

2.1 Data Types and Attributes

A data is a known fact that can be recorded and have implicit (imbedded) meaning. For example, the names, eye color, telephone numbers, and addresses of the students of a class. So, data is a collection of data objects and their attributes.

A collection of attributes describe an object

Attribute

An attribute is a property or characteristic of an object. Attribute values are numbers or symbols assigned to an attribute. Different attributes can be mapped to the same set of values. Example: Attribute values for ID and age are integers but properties of attribute values can be different. ID has no limit but age has a maximum and minimum value.

Properties and types of attribute values:

The type of attribute depends on which of the following properties it possesses:

Distinctness: = and \neq

Order: $<$, \leq , $>$ and \geq

Addition: + and –

Multiplication: * and /

A simple way to specify the type of an attribute is to identify the properties of numbers that corresponds to those properties. Given these properties we can define following types of attributes:

i Categorical Attributes

Categorical attributes are also called qualitative attributes. Categorical attributes lack most of the properties of numbers even if they are represented by numbers. If these attributes are integers, they should be treated more like symbols.

a. Nominal Attributes:

In a nominal scale of measurement numbers or names are assigned to objects where the different values indicate different objects. Those attributes have no real meaning other than differentiating between objects. For example student roll numbers. Because the roll number that a student has

provides no insight into the student's grade or IQ or anything like that it just simply differentiates between students. So roll number 23 and roll number 25 doesn't mean that the student who has 25 is better or smarter or anything like that it just means they're not the same student as number 23.

b. Ordinal Attribute

Ordinal attributes are the values which have meaningful order of ranking between