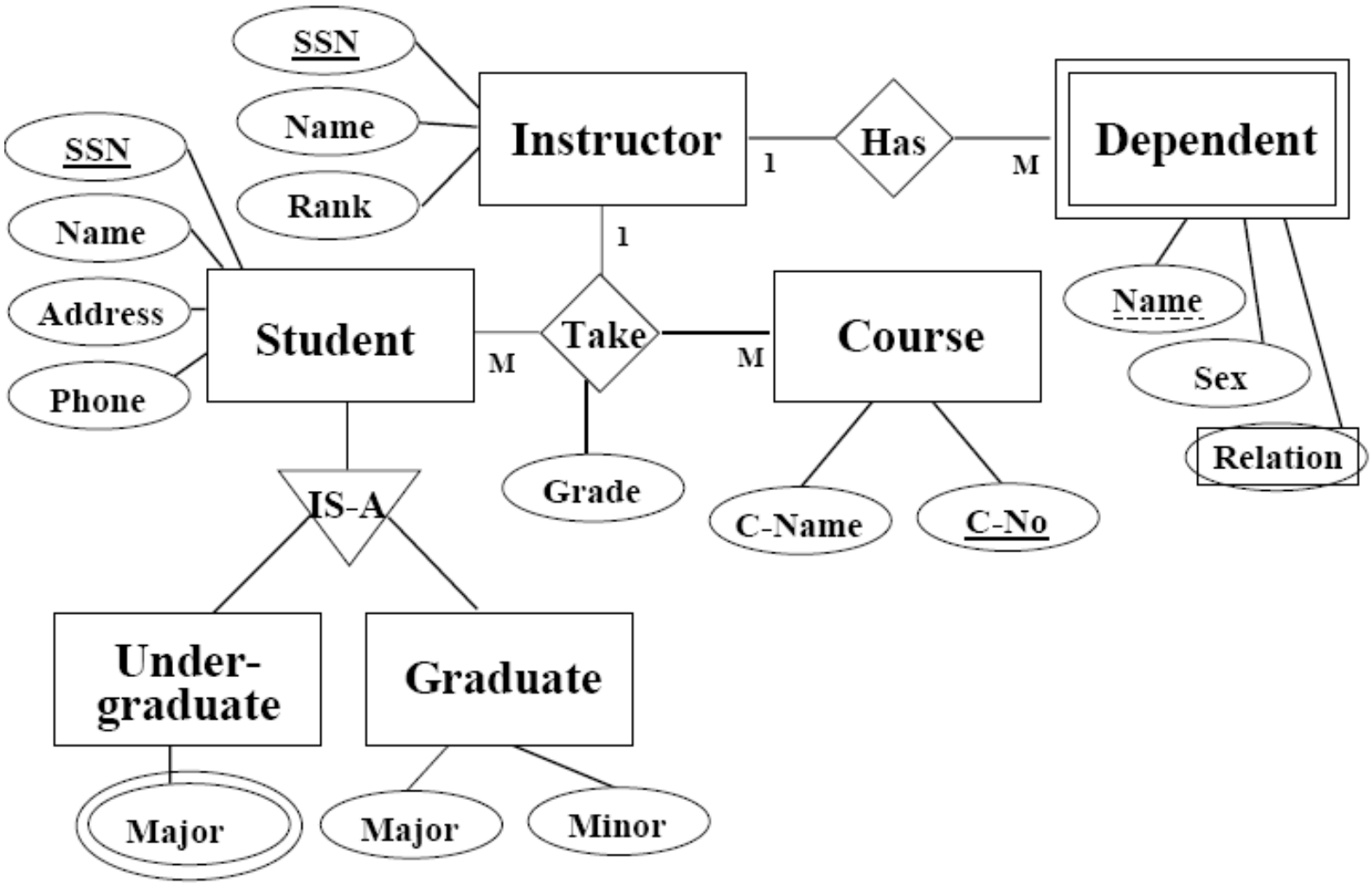
Example of business dimensions



ER Modelling

- Logical design technique that seeks to eliminate data redundancy
- Focusses on the microscopic relationships among data elements
- Emphasis on **Normalization** -- a technique of minimizing the insertion, deletion and update anomalies through eliminating the redundant data
- Perfect for OLTP systems
- Responsible for success of transaction processing in Relational Databases

Transactional ER system is not user friendly for retrieving data



Problems with ER Model

ER models are NOT suitable for DW?

- Difficult to understand or remember an ER Model since there are lot of details
- Many DWs have failed because of overly complex ER designs
- Denormalization required--To achieve the faster execution of the queries redundancy is added to the data
- Data retrieval becomes difficult due to normalization
- Making the search and analysis faster
- Not optimized for complex, ad-hoc queries
- Browsing becomes difficult

Design Requirements

- Design of the DW must directly reflect the way the managers look at the business
- Should capture the measurements of importance
- Must facilitate data analysis, i.e., answering business questions
- DW requires a subject-oriented schema that facilitates online data analysis
- Data model for a data warehouse is a multidimensional model, which can exist in the form of a star schema, a snowflake schema, or a fact constellation schema

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

Star Schema

- Most common modeling paradigm is the star schema

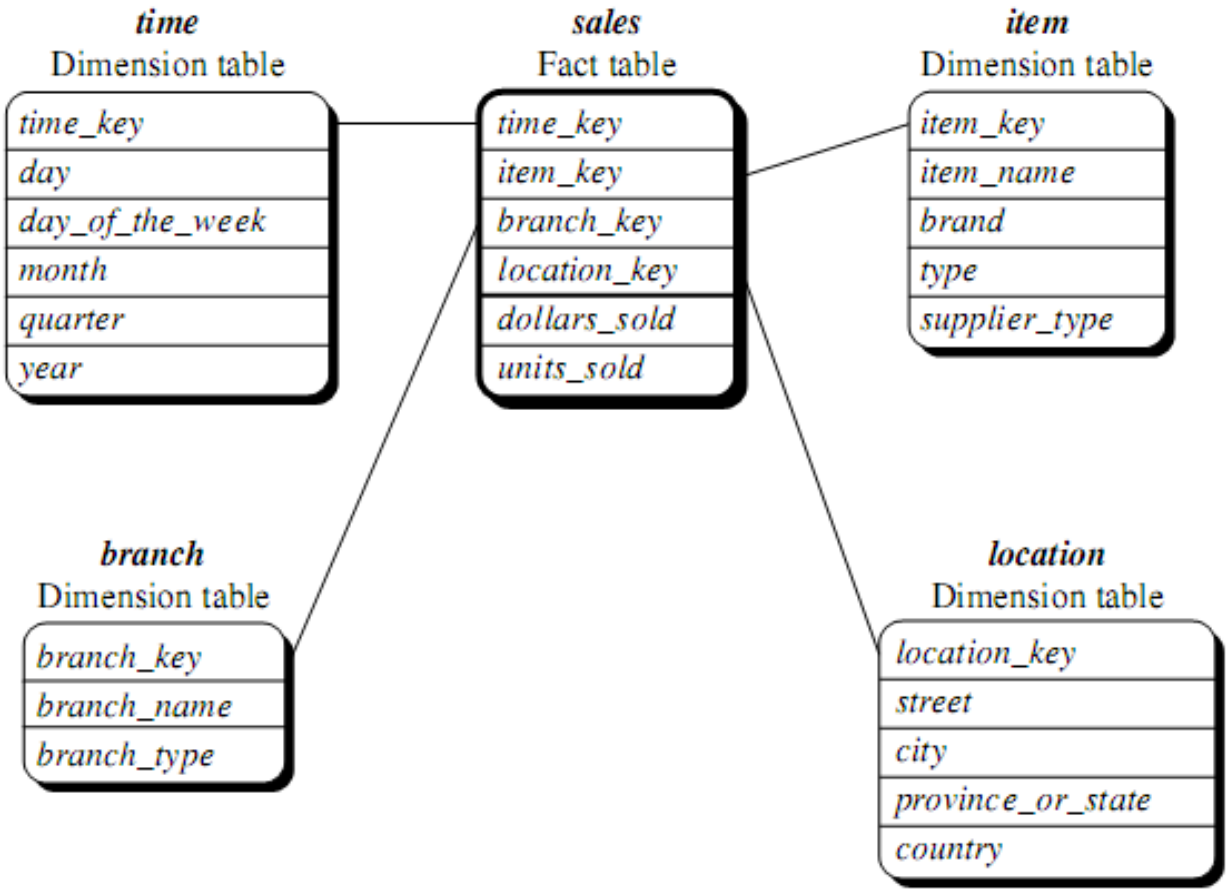
- Contains

(1) a large central table **(fact table)** containing the bulk of the data and points to each dimension

(2) a set of smaller attendant tables **(dimension tables)**, one for each dimension and contains a set of attributes

