Data Warehousing

Lecture 2 Modelling of Data Warehouses and OLAP

CITS3401
CITS5504

Zeyi Wen

Computer Science and Software Engineering

School of Maths, Physics and Computing

Acknowledgement: The lecture slides are based on online sources.

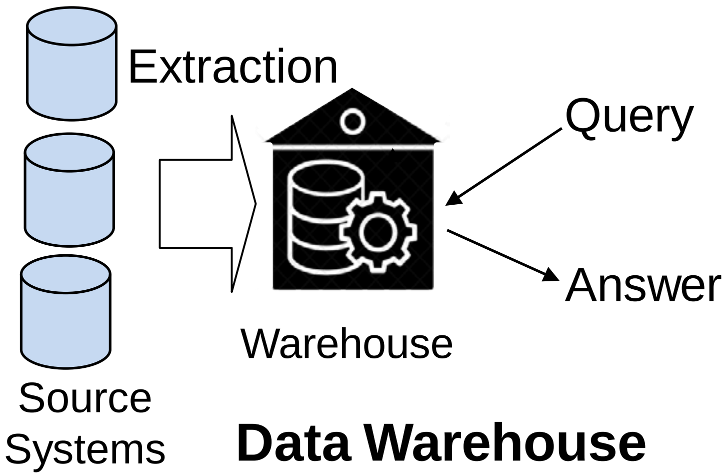
Lecture Outline

THE UNIVERSITY OF WESTERN AUSTRALIA

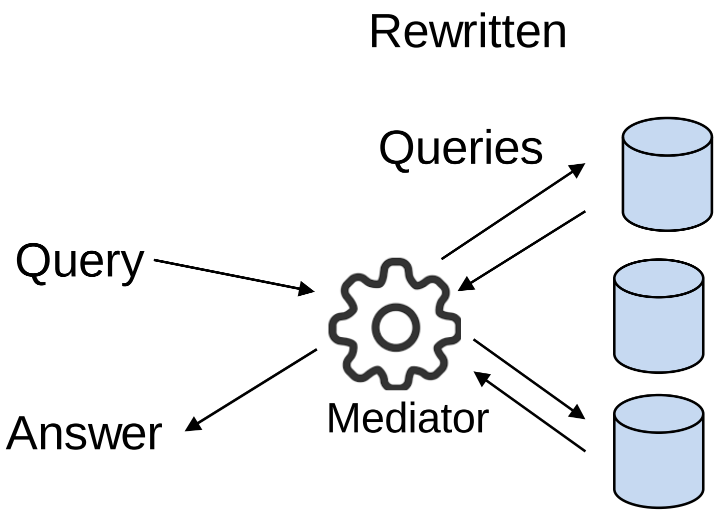
- Important Application of Data Warehouse
- Storing Data in Data Warehouse
- Fact Tables and Dimension Tables
- Schema of a Data Warehouse
  - ➢ Star, Snowflakes, Fact Constellations
- OLAP Operations
  - ➢ Roll up, Drill down, Slice & Dice, Pivot

Federated Databases

- **An alternative to data warehouses**
- **Data warehouse**
  - Create a copy of all the data
  - Execute queries against the copy
- **Federated database**
  - Pull data from source systems as needed to answer queries
- **"lazy" vs. "eager" data integration**



Data Warehouse



Federated Database

Warehouse vs. Federation

- Advantages of federated databases:
  - No redundant copying of data
  - Queries see “real-time” view of evolving data
  - More flexible security policy

- Disadvantages of federated databases:
  - Analysis queries place extra load on operational DB systems
  - Query optimisation is hard to do well
  - Historical data may not be available
  - Complex “wrappers” needed to mediate between analysis server and source systems

- Data warehouses are much more common in practice
  - Better performance
  - Lower complexity
  - Slightly out-of-date data is acceptable

3 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

- **Data mining**
  - knowledge discovery from hidden patterns
  - supports associations, constructing analytical models, performing prediction, and presenting the mining results using visualisation tools.

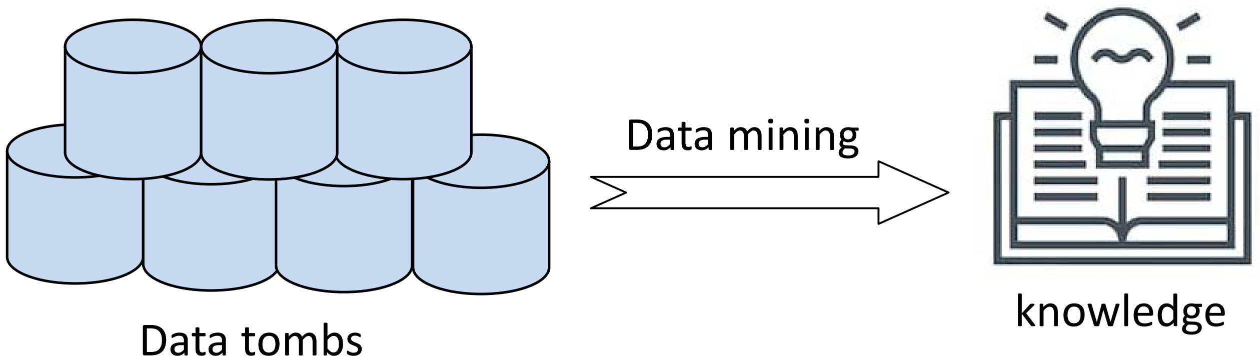
Why Data Mining?

- The Explosive Growth of Data: from terabytes to petabytes
  - Data explosion
    - Capability of generating, collecting, storing and managing data has grown tremendously in the last 50 years.
  - Large number of data sources
    - Automated data collection tools, database systems,
    - Business: Web, e-commerce, transactions, stocks, …
    - Science: Remote sensing, bioinformatics, scientific simulation, …
    - Everyone: news, digital cameras, YouTube
  - Far exceeded human ability for comprehension.

- We are drowning in data, but starving for knowledge!
  - Abundance of data and data archives are seldom visited.
  - Manual knowledge extraction is prone to biases and errors, and is extremely costly and time consuming.

What Data Mining Does?

- Data mining: Automated and scalable analysis of massive data
- Perform data analysis and uncover important data patterns,
- Contribute greatly to business strategies, knowledge bases, and scientific and medical research.



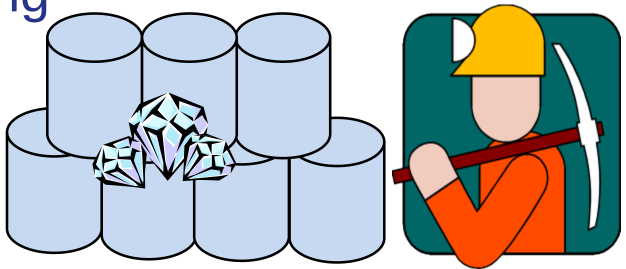
What is Data Mining?

- Data mining (knowledge discovery from data)
  - Extraction of interesting (non-trivial, implicit, previously unknown and potentially useful) patterns or knowledge from huge amount of data
  - Data mining: a misnomer? (Knowledge Mining from data)

- Alternative names
  - Knowledge discovery (mining) in databases (KDD), knowledge extraction, data/pattern analysis, data archeology, data dredging, information harvesting, business intelligence, etc.

- Watch out: Is everything "data mining"?
  - Simple search and query processing

Community for Data Mining: https://www.kdd.org/



Data Mining in Different Data Sources

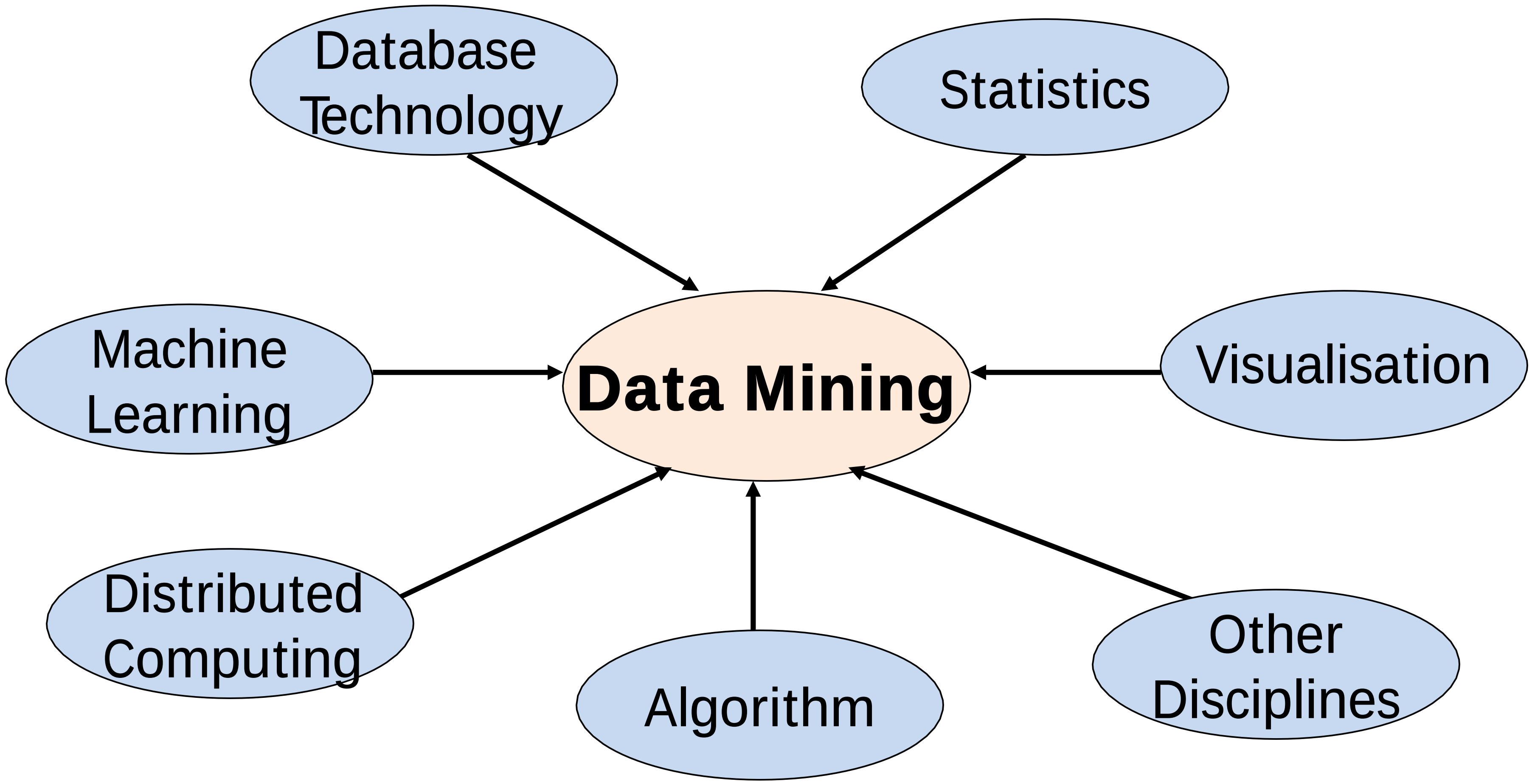
- Structured and semi-structured data from
  - Relational database
  - Data Warehouse,
  - Transactional database

Relatively easy

Hard

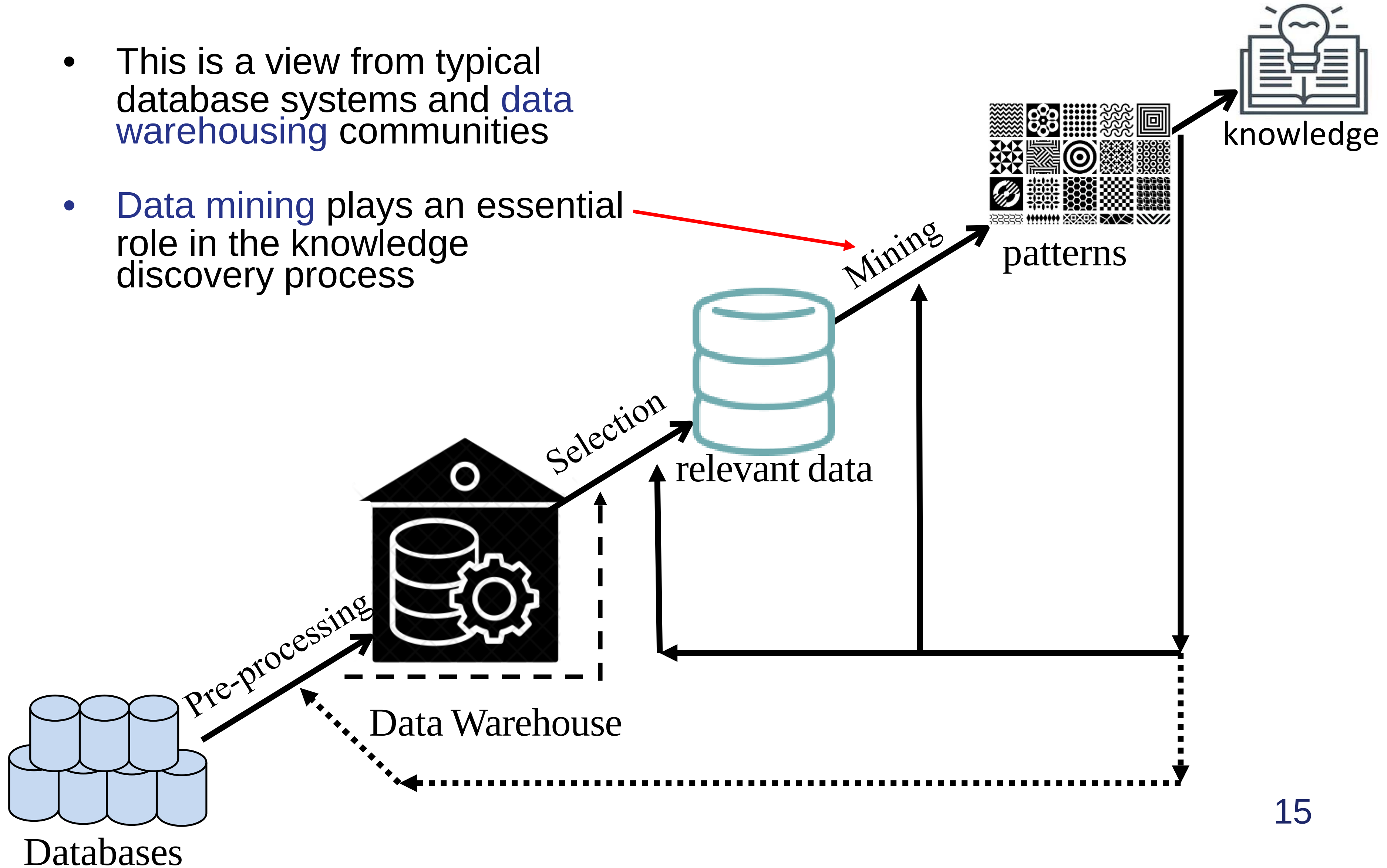
- Unstructured data
  - Data streams and sensor data
  - Text data
  - Time-series data, temporal data, sequence data (incl. bio-sequences)
  - Graphs, social networks and information networks
  - Spatial data, spatiotemporal data and multimedia data

