Data Warehouse and Data Mining

Dhruv Gupta

16-January-2023

NTNU | Norwegian University of Science and Technology

1. QA and References
   - Answers to Previous Questions
   - References for Today's Lecture

2. Origins of Data Warehouses
   - Data Analysis and Visualization
   - Data Extraction using SQL
   - Problems with SQL for Data Analysis
   - The Data Cube

3. Data Cubes
   - Multi-dimensional Data Model
   - Dimensions and Concept Hierarchies
   - Schema Design
   - Operators

4. Summary

Answers

1. Lecture plan errors.
   - Will be corrected by end of this week.
2. TDT4145 pre-requisite?
   - It is not a formal requirement. However, for basics regarding databases and SQL we expect the student, that has not had the course, to spend time on their own to fill in the gaps.
3. Tutorial sessions?
   - Currently, we are planning on office hours with the tutors for helping out with the assignments.
4. Programming language used in assignments?
   - Past assignments have used Python. We expect this to be the case in this year's course also.
   - Alternatively, we may plan to have to code (irrespective of language) and results submission.

Amazon Go — Computer Vision in Data Mining

Image source: https://techcrunch.com/2018/01/21/inside-amazons-surveillance-powered-no-checkout-convenience-store/

Amazon Go — Computer Vision in Data Mining

Image source: https://techcrunch.com/2018/01/21/inside-amazons-surveillance-powered-no-checkout-convenience-store/

Amazon Go — Computer Vision in Data Mining

Image source: https://techcrunch.com/2018/01/21/inside-amazons-surveillance-powered-no-checkout-convenience-store/

Amazon Go — Computer Vision in Data Mining

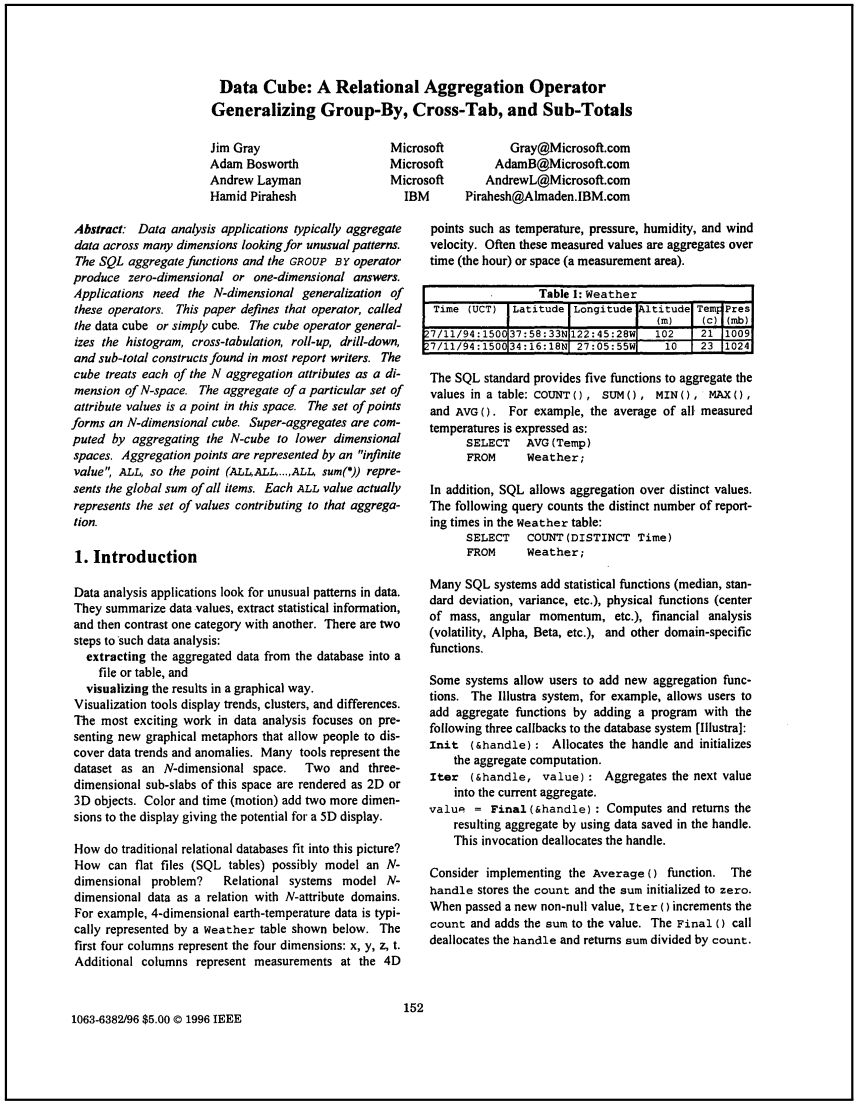
Image source: https://techcrunch.com/2018/01/21/inside-amazons-surveillance-powered-no-checkout-convenience-store/

Amazon Go — Computer Vision in Data Mining

Image source: https://techcrunch.com/2018/01/21/inside-amazons-surveillance-powered-no-checkout-convenience-store/

References for "Origins of Data Warehouses"

1. Conference Article:
   Gray et al. *"Data Cube: A Relational Aggregation Operator Generalizing Group-By, Cross-Tab, and Sub-Totals"*. ICDE'96.
2. Extended Article:
   `https://arxiv.org/abs/cs/0701155`
3. All text and images for "Origins of Data Warehouse" are based on the arXiv preprint.



Data Cube: A Relational Aggregation Operator Generalizing Group-By, Cross-Tab, and Sub-Totals

Jim Gray — Microsoft — Gray@Microsoft.com
Adam Bosworth — Microsoft — AdamB@Microsoft.com
Andrew Layman — Microsoft — AndrewL@Microsoft.com
Hamid Pirahesh — IBM — Pirahesh@Almaden.IBM.com

Abstract: *Data analysis applications typically aggregate data across many dimensions looking for unusual patterns. The SQL aggregate functions and the* `GROUP BY` *operator produce zero-dimensional or one-dimensional answers. Applications need the N-dimensional generalization of these operators. This paper defines that operator, called the* data cube *or simply* cube. *The cube operator generalizes the histogram, cross-tabulation, roll-up, drill-down, and sub-total constructs found in most report writers. The cube treats each of the N aggregation attributes as a dimension of N-space. The aggregate of a particular set of attribute values is a point in this space. The set of points forms an N-dimensional cube. Super-aggregates are computed by aggregating the N-cube to lower dimensional spaces. Aggregation points are represented by an "infinite value",* `ALL`*, so the point (ALL,ALL,...,ALL, sum(*)) represents the global sum of all items. Each* `ALL` *value actually represents the set of values contributing to that aggregation.*

1. Introduction

Data analysis applications look for unusual patterns in data. They summarize data values, extract statistical information, and then contrast one category with another. There are two steps to such data analysis:

- **extracting** the aggregated data from the database into a file or table, and
- **visualizing** the results in a graphical way.

Visualization tools display trends, clusters, and differences. The most exciting work in data analysis focuses on presenting new graphical metaphors that allow people to discover data trends and anomalies. Many tools represent the dataset as an *N*-dimensional space. Two and three-dimensional sub-slabs of this space are rendered as 2D or 3D objects. Color and time (motion) add two more dimensions to the display giving the potential for a 5D display.

How do traditional relational databases fit into this picture? How can flat files (SQL tables) possibly model an *N*-dimensional problem? Relational systems model *N*-dimensional data as a relation with *N*-attribute domains. For example, 4-dimensional earth-temperature data is typically represented by a `Weather` table shown below. The first four columns represent the four dimensions: x, y, z, t. Additional columns represent measurements at the 4D points such as temperature, pressure, humidity, and wind velocity. Often these measured values are aggregates over time (the hour) or space (a measurement area).

Table 1: `Weather`

Time (UCT)	Latitude	Longitude	Altitude (m)	Temp (c)	Pres (mb)
27/11/94:1500	37:58:33N	122:45:28W	102	21	1009
27/11/94:1500	34:16:18N	27:05:55W	10	23	1024

The SQL standard provides five functions to aggregate the values in a table: `COUNT()`, `SUM()`, `MIN()`, `MAX()`, and `AVG()`. For example, the average of all measured temperatures is expressed as:

```
SELECT   AVG(Temp)
FROM     Weather;
```

In addition, SQL allows aggregation over distinct values. The following query counts the distinct number of reporting times in the `Weather` table:

```
SELECT   COUNT(DISTINCT Time)
FROM     Weather;
```

Many SQL systems add statistical functions (median, standard deviation, variance, etc.), physical functions (center of mass, angular momentum, etc.), financial analysis (volatility, Alpha, Beta, etc.), and other domain-specific functions.

Some systems allow users to add new aggregation functions. The Illustra system, for example, allows users to add aggregate functions by adding a program with the following three callbacks to the database system [Illustra]:

`Init (&handle)`: Allocates the handle and initializes the aggregate computation.

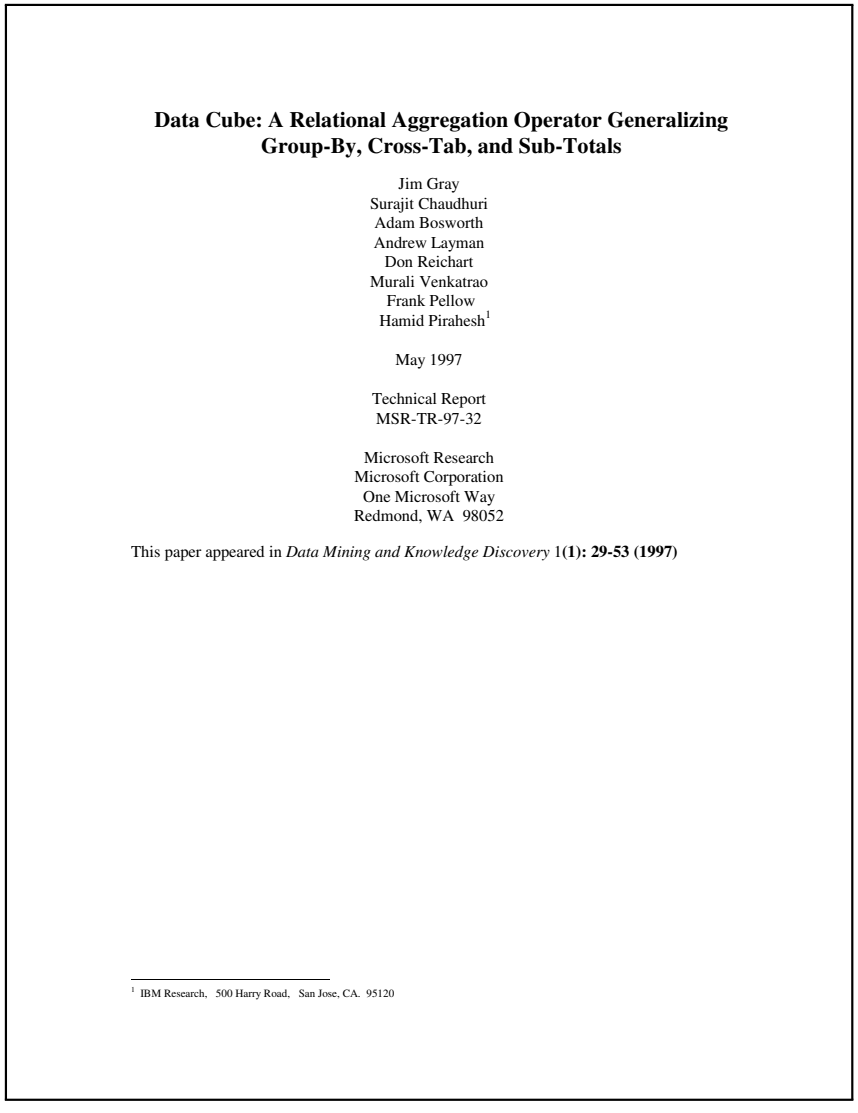
`Iter (&handle, value)`: Aggregates the next value into the current aggregate.

`value = Final(&handle)`: Computes and returns the resulting aggregate by using data saved in the handle. This invocation deallocates the handle.

Consider implementing the `Average()` function. The `handle` stores the `count` and the `sum` initialized to `zero`. When passed a new non-null value, `Iter()` increments the `count` and adds the `sum` to the value. The `Final()` call deallocates the `handle` and returns `sum` divided by `count`.

References for "Origins of Data Warehouses"

1. Conference Article:
   Gray et al. *"Data Cube: A Relational Aggregation Operator Generalizing Group-By, Cross-Tab, and Sub-Totals"*. ICDE'96.
2. Extended Article:
   `https://arxiv.org/abs/cs/0701155`
3. All text and images for "Origins of Data Warehouse" are based on the arXiv preprint.