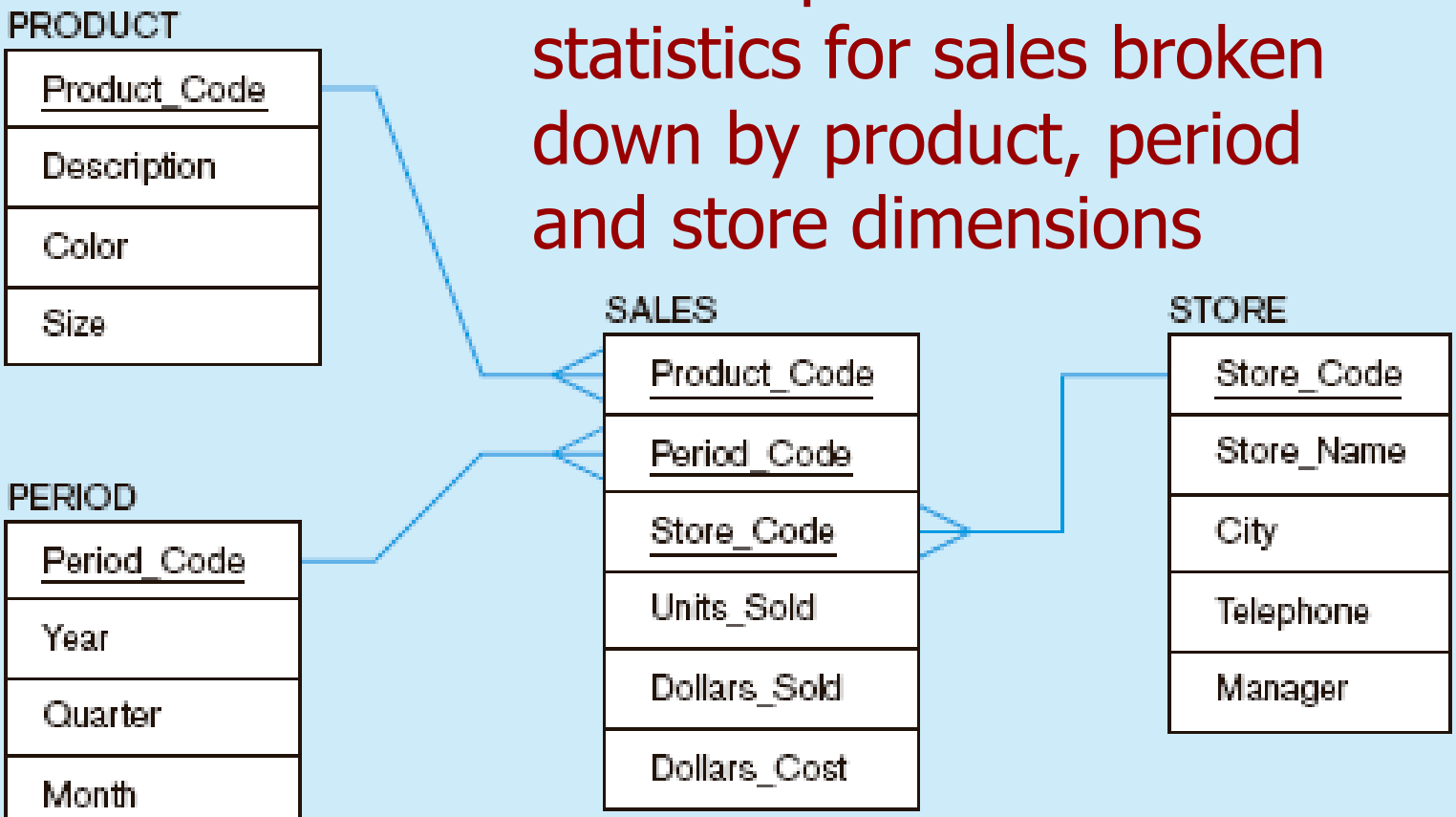
Star schema example

Sales are considered along **four dimensions: time, item, branch, and location**. Schema contains a central **fact table for sales** that contains keys to each of the four dimensions, along with two measures: dollars sold and units sold

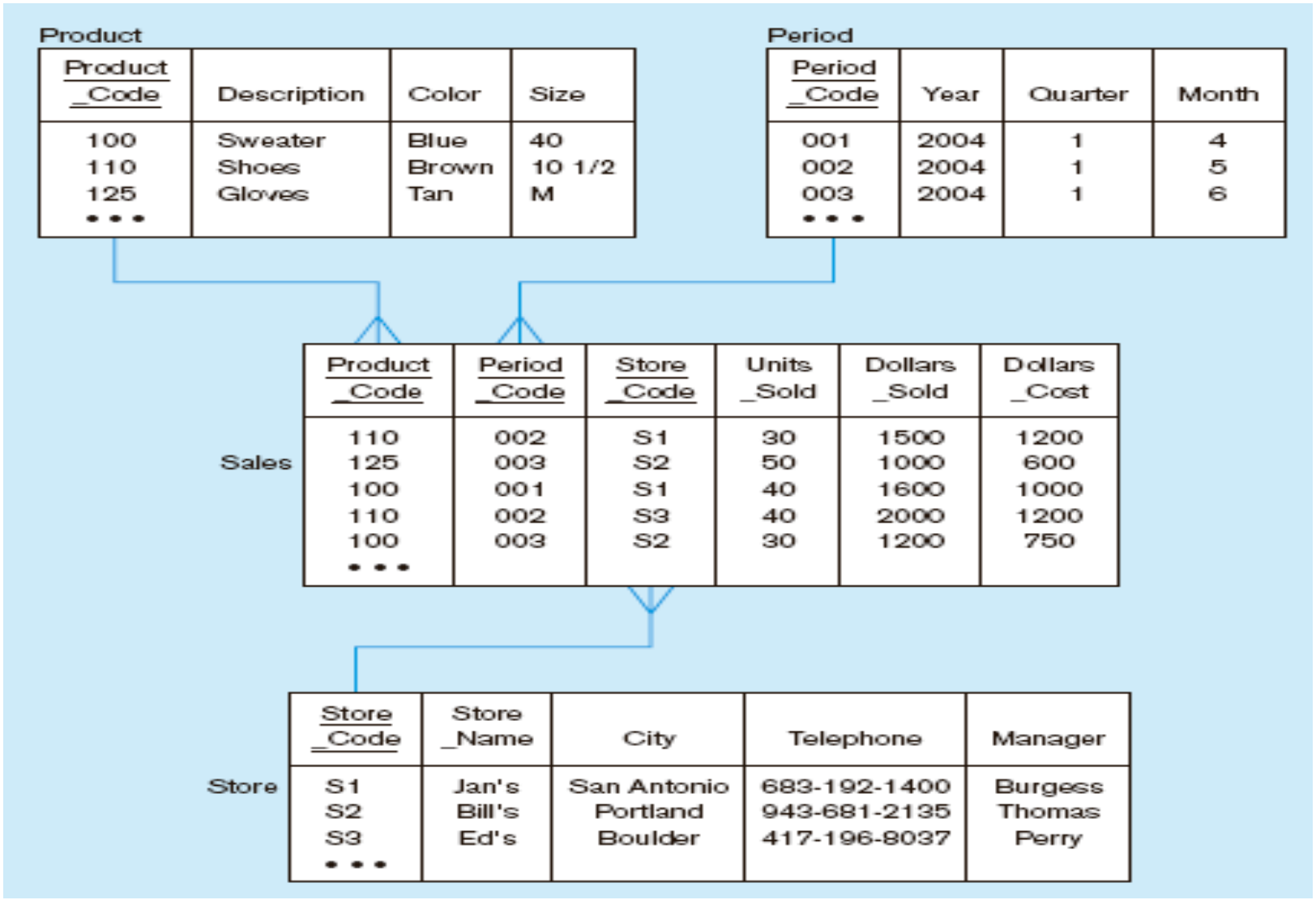


Star schema example

Fact table provides statistics for sales broken down by product, period and store dimensions



Star schema with sample data



STAR Schema keys

- Primary Keys-Uniquely identifies a record in dimension table
- Surrogate Keys-
  - System generated sequence numbers
  - Do not have any built in meanings.

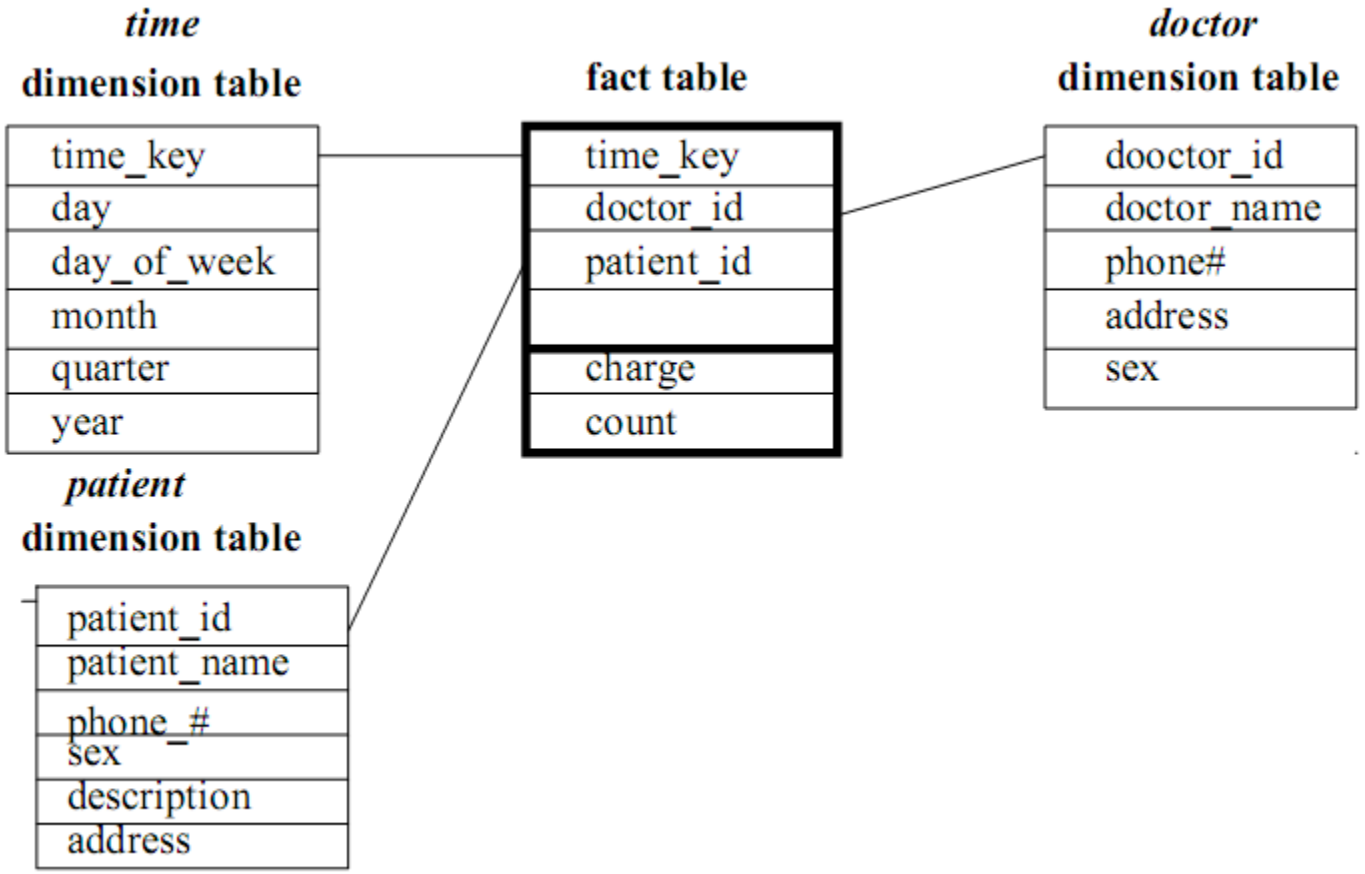
Advantages of STAR Schema

- Easy for users to understand
- Optimizes navigation
- Most suitable for complex query processing