Data Mining: Confluence of Multiple Disciplines

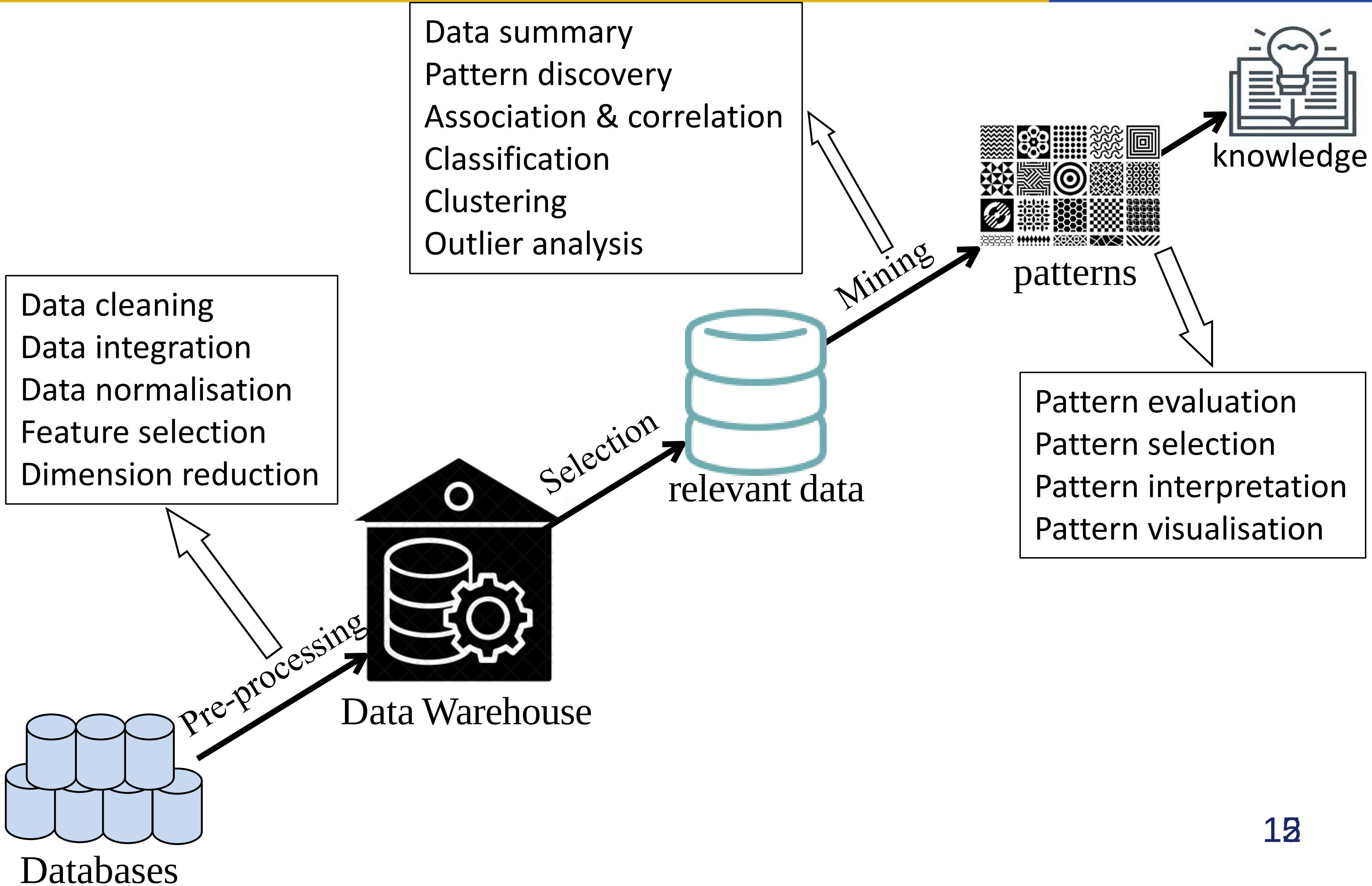


Steps of Knowledge Discovery from Data (KDD) Process

- This is a view from typical database systems and data warehousing communities
- Data mining plays an essential role in the knowledge discovery process



Techniques in the Process



Coupling DM with DB/DW Systems

- No coupling
  - ➢ flat file processing,
  - ➢ is a poor design as may spend much time in preprocessing.
- Loose coupling
  - ➢ Fetching data from DB/DW. Mining does not explore data structure and optimisation methods provided by databases & Data Warehouse. Difficult for high scalability.
- Semi-tight coupling—enhanced DM performance
  - – Provide efficient implementation for a few data mining primitives in a DB/DW system, e.g. sorting, indexing, aggregation, histogram analysis, multiway join, precomputation of some statistical functions
- Tight coupling—uniform processing environment
  - – DM is smoothly integrated into a DB/DW system, mining query is optimised based on mining query, indexing, query processing methods, etc.

Lecture Outline

- Important Applications of Data Warehouse
- Storing Data in Data Warehouse
- Fact Tables and Dimension Tables
- Schema of a Data Warehouse
  - Star, Snowflakes, Fact Constellations
- OLAP Operations
  - Roll up, Drill down, Slice & Dice, Pivot

Storing Data in Data Warehouse

- Data Warehouse is a more advanced database management system on storing large amount of data.
- Data Warehouse is built on top of a database management system (e.g. RDBMS like SQL Server)
- Data Warehouses store data using tables like DB systems.
- Data Cube is used to support OLAP operations.

Multi-dimensional View of Data (2-D)

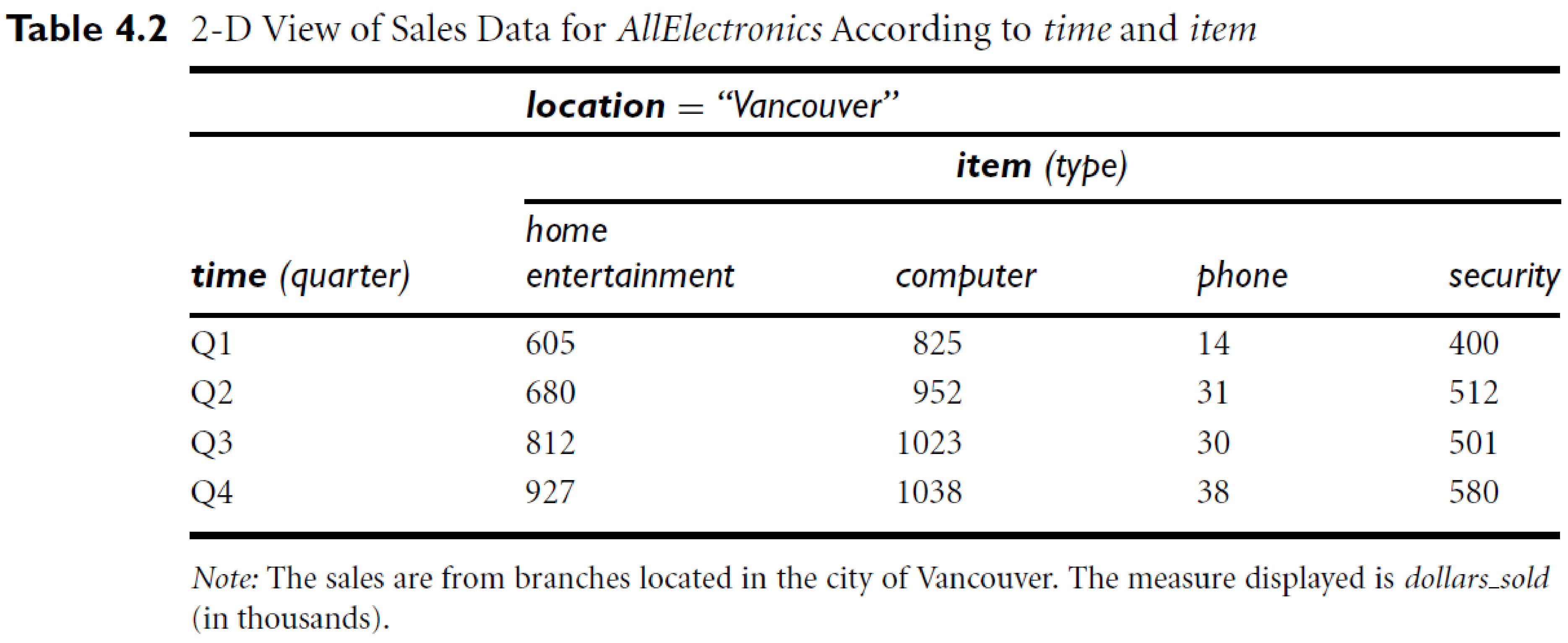


Table 4.2 2-D View of Sales Data for *AllElectronics* According to *time* and *item*

	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).

Multi-dimensional View of Data (3-D)

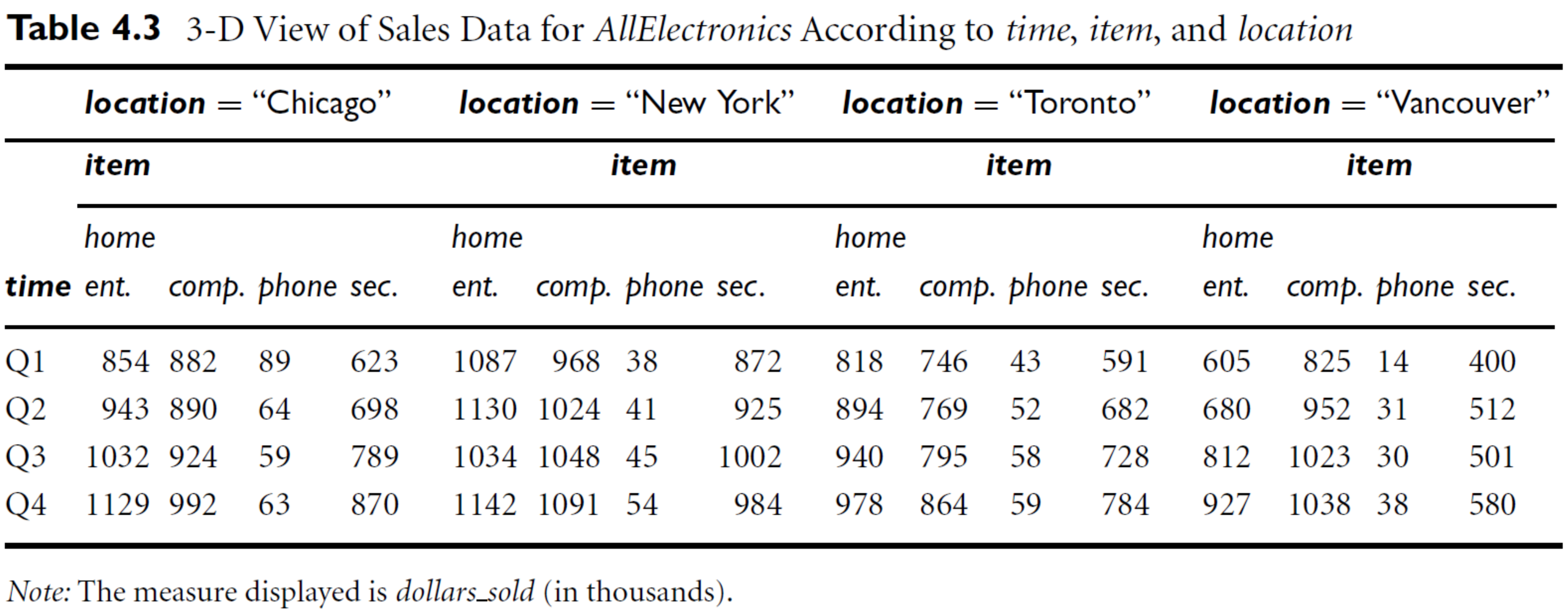


Table 4.3 3-D View of Sales Data for *AllElectronics* According to *time*, *item*, and *location*

	location = "Chicago"				***location*** = "New York"				***location*** = "Toronto"				***location*** = "Vancouver"			
	item				***item***				***item***				***item***			
time	*home ent.*	*comp.*	*phone*	*sec.*	*home ent.*	*comp.*	*phone*	*sec.*	*home ent.*	*comp.*	*phone*	*sec.*	*home 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).

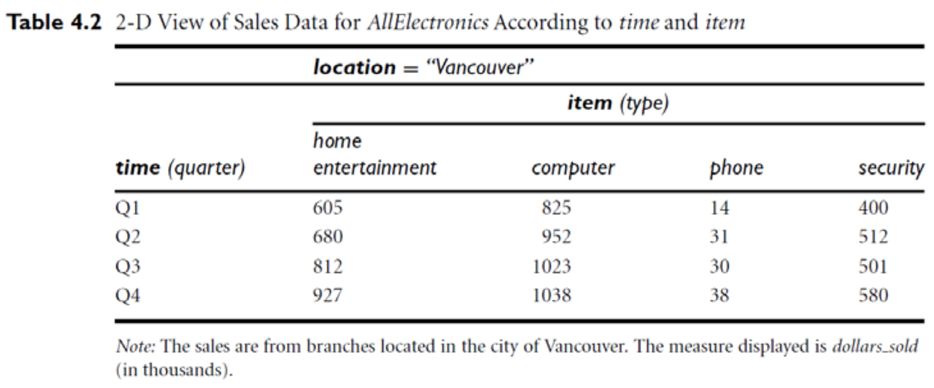
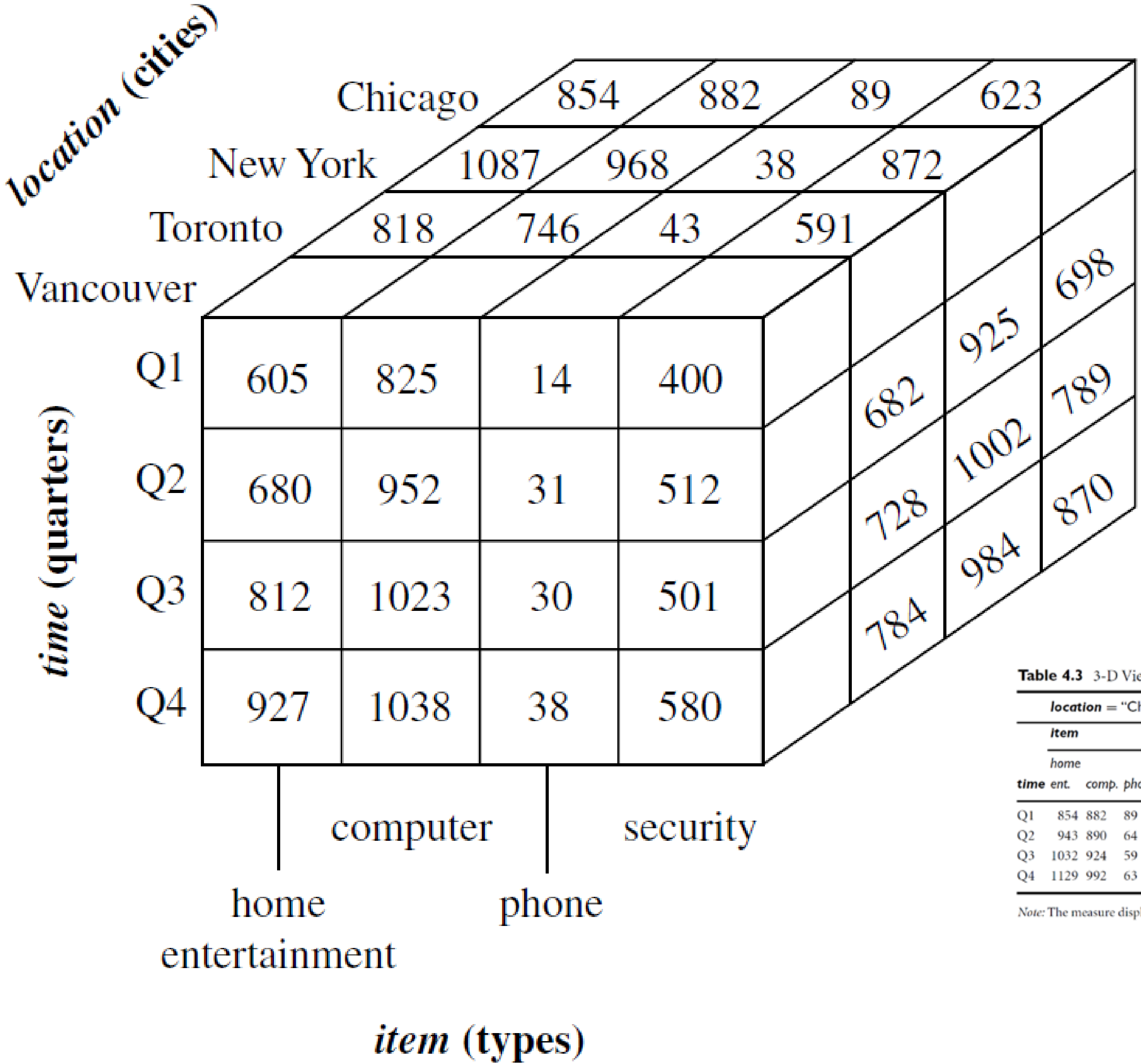


Table 4.2 2-D View of Sales Data for *AllElectronics* According to *time* and *item*

	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).