Data Cube: A Relational Aggregation Operator Generalizing Group-By, Cross-Tab, and Sub-Totals

Jim Gray
Surajit Chaudhuri
Adam Bosworth
Andrew Layman
Don Reichart
Murali Venkatrao
Frank Pellow
Hamid Pirahesh[1]

May 1997

Technical Report
MSR-TR-97-32

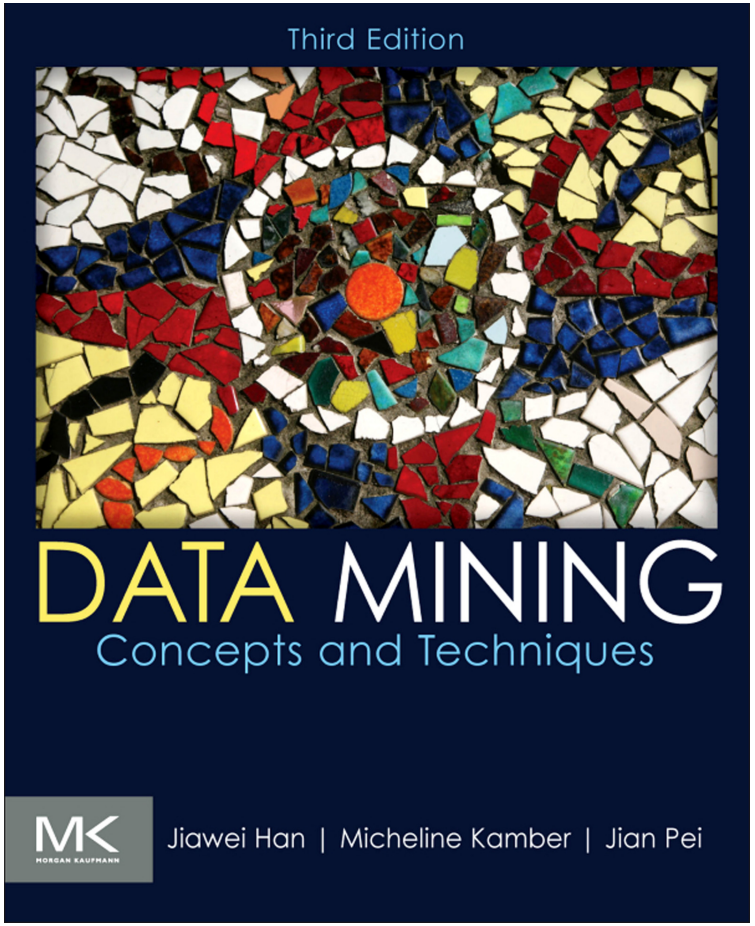
Microsoft Research
Microsoft Corporation
One Microsoft Way
Redmond, WA 98052

This paper appeared in *Data Mining and Knowledge Discovery* 1**(1): 29-53 (1997)**

[1] IBM Research, 500 Harry Road, San Jose, CA. 95120

References for "Data Cubes"

1. Book: Han et al. *"Data Mining Concepts and Techniques"*, 3rd Edition, 2012, Morgan Kaufmann Publishers.
2. All text and images for "Data Cubes" are based on the book by Han et al.



1. QA and References
   - Answers to Previous Questions
   - References for Today's Lecture

2. Origins of Data Warehouses
   - Data Analysis and Visualization
   - Data Extraction using SQL
   - Problems with SQL for Data Analysis
   - The Data Cube

3. Data Cubes
   - Multi-dimensional Data Model
   - Dimensions and Concept Hierarchies
   - Schema Design
   - Operators

4. Summary

Data Analysis

- Data analysis applications look for unusual patterns in data.
- Data analysis applications typically aggregate data across many dimensions looking for unusual patterns.
- They categorize data values and trends, extract statistical information, and then contrast one category with another.

All text from: Gray et al. "Data Cube: A Relational Aggregation Operator Generalizing Group-By, Cross-Tab, and Sub-Totals". ICDE'96.

Data Analysis

- There are four steps to such data analysis:
  - Formulating a query that extracts relevant data from a large database.
  - Extracting the aggregated data from the database into a file or table.
  - Visualizing the results in a graphical way.
  - Analyzing the results and formulating a new query.
- Data analysis tools often try to identify a subspace of the N-dimensional space which is "interesting" (e.g., discriminating attributes of the data set).

All text from: Gray et al. "Data Cube: A Relational Aggregation Operator Generalizing Group-By, Cross-Tab, and Sub-Totals". ICDE'96.

Visualization

- Visualization tools display data trends, clusters, and differences.
- Visualization focuses on presenting new graphical metaphors that allow people to discover data trends and anomalies.
- These visualization and data analysis tools represent the dataset as an N-dimensional space.
- Visualization tools render two and three-dimensional sub-slabs of this space as 2D or 3D objects.
- Color and time (motion) add two more dimensions to the display giving the potential for a 5D display.

All text from: Gray et al. "Data Cube: A Relational Aggregation Operator Generalizing Group-By, Cross-Tab, and Sub-Totals". ICDE'96.

Data Analysis and Visualization

- Thus, visualization as well as data analysis tools do "dimensionality reduction" , often by summarizing data along the dimensions that are left out.

Example

For example, in trying to analyze car sales, we might focus on the role of model, year and color of the cars in sale. Thus, we ignore the differences between two sales along the dimensions of date of sale or dealership but analyze the totals sale for cars by model, by year and by color only.

- Along with summarization and dimensionality reduction, data analysis applications use constructs such as histogram, cross-tabulation, subtotals, roll-up and drill-down extensively.

All text from: Gray et al. "Data Cube: A Relational Aggregation Operator Generalizing Group-By, Cross-Tab, and Sub-Totals". ICDE'96.

Data Analysis and Visualization

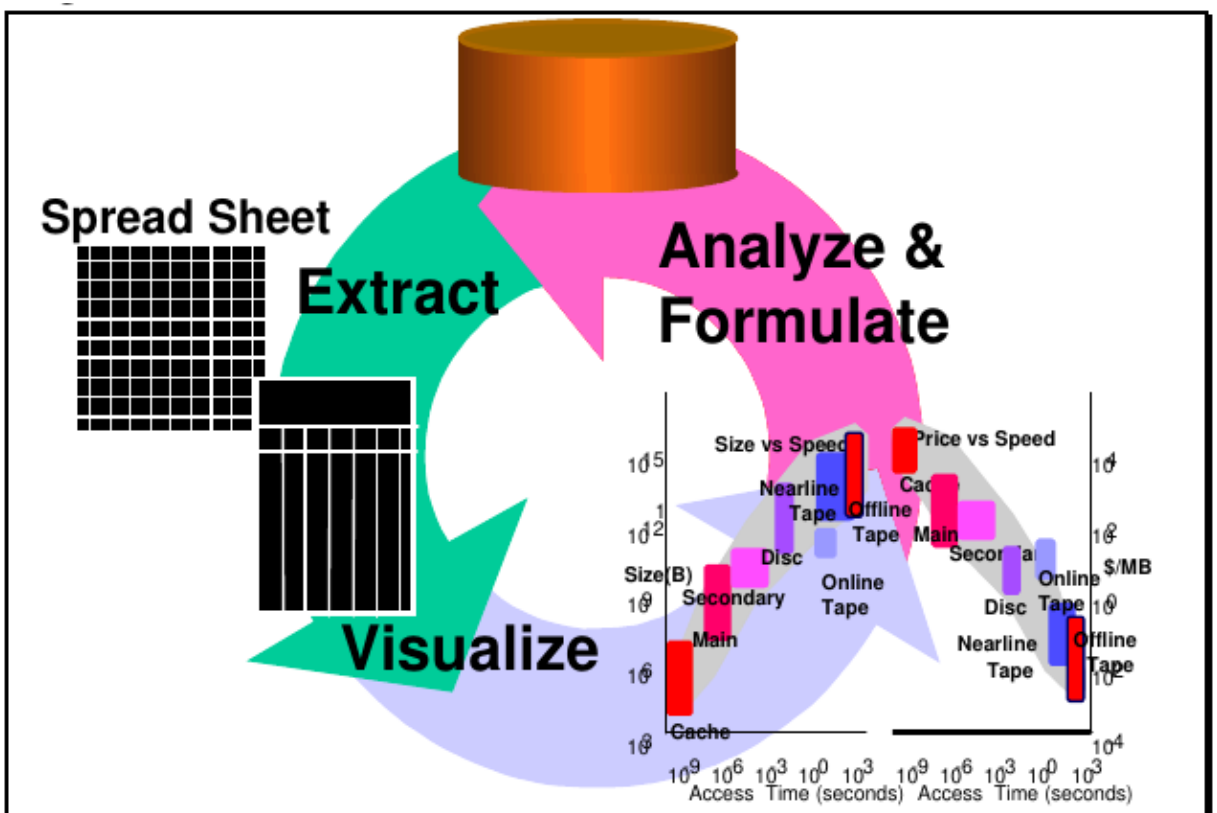


Figure 1: Data analysis tools facilitate the Extract-Visualize-Analyze loop. The cube and roll-up operators along with system and user-defined aggregates are part of the extraction process.

Image from: Gray et al. "Data Cube: A Relational Aggregation Operator Generalizing Group-By, Cross-Tab, and Sub-Totals". ICDE'96.

Data Extraction using SQL

1. How do traditional relational databases fit into this multi-dimensional data analysis picture?
2. How can 2D flat files (SQL tables) model an N-dimensional problem?
3. Further more, how do the relational systems support the ability to support operations over N-dimensional representation that are central to visualization and data analysis programs?

All text from: Gray et al. "Data Cube: A Relational Aggregation Operator Generalizing Group-By, Cross-Tab, and Sub-Totals". ICDE'96.

Data Extraction using SQL

1. The answer to the first question is that relational systems model N-dimensional data as a relation with N-attribute domains.
2. Visualization and data analysis tools extensively use dimensionality reduction (aggregation) for better comprehensibility.
3. Often data along the other dimensions that are not included in a "2-D" representation are summarized via aggregation in the form of histogram, cross-tabulation, subtotals etc.
4. In SQL Standard, we depend on aggregate functions and the GROUP BY operator to support aggregation.
5. The SQL standard, provides five functions to aggregate the values in a table: COUNT(), SUM(), MIN(), MAX(), and AVG().

All text from: Gray et al. "Data Cube: A Relational Aggregation Operator Generalizing Group-By, Cross-Tab, and Sub-Totals". ICDE'96.

Data Extraction using SQL

Table 1: `Weather`

Time (UCT)	Latitude	Longitude	Altitude (m)	Temp (c)	Pres (mb)
96/6/1:1500	37:58:33N	122:45:28W	102	21	1009
	many more rows like the ones above and below				
96/6/7:1500	34:16:18N	27:05:55W	10	23	1024

Image from: Gray et al. "Data Cube: A Relational Aggregation Operator Generalizing Group-By, Cross-Tab, and Sub-Totals". ICDE'96.

Data Extraction using SQL

Table 1: `Weather`

Time (UCT)	Latitude	Longitude	Altitude (m)	Temp (c)	Pres (mb)
96/6/1:1500	37:58:33N	122:45:28W	102	21	1009
		many more rows like the ones above and below			
96/6/7:1500	34:16:18N	27:05:55W	10	23	1024

Example

Find average of all measured temperatures.

```
SELECT  AVG(Temp)
FROM    Weather;
```

All text from: Gray et al. "Data Cube: A Relational Aggregation Operator Generalizing Group-By, Cross-Tab, and Sub-Totals". ICDE'96.

Data Extraction using SQL

Table 1: `Weather`

Time (UCT)	Latitude	Longitude	Altitude (m)	Temp (c)	Pres (mb)
96/6/1:1500	37:58:33N	122:45:28W	102	21	1009
		many more rows like the ones above and below			
96/6/7:1500	34:16:18N	27:05:55W	10	23	1024

Example

Count distinct number of reporting times in the Weather table.

```
SELECT  COUNT(DISTINCT Time)
FROM    Weather;
```

All text from: Gray et al. "Data Cube: A Relational Aggregation Operator Generalizing Group-By, Cross-Tab, and Sub-Totals". ICDE'96.

Data Extraction using SQL

Table 1: `Weather`

Time (UCT)	Latitude	Longitude	Altitude (m)	Temp (c)	Pres (mb)
96/6/1:1500	37:58:33N	122:45:28W	102	21	1009
		many more rows like the ones above and below			
96/6/7:1500	34:16:18N	27:05:55W	10	23	1024

- Aggregate functions return a single value. Using GROUP BY construct, SQL can also create a table of many aggregate values indexed by a set of attributes.

Example

Find average temperature for each reporting time and altitude.

```
SELECT  Time, Altitude, AVG(Temp)
FROM    Weather
GROUP BY Time, Altitude;
```

All text from: Gray et al. "Data Cube: A Relational Aggregation Operator Generalizing Group-By, Cross-Tab, and Sub-Totals". ICDE'96.

Data Extraction using SQL

- GROUP BY is an unusual relational operator: It partitions the relation into disjoint tuple sets and then aggregates over each set.

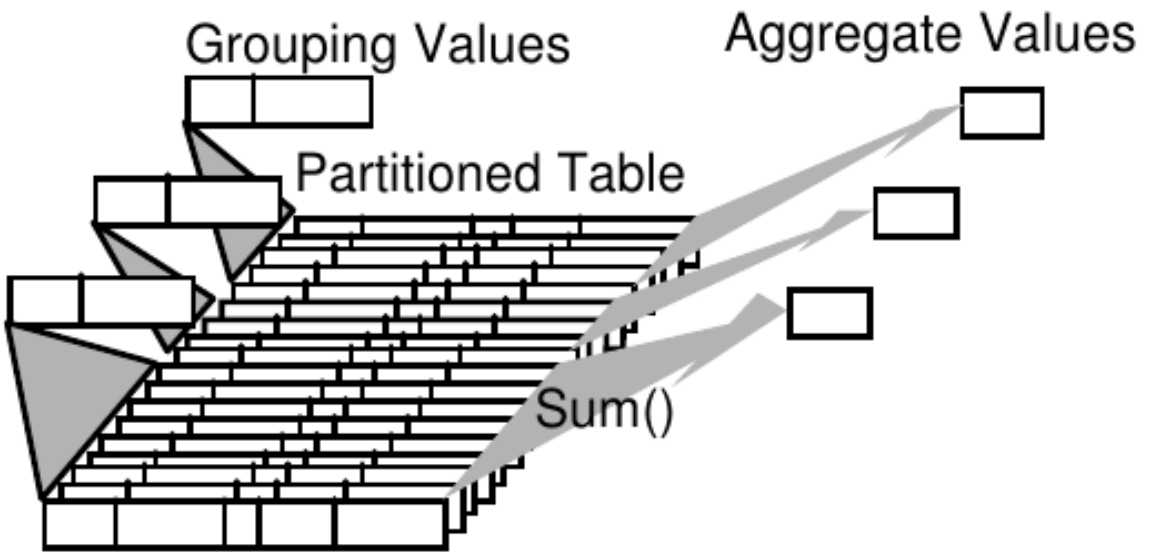


Figure 2: The `GROUP BY` relational operator partitions a table into groups. Each group is then aggregated by a function. The aggregation function summarizes some column of groups returning a value for each group.