Exercise

- 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.
- Draw a schema diagram for the above data warehouse

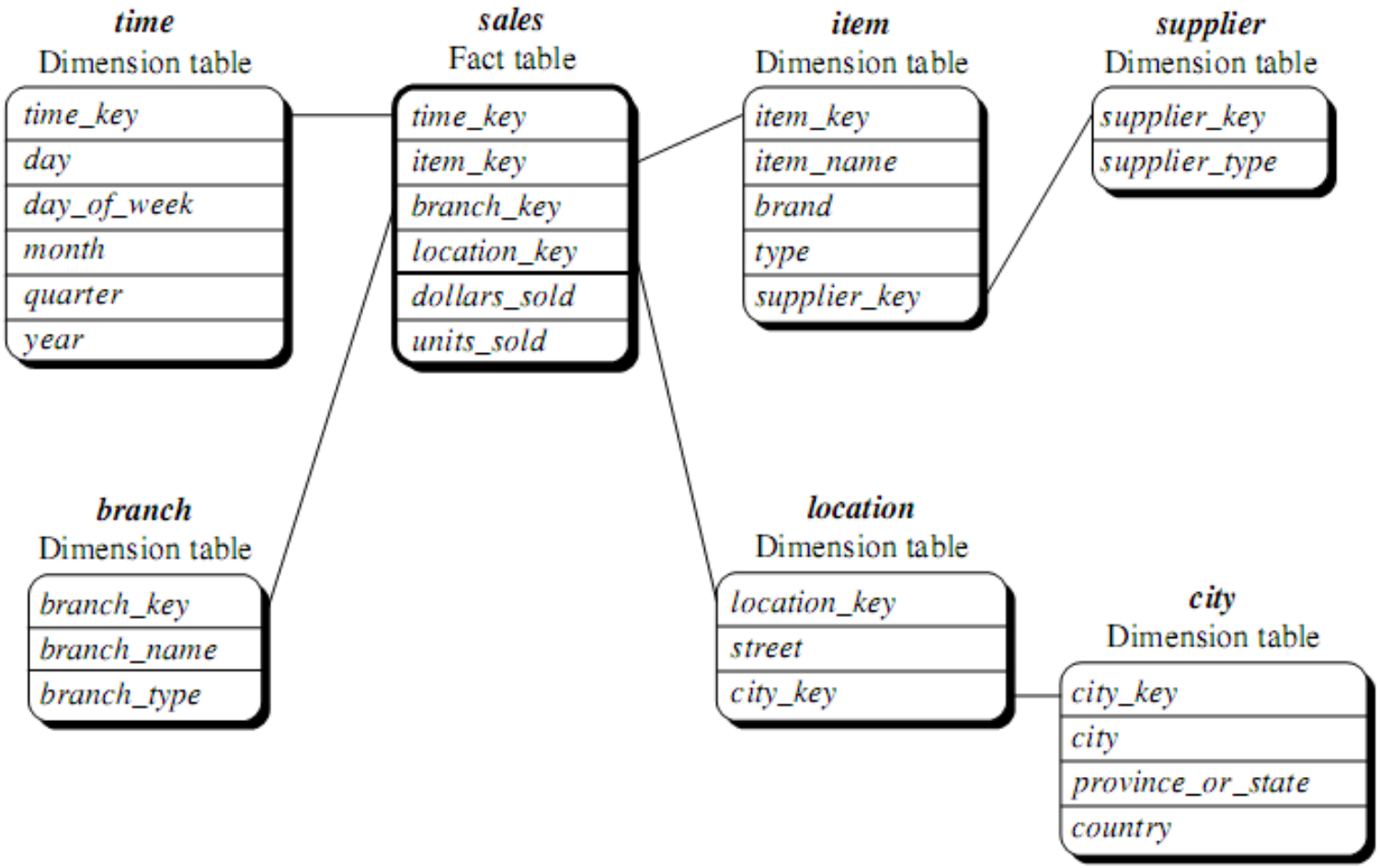
Exercise


time
dimension table
time_key
day
day_of_week
month
quarter
year
patient
dimension table
patient_id
patient_name
phone_#
sex
description
address
fact table
time_key
doctor_id
patient_id
charge
count
doctor
dimension table
dooctor_id
doctor_name
phone#
address
sex

Snowflake Schema

- Variant of the star schema model, where some dimension tables are normalized
- Further splitting the data into additional tables
- Resulting schema graph forms a shape similar to a snowflake
- Eg. location dimension table contains the attribute set {location_ key, street, city, province_or_state, country}.
- For instance, “Pune” is a city in the state of Maharashtra, India.
- Entries for such different streets in same city will create redundancy among the attributes province_or_state and country; i.e, (...., Pune, Maharashtra, India) and (...., Pune, Maharashtra, India)

Snowflake Schema



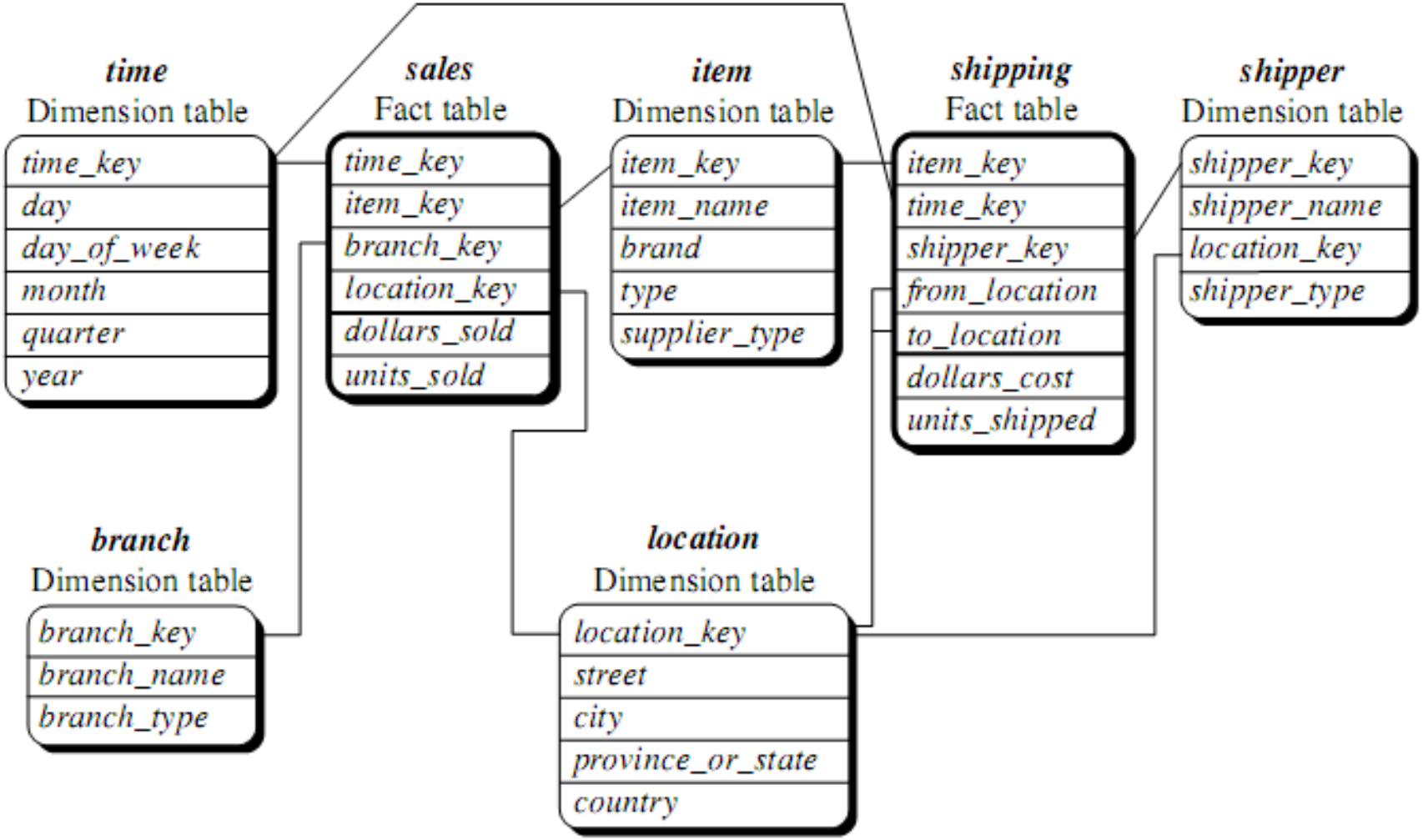
Snowflake Schema

- Advantages
  - Saves storage space
  - Reduces redundancies
- Disadvantages
  - Degrades performance because more joins needed to execute a query
  - System performance adversely affected

Fact Constellation

- Sophisticated applications may require **multiple fact tables** to share dimension tables
- This kind of schema can be viewed as a collection of stars, and hence is called a galaxy schema or a fact constellation schema
- Eg. two fact tables, sales and shipping
- Fact constellation schema allows dimension tables to be shared between fact tables.
- Eg. the dimensions tables for time, item, and location are shared between the sales and shipping fact tables.