Multi-dimensional View of Data (Data Cube)



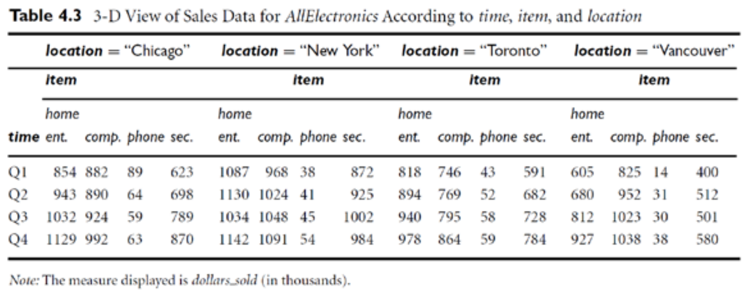


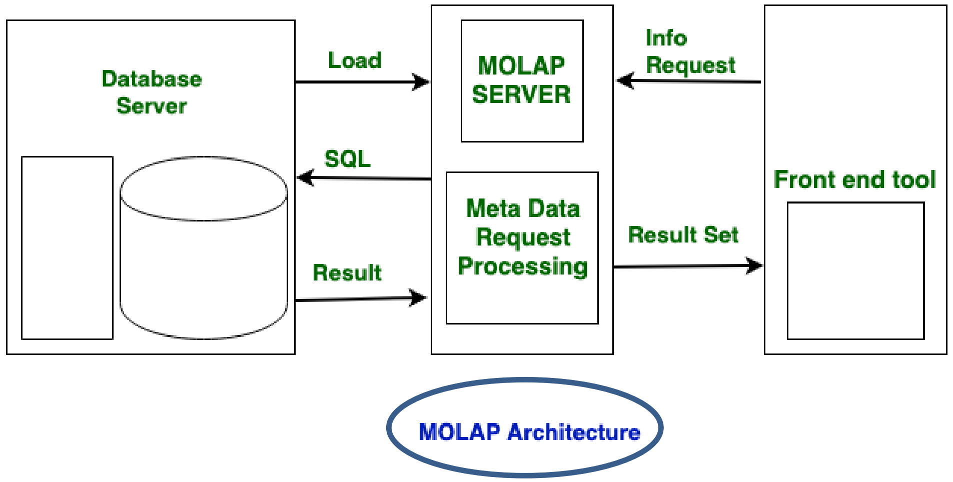
Table 4.3 3-D View of Sales Data for *AllElectronics* According to *time*, *item*, and *location*

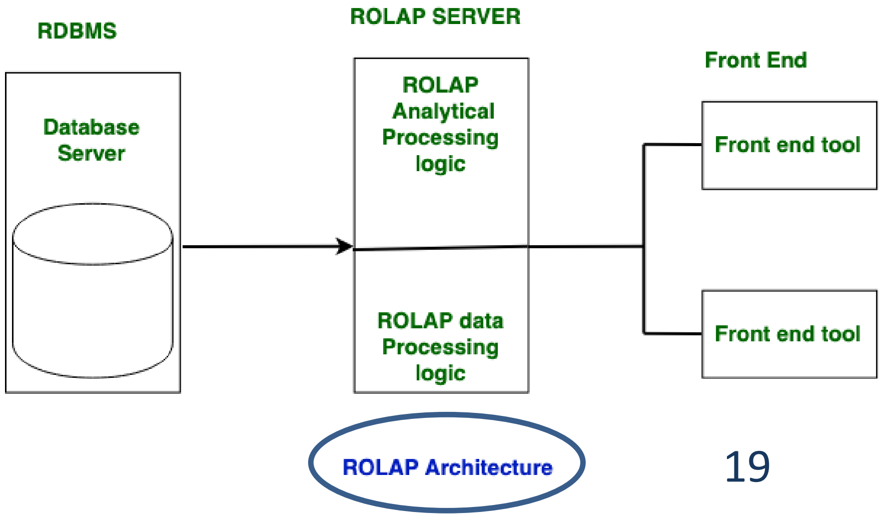
	location = "Chicago"				***location*** = "New York"				***location*** = "Toronto"				***location*** = "Vancouver"			
	item				***item***				***item***				***item***			
time	*home ent.*	*comp.*	*phone*	*sec.*	*home ent.*	*comp.*	*phone*	*sec.*	*home ent.*	*comp.*	*phone*	*sec.*	*home ent.*	*comp.*	*phone*	*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).

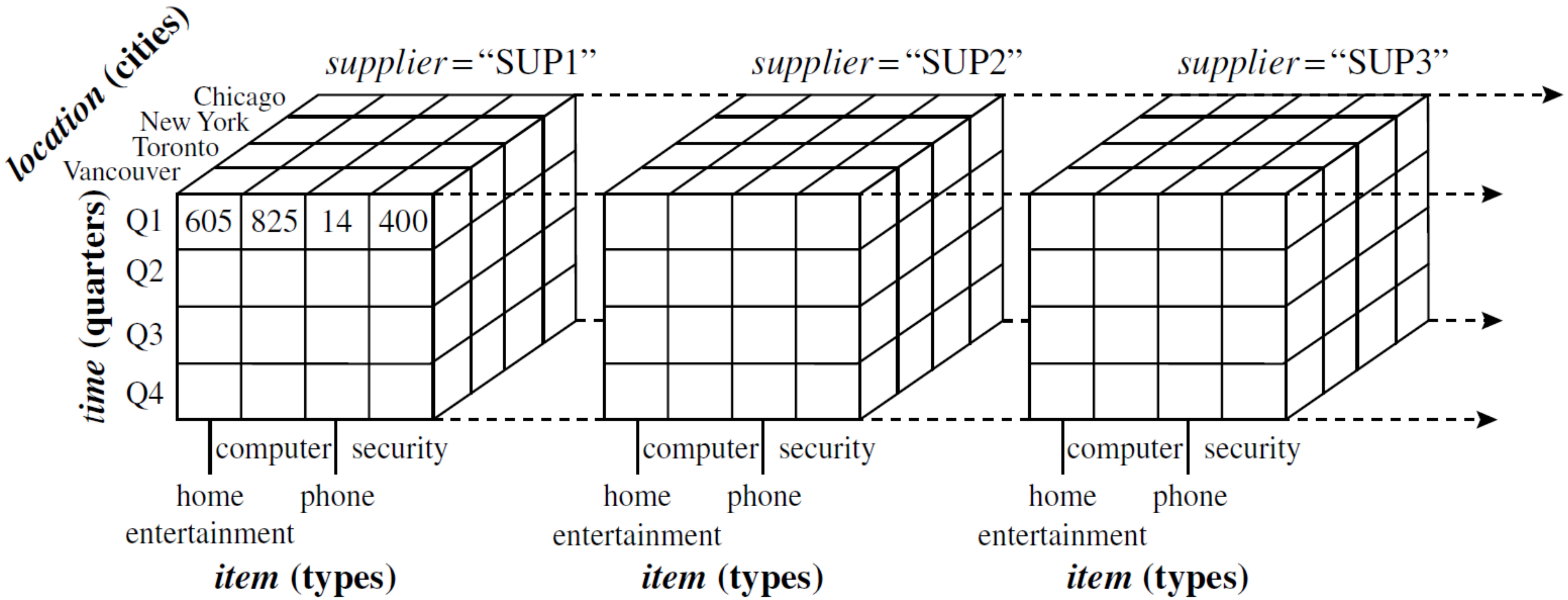
Data Cube is a Metaphor

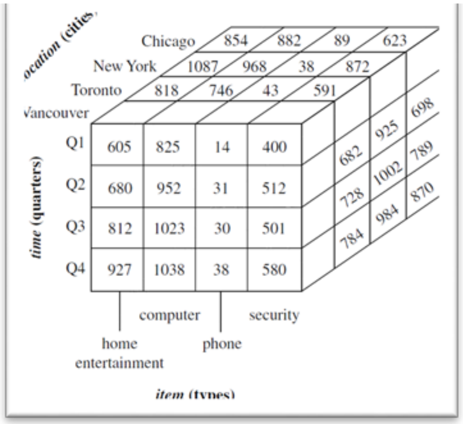
- Data cube is a metaphor for multi-dimentional data storage.
- The term hypercube is sometimes used, especially for data with more than three dimensions.
- Actual storage of such data may be different from the logical representation.





Data Cube: Don’t confine Data to 3-D

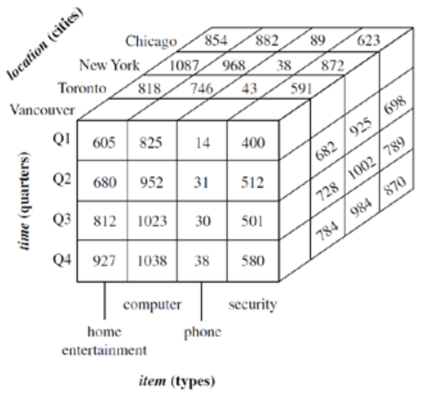




5-D data cube: a series of 4-D data cubes

From Tables to Data Cubes

- A data warehouse is based on a multi-dimensional data model which views data in the form of a data cube
- A data cube, is organised around a central theme, such as sales, allows data to be modelled and viewed in multiple dimensions
  - Dimension tables, such as item (item_name, brand, type), or time (day, week, month, quarter, year) or location (branch, city, state, country)
  - Fact table contains measures of central theme (such as dollars_sold, units sold) and keys to each of the related dimension tables



Lecture Outline

- Important Applications of Data Warehouse
- Storing Data in Data Warehouse
- Fact Tables and Dimension Tables
- Schema of a Data Warehouse
  - Star, Snowflakes, Fact Constellations
- OLAP Operations
  - Roll up, Drill down, Slice & Dice, Pivot

Fact Tables

- Each fact table contains measurements (e.g. dollar_sold) about a process of interest.

- Each fact row contains two things:
  - – Numerical measure columns
  - – Foreign keys to dimension tables

- Properties of fact tables:
  - – Very big
    - Often millions or billions of rows
  - – Narrow
    - Small number of columns
  - – Often append new rows to the fact table
    - New events in the world → new rows in the fact table

- Uses of fact tables:
  - – Obtain measurements from the fact table
  - – Aggregate measurements from columns of the fact table.

Grain of a fact table

- Grain of a fact table = the meaning of one fact table row
- Determines the maximum level of detail of the warehouse
- Example grain statements: (*one fact row represents a…*)
  - Line item from a cash register receipt
  - Boarding pass to get on a flight
  - Daily snapshot of inventory level for a product in a warehouse
  - Sensor reading per minute for a sensor
  - Student enrolled in a course
- Finer-grained fact tables:
  - are more expressive
  - have more rows
- Trade-off between performance and expressiveness
  - Rule of thumb: Error in favor of expressiveness
  - Pre-computed aggregates can solve performance problems

Measures

- A data cube measure is a numeric function that can be evaluated at each point in the data cube space.

- A measure value is computed for a given point by aggregating the data corresponding to the respective dimension–value pairs defining the given point.

Types of Measures

- **Types of measures**
  - Distributive:
    - An aggregate function is distributive if it can be computed in a distributed manner by applying the same function on partitioned sets.
    - count(), min(), and max() are distributive aggregate functions.
  - Algebraic:
    - An aggregate function is algebraic if it can be computed by an algebraic function with *M* arguments (where *M* is a bounded positive integer), each of which is obtained by applying a *distributive* aggregate function.
    - avg() (average) can be computed by sum()/count(), where both sum() and count() are distributive
    - standard_deviation().
  - Holistic:
    - median(), mode(), and rank().

Dimension Tables

- Each one corresponds to a real-world object or concept.
  - Examples: Customer, Product, Date, Employee, Region, Store, Promotion, Vendor, Partner, Account, Department
- Properties of dimension tables:
  - Contain many descriptive columns
    - Dimension tables are wide (dozens of columns)
  - Generally don’t have too many rows
    - At least in comparison to the fact tables
    - Usually < 1 million rows
  - Contents are relatively static
    - Almost like a lookup table
- Uses of dimension tables
  - Filters are based on dimension attributes
  - Grouping columns are dimension attributes
  - Fact tables are referenced through dimensions

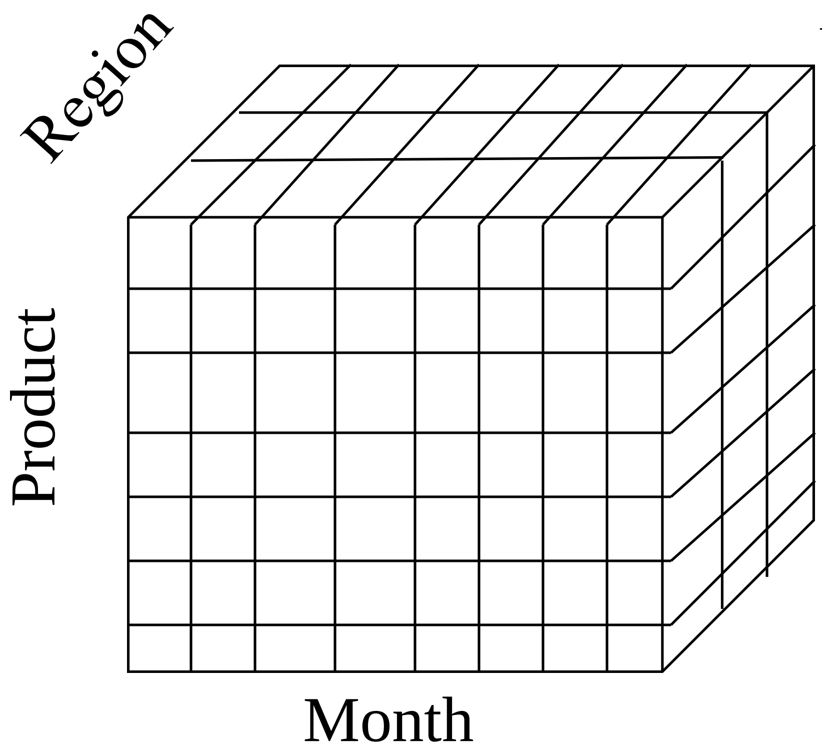
Dimension Tables

- Determine a candidate key based on the grain statement.
  - Example: a student enrolled in a course
  - (Course, Student, Term) is a candidate key

- Add other relevant dimensions that are functionally determined by the candidate key.
  - For example, Instructor and Classroom
    - Assuming each course has a single instructor!