All text from: Gray et al. "Data Cube: A Relational Aggregation Operator Generalizing Group-By, Cross-Tab, and Sub-Totals". ICDE'96.

Problems with GROUP BY

- Certain common forms of data analysis are difficult with these SQL aggregation constructs.
- Four common problems are:
  1. Histograms,
  2. Roll-up,
  3. Totals and Sub-Totals for drill-downs,
  4. Cross Tabulations.

All text from: Gray et al. "Data Cube: A Relational Aggregation Operator Generalizing Group-By, Cross-Tab, and Sub-Totals". ICDE'96.

Problems with GROUP BY

- The standard SQL GROUP BY operator does not allow a direct construction of histograms (aggregation over computed categories).
- For example, for queries based on the Weather table, it would be nice to be able to group times into days, weeks, or months, and to group locations into areas (e.g., US, Canada, Europe,...).

```
SELECT day, nation, MAX(Temp)
FROM Weather
GROUP BY Day(Time) AS day,
                Nation(Latitude , Longitude) AS nation;
```

All text from: Gray et al. "Data Cube: A Relational Aggregation Operator Generalizing Group-By, Cross-Tab, and Sub-Totals". ICDE'96.

Problems with GROUP BY

- In standard SQL, histograms are computed indirectly from a table-valued expression which is then aggregated.
- The following statement demonstrates this SQL92 construct using nested queries.

```
SELECT day, nation, MAX(Temp)
FROM (
        SELECT Day(Time) AS day,
             Nation(Latitude, Longitude) AS nation,
             Temp
        FROM Weather
      )AS foo
 GROUP BY day, nation;
```

All text from: Gray et al. "Data Cube: A Relational Aggregation Operator Generalizing Group-By, Cross-Tab, and Sub-Totals". ICDE'96.

Problems with GROUP BY

- A more serious problem, relates to roll-ups using totals and sub-totals for drill-down reports.
- Reports commonly aggregate data at a coarse level, and then at successively finer levels.

All text from: Gray et al. "Data Cube: A Relational Aggregation Operator Generalizing Group-By, Cross-Tab, and Sub-Totals". ICDE'96.

Problems with GROUP BY

SALES			
Model	Year	Color	Sales
Chevy	1990	red	5
Chevy	1990	white	87
Chevy	1990	blue	62
Chevy	1991	red	54
Chevy	1991	white	95
Chevy	1991	blue	49
Chevy	1992	red	31
Chevy	1992	white	54
Chevy	1992	blue	71
Ford	1990	red	64
Ford	1990	white	62
Ford	1990	blue	63
Ford	1991	red	52
Ford	1991	white	9

All text from: Gray et al. "Data Cube: A Relational Aggregation Operator Generalizing Group-By, Cross-Tab, and Sub-Totals". ICDE'96.

Problems with GROUP BY

- The report shows data aggregated at three levels.
- Going up the levels is called rolling-up the data.
- Going down is called drilling-down into the data.
- Data aggregated at each distinct level produces a sub-total.

Table 3.a: Sales Roll Up by Model by Year by Color

Model	Year	Color	Sales by Model by Year by Color	Sales by Model by Year	Sales by Model
Chevy	1994	black	50		
		white	40		
				90	
	1995	black	85		
		white	115		
				200	
					290

All from: Gray et al. "Data Cube: A Relational Aggregation Operator Generalizing Group-By, Cross-Tab, and Sub-Totals". ICDE'96.

Problems with GROUP BY

- Table 3.a suggests creating 2^N aggregation columns for a roll-up of N elements.
- The representation of Table 3.a is not relational because the empty cells (presumably NULL values), cannot form a key.

Table 3.a: Sales Roll Up by Model by Year by Color

Model	Year	Color	Sales by Model by Year by Color	Sales by Model by Year	Sales by Model
Chevy	1994	black	50		
		white	40		
				90	
	1995	black	85		
		white	115		
				200	
					290

All from: Gray et al. "Data Cube: A Relational Aggregation Operator Generalizing Group-By, Cross-Tab, and Sub-Totals". ICDE'96.

Problems with GROUP BY

- Indeed, Chris Date (a database researcher) recommends this approach.
- The approach recommended by Date is reminiscent of pivot tables found in Excel.

Table 3.b: Sales Roll-Up by Model by Year by Color as recommended by Chris Date [Date1].

Model	Year	Color	Sales	Sales by Model by Year	Sales by Model
Chevy	1994	black	50	90	290
Chevy	1994	white	40	90	290
Chevy	1995	black	85	200	290
Chevy	1995	white	115	200	290

All text from: Gray et al. "Data Cube: A Relational Aggregation Operator Generalizing Group-By, Cross-Tab, and Sub-Totals". ICDE'96.

Problems with GROUP BY

- Indeed, Chris Date (a database researcher) recommends this approach.
- The approach recommended by Date is reminiscent of pivot tables found in Excel.

Table 4: An Excel pivot table representation of Table 3 with Ford sales data included.

Sum	Year	Color					
Sales	1994		1994 Total	1995		1995 Total	Grand Total
Model	black	white		black	white		
Chevy	50	40	90	85	115	200	290
Ford	50	10	60	85	75	160	220
Grand Total	100	50	150	170	190	360	510

All text from: Gray et al. "Data Cube: A Relational Aggregation Operator Generalizing Group-By, Cross-Tab, and Sub-Totals". ICDE'96.

Problems with GROUP BY

- Table 4 an alternative representation of Table 3a (with Ford Sales data included) that illustrates how a pivot table in Excel can present the Sales data by Model, by Year, and then by Color.
- The pivot operator transposes a spreadsheet: typically aggregating cells based on values in the cells.
- Rather than just creating columns based on subsets of column names, pivot creates columns based on subsets of column values.
- This is a much larger set if one pivots on two columns containing N and M values, the resulting pivot table has NxM values.

Table 4: An Excel pivot table representation of Table 3 with Ford sales data included.

Sum	Year	Color					
Sales	1994		1994 Total	1995		1995 Total	Grand Total
Model	black	white		black	white		
Chevy	50	40	90	85	115	200	290
Ford	50	10	60	85	75	160	220
Grand Total	100	50	150	170	190	360	510

All text from: Gray et al. "Data Cube: A Relational Aggregation Operator Generalizing Group-By, Cross-Tab, and Sub-Totals". ICDE'96.

Problems with GROUP BY

- Rather than extend the result table to have many new columns, a more conservative approach prevents the exponential growth of columns by overloading column values.
- The idea is to introduce an ALL value.
- Table 5.a demonstrates this relational & more convenient representation.
- Dummy value "ALL" has been filled in the super aggregation items.

Table 5.a: Sales Summary

Model	Year	Color	Units
Chevy	1994	black	50
Chevy	1994	white	40
Chevy	1994	ALL	90
Chevy	1995	black	85
Chevy	1995	white	115
Chevy	1995	ALL	200
Chevy	ALL	ALL	290

All text from: Gray et al. "Data Cube: A Relational Aggregation Operator Generalizing Group-By, Cross-Tab, and Sub-Totals". ICDE'96.

Problems with GROUP BY

- Since Table 5.a is a relation, it is not surprising that it can be built using standard SQL.
- The SQL statement to build this SalesSummary table from the raw Sales data is:

```
 SELECT 'ALL', 'ALL', 'ALL', SUM(Sales)
     FROM Sales
     WHERE Model = 'Chevy'
UNION
SELECT Model, 'ALL', 'ALL', SUM(Sales)
     FROM Sales
     WHERE Model = 'Chevy'
     GROUP BY Model
UNION
SELECT Model, Year, 'ALL', SUM(Sales)
     FROM Sales
     WHERE Model = 'Chevy'
     GROUP BY Model, Year
UNION
SELECT Model, Year, Color, SUM(Sales)
     FROM Sales
     WHERE Model = 'Chevy'
     GROUP BY Model, Year, Color;
```

All text from: Gray et al. "Data Cube: A Relational Aggregation Operator Generalizing Group-By, Cross-Tab, and Sub-Totals". ICDE'96.

Problems with GROUP BY

- This is a simple 3-dimensional roll-up.
- Aggregating over N dimensions requires N such unions.
- Roll-up is asymmetric – notice that Table 5.a aggregates sales by year but not by color. These rows are:

Table 5.b: Sales Summary rows missing form Table 5.a to convert the roll-up into a cube.

Model	Year	Color	Units
Chevy	ALL	black	135
Chevy	ALL	white	155

- These additional rows could be captured by adding the following clause to the SQL statement above:

```
UNION
SELECT Model, 'ALL', Color, SUM(Sales)
      FROM Sales
      WHERE Model = 'Chevy'
      GROUP BY Model, Color;
```

All text from: Gray et al. "Data Cube: A Relational Aggregation Operator Generalizing Group-By, Cross-Tab, and Sub-Totals". ICDE'96.

Problems with GROUP BY

- The symmetric aggregation result is a table called a cross-tabulation, or cross tab for short.
- Tables 5.a and 5.b are the relational form of the cross-tabs, but cross tab data is routinely displayed in the more compact format of Table 6.a.

Table 6.a: Chevy Sales Cross Tab			
Chevy	**1994**	**1995**	**total** (ALL)
black	50	85	135
white	40	115	155
total (ALL)	90	200	290

All text from: Gray et al. "Data Cube: A Relational Aggregation Operator Generalizing Group-By, Cross-Tab, and Sub-Totals". ICDE'96.

Problems with GROUP BY

- This cross tab is a two-dimensional aggregation.
- If other automobile models are added, it becomes a 3D aggregation.
- For example, data for Ford products adds an additional cross tab plane.
- The cross-tab-array representation (Table 6.a, 6.b) is equivalent to the relational representation using the ALL value.
- Both generalize to an N-dimensional cross tab.

Table 6b: Ford Sales Cross Tab			
Ford	**1994**	**1995**	**total** (ALL)
black	50	85	135
white	10	75	85
total (ALL)	60	160	220

All text from: Gray et al. "Data Cube: A Relational Aggregation Operator Generalizing Group-By, Cross-Tab, and Sub-Totals". ICDE'96.

Problems with GROUP BY

- The representation suggested by Table 5 and unioned GROUP BYs “solve” the problem of representing aggregate data in a relational data model.
- The problem remains that expressing roll-up, and cross-tab queries with conventional SQL is daunting.
- A six dimension cross-tab requires a 64-way union of 64 different GROUP BY operators to build the underlying representation.

All text from: Gray et al. "Data Cube: A Relational Aggregation Operator Generalizing Group-By, Cross-Tab, and Sub-Totals". ICDE’96.

Problems with GROUP BY

- There is another very important reason why it is inadequate to use GROUP BYs.
- The resulting representation of aggregation is too complex to analyze for optimization.
- On most SQL systems this will result in 64 scans of the data, 64 sorts or hashes, and a long wait.

All text from: Gray et al. "Data Cube: A Relational Aggregation Operator Generalizing Group-By, Cross-Tab, and Sub-Totals". ICDE'96.

The Data Cube

- The generalization of group by, roll-up and cross-tab ideas seems obvious: Figure 3 shows the concept for aggregation up to 3-dimensions.
- The traditional GROUP BY generates the N-dimensional data cube core.
- The N-1 lower-dimensional aggregates appear as points, lines, planes, cubes, or hyper-cubes hanging off the data cube core.
- Creating a data cube requires generating the power set (set of all subsets) of the aggregation columns.

All text from: Gray et al. "Data Cube: A Relational Aggregation Operator Generalizing Group-By, Cross-Tab, and Sub-Totals". ICDE'96.

The Data Cube

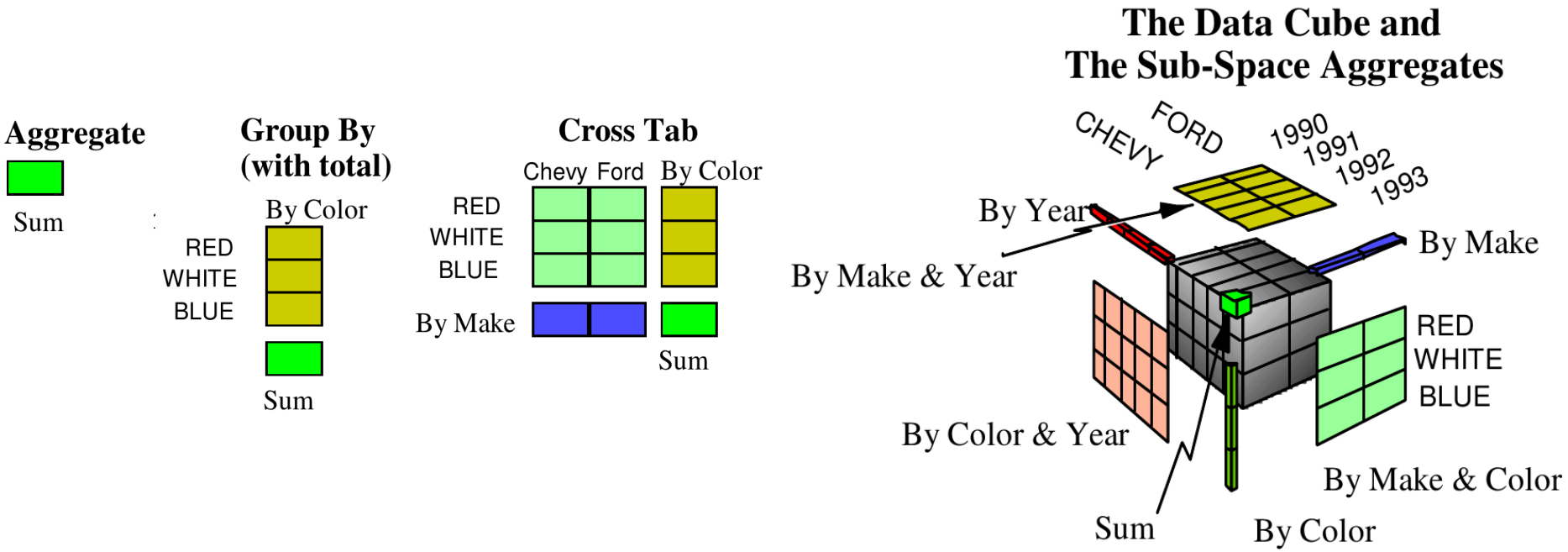


Figure 3: The CUBE operator is the N-dimensional generalization of simple aggregate functions. The 0D data cube is a point. The 1D data cube is a line with a point. The 2D data cube is a cross tabulation, a plane, two lines, and a point. The 3D data cube is a cube with three intersecting 2D cross tabs.