Example of Fact Constellation



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

Modeling of Data Warehouses

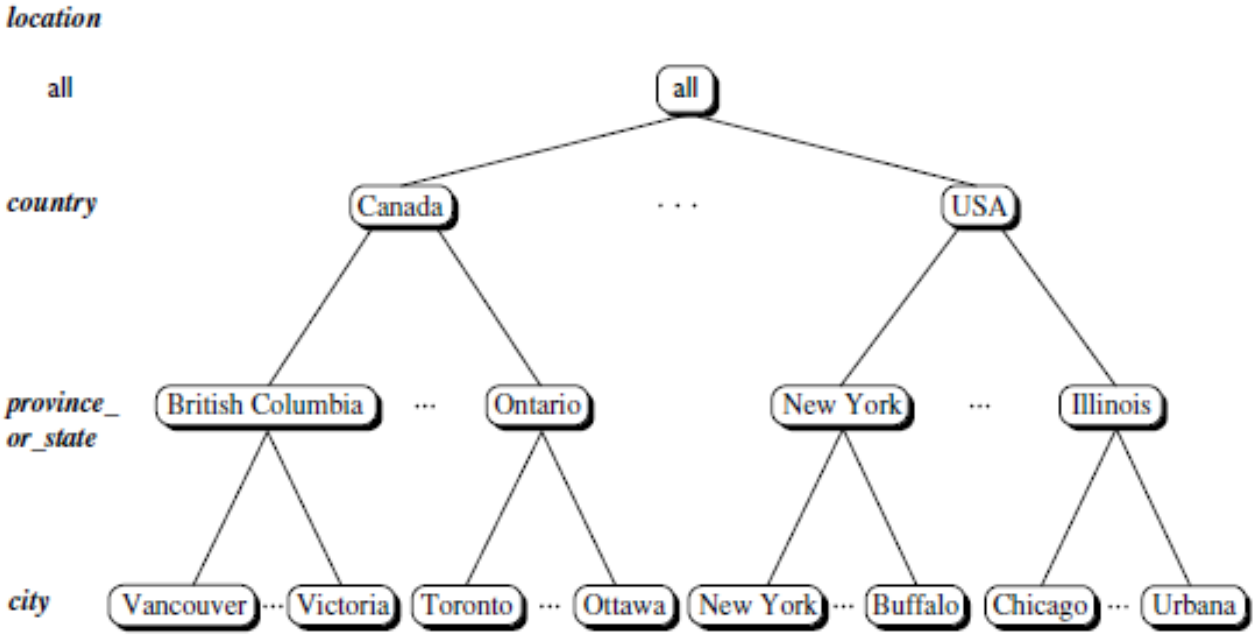
- Enterprise-wide data warehouses, the fact constellation schema is commonly used, since it can model multiple, interrelated subjects
- For data marts, the star or snowflake schema is commonly used, since both are geared toward modeling selected subjects
- Star schema is more popular and efficient

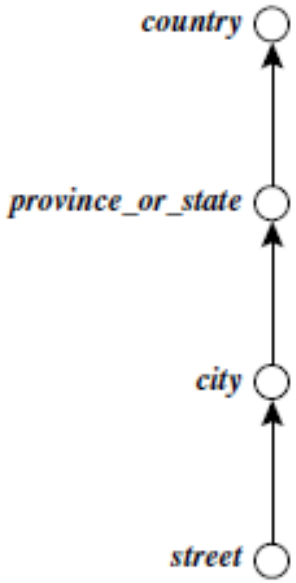
Concept Hierarchy: Dimensions

- Sequence of mappings from a set of low-level concepts to higher-level, more general concepts
- Concept hierarchies allow data to be handled at varying levels of abstraction
- Eg. suppose that the dimension *location* is described by the attributes *street, city, province_or_state* and *country*.
- These attributes are related by a total order, forming a concept hierarchy such as

  "street < city < province or state < country"

Concept Hierarchy: Dimension (location)





Measures of Data Cube

- **Measure** is a numeric function that can be evaluated at each point in the data cube space
- Measure value is computed for a given point by aggregating the data corresponding to the respective dimension–value pairs defining the given point
- Three Categories: based on the kind of aggregate functions used

1. **Distributive**
   - Can be computed in a distributed manner
   - Suppose the data are partitioned into *n* sets.
   - We apply the function to each partition, resulting in *n* aggregate values.
   - If the result derived by applying the function to the *n* aggregate values is the same as that derived by applying the function to the entire data set (without partitioning), the function can be computed in a distributed manner
   - E.g., count(), sum(), etc

Measures of Data Cube: Three Categories

2. Algebraic:

- Can be computed by an algebraic function with *M* arguments, each of which is obtained by applying a distributive aggregate function
- avg() can be computed by sum()/count(), where both sum() and count() are distributive aggregate functions

3. Holistic:

- Cannot be described with an algebraic function with M arguments
- Eg. median() function, which cannot be described by an algebraic expression or calculated in a distributive manner
- Result of any holistic function cannot be computed using pre-aggregated sub-parts and instead generally requires **access to all the individual elements**

Typical OLAP Operations

Roll-up (*drill-up*):

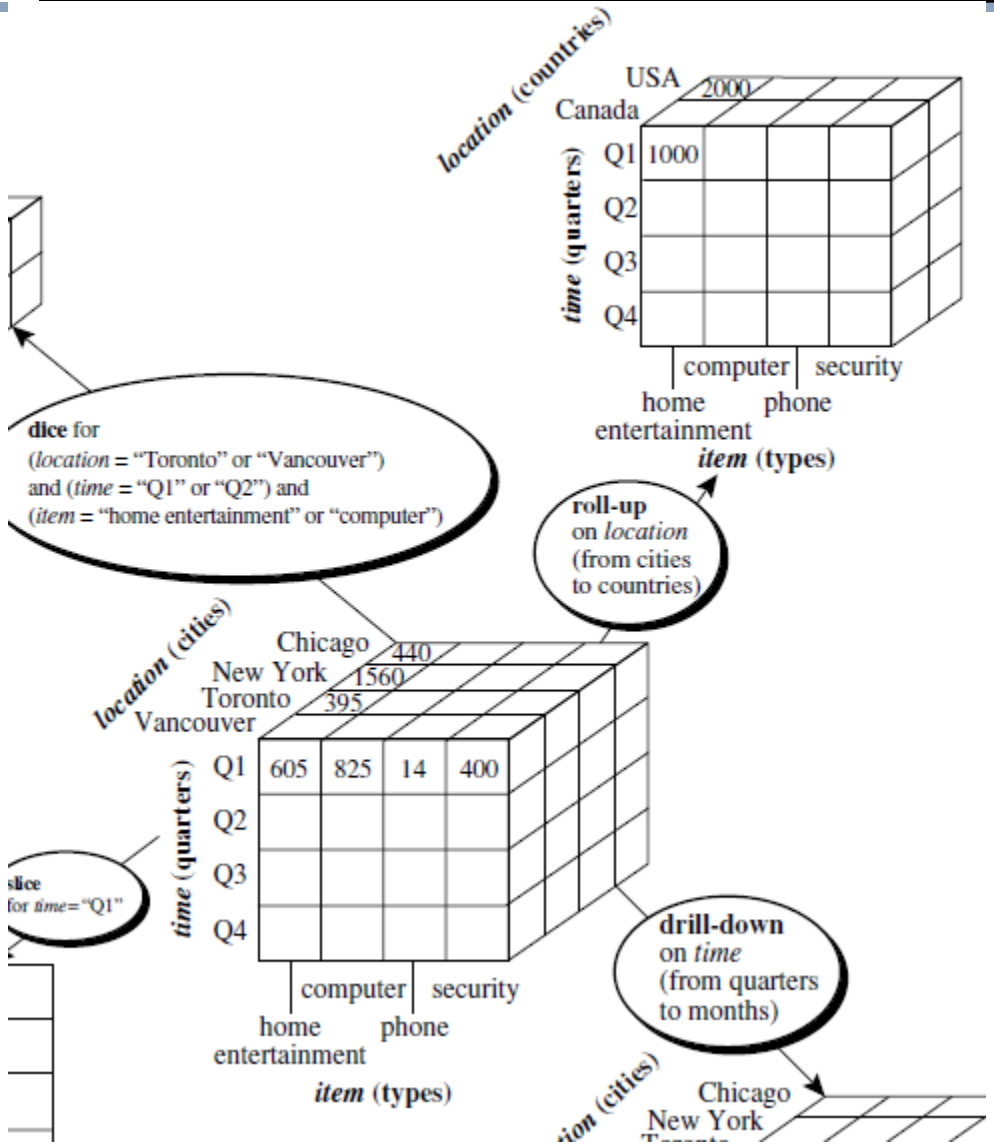
- Performs aggregation on a data cube, either by *climbing up a concept hierarchy* for a dimension or by *dimension reduction*

- ***Eg.*** roll-up operation shown aggregates the data by ascending the *location* hierarchy from the level of *city* to the level of *country*
  i.e. grouping the data by city to grouping the data by country

- When roll-up is performed by dimension reduction, one or more dimensions are removed from the given cube.

Eg. consider a sales data cube containing only the *location* and *time* dimensions. Roll-up may be performed by removing, say, the *time* dimension, resulting in an aggregation of the total sales by location, rather than by location and by time.