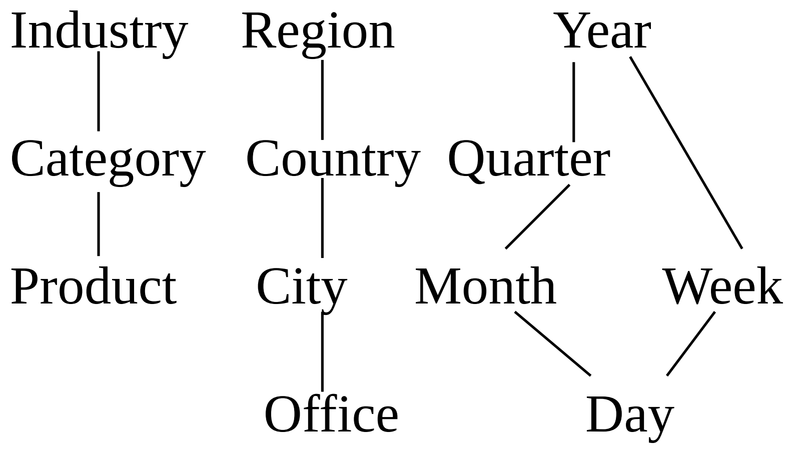
Concept Hierarchy in Dimensions

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



Dimensions: Product, Location, Time

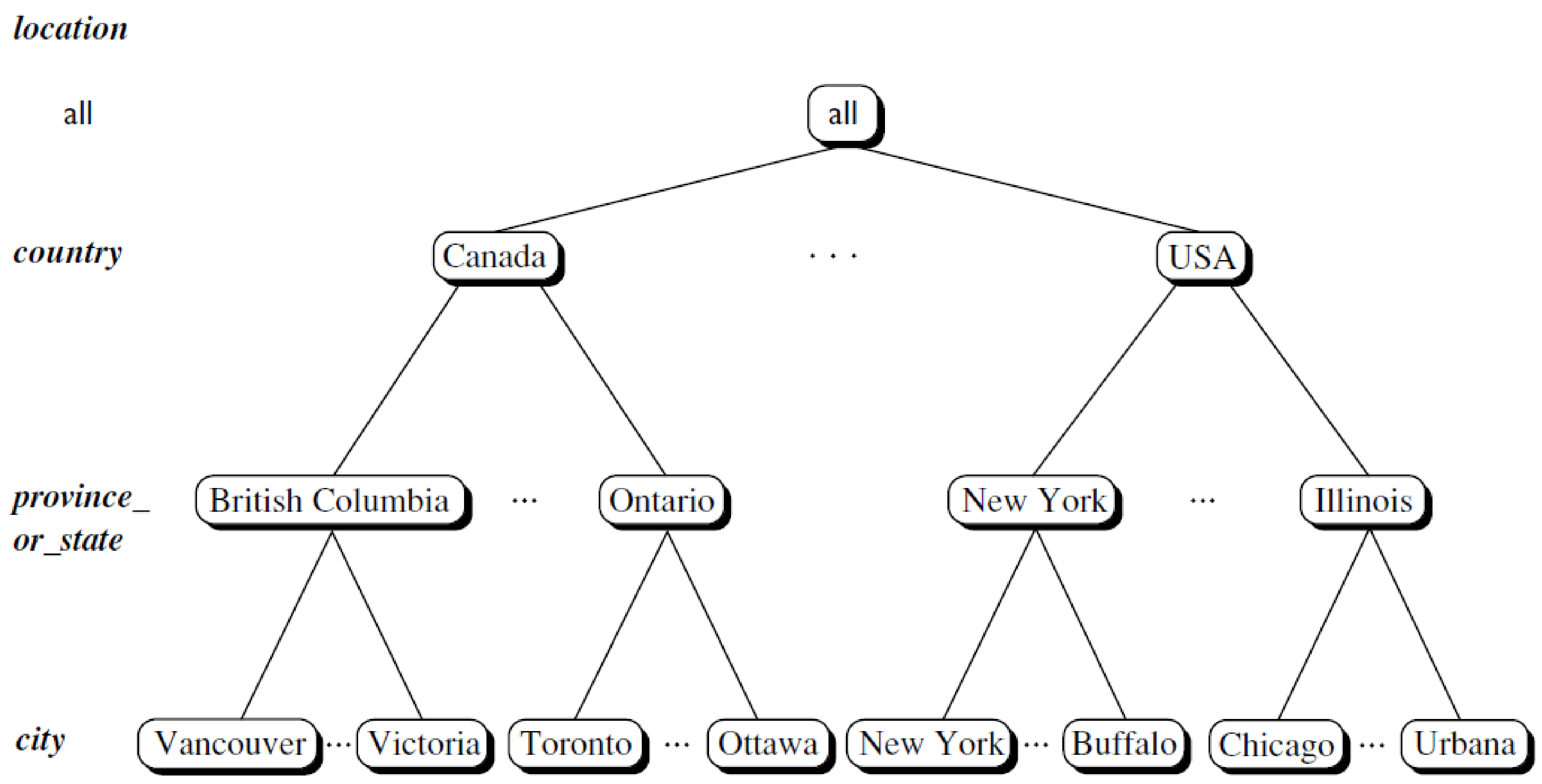
Hierarchical summarisation paths



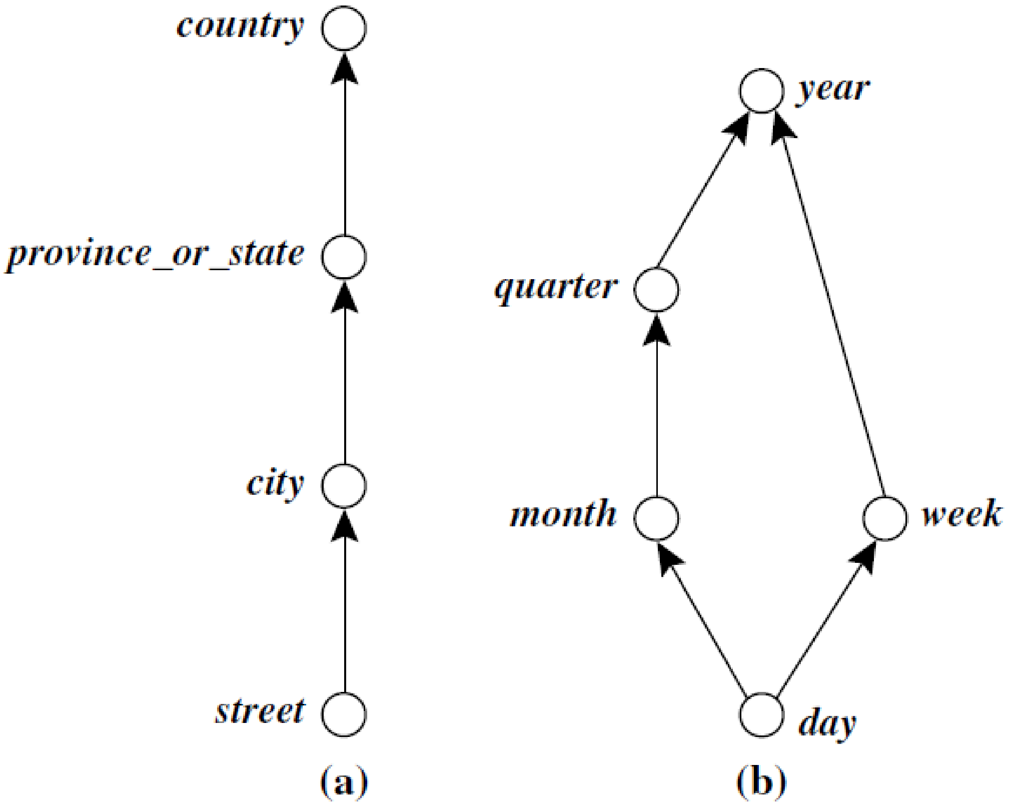
The Role of Concept Hierarchies

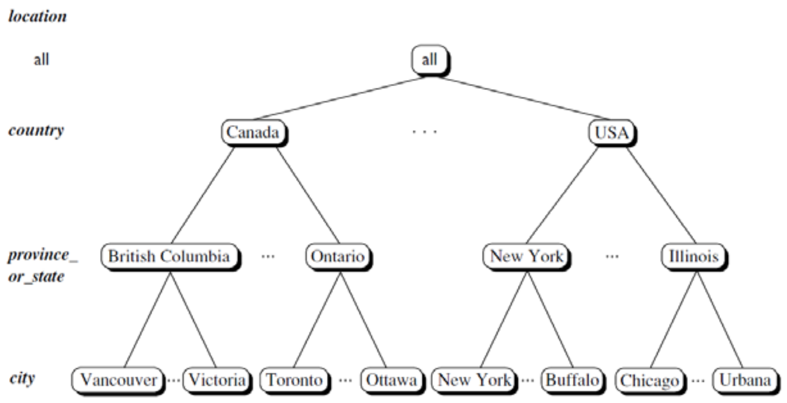
- **Concept Hierarchy**
  - Defines a sequence of mappings from a set of low-level concepts to high-level, more general concepts.
- **Schema Hierarchy**
  - A concept hierarchy that is a total or partial order among attributes in a database schema
    - Total order: street < city < province_or_state < country
    - Partial order: day < {month < quarter; week} < year
- **Set-grouping Hierarchy**
  - defined by discretising or grouping values for a given dimension or attribute.

Example Concept Hierarchies

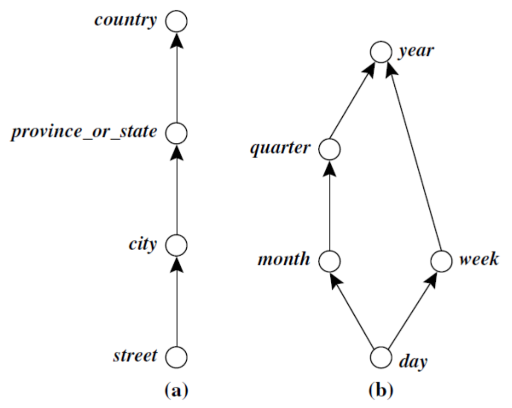


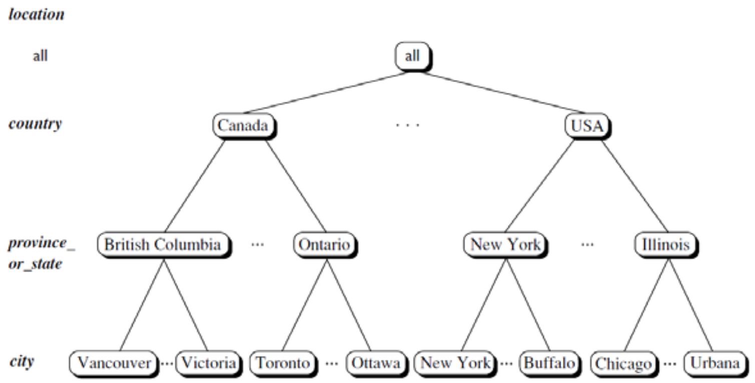
Example Concept Hierarchies: Schema Hierachy

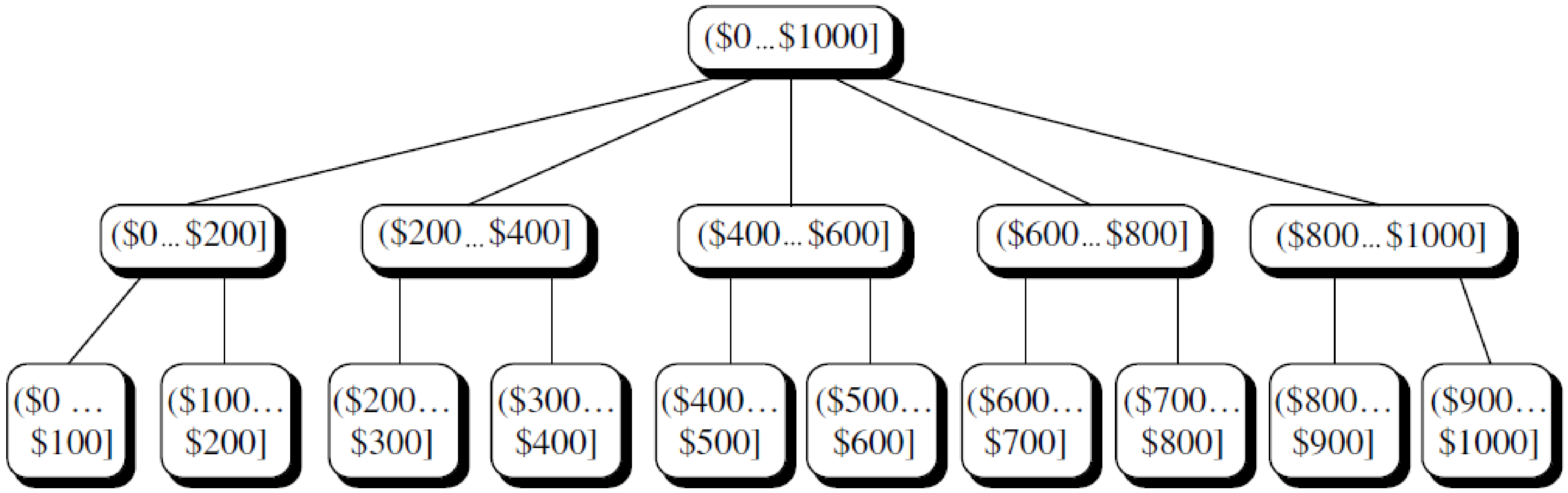




Example Concept Hierarchies: Set-grouping Hierarchy







A concept hierarchy for *price*.

Facts vs. Dimension Tables

Facts	Dimensions
• Narrow	• Wide
• Big (many rows)	• Small (few rows)
• Numeric	• Descriptive
• Growing over time	• Static