Image from: Gray et al. "Data Cube: A Relational Aggregation Operator Generalizing Group-By, Cross-Tab, and Sub-Totals". ICDE'96.

1. QA and References
   - Answers to Previous Questions
   - References for Today's Lecture

2. Origins of Data Warehouses
   - Data Analysis and Visualization
   - Data Extraction using SQL
   - Problems with SQL for Data Analysis
   - The Data Cube

3. Data Cubes
   - Multi-dimensional Data Model
   - Dimensions and Concept Hierarchies
   - Schema Design
   - Operators

4. Summary

Multi-dimensional Data Model

- A data cube allows data to be modeled and viewed in multiple dimensions.
- It is defined by dimensions and facts.
- Dimensions are the perspectives or entities with respect to which an organization wants to keep records. Each dimension may have a table associated with it, called a dimension table, which further describes the dimension.
- A 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. Think of them as the quantities by which we want to analyze relationships between dimensions.
- The fact table contains the names of the facts, or measures, as well as keys to each of the related dimension tables.

All text from: Han et al. "Data Mining Concepts and Techniques", 3rd Edition, 2012, Morgan Kaufmann Publishers.

Multi-dimensional Data Model

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).

Image from: Han et al. "Data Mining Concepts and Techniques", 3rd Edition, 2012, Morgan Kaufmann Publishers.

Multi-dimensional Data Model

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***			
	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).

Image from: Han et al. "Data Mining Concepts and Techniques", 3rd Edition, 2012, Morgan Kaufmann Publishers.

Multi-dimensional Data Model

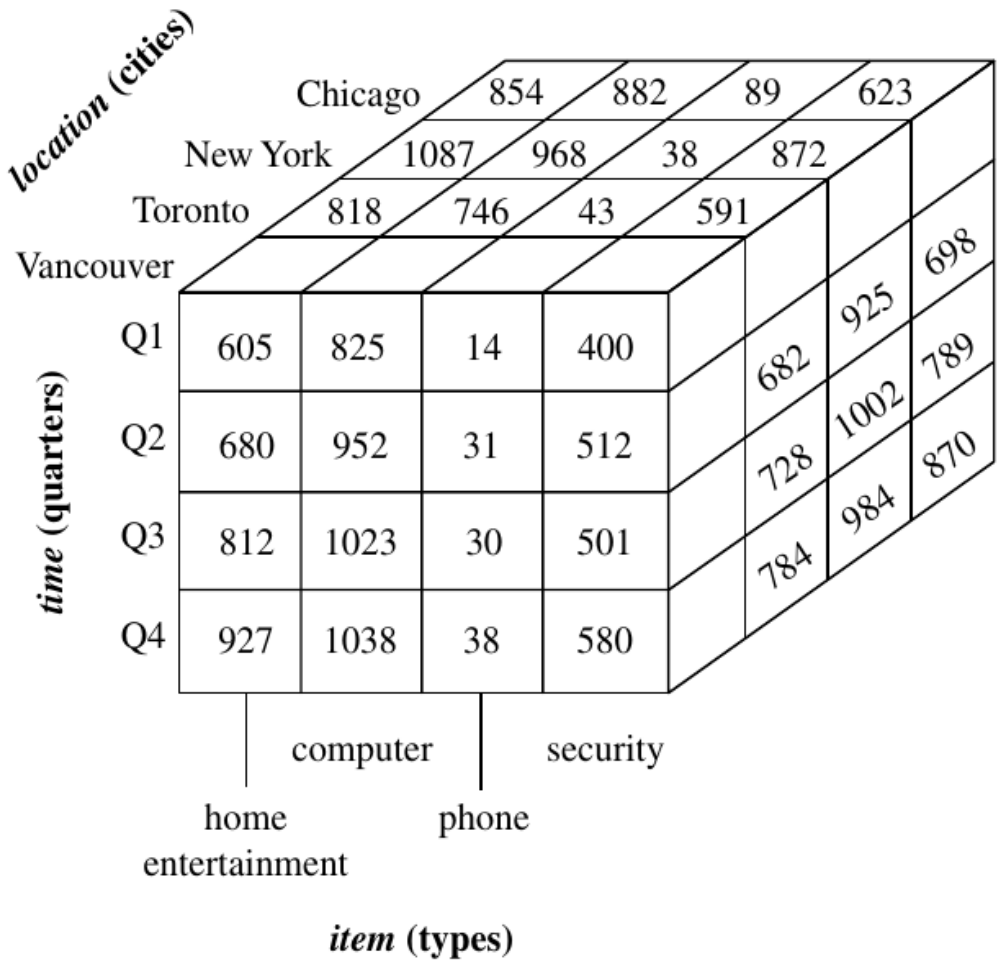


Figure 4.3 A 3-D data cube representation of the data in Table 4.3, according to *time*, *item*, and *location*. The measure displayed is *dollars_sold* (in thousands).

Image from: Han et al. "Data Mining Concepts and Techniques", 3rd Edition, 2012, Morgan Kaufmann Publishers.

Multi-dimensional Data Model

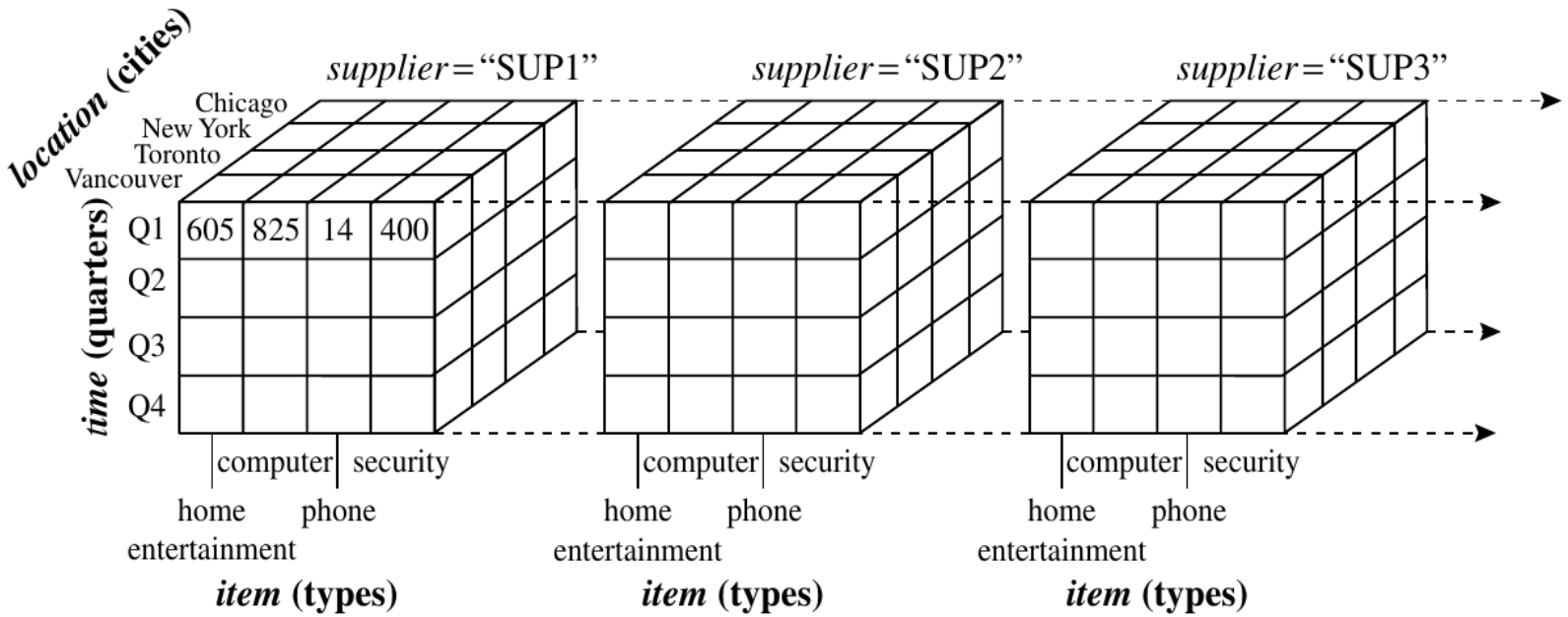


Figure 4.4 A 4-D data cube representation of sales data, according to *time*, *item*, *location*, and *supplier*. The measure displayed is *dollars_sold* (in thousands). For improved readability, only some of the cube values are shown.

Image from: Han et al. "Data Mining Concepts and Techniques", 3rd Edition, 2012, Morgan Kaufmann Publishers.

Multi-dimensional Data Model

- A data cube like those shown in Figures 4.3 and 4.4 is often referred to as a cuboid.
- Given a set of dimensions, we can generate a cuboid for each of the possible subsets of the given dimensions.
- The result would form a lattice of cuboids, each showing the data at a different level of summarization, or group-by.
- The lattice of cuboids is then referred to as a data cube.

All text from: Han et al. "Data Mining Concepts and Techniques", 3rd Edition, 2012, Morgan Kaufmann Publishers.

Multi-dimensional Data Model

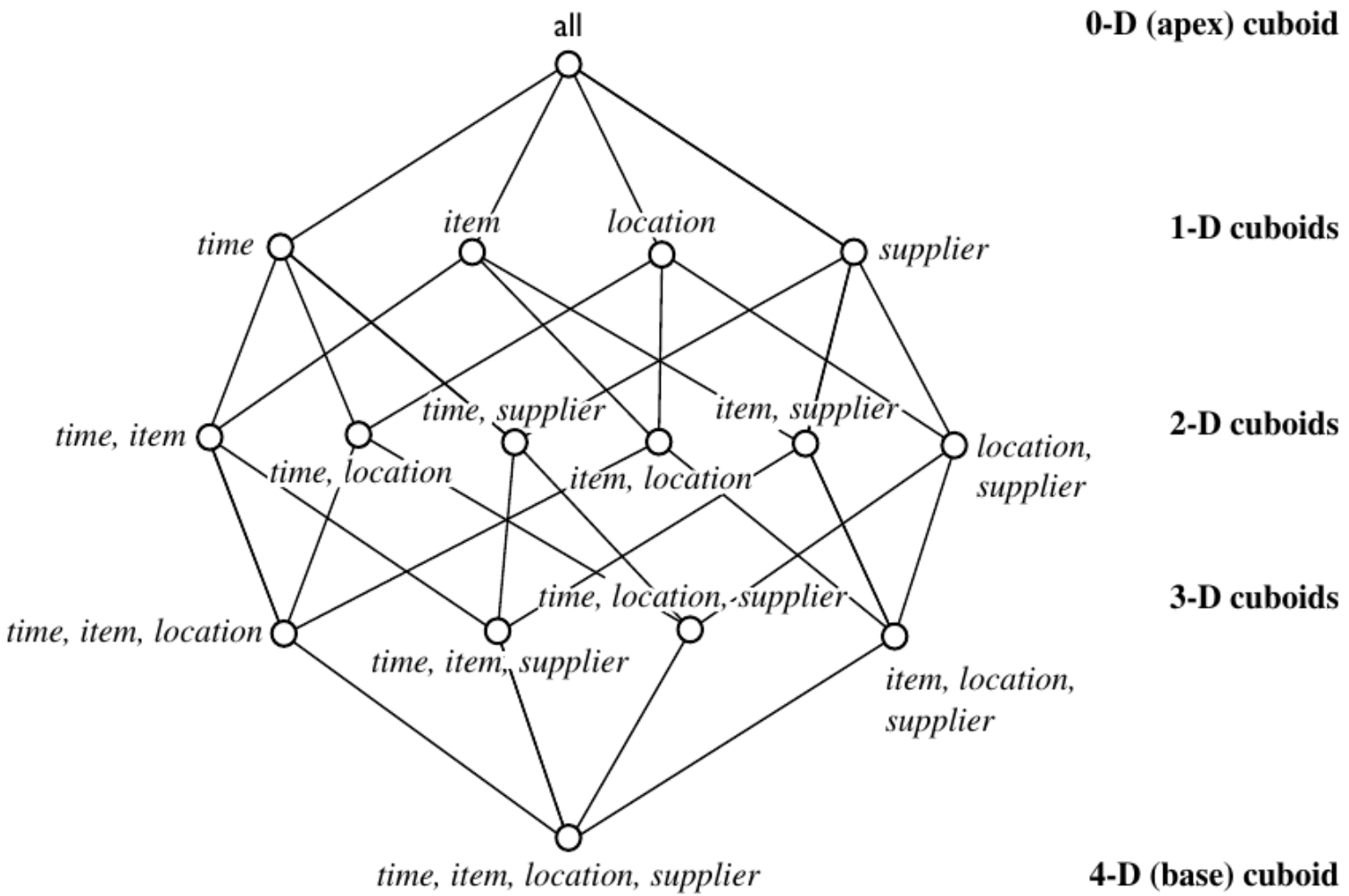


Figure 4.5 Lattice of cuboids, making up a 4-D data cube for *time*, *item*, *location*, and *supplier*. Each cuboid represents a different degree of summarization.

Image from: Han et al. "Data Mining Concepts and Techniques", 3rd Edition, 2012, Morgan Kaufmann Publishers.

Dimensions and Concept Hierarchies

- A concept hierarchy defines a sequence of mappings from a set of low-level concepts to higher-level, more general concepts.

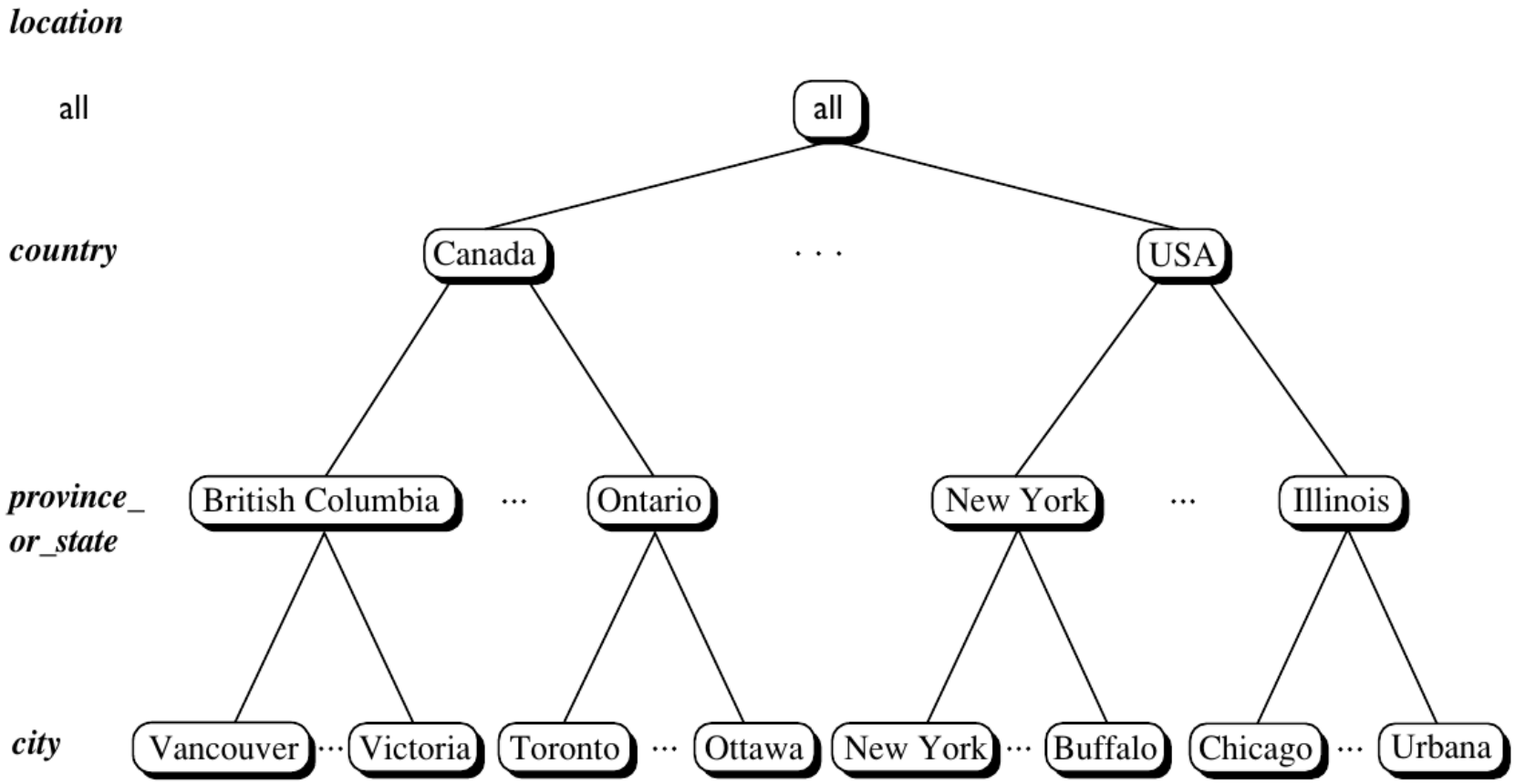


Figure 4.9 A concept hierarchy for *location*. Due to space limitations, not all of the hierarchy nodes are shown, indicated by ellipses between nodes.

Text & image from: Han et al. "Data Mining Concepts and Techniques", 3rd Edition, 2012, Morgan Kaufmann Publishers.

Dimesions and Concept Hierarchies

- Concept hierarchies may also be defined by discretizing or grouping values for a given dimension or attribute, resulting in a set-grouping hierarchy.

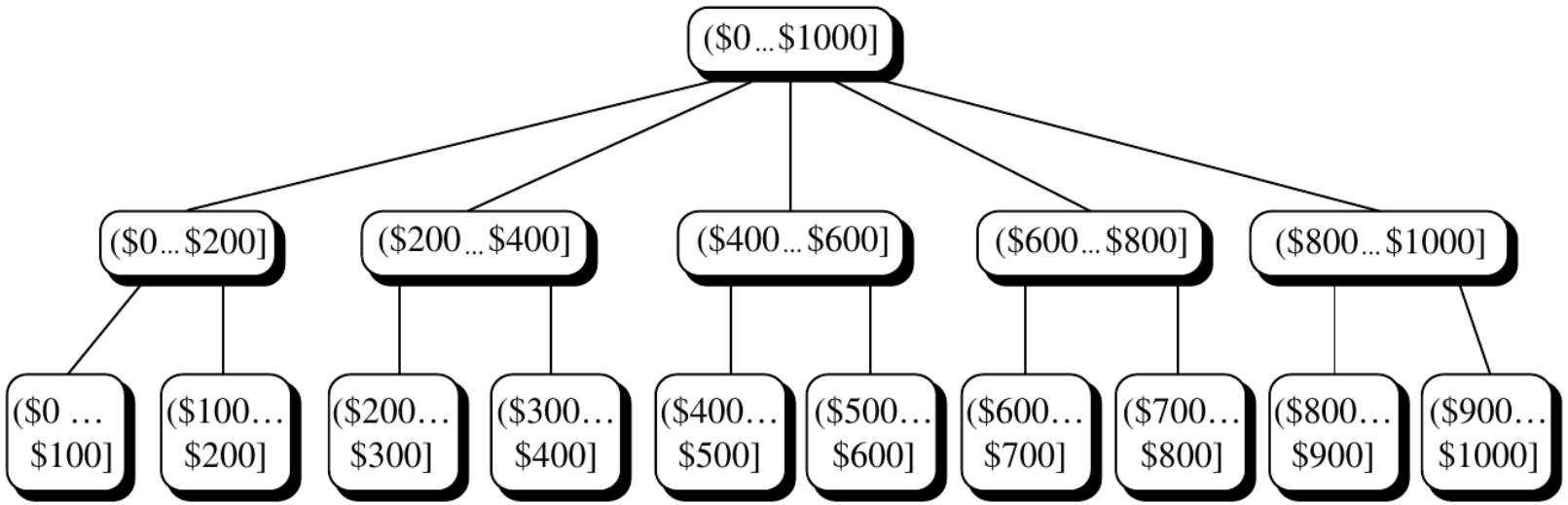


Figure 4.11 A concept hierarchy for *price*.

Text & image from: Han et al. "Data Mining Concepts and Techniques", 3rd Edition, 2012, Morgan Kaufmann Publishers.

Schema Design

- The entity-relationship data model is commonly used in the design of relational databases, where a database schema consists of a set of entities and the relationships between them.
- Such a data model is appropriate for online transaction processing.
- A data warehouse, however, requires a concise, subject-oriented schema that facilitates online data analysis.
- The most popular data model for a data warehouse is a multidimensional model, which can exist in the form of:
  1. a star schema,
  2. a snowflake schema, or
  3. a fact constellation schema.

All text from: Han et al. "Data Mining Concepts and Techniques", 3rd Edition, 2012, Morgan Kaufmann Publishers.

Schema Design — Star Schema

- Star Schema contains:
  1. a large central table (fact table) containing the bulk of the data, with no redundancy,
  2. a set of smaller attendant tables (dimension tables), one for each dimension.
- The schema graph resembles a starburst, with the dimension tables displayed in a radial pattern around the central fact table.

All text from: Han et al. "Data Mining Concepts and Techniques", 3rd Edition, 2012, Morgan Kaufmann Publishers.

Schema Design — Star Schema

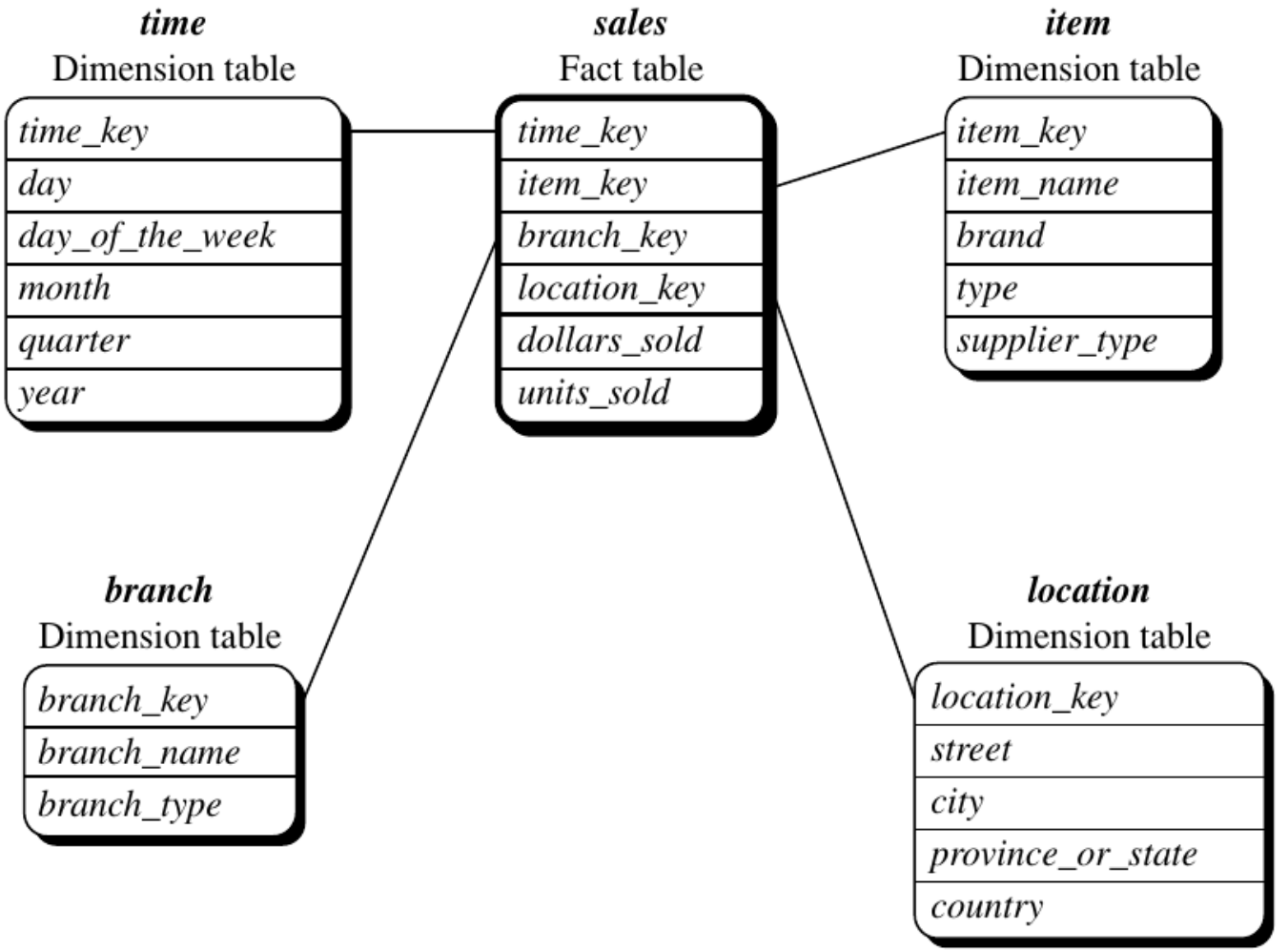


Figure 4.6 Star schema of *sales* data warehouse.

Image from: Han et al. "Data Mining Concepts and Techniques", 3rd Edition, 2012, Morgan Kaufmann Publishers.

Schema Design — Snowflake Schema

- Snowflake Schema is a variant of the star schema model, where some dimension tables are normalized, thereby further splitting the data into additional tables.
- The resulting schema graph forms a shape similar to a snowflake.

All text from: Han et al. "Data Mining Concepts and Techniques", 3rd Edition, 2012, Morgan Kaufmann Publishers.

Schema Design — Snowflake Schema

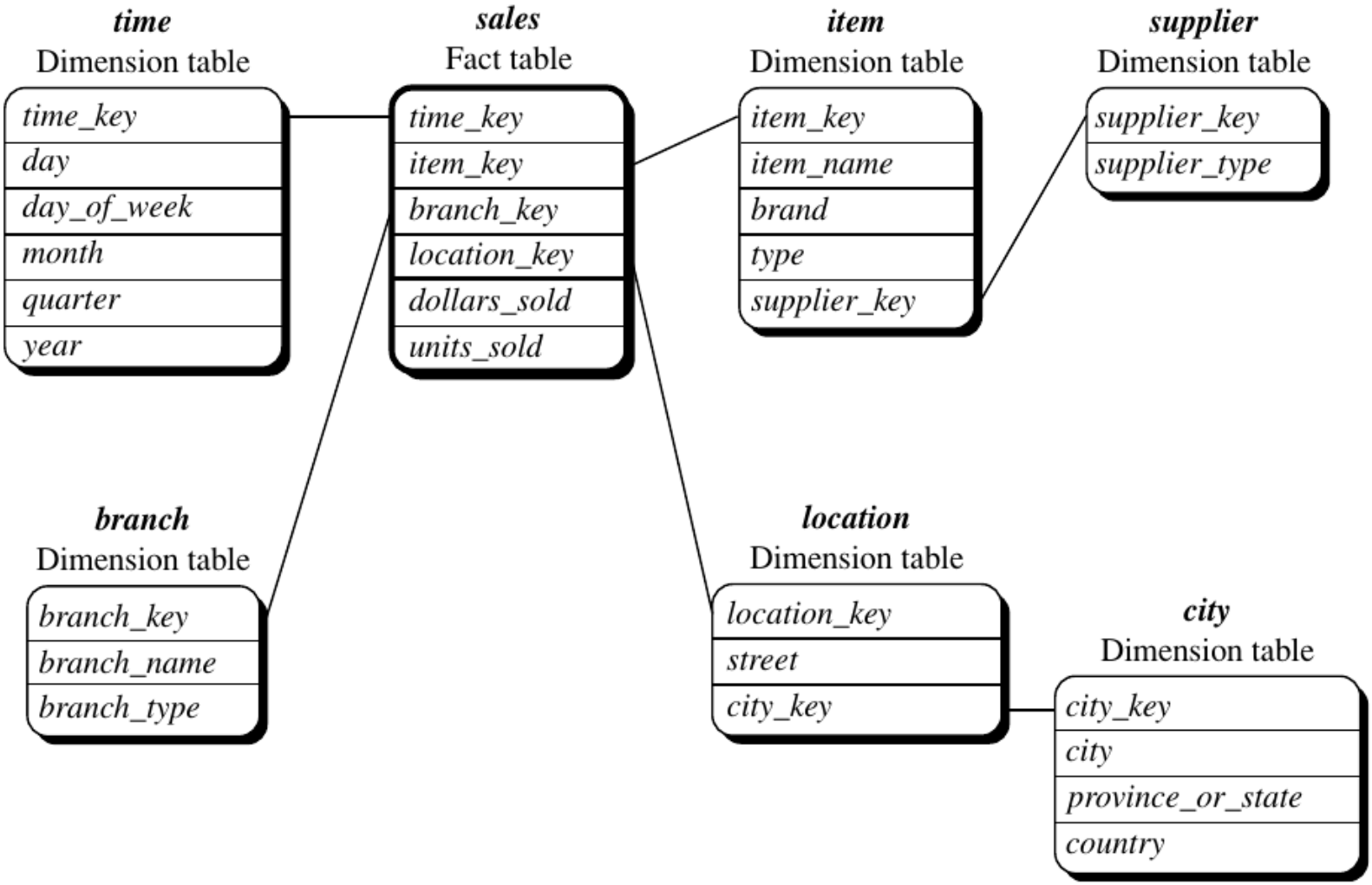


Figure 4.7 Snowflake schema of a *sales* data warehouse.

Image from: Han et al. "Data Mining Concepts and Techniques", 3rd Edition, 2012, Morgan Kaufmann Publishers.

Star versus Snowflake Schema

- The major difference between the snowflake and star schema models is that the dimension tables of the snowflake model may be kept in normalized form to reduce redundancies.
- Such a table is easy to maintain and saves storage space.
- However, this space savings is negligible in comparison to the typical magnitude of the fact table.
- Furthermore, the snowflake structure can reduce the effectiveness of browsing, since more joins will be needed to execute a query.
- Consequently, the system performance may be adversely impacted.
- Hence, although the snowflake schema reduces redundancy, it is not as popular as the star schema in data warehouse design.

Fact Constellation Schema

- 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.

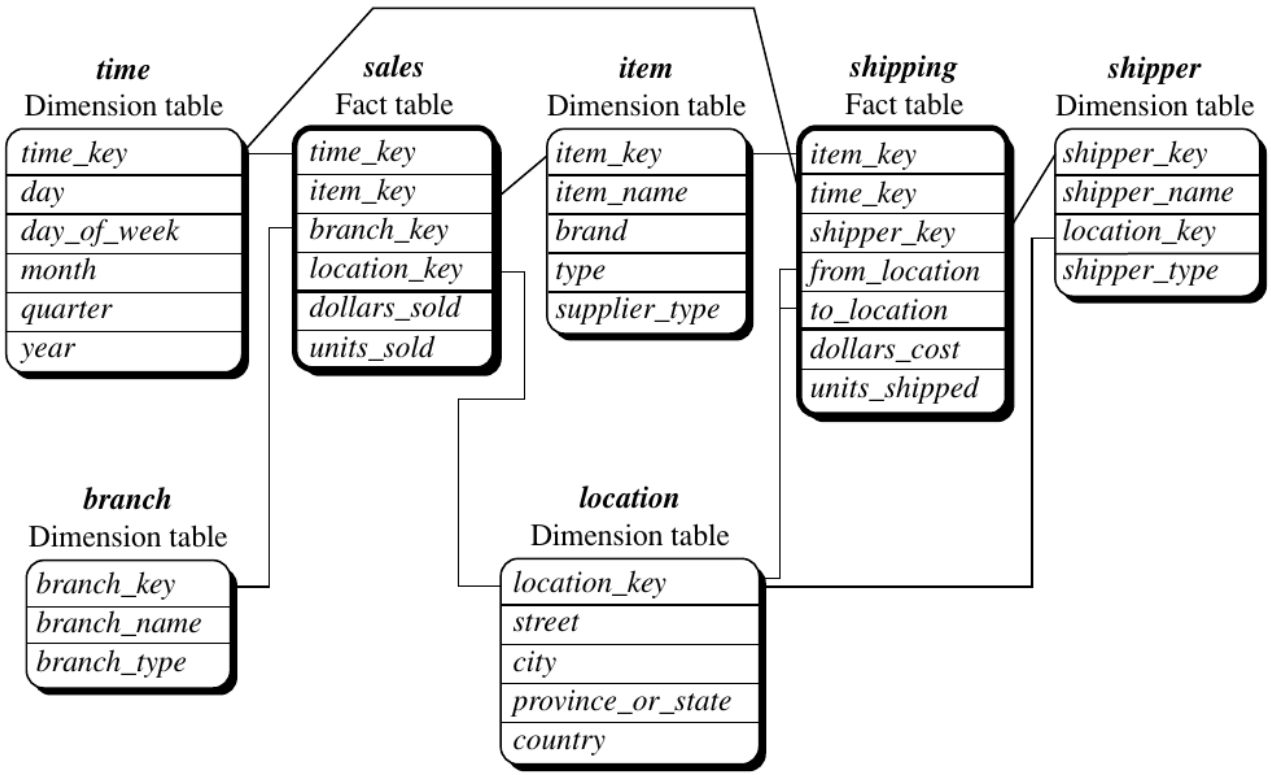


Figure 4.8 Fact constellation schema of a sales and shipping data warehouse.

Text & image: Han et al. "Data Mining Concepts and Techniques", 3rd Edition, 2012, Morgan Kaufmann Publishers.

Operators

- A number of data cube operations exist to materialize these different views, allowing interactive querying and analysis of the data at hand.
  1. Drill Down
  2. Roll Up
  3. Dice
  4. Pivot
  5. Slice

All text from: Han et al. "Data Mining Concepts and Techniques", 3rd Edition, 2012, Morgan Kaufmann Publishers.

Operators — Drill Down

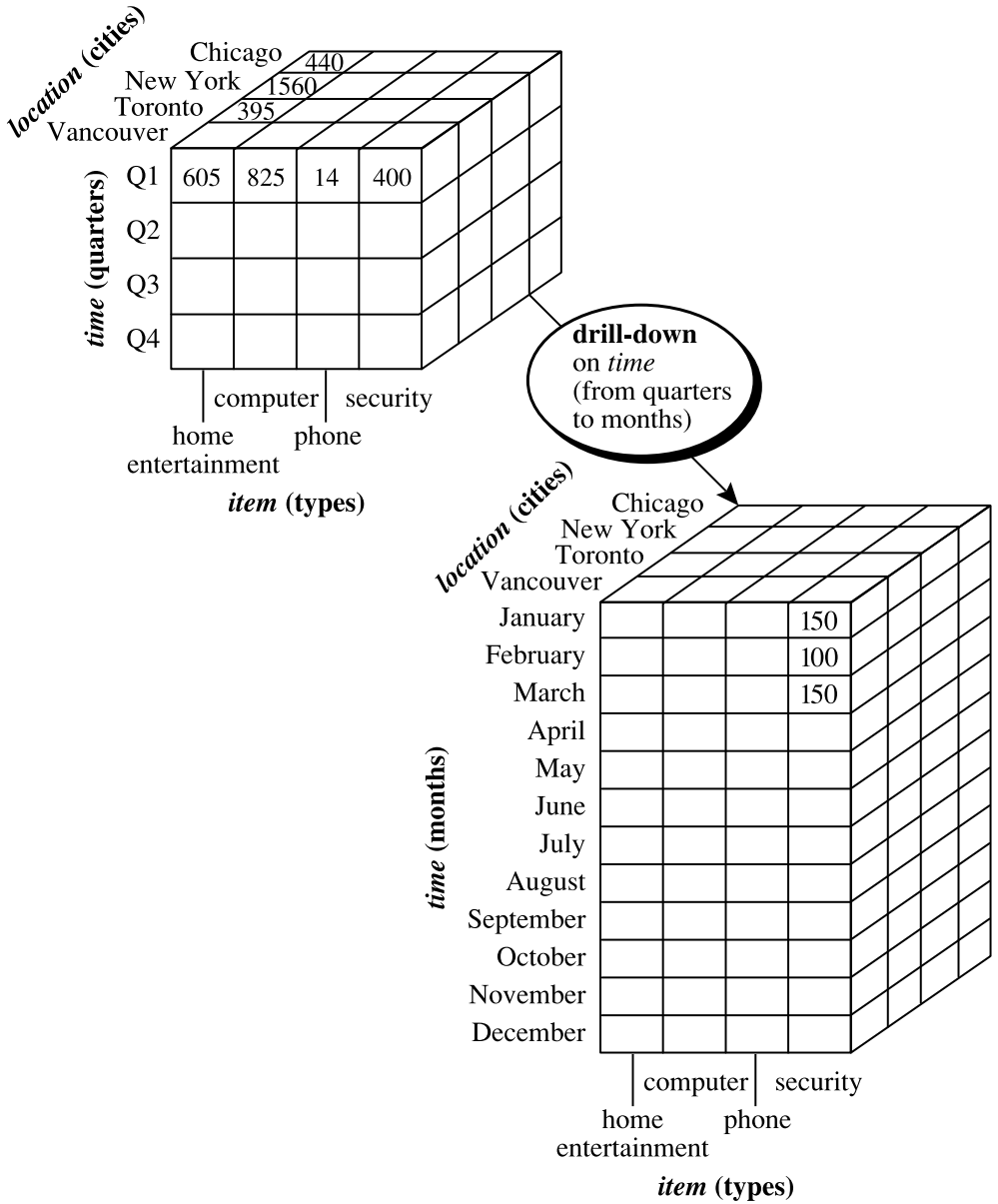


Image from: Han et al. "Data Mining Concepts and Techniques", 3rd Edition, 2012, Morgan Kaufmann Publishers.

Operators — Roll Up

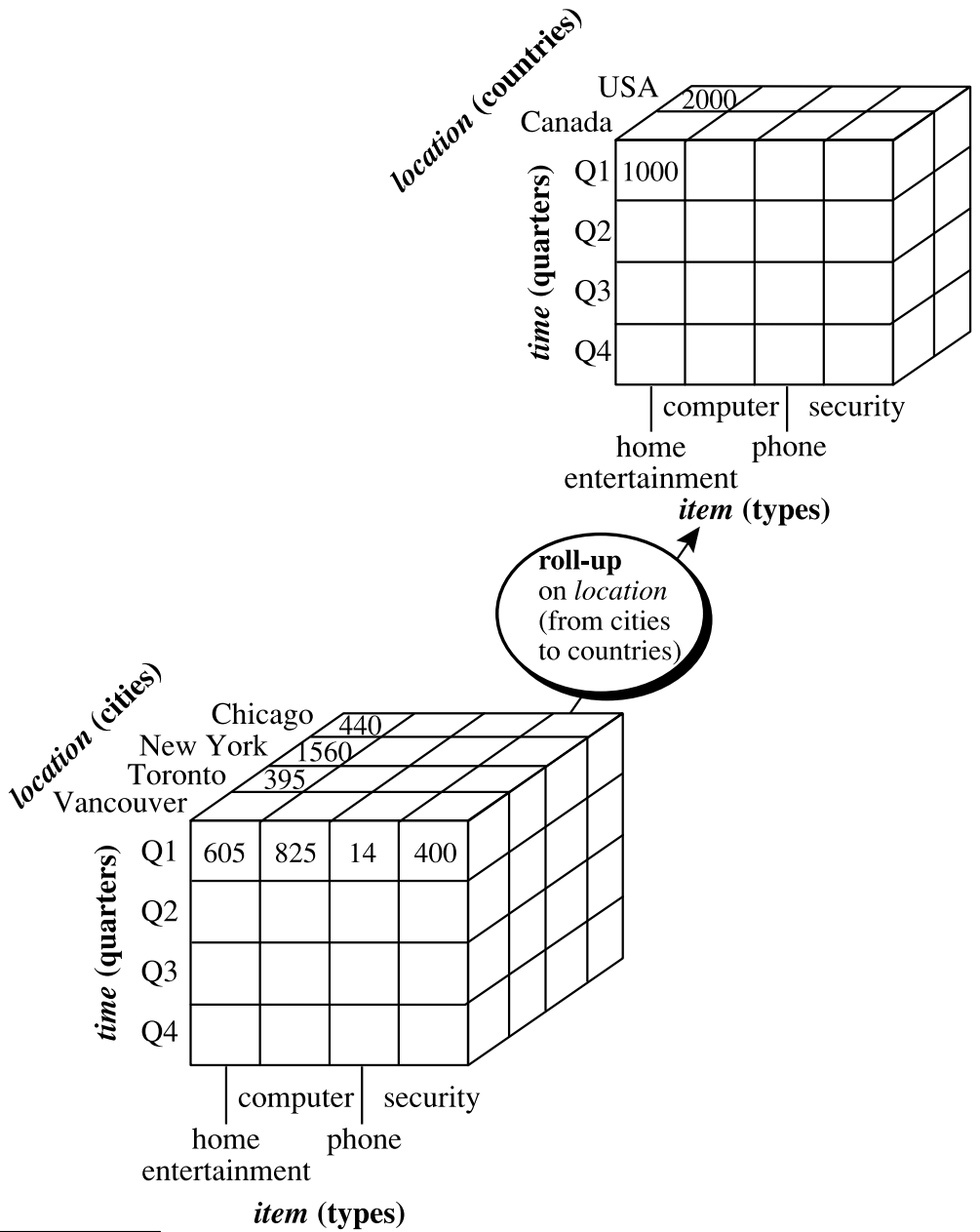


Image from: Han et al. "Data Mining Concepts and Techniques", 3rd Edition, 2012, Morgan Kaufmann Publishers.

Operators — Dice

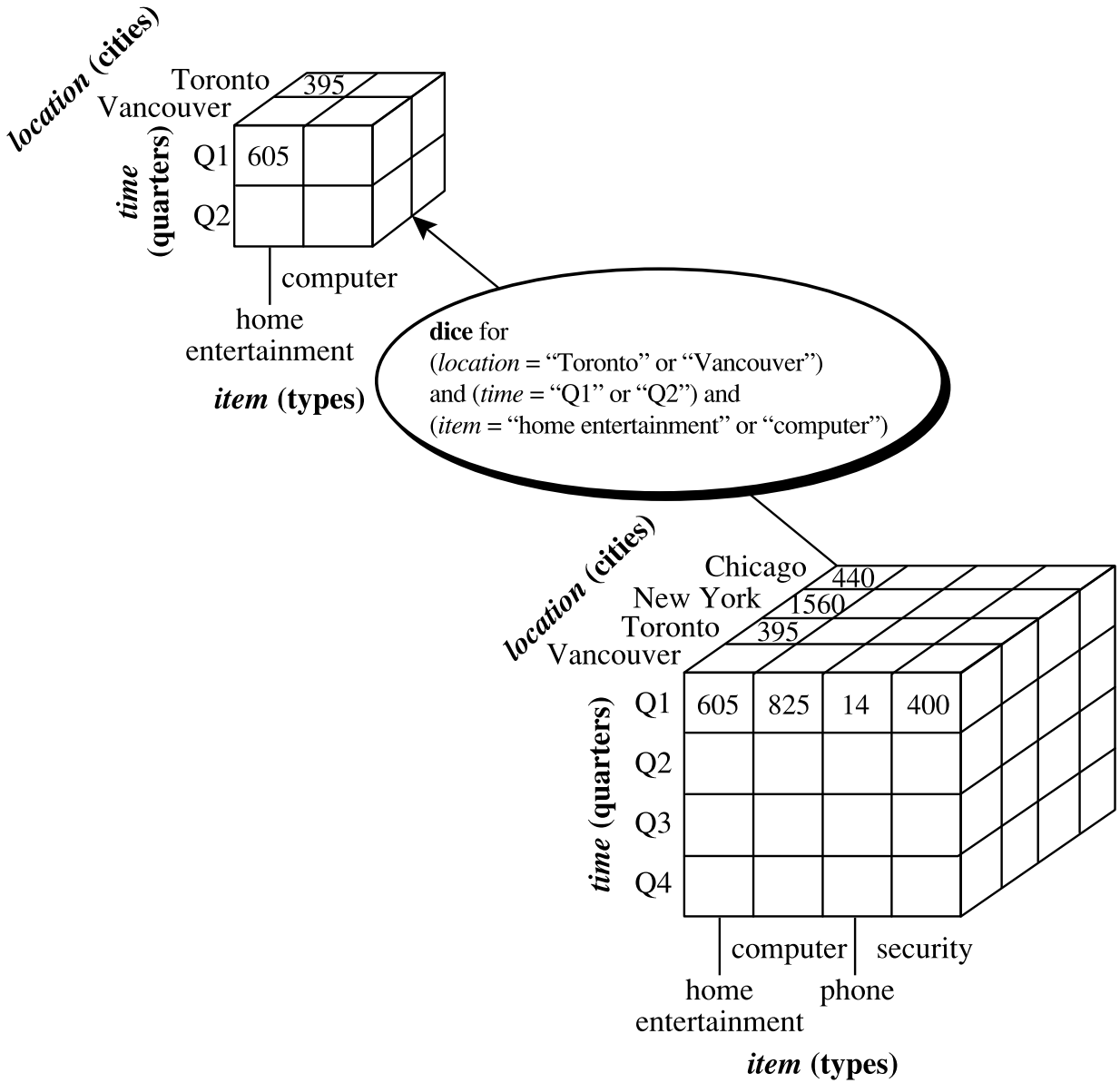


Image from: Han et al. "Data Mining Concepts and Techniques", 3rd Edition, 2012, Morgan Kaufmann Publishers.

Operators — Slice and Pivot

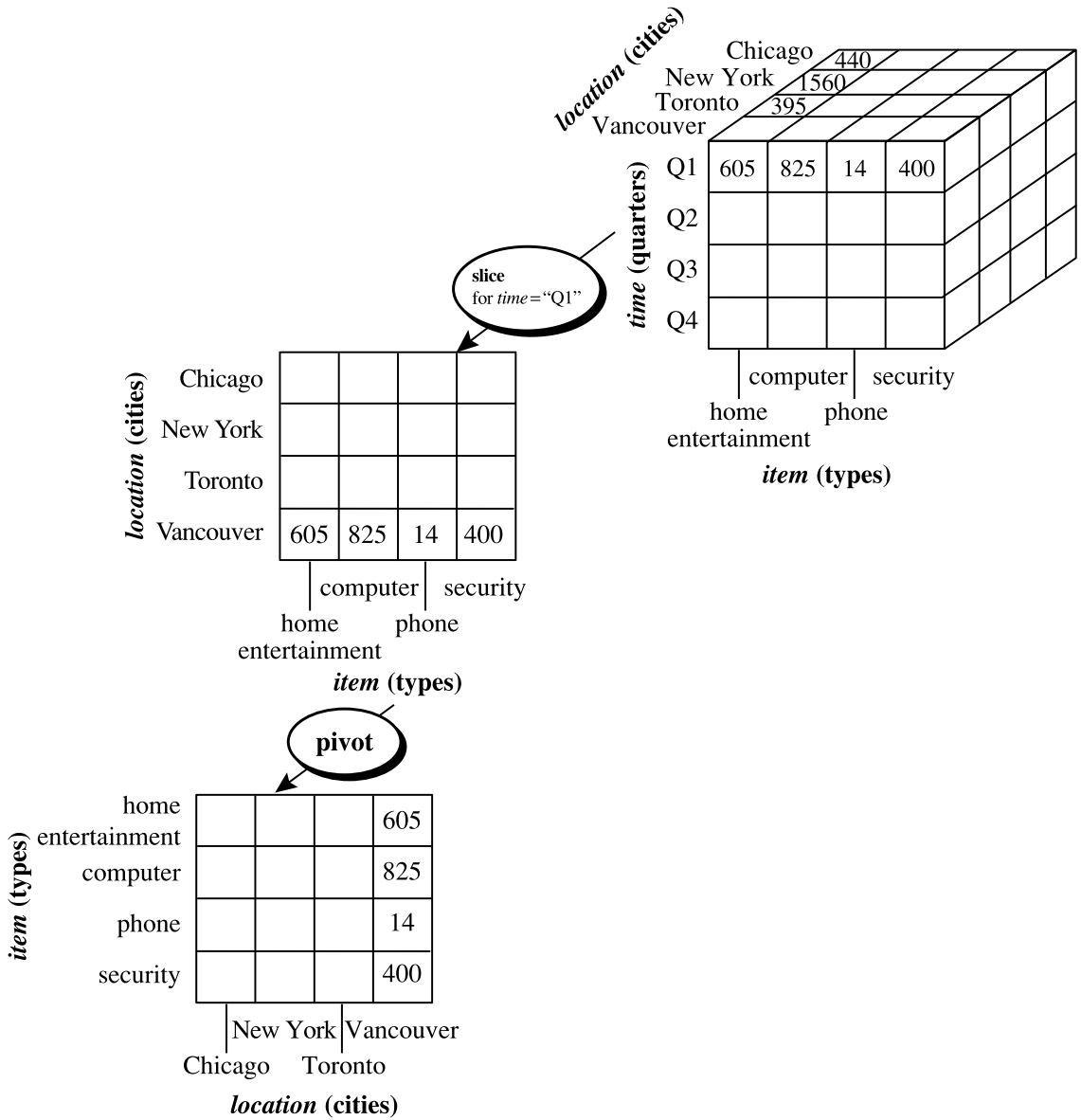


Image from: Han et al. "Data Mining Concepts and Techniques", 3rd Edition, 2012, Morgan Kaufmann Publishers.

Starnet Query Model

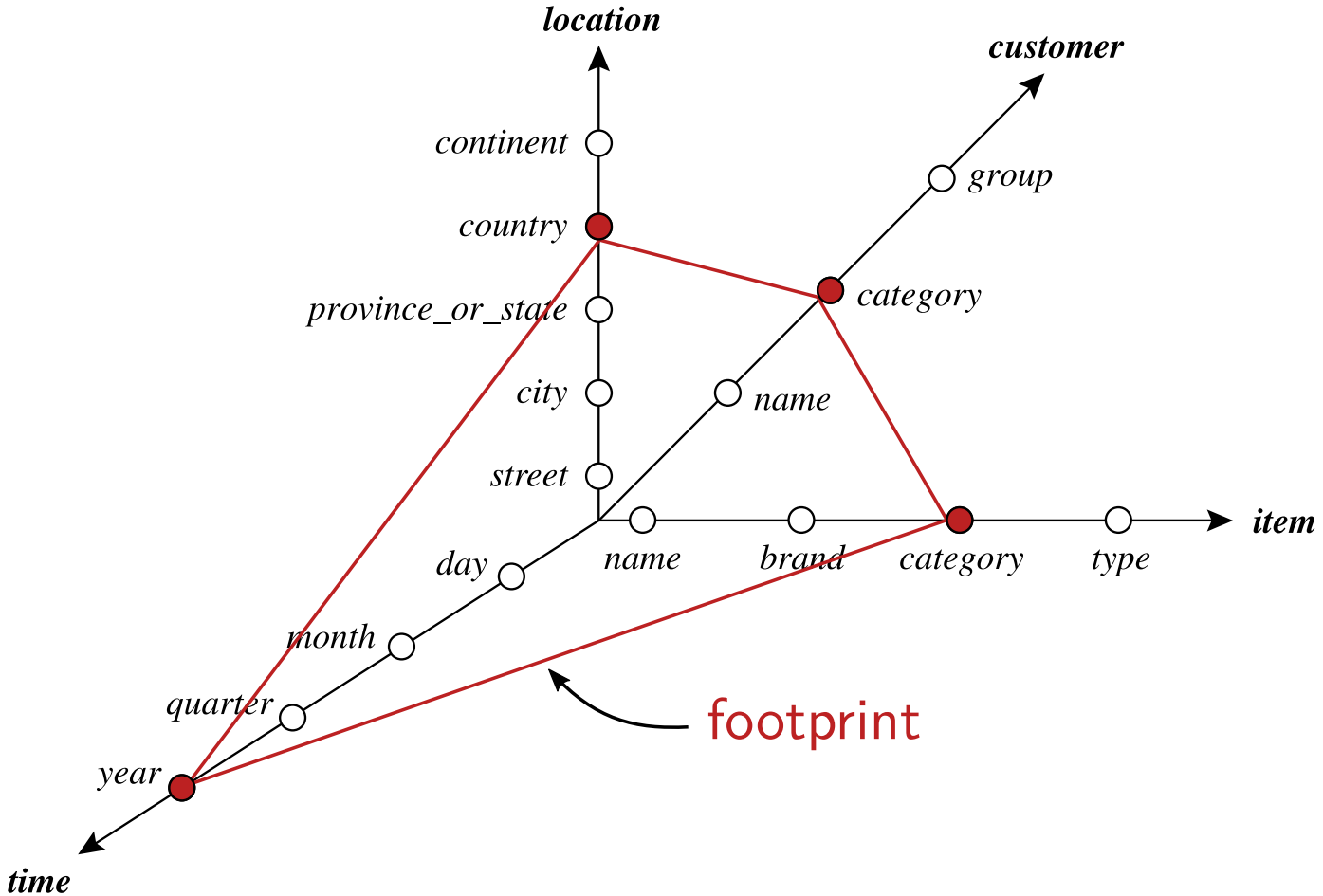


Figure 4.13 A starnet model of business queries.

Image from: Han et al. "Data Mining Concepts and Techniques", 3rd Edition, 2012, Morgan Kaufmann Publishers.

1. QA and References
   - Answers to Previous Questions
   - References for Today's Lecture

2. Origins of Data Warehouses
   - Data Analysis and Visualization
   - Data Extraction using SQL
   - Problems with SQL for Data Analysis
   - The Data Cube

3. Data Cubes
   - Multi-dimensional Data Model
   - Dimensions and Concept Hierarchies
   - Schema Design
   - Operators

4. Summary

Summary — Data Analysis and Visualization

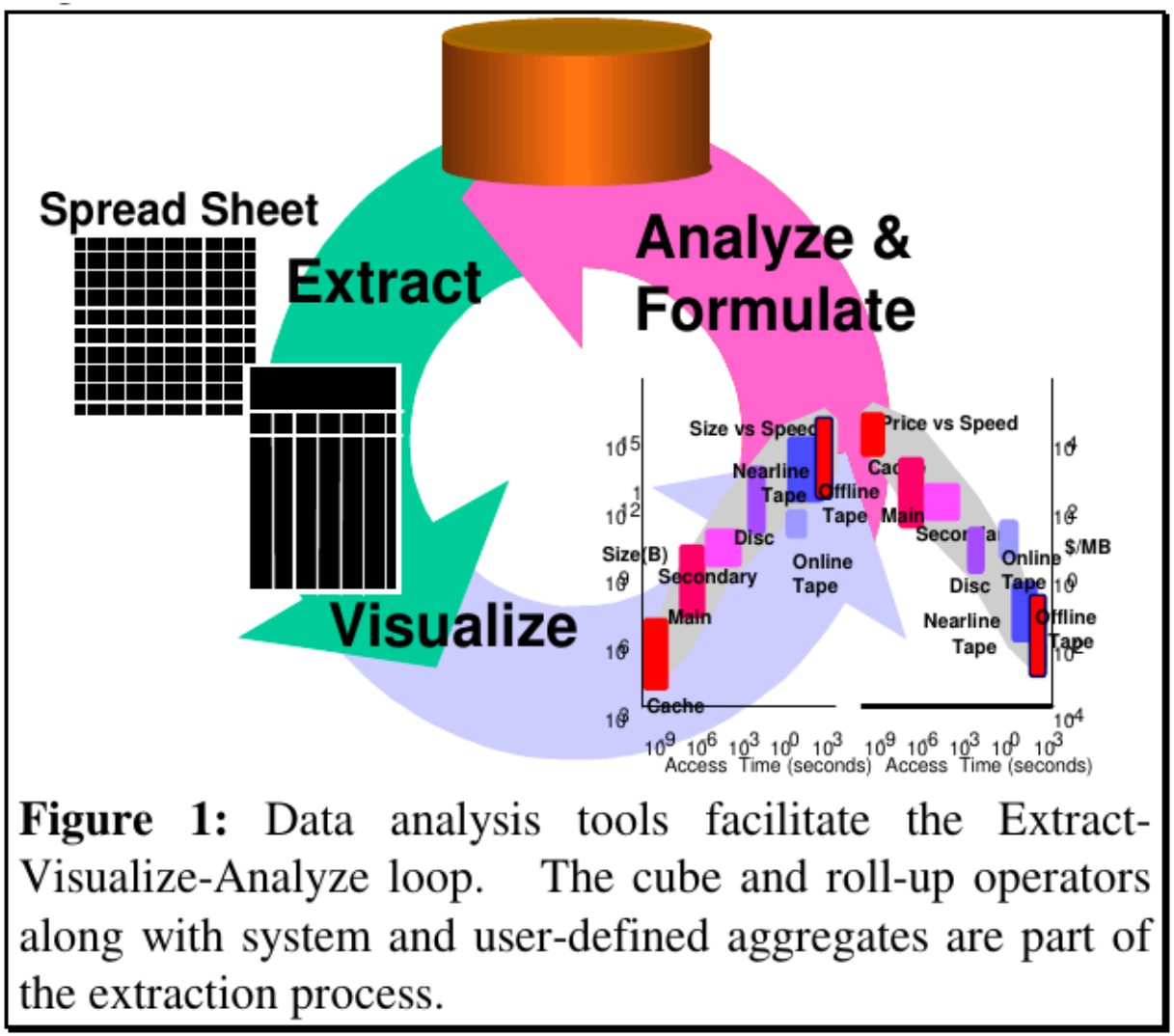


Figure 1: Data analysis tools facilitate the Extract-Visualize-Analyze loop. The cube and roll-up operators along with system and user-defined aggregates are part of the extraction process.

Image from: Gray et al. "Data Cube: A Relational Aggregation Operator Generalizing Group-By, Cross-Tab, and Sub-Totals". ICDE'96.

Summary — The Data Cube

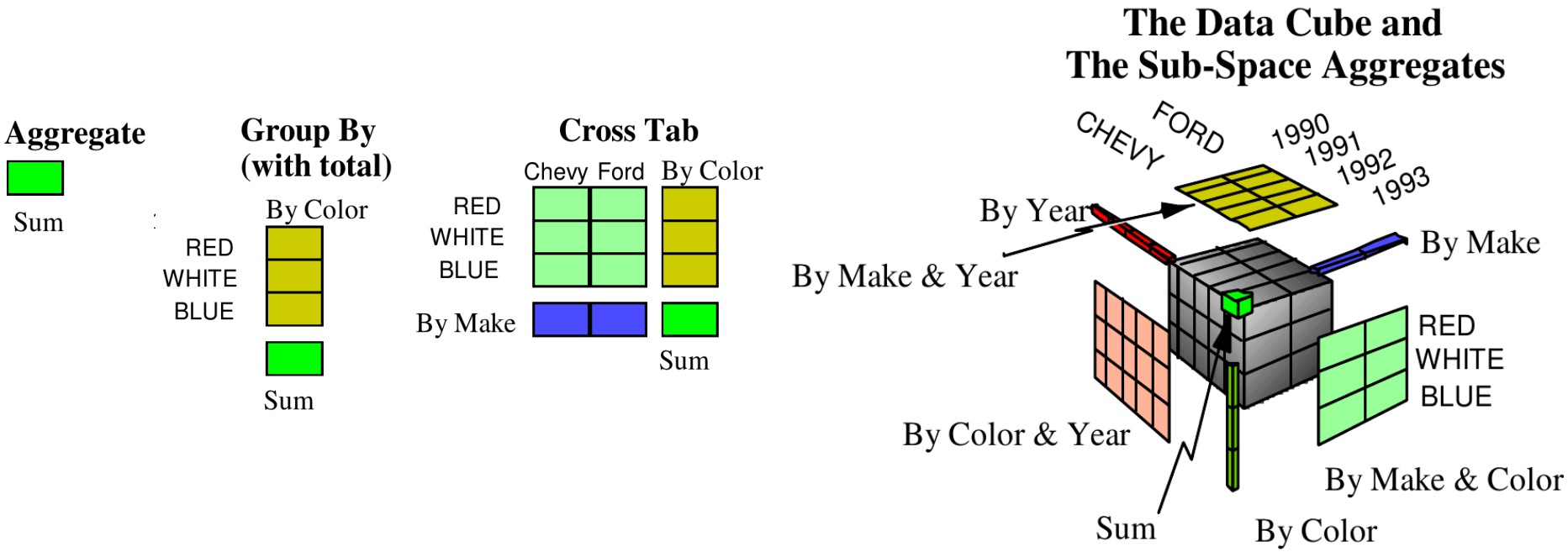


Figure 3: The CUBE operator is the N-dimensional generalization of simple aggregate functions. The 0D data cube is a point. The 1D data cube is a line with a point. The 2D data cube is a cross tabulation, a plane, two lines, and a point. The 3D data cube is a cube with three intersecting 2D cross tabs.

Image from: Gray et al. "Data Cube: A Relational Aggregation Operator Generalizing Group-By, Cross-Tab, and Sub-Totals". ICDE'96.

Summary — Fact Constellation Schema

- 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.

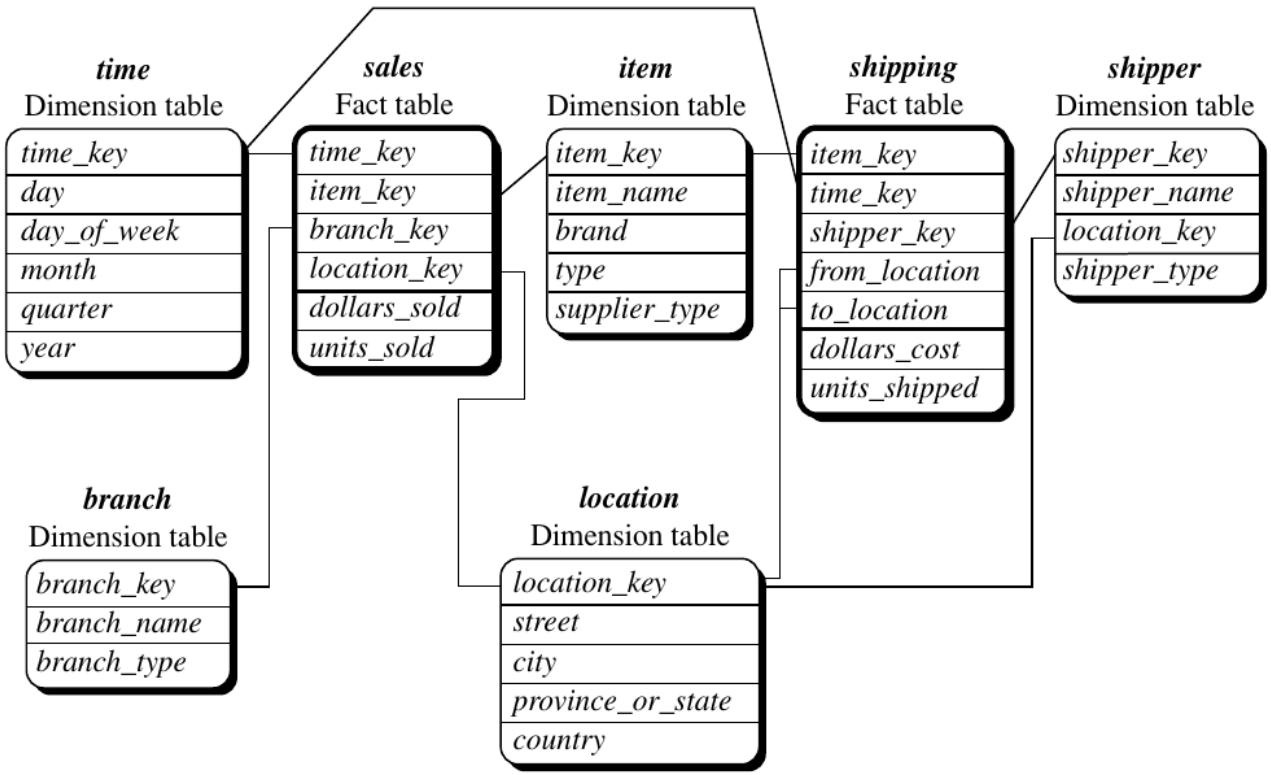


Figure 4.8 Fact constellation schema of a sales and shipping data warehouse.

Text & image: Han et al. "Data Mining Concepts and Techniques", 3rd Edition, 2012, Morgan Kaufmann Publishers.

Summary — Drill Down Operator

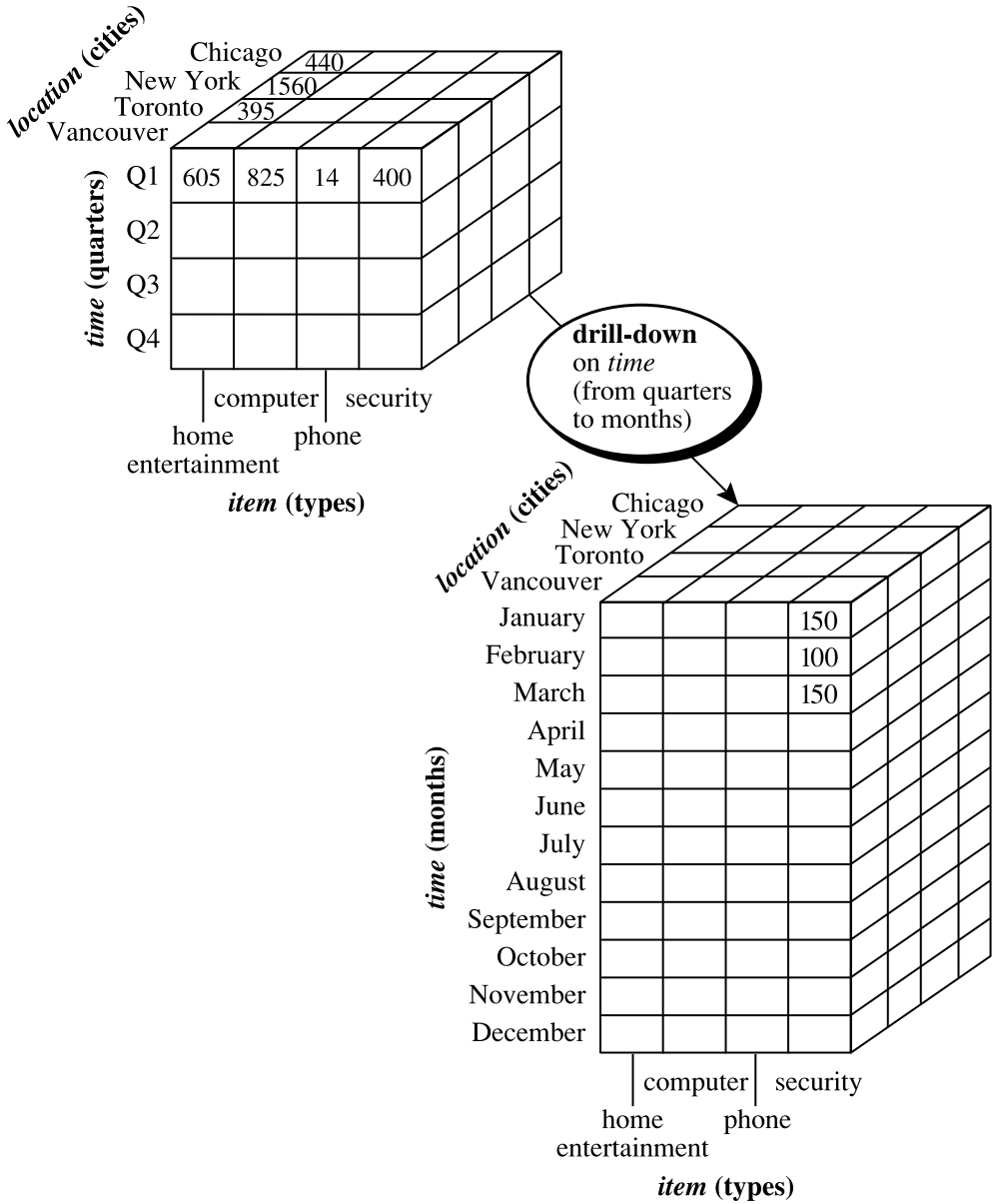


Image from: Han et al. "Data Mining Concepts and Techniques", 3rd Edition, 2012, Morgan Kaufmann Publishers.