Typical OLAP Operations

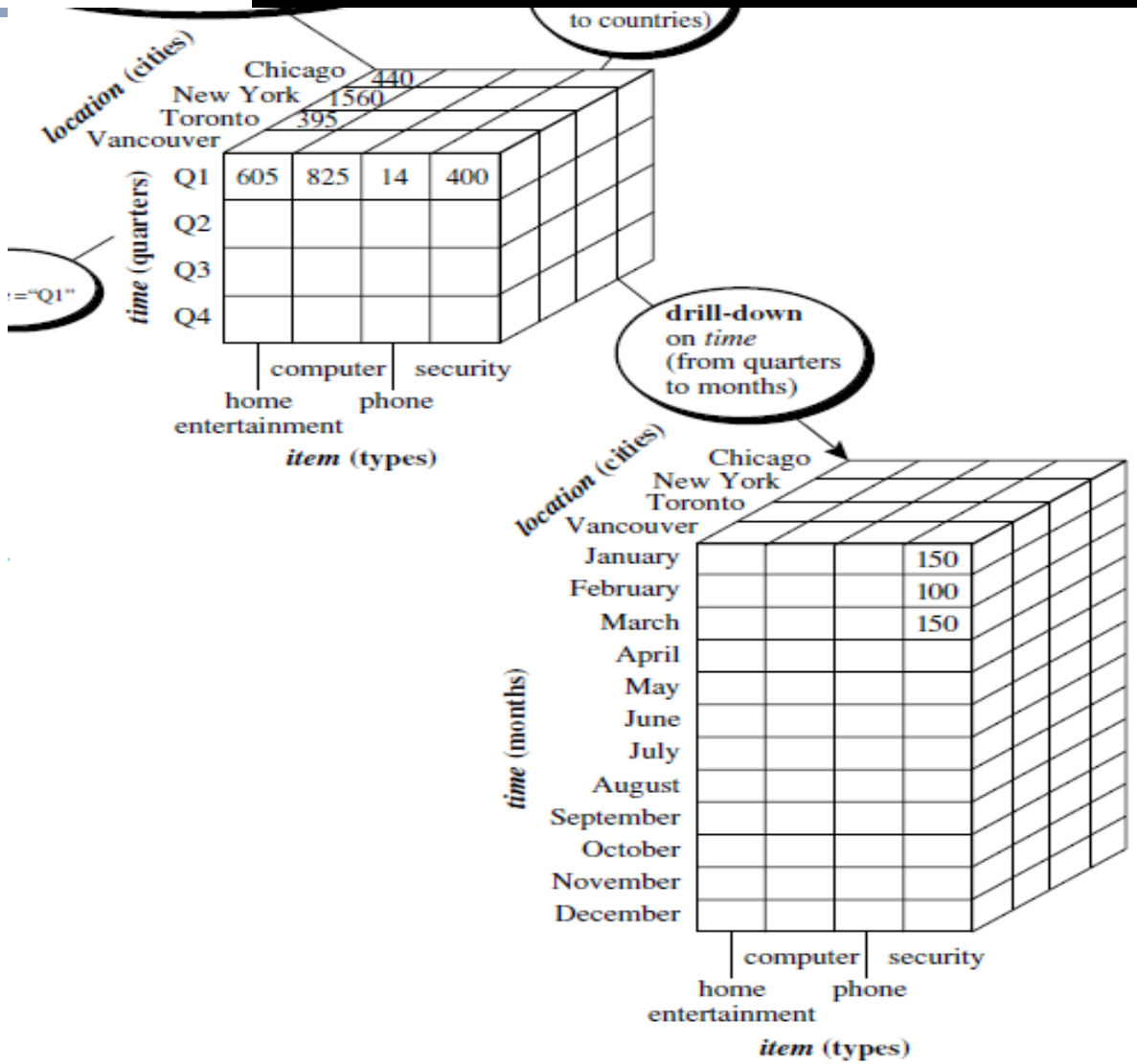


Typical OLAP Operations

Drill-down:

- Drill-down is the reverse of roll-up
- Navigates from less detailed data to more detailed data.
- Drill-down can be realized by either *stepping down a concept hierarchy* for a dimension or *introducing additional dimensions*.
- Drill-down operation performed on the central cube by stepping down a concept hierarchy for *time* defined as *"day < month < quarter < year."*
- Drill-down occurs by descending the *time* hierarchy from the level of *quarter* to the more detailed level of *month* i.e. resulting data cube details the total sales per month rather than summarizing them by quarter.
- Because a drill-down adds more detail to the given data, it can also be performed by adding new dimensions to a cube. Eg. a drill-down on the cube can occur by introducing an additional dimension, such as *customer group*.

Typical OLAP Operations



Typical OLAP Operations

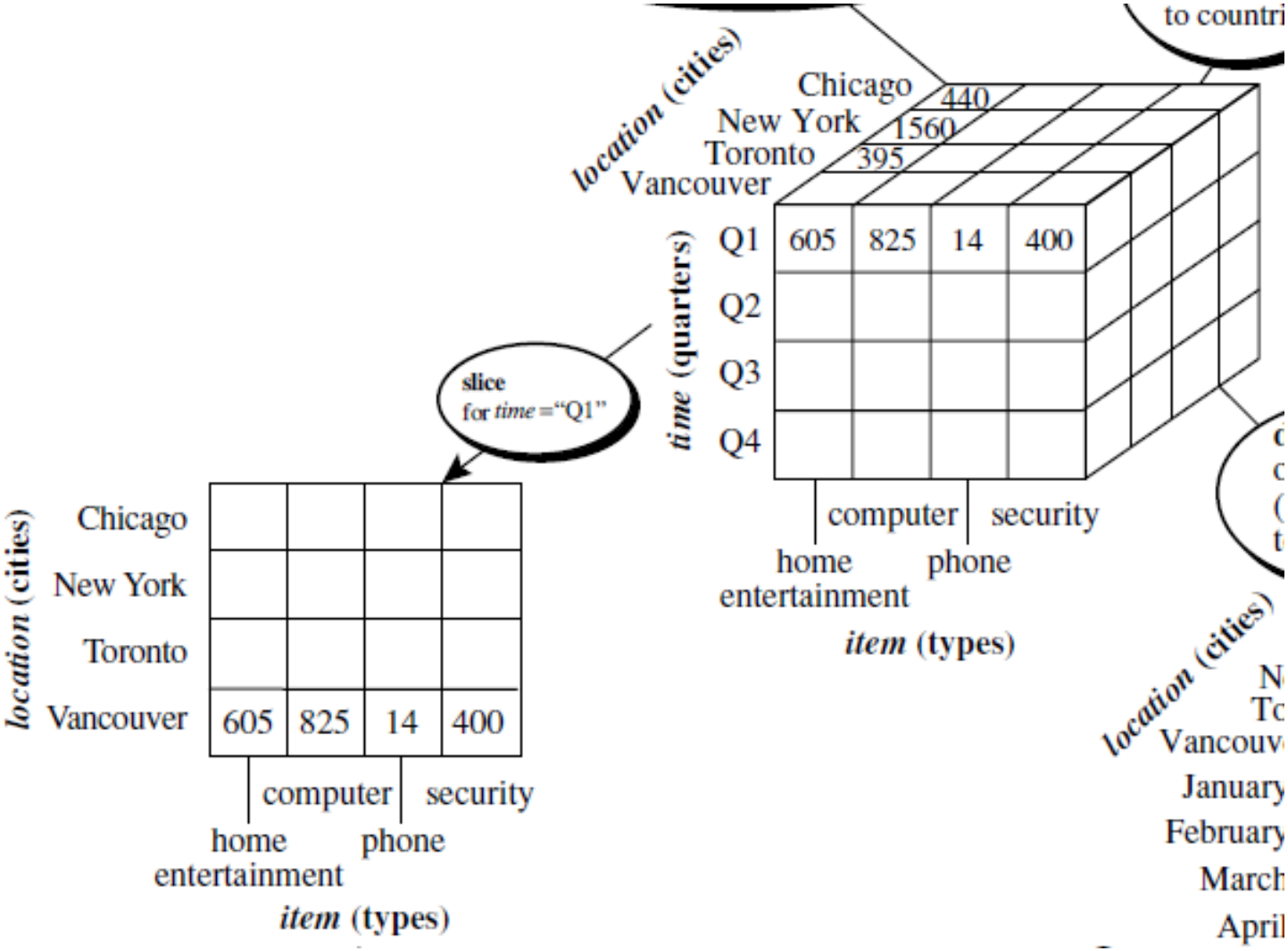
Slice :

- Performs a selection on one dimension of the given cube, resulting in a subcube
- Eg. a slice operation where the sales data are selected from the central cube for the dimension *time* using the criterion *time* = "Q1."

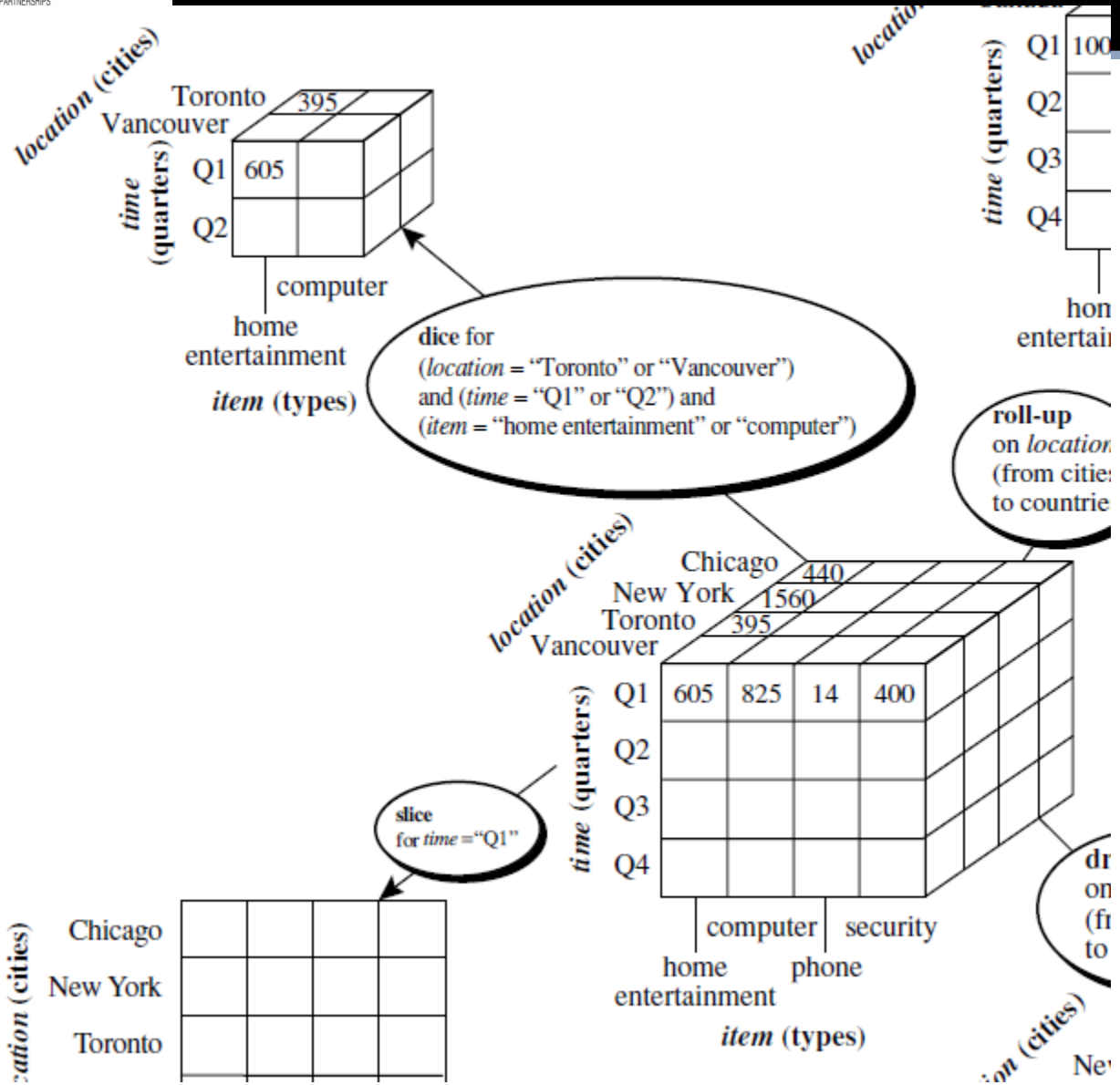
Dice:

- *Dice* operation defines a subcube by performing a selection on two or more dimensions
- Eg. Dice operation on the central cube based on the following selection criteria that involve three dimensions: (*location* ="Toronto" or "Vancouver") and (*time* = "Q1" or "Q2") and (item = "home entertainment" or "computer")

Typical OLAP Operations



Typical OLAP Operations

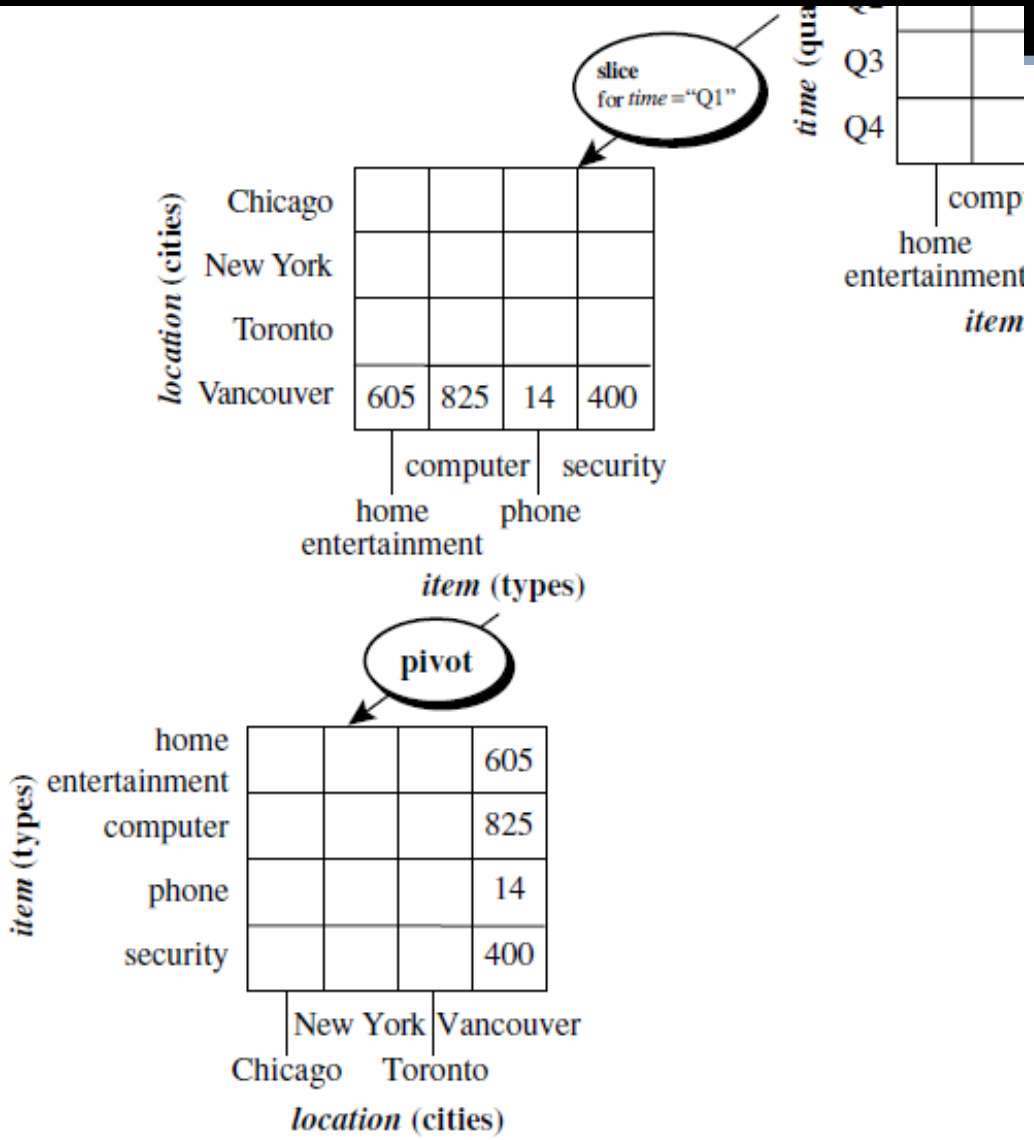


Typical OLAP Operations

Pivot (rotate):

- *Pivot* is a visualization operation that rotates the data axes in view to provide an alternative data presentation.
- Eg. Pivot operation where the *item* and *location* axes in a 2-D slice are rotated.
- Other examples include rotating the axes in a 3-D cube

Typical OLAP Operations



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*
- 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)*

Exercise

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) 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?

Exercise

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?

The operations to be performed are:

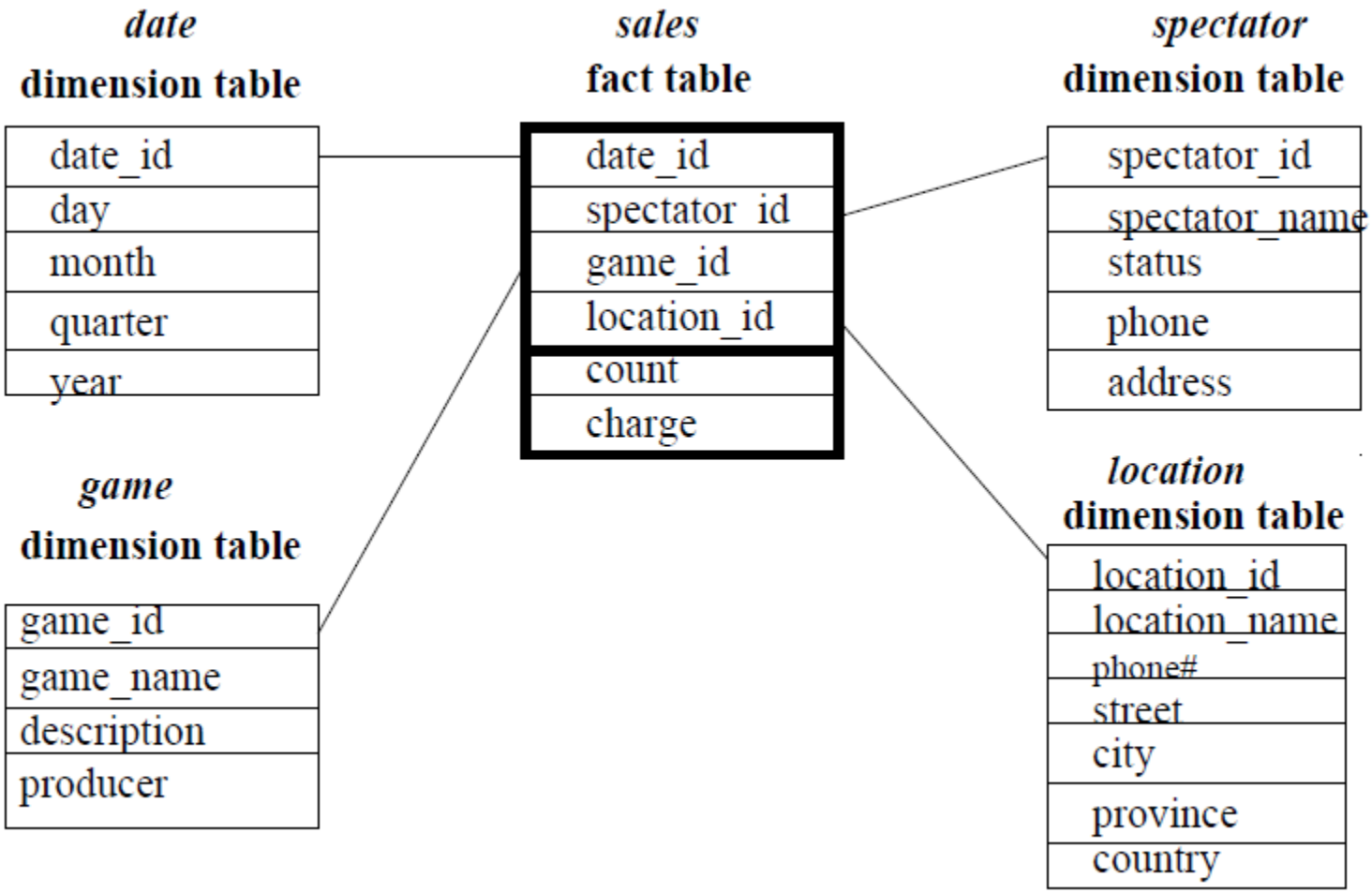
1. Roll-up on *time* from *day* to *year*
2. Roll-up on *patient* from individual patient to all
3. Slice for *time=2004*.

Exercise

Suppose that a data warehouse consists of the four dimensions, *date, spectator, location*, and *game*, and the two measures, *count* and *charge*, where *charge* is the fare that a spectator pays when watching a game on a given date. Spectators may be students, adults, or seniors, with each category having its own charge rate.

Draw a *star schema* diagram for the data warehouse

STAR Schema


date
dimension table
date_id
day
month
quarter
year
sales
fact table
date_id
spectator_id
game_id
location_id
count
charge
spectator
dimension table
spectator_id
spectator_name
status
phone
address
game
dimension table
game_id
game_name
description
producer
location
dimension table
location_id
location_name
phone#
street
city
province
country

Exercise

Suppose that a data warehouse consists of the four dimensions, *date, spectator, location*, and *game*, and the two measures, *count* and *charge*, where *charge* is the fare that a spectator pays when watching a game on a given date. Spectators may be students, adults, or seniors, with each category having its own charge rate.

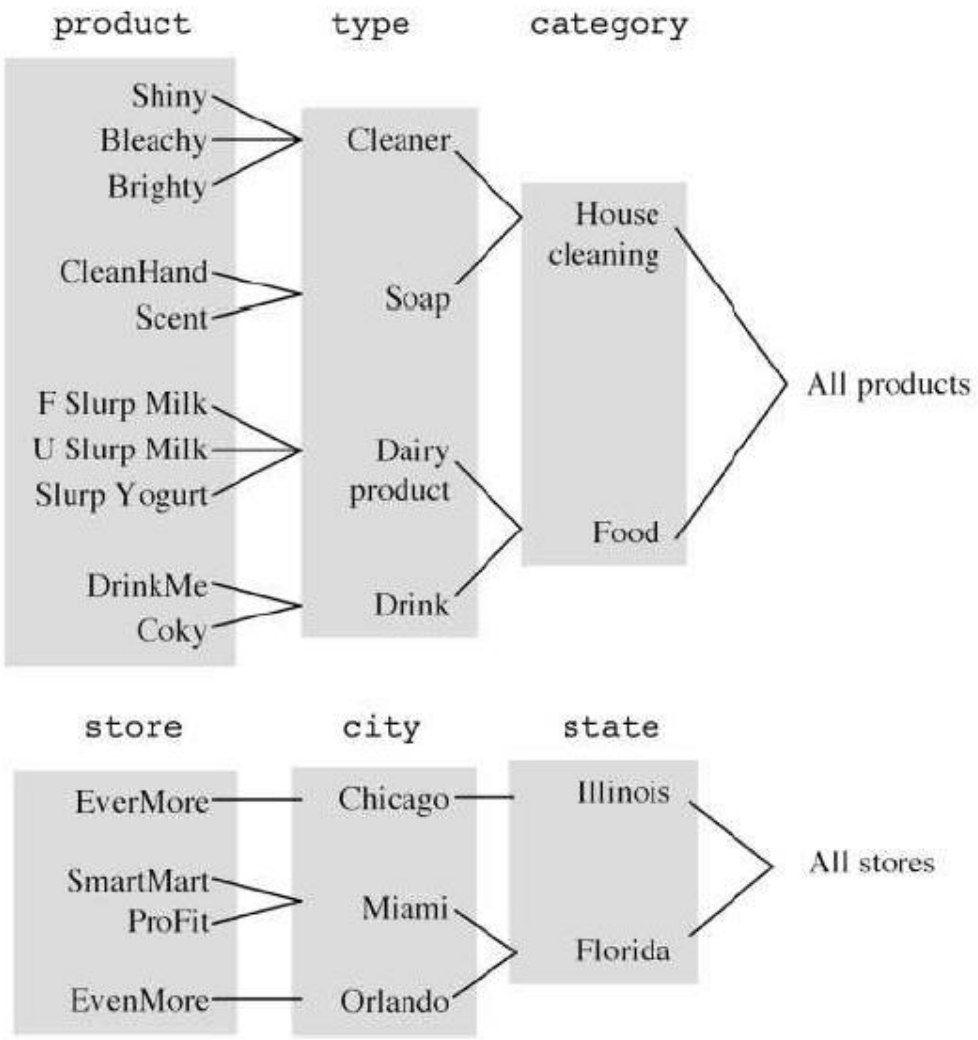
(a) Starting with the base cuboid [*date, spectator, location, game*], what specific *OLAP operations* should one perform in order to list the total charge paid by student spectators at GM_Place in 2004?

Exercise

a) Starting with the base cuboid [*date, spectator, location, game*], what specific *OLAP operations* should one perform in order to list the total charge paid by student spectators at GM_Place in 2004?

1. Roll-up on date from date_id to year.
2. Roll-up on game from game_id to all.
3. Roll-up on location from location_id to location_name.
4. Roll-up on spectator from spectator_id to status.
5. Dice with status="students", location_name="GM_Place", and year=2004.

Roll up (drill-up):



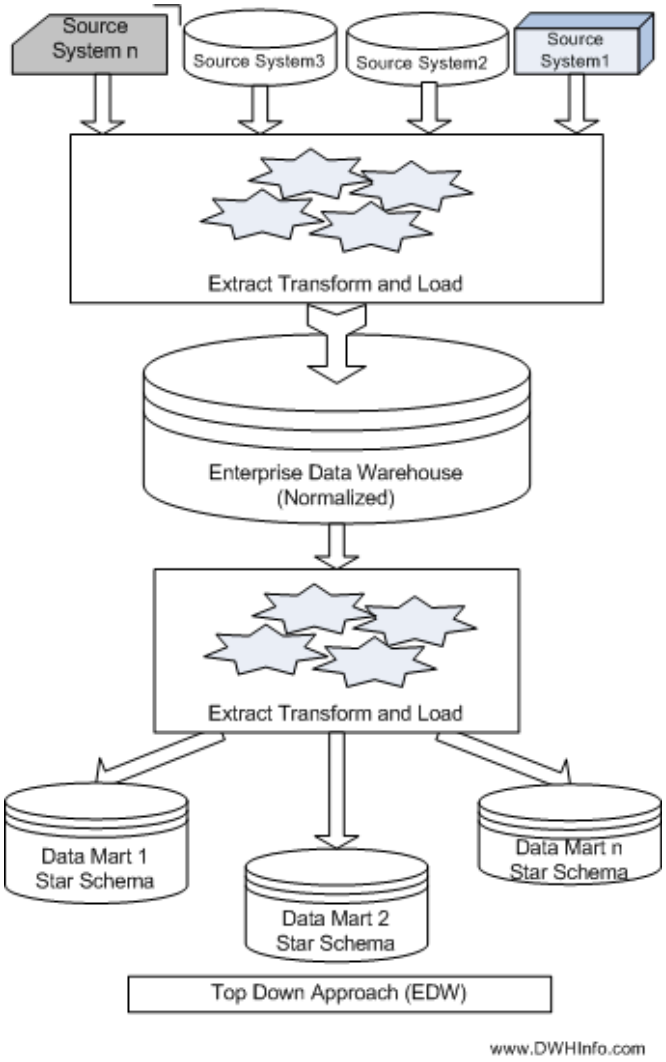
Design of Data Warehouse: A Business Analysis Framework

- 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
  - 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

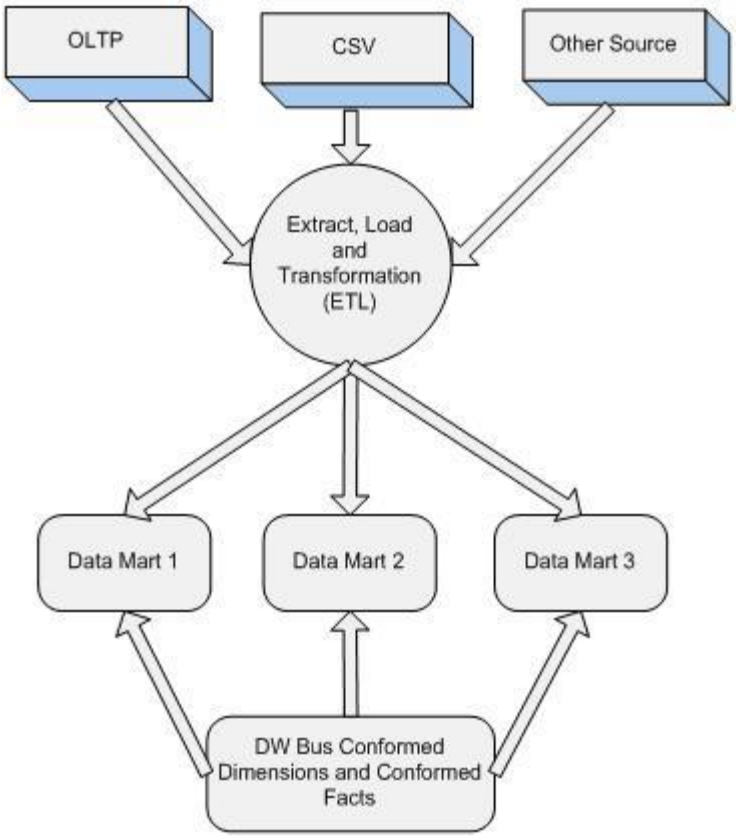
Top Down Approach



Top Down Approach

- **Advantages Of Top Down Design :**
  - It is easier to maintain Top Down Design
  - Provides consistent dimensional views of data across data marts, as all data marts are loaded from the data warehouse.
  - This approach is robust against business changes. Creating a new data mart from the data warehouse is very easy.
  - Initial cost is high but subsequent project development cost is lower
- **Disadvantages Of Top Down Design :**
  - It represents a very large project and the cost of implementing the project is significant.
  - It is time consuming and more time required for initial set up.
  - This technique is inflexible to changing departmental needs.
  - Highly skilled people required for set up

Bottom Up Approach



- **Advantages Of Bottom Up Design :**
  - This model contains consistent data marts and these data marts can be delivered quickly.
  - The data marts are created first to provide reporting capability
  - It is easier to extend the data warehouse as it can easily accommodate new business units. It is just creating new data marts and then integrating with other data marts.
  - This Approach take less time. Initial set up is very quickly
- **Disadvantage of Bottom Up Design :**
  - Initial cost is low but each subsequent phase will cost same
  - The positions of the data warehouse and the data marts are reversed in the bottom-up approach design.
  - It is difficult to maintain and often redundant and subject to revisions

Top-Down Design Approach	Bottom-Up Design Approach
Breaks the vast problem into smaller subproblems.	Solves the essential low-level problem and integrates them into a higher one.
Inherently architected- not a union of several data marts.	Inherently incremental; can schedule essential data marts first.
Single, central storage of information about the content.	Departmental information stored.
Centralized rules and control.	Departmental rules and control.
It includes redundant information.	Redundancy can be removed.

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

OLAP Server Architectures

- Relational OLAP (ROLAP)
- Multidimensional OLAP (MOLAP)
- Hybrid OLAP (HOLAP) (e.g., Microsoft SQLServer)
- Specialized SQL servers (e.g., Redbricks)
  - Specialized support for SQL queries over star/snowflake schemas

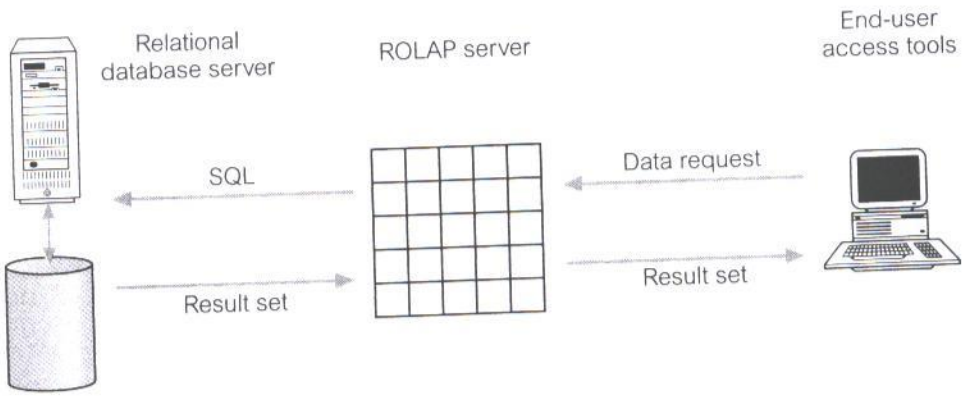
Relational OLAP (ROLAP)

- 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
- 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.

Relational OLAP (ROLAP)

- 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.
  - Development of an option to create persistent multi-dimensional structures, together with facilities o assist in the administration of these structures.

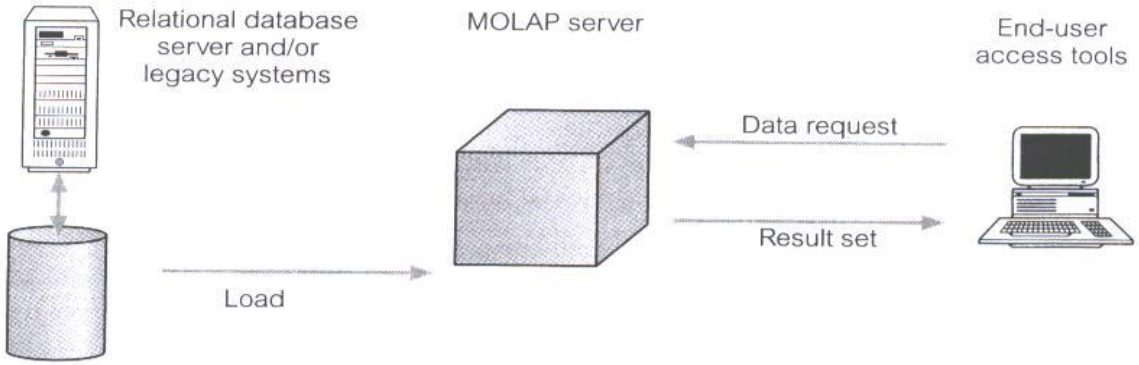
Relational OLAP (ROLAP)



Multidimensional OLAP (MOLAP)

- MOLAP tools use specialized data structures and multi-dimensional database management systems (MDDBMS) to organize, navigate, and analyze data.
- Sparse array-based multidimensional storage engine that minimize the disk space requirements through sparse data management.
- Fast indexing to pre-computed summarized data
- 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.
  - MOLAP products require a different set of skills and tools to build and maintain the database.

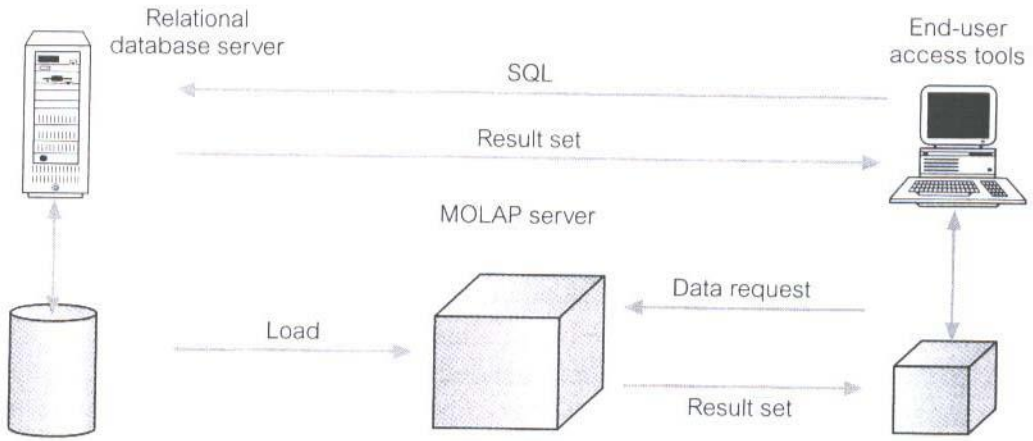
Multidimensional OLAP (MOLAP)



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.
- Flexibility, e.g., low level: relational, high-level: array
- 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 of data consistency among users.
  - Only a limited amount of data can be efficiently maintained.

Hybrid OLAP (HOLAP)



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
  - How many cuboids in an n-dimensional cube with L levels?

- Materialization of data cube $T = \prod_{i=1}^{n} (L_i + 1)$
  - 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.

This approach involves pre-computing and storing the data cube in a database. This can be done using a materialized view, which is a pre-computed table that is based on a SELECT statement.

Advantage: The advantage of this approach is that data cube queries can be answered quickly since the data is already pre-computed and stored in the database.

Disadvantage: The disadvantage is that the materialized view needs to be updated regularly to reflect changes in the underlying data.

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
- **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?

Full materialization refers to the computation of all of the cuboids in the lattice defining a data cube. It typically requires an excessive amount of storage space,particularly as the number of dimensions and size of associated concept hierarchies grow. This problem is known as the curse of dimensionality. Alternatively, partial materialization is the selective computation of a subset of the cuboids or subcubes in the lattice. For example, an iceberg cube is a data cube that stores only those cube cells that have an aggregate value (e.g., count) above some minimum support threshold.

OLAP query processing can be made more efficient with the use of indexing techniques. In bitmap indexing, each attribute has its own bitmap index table.

Bitmap indexing reduces join, aggregation, and comparison operations to bit arithmetic.

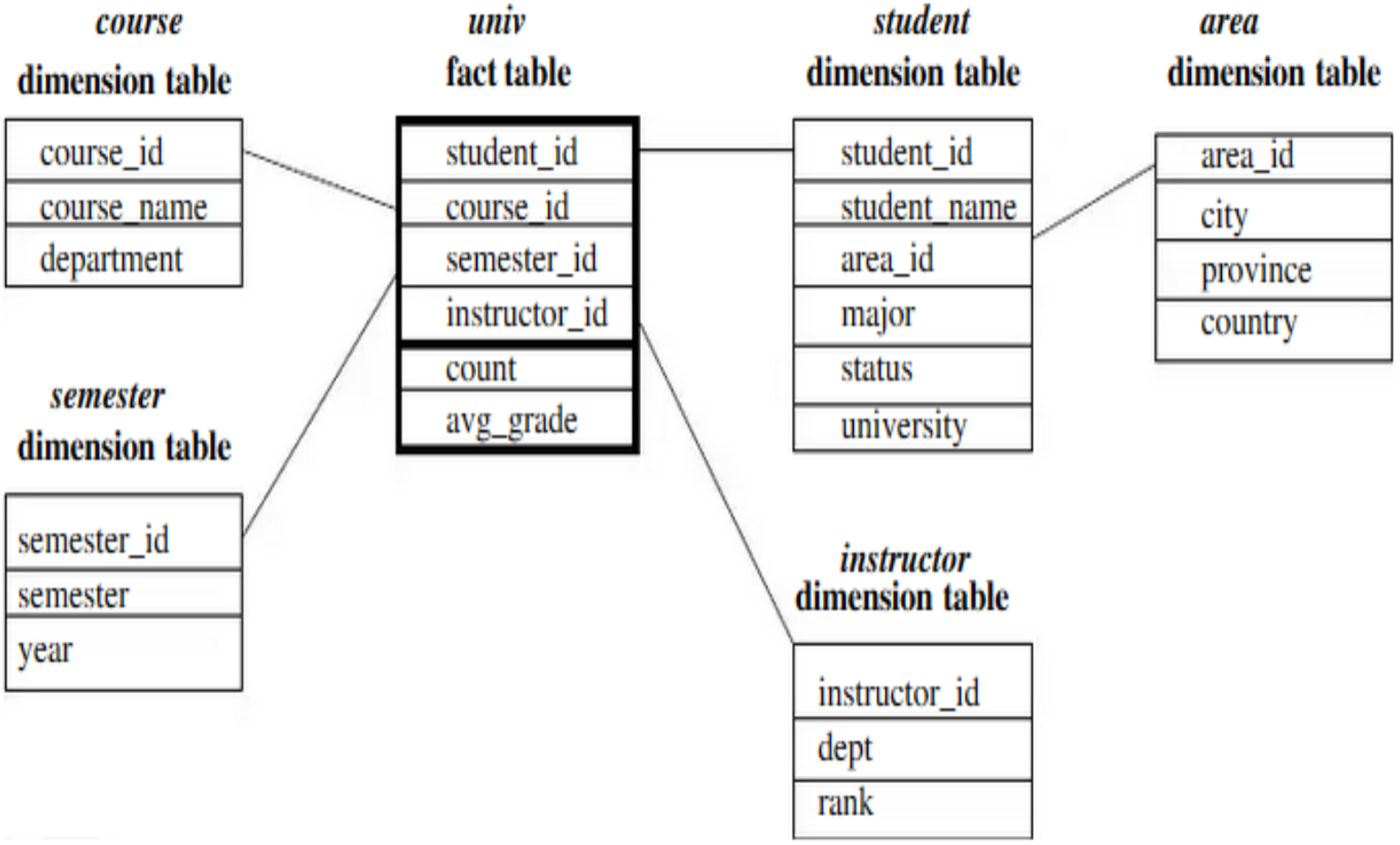
Join indexing registers the joinable rows of two or more relations from a relational database, reducing the overall cost of OLAP join operations. Bitmapped join index-
ing, which combines the bitmap and join index methods, can be used to furtherspeed up OLAP query processing

Case Study

- **Suppose that a data warehouse for Big University consists of the following four dimensions: student, course, semester, and instructor , and two measures count and avg grade . When at the lowest conceptual level (e.g.,for a given student, course, semester, and instructor combination), the avg grade measure stores the actualcourse grade of the student. At higher conceptual levels, avg grade stores the average grade for the givencombination.**

(a) Draw a snowflake schema diagram for the data warehouse.
(b) Starting with the base cuboid [student,course,semester,instructor], what specific OLAP operations (e.g., roll-up from semester to year) should one perform in order to list the average grade of CS coursesfor each Big University student.
(c) If each dimension has five levels (including all), such as " student < major < status < university < all ",how many cuboids will this cube contain (including the base and apex cuboids)?

Snowflake Schema diagram



- **Starting with the base cuboid [student,course,semester,instructor], what specific OLAP operations (e.g., roll-up from semester to year) should one perform in order to list the average grade of CS courses for each Big University student. The specific OLAP operations to be performed are:**

• Roll-up on course from course id to department .
• Roll-up on student from student id to university.
• Dice on course, student with department="CS" and university = "Big University".
• Drill-down on student from university to student name

- **(c) If each dimension has five levels (including all), such as student < major < status < university < all,how many cuboids will this cube contain (including the base and apex cuboids)**
- This cube will contain 54= 625 cuboids