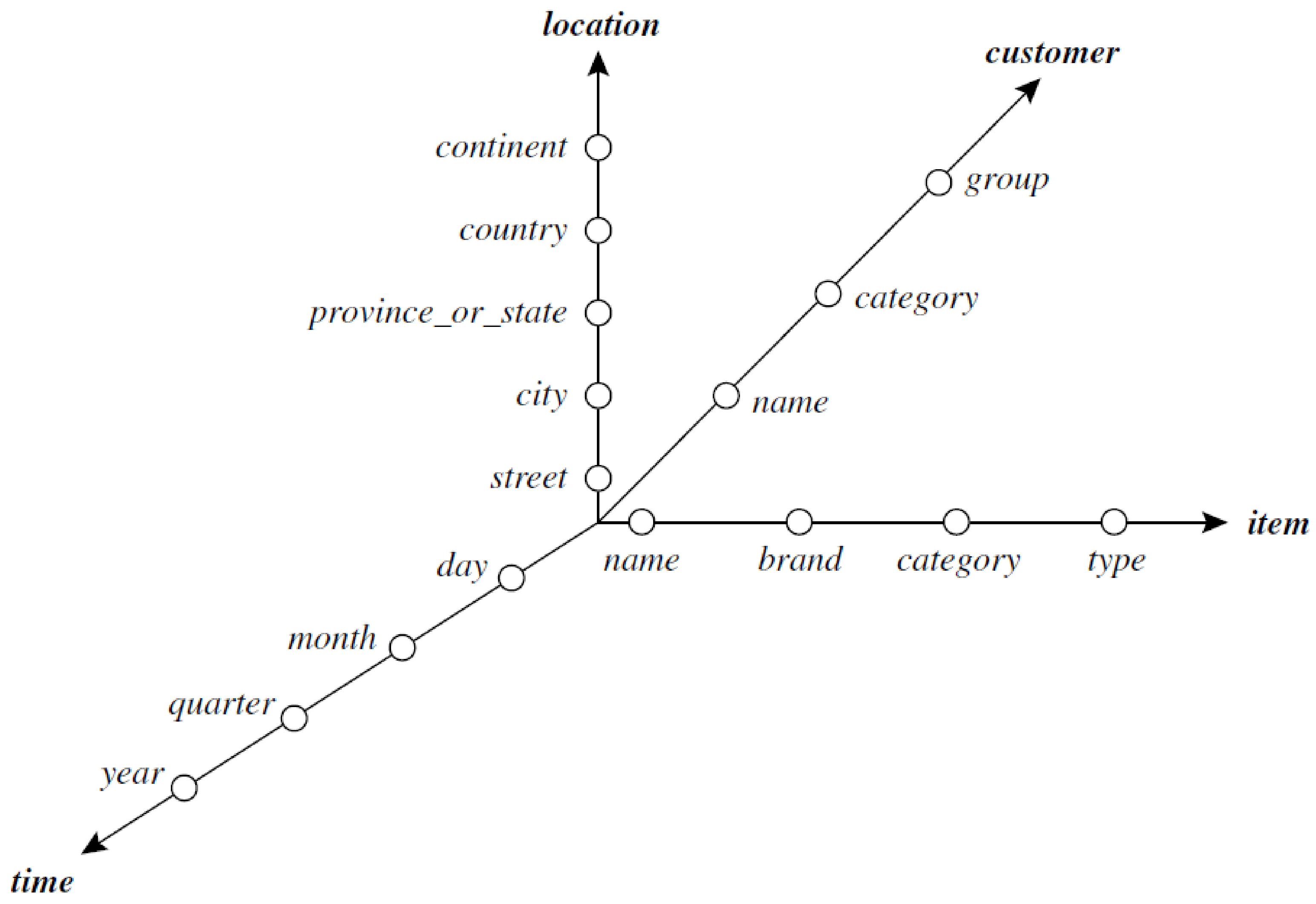
Multi-Dimensional Data Model

- Data Cube:
  - base cube, apex cube
  - concept of hierarchies

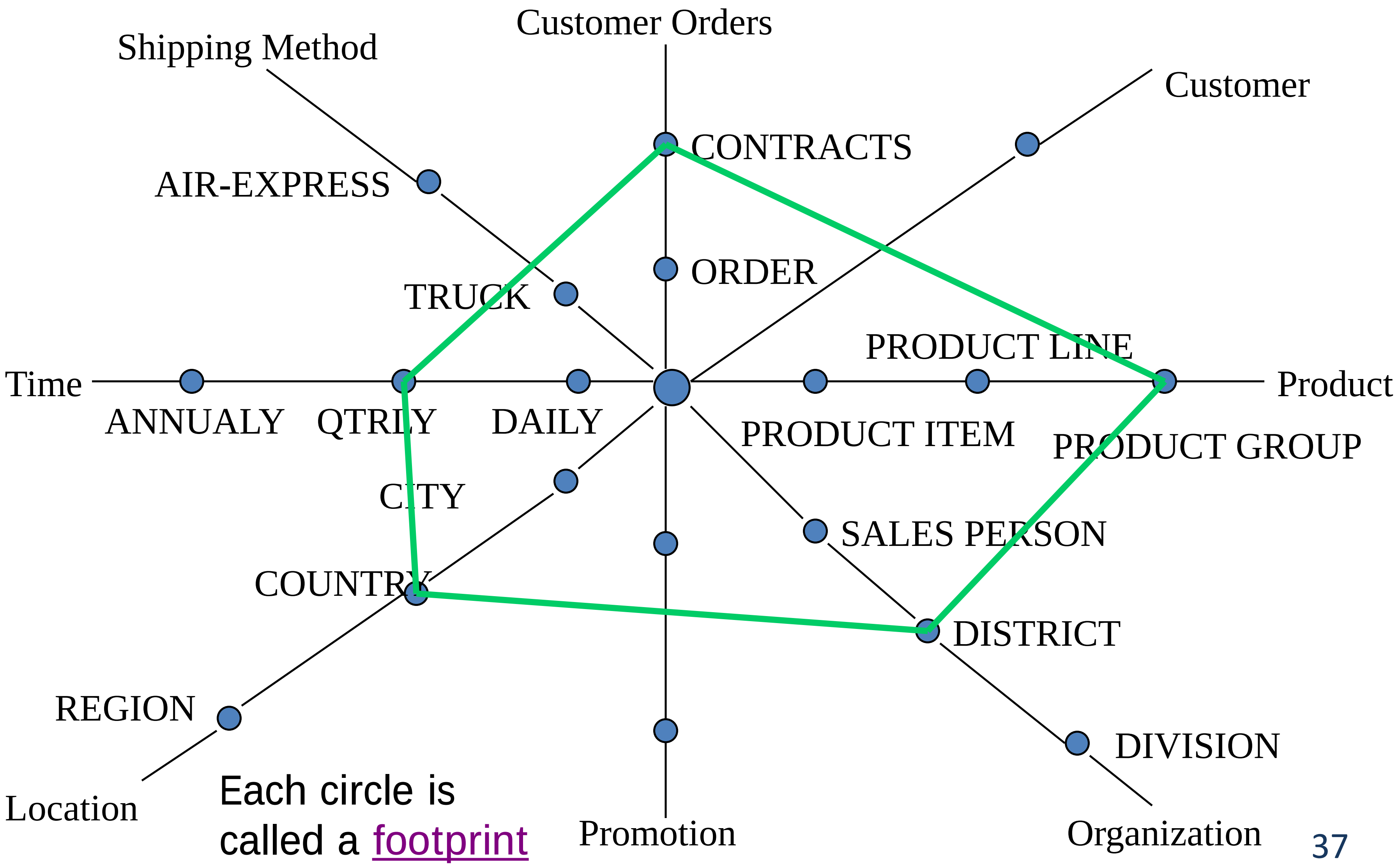
- Schemas:
  - Star, Snowflakes, Fact Constellations

- OLAP Operations:
  - Roll up, Drill down, Slice & Dice, Pivot

A Starnet Model of Business Queries



Granularity of viewing the data warehouse



Each circle is called a footprint

Data Warehouse Design Template

Kimball's four steps

- Identify a business process to model
  - E.g. orders, invoices, shipments, sales …
- Determine the grain of the business process
  - E.g. individual transactions, individual daily snapshots
- Choose the dimensions that apply to fact table rows
  - Example dimensions are time, item, customer, supplier, transaction type and status
- Identify the measure that populates fact table rows
  - Typical measures are numeric additive quantities like dollars_sold and units_sold

Logical Data Warehouse Design

- **Logical design vs. physical design:**
  - Logical design = conceptual organisation for the database
    - Create an abstraction for a real-world process
  - Physical design = how is the data stored
    - Select data structures (tables, indexes, materialised views)
    - Organise data structures on disk

- **Three main goals for logical design:**
  - Simplicity
  - Expressiveness
  - Performance

Goals for Logical Design

- **Simplicity**
  - Users should understand the design
  - Data model should match users’ conceptual model
  - Queries should be easy and intuitive to write
- **Expressiveness**
  - Include enough info. to answer important queries
  - Include all relevant data (without irrelevant data)
- **Performance**
  - An efficient physical design should be possible

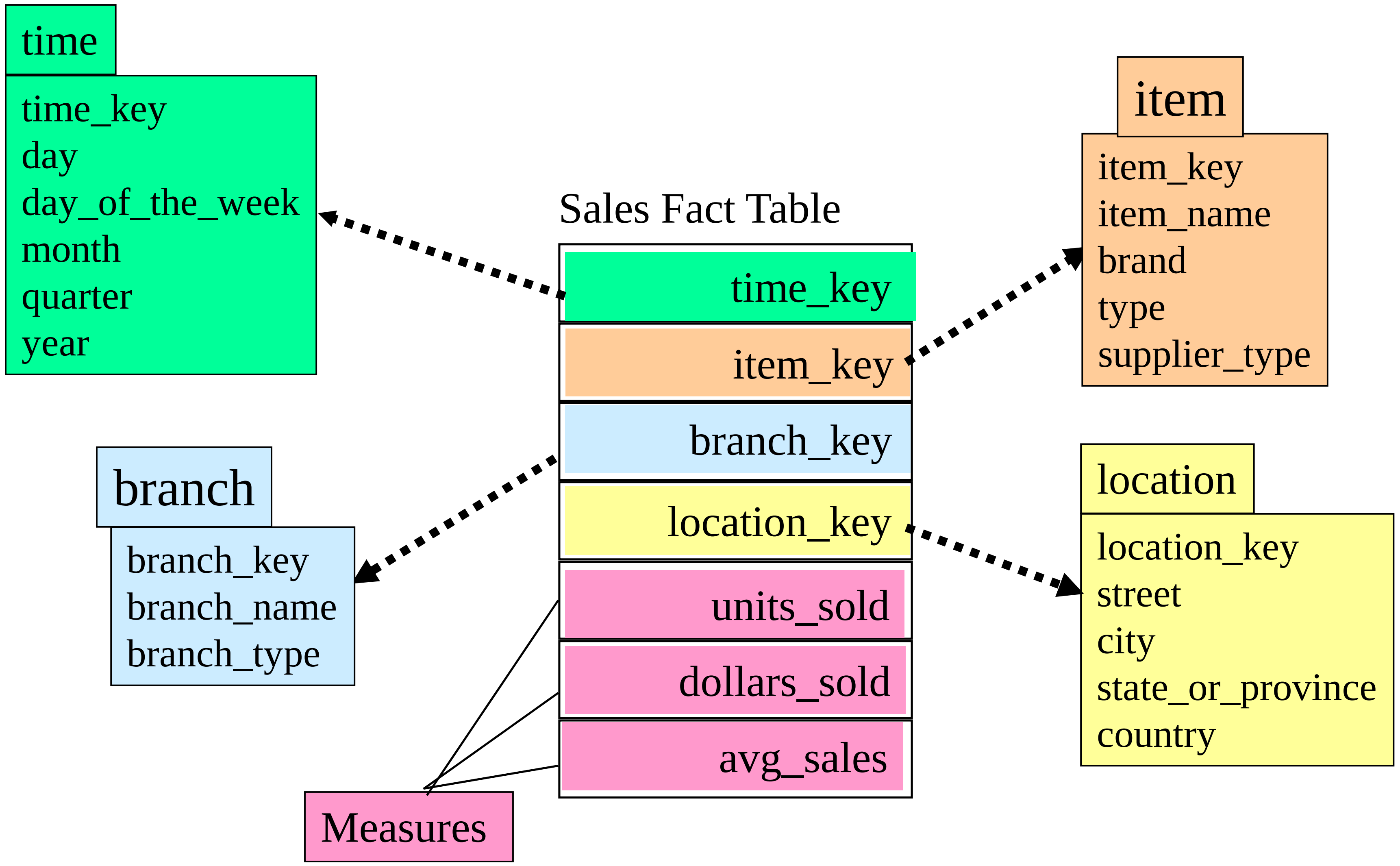
Lecture Outline

- Important Applications of Data Warehouse
- Storing Data in Data Warehouse
- Fact Tables and Dimension Tables
- Schema of a Data Warehouse
  - ➢ Star, Snowflakes, Fact Constellations
- OLAP Operations
  - ➢ Roll up, Drill down, Slice & Dice, Pivot

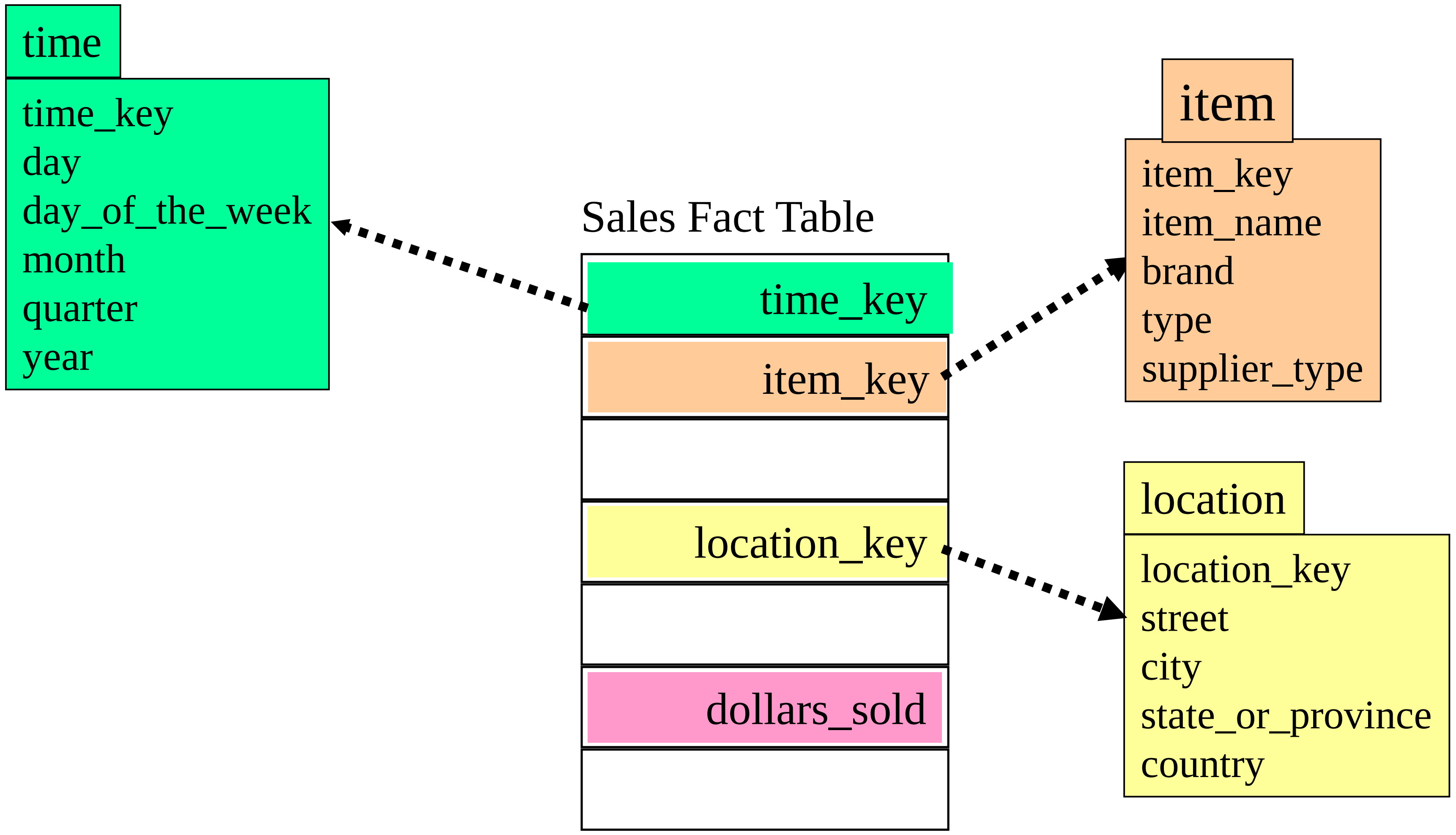
Schema

- **Star Schema**
  - A fact table in the middle connected to a set of dimension tables
- **Snowflake Schema**
  - Some dimensional hierarchy is normalised into a set of smaller dimension tables, forming a shape similar to snowflake.
  - Reduces redundancy at the cost of efficiency.
- **Galaxy schema (Fact Constellation)**
  - Multiple fact tables share dimension tables
  - Viewed as a collection of stars - Galaxy schema

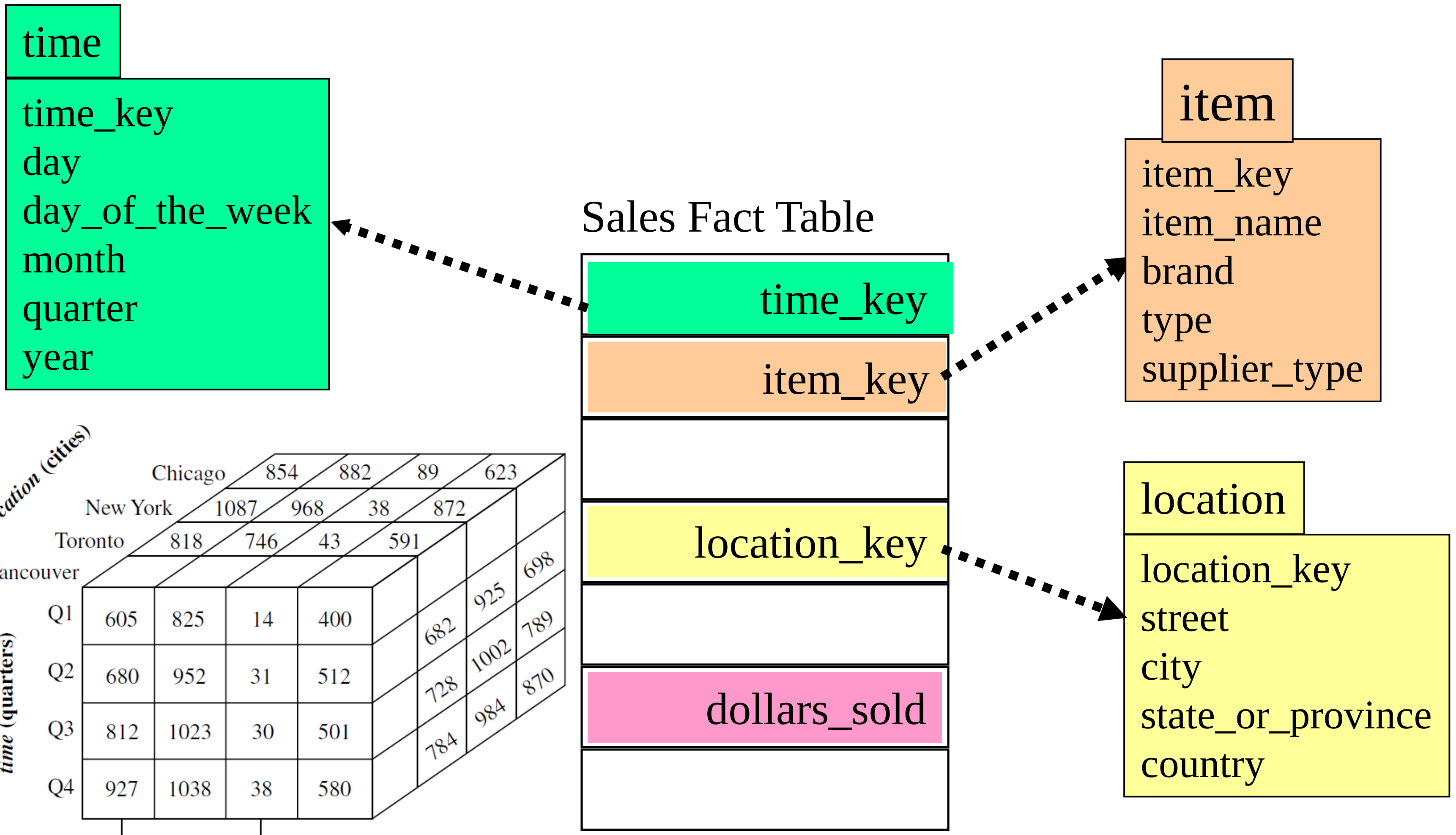
Star Schema

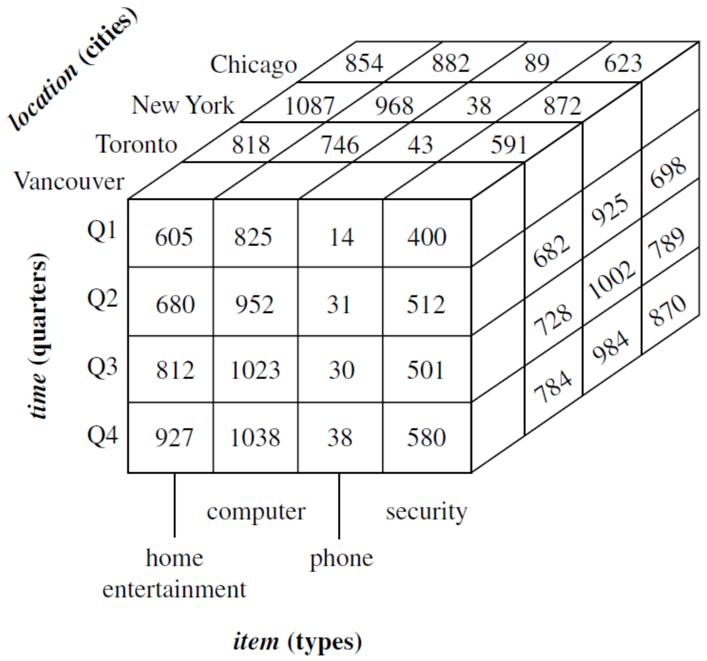

time
time_key
day
day_of_the_week
month
quarter
year
Sales Fact Table
time_key
item_key
branch_key
location_key
units_sold
dollars_sold
avg_sales
item
item_key
item_name
brand
type
supplier_type
branch
branch_key
branch_name
branch_type
location
location_key
street
city
state_or_province
country
Measures

Star Schema: Ignore some fields

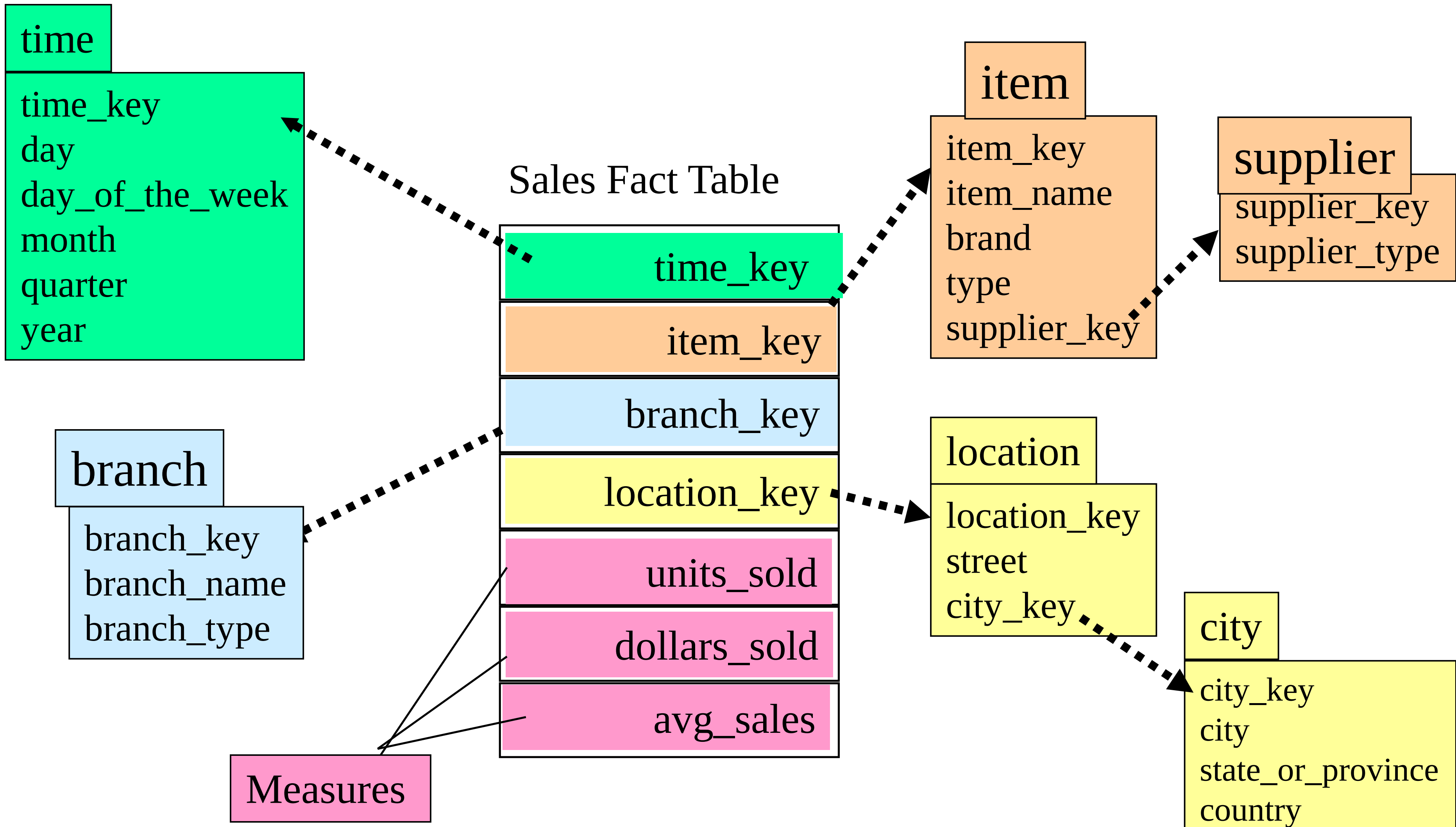


Star Schema: Representing Data Cube with Four Tables



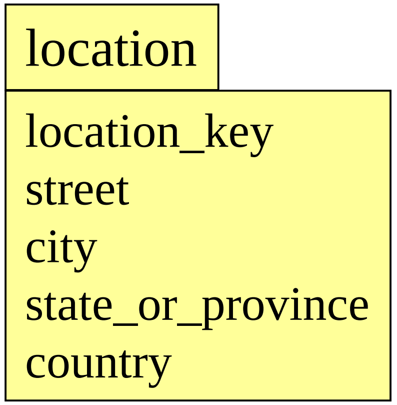


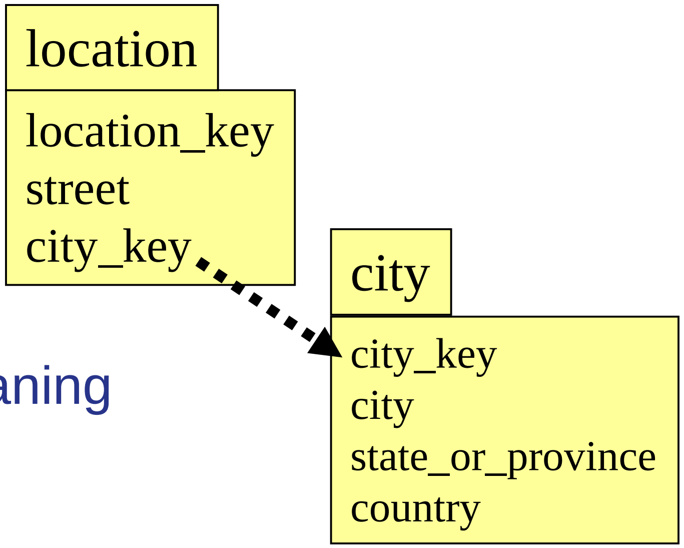
Snowflake Schema



Star Schema v.s. Snowflake Schema

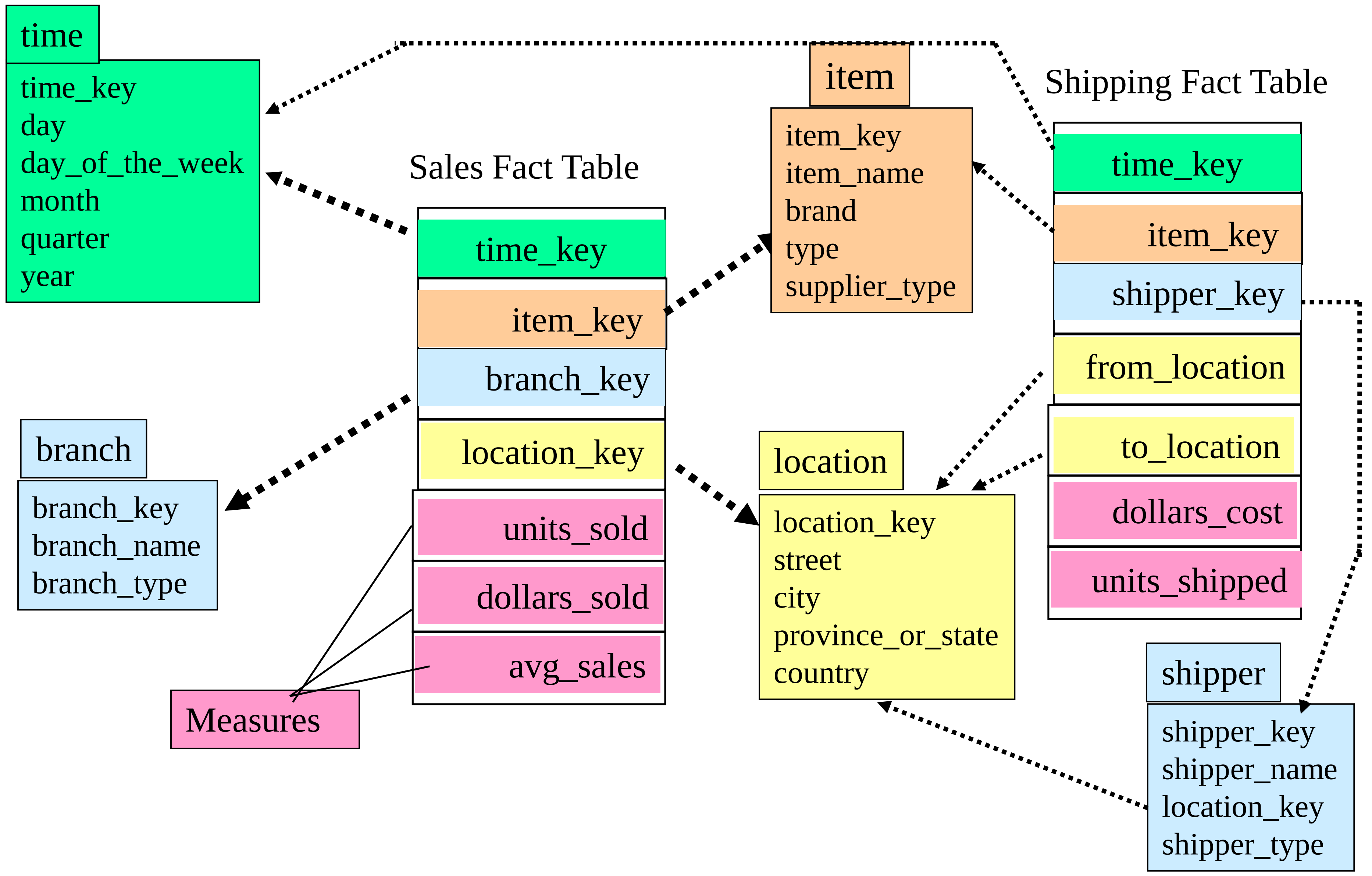
- **Star Schema**
  - Fewer tables, faster when browsing data
  - Has more redundant information
- **Snowflake Schema**
  - More tables, slower when browsing data
  - Reduces redundancy





- **Redundancy means:**
  - More storage
  - More work in data integration and cleaning

Fact Constellation – Galaxy Schema



Lecture Outline

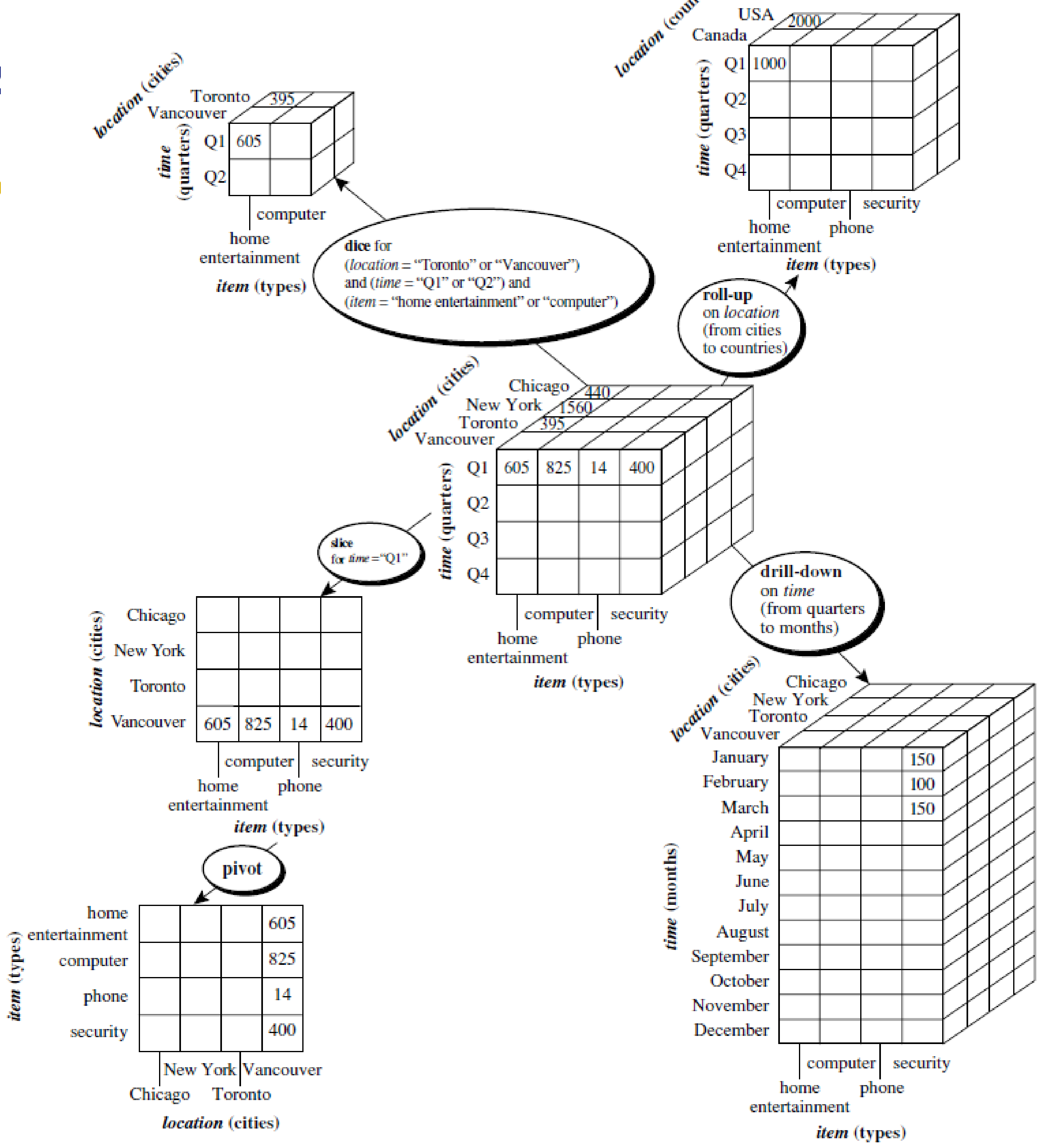
- Important Applications of Data Warehouse
- Storing Data in Data Warehouse
- Fact Tables and Dimension Tables
- Schema of a Data Warehouse
  - ➢ Star, Snowflakes, Fact Constellations
- OLAP Operations
  - ➢ Roll up, Drill down, Slice & Dice, Pivot

Online analytical processing (**OLAP**) is a technique of analysing data to look for insights.

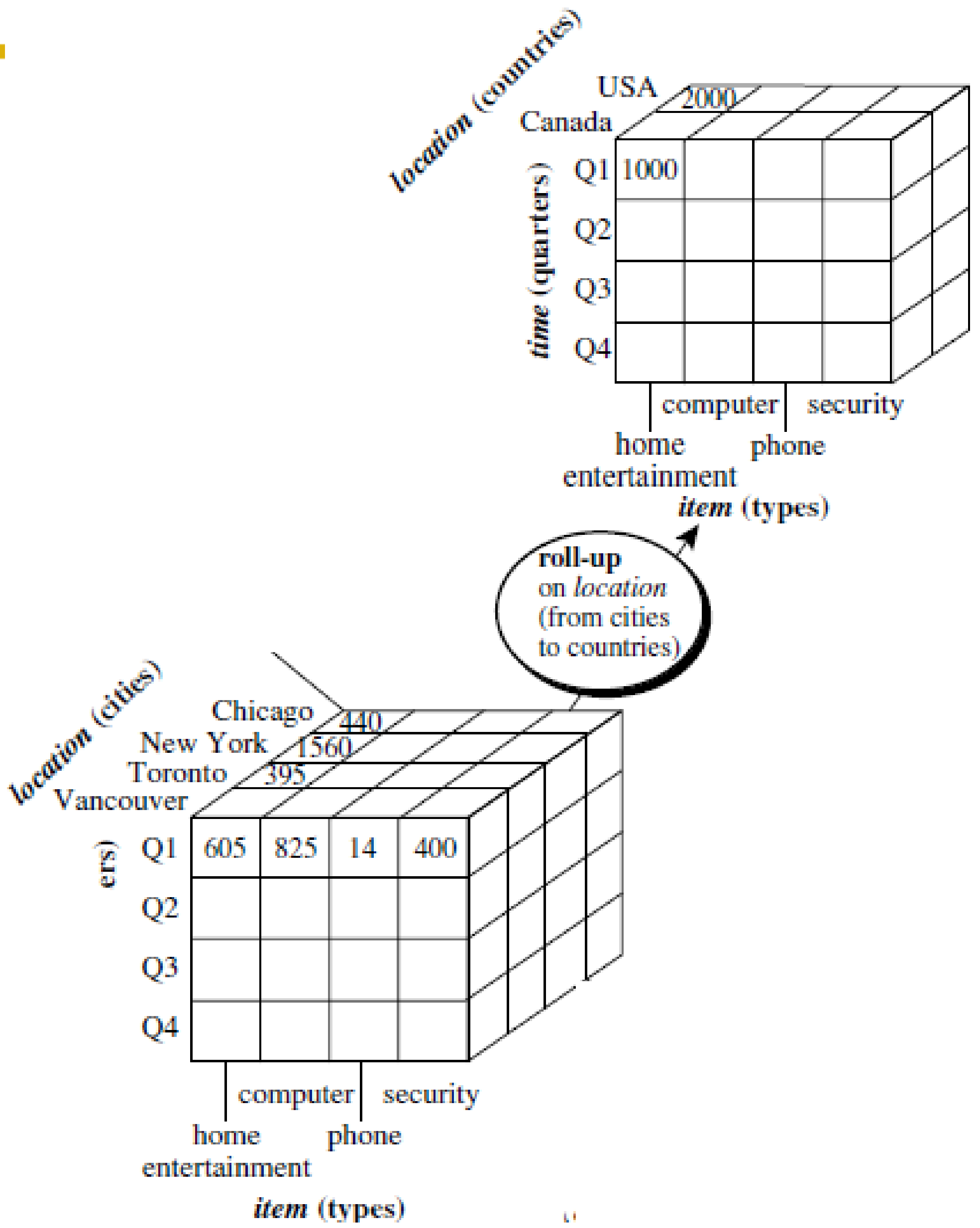
Typical OLAP Operations

- **Roll up (drill up): summarise 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, visualisation, 3D to series of 2D planes.*
- **Other operations (aside)**
  - *drill across: involving (across) multiple fact tables*
  - *drill through: through the bottom level of the cube to its back-end relational tables (using SQL)*

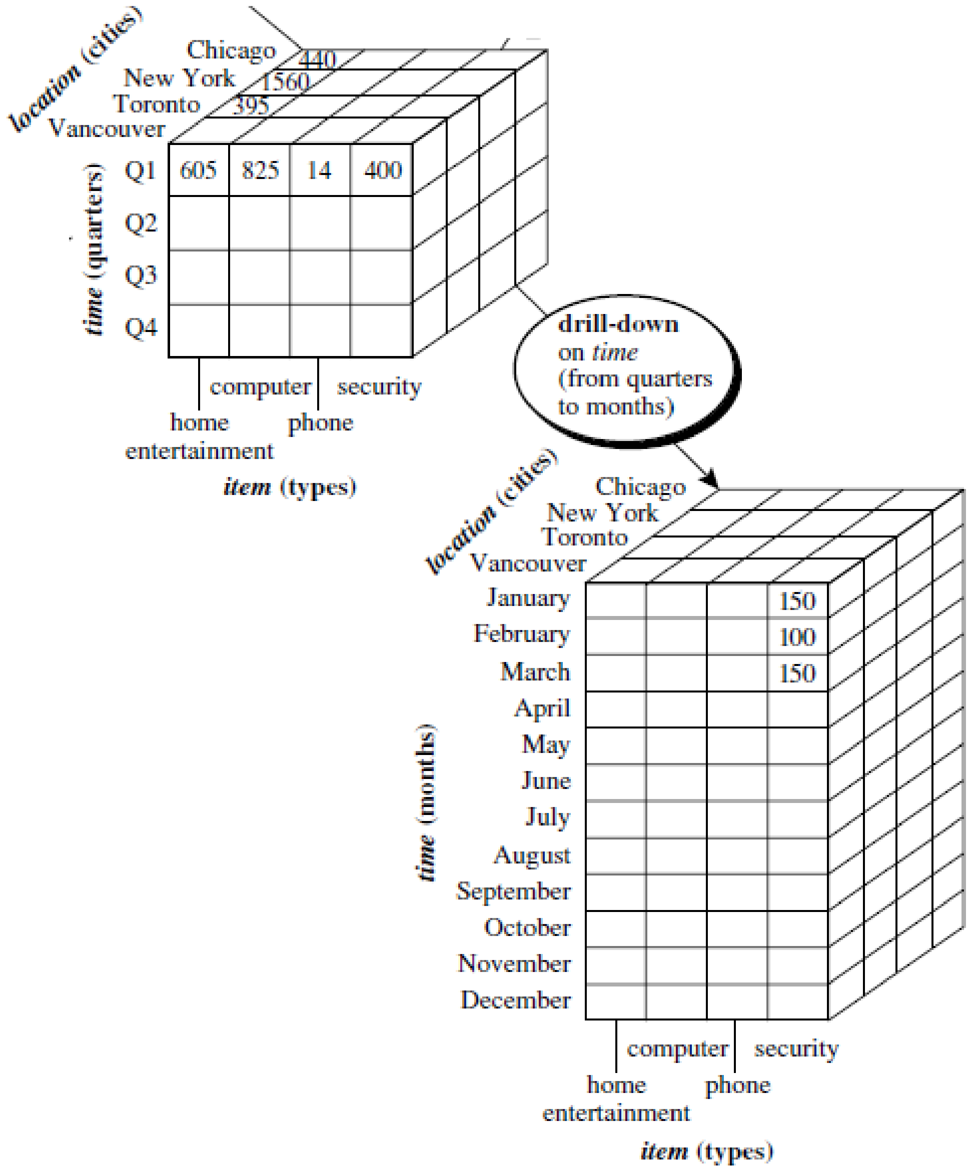
Example of OL Operations



Example of OLAP Operations (roll-up)



Example of OLAP Operations (drill-down)



Roll Up and Drill Down

Autos Sold

	VIC	NSW	WA	Total
Jul	45	33	30	108
Aug	50	36	42	128
Sep	38	31	40	109

$\Rightarrow$ Roll up by Month

Autos Sold

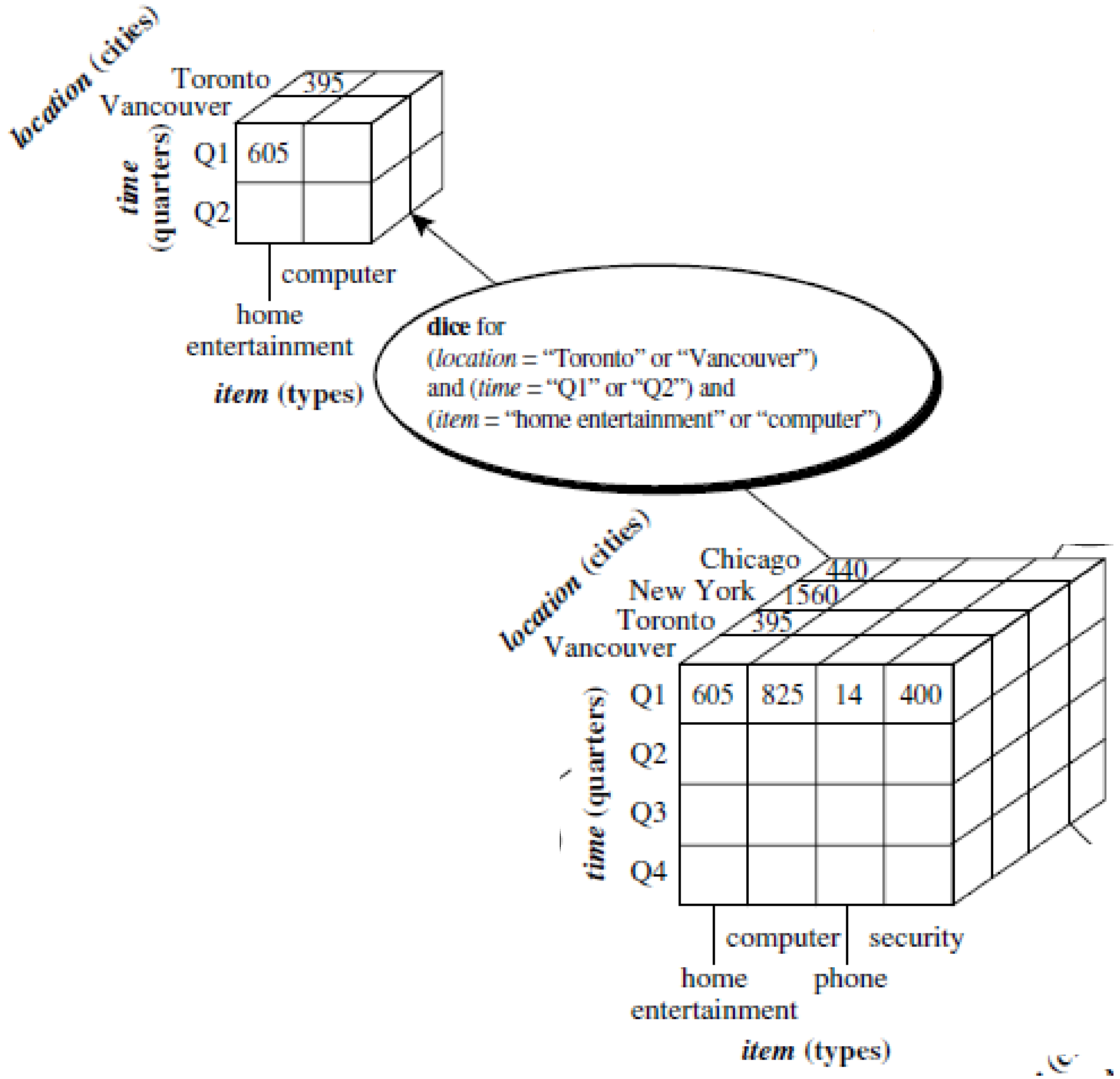
VIC	NSW	WA	Total
133	100	112	345

$\Downarrow$ Drill down by Color

Autos Sold

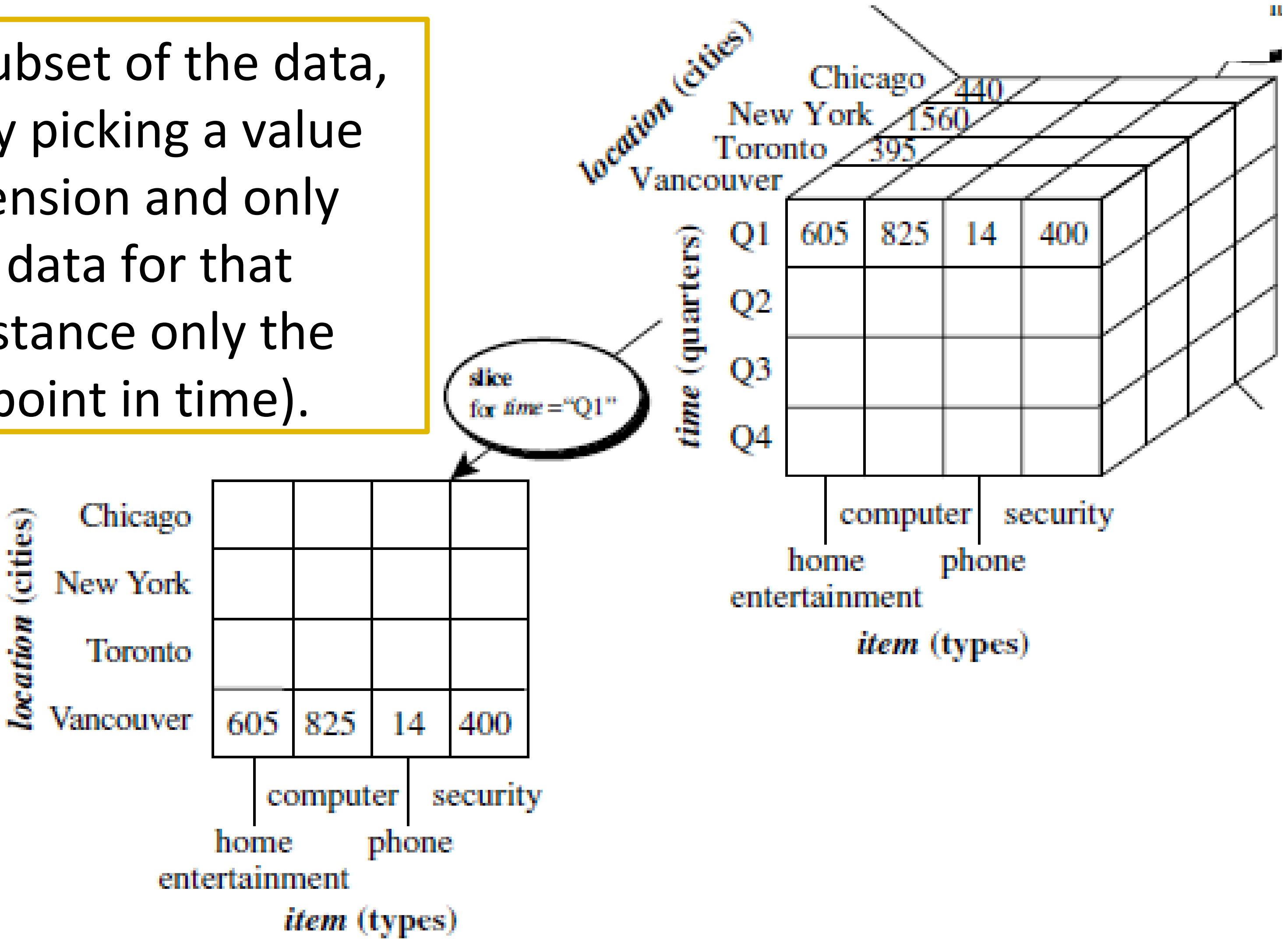
	VIC	NSW	WA	Total
Red	40	29	40	109
Blue	45	31	37	113
Gray	48	40	35	123

Example of OLAP Operations (dice)

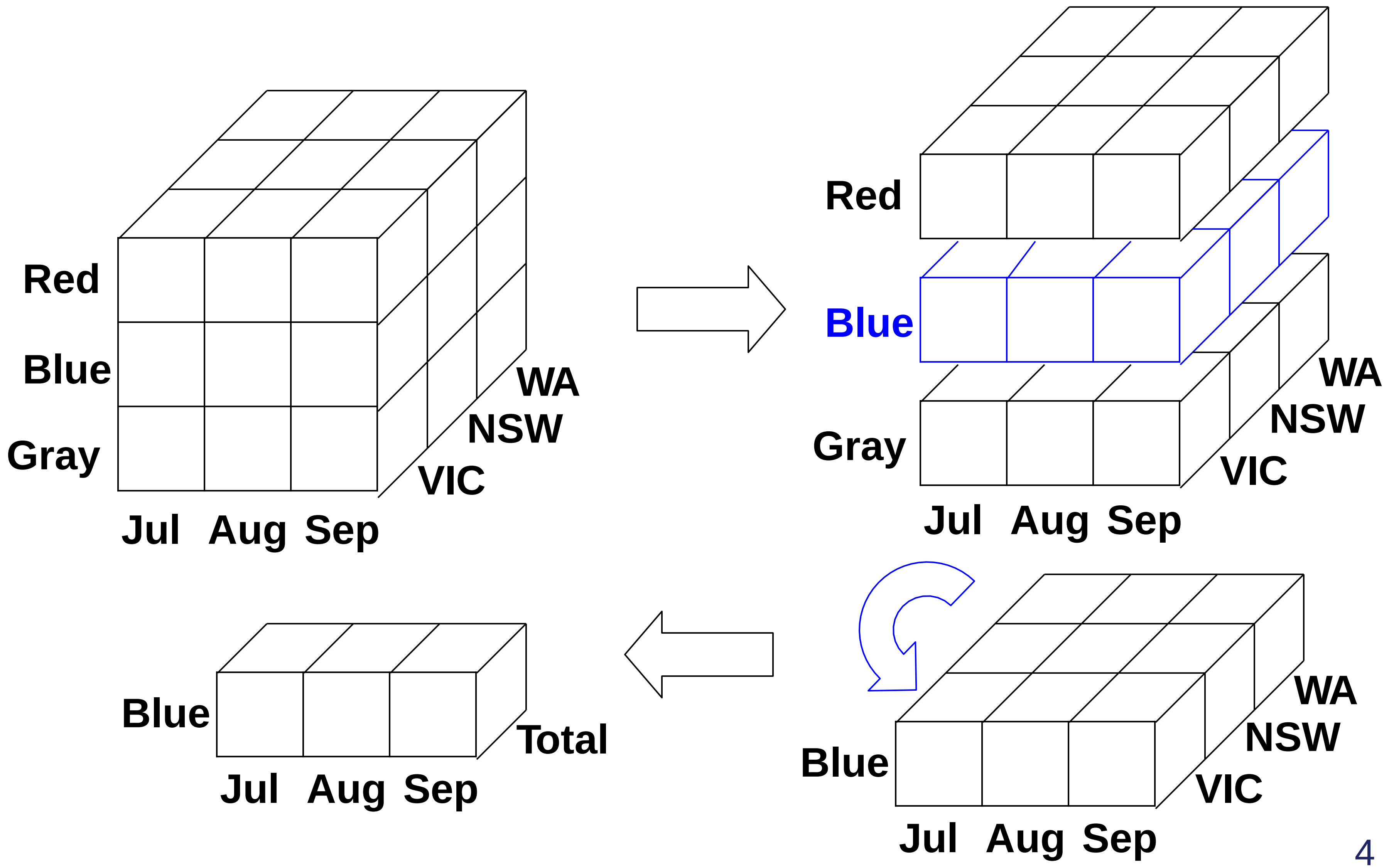


Example of OLAP Operations (slice)

A Slice is a subset of the data, generated by picking a value for one dimension and only showing the data for that value (for instance only the data at one point in time).



Slicing


Red
Blue
Gray
Jul
Aug
Sep
WA
NSW
VIC
Red
Blue
Gray
Jul
Aug
Sep
WA
NSW
VIC
Blue
Jul
Aug
Sep
WA
NSW
VIC
Blue
Jul
Aug
Sep
Total

References

- Some slides are adapted from
  - http://web.stanford.edu/class/cs345/
  - https://hanj.cs.illinois.edu/bk3/bk3_slidesindex.htm

- Readings
  - Chapter 4.2 of Han et al.'s book
  - Chapter 5 of Rainardi's book
  - Drill down v.s. drill through
  - An example of drill across (next page)

Class Representative

Education Council President of the Student Guild calls for student representative for this class.

Link to Class Rep EOI form:
https://forms.office.com/Pages/ResponsePage.aspx?id=xpKj6_peiE-83BzOEVsl73f62np-t2hBraYe4hqm3rJUNVZXMjhZOU5ZNURHRDNFMFlYVjIyMFNOUi4u

Additional Slides

DMQL Examples

- Not examinable content
- Help you better understand Data Warehouse, fact tables and dimension tables

Drill-Across Example (optional)

Question: How did actual sales diverge from forecasted sales in Sep 19?
Drill-across between “Forecast” and “Sales”

- Step 1: Query Forecast fact
  - Group by Brand Name, District Name
  - Filter on MonthAndYear =‘Sep 19’
  - Calculate SUM(ForecastAmt)
  - Query result has schema (Brand Name, District Name, ForecastAmt)
- Step 2: Query Sales fact
  - Group by Brand Name, District Name
  - Filter on MonthAndYear =‘Sept 19’
  - Calculate SUM(TotalSalesAmt)
  - Query result has schema (Brand Name, District Name, TotalSalesAmt)
- Step 3: Combine query results
  - Join Result 1 and Result 2 on Brand Name and District Name
  - Result has schema (Brand Name, District Name, ForecastAmt, TotalSalesAmt)

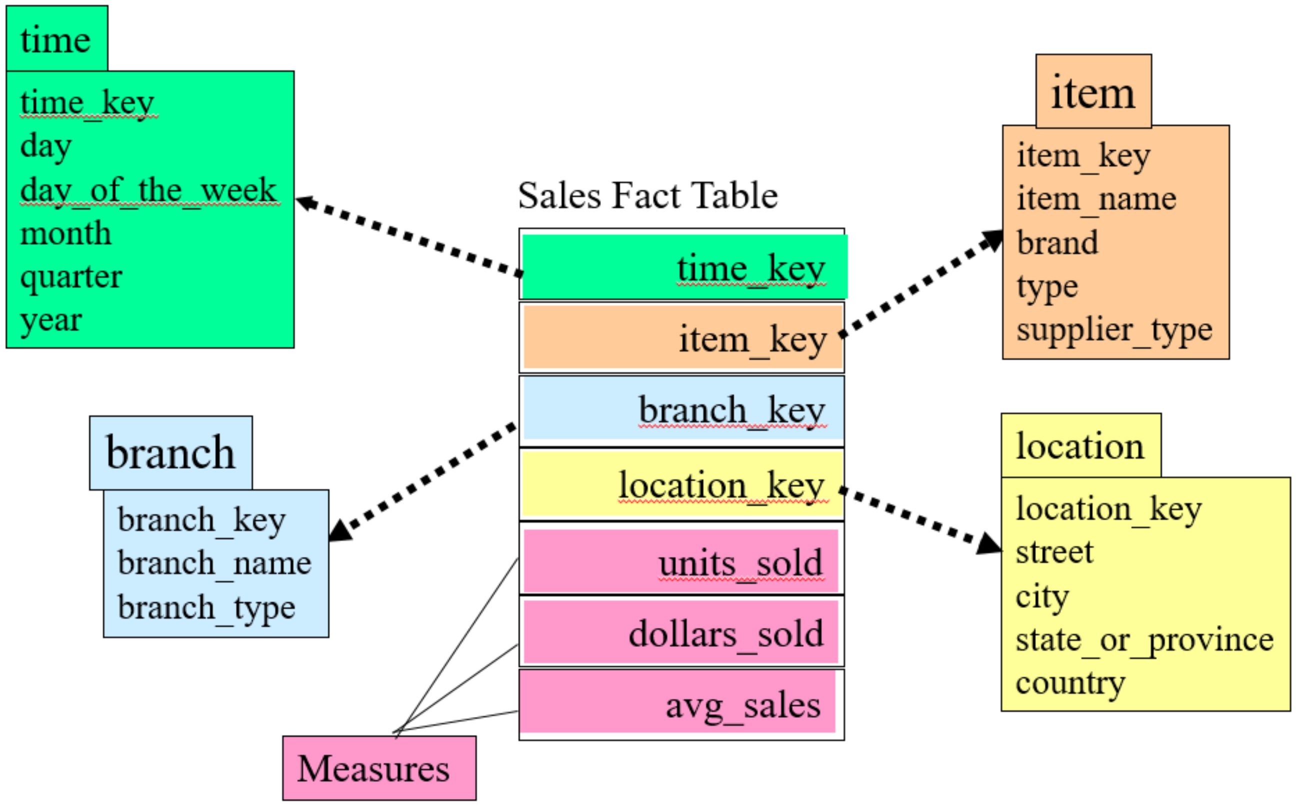
Cube Definition Syntax (BNF) in DMQL

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

Defining Star Schema in DMQL

define cube sales_star [time, item, branch, location]:

dollars_sold = sum(sales_in_dollars), avg_sales = avg(sales_in_dollars), units_sold = count(*)



Defining Star Schema in DMQL

define cube **sales_star [time, item, branch, location]:**

dollars_sold = sum(sales_in_dollars), avg_sales = avg(sales_in_dollars), units_sold = count(*)

define dimension **time** **as** **(time_key, day, day_of_week, month, quarter, year)**

define dimension **item** **as** **(item_key, item_name, brand, type, supplier_type)**

define dimension **branch** **as** **(branch_key, branch_name, branch_type)**

define dimension **location** **as** **(location_key, street, city, province_or_state, country)**

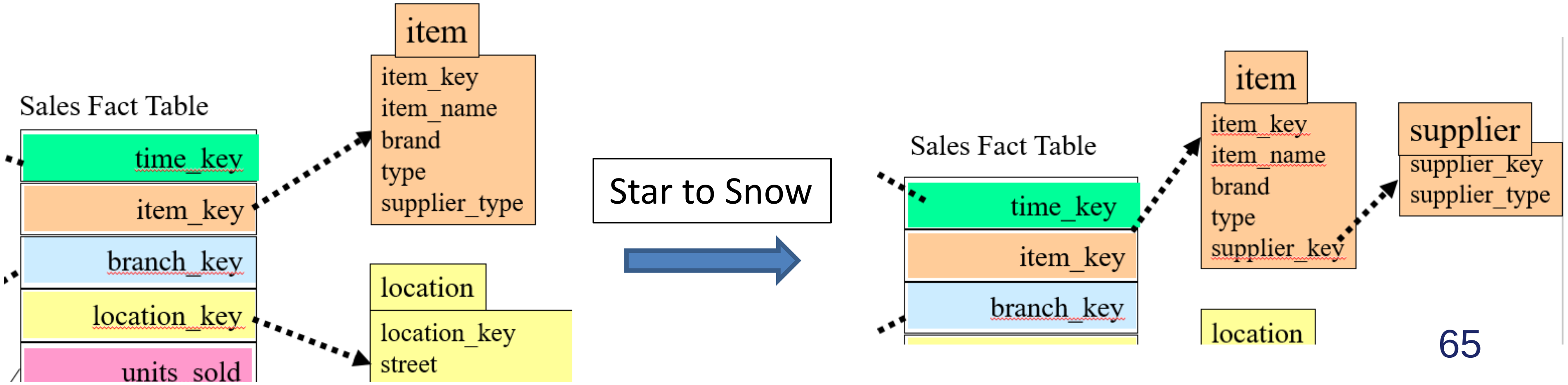
Defining Snowflake Schema in DMQL

define cube sales_snowflake [time, item, branch, location]:

dollars_sold = sum(sales_in_dollars), avg_sales = avg(sales_in_dollars), units_sold = count(*)

define dimension time as (time_key, day, day_of_week, month, quarter, year)

define dimension item as (item_key, item_name, brand, type, supplier(supplier_key, supplier_type))



Defining Snowflake Schema in DMQL

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

Defining Fact Constellation in DMQL

define cube sales [time, item, branch, location]:

dollars_sold = sum(sales_in_dollars), avg_sales = avg(sales_in_dollars), units_sold = count(*)

define dimension time as (time_key, day, day_of_week, month, quarter, year)

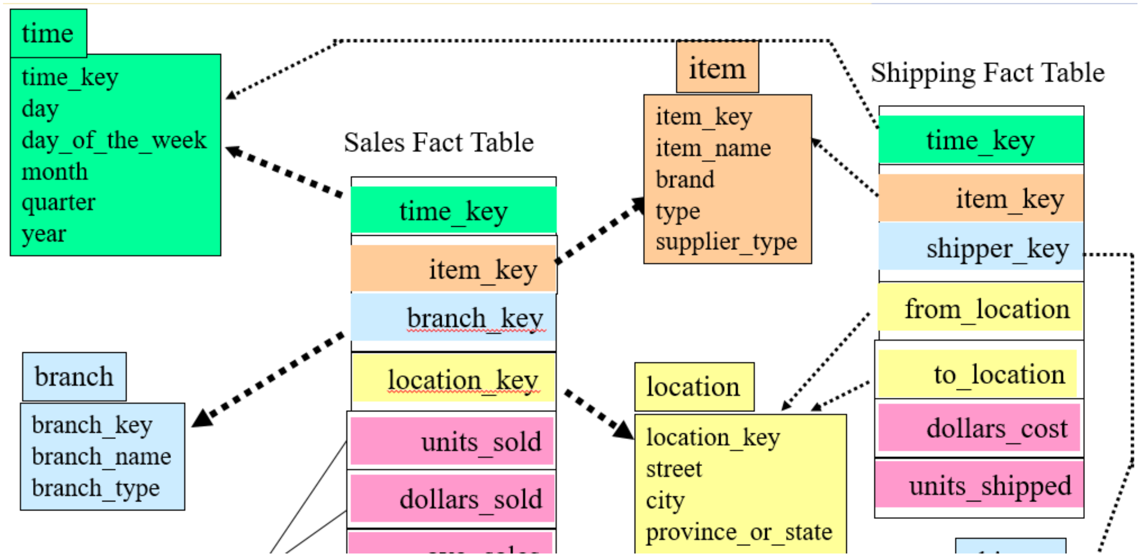
define dimension item as (item_key, item_name, brand, type, supplier_type)

define dimension branch as (branch_key, branch_name, branch_type)

define dimension location as (location_key, street, city, province_or_state, country)

define cube shipping [time, item, shipper, from_location, to_location]:

dollar_cost = sum(cost_in_dollars), unit_shipped = count(*)



Defining Fact Constellation in DMQL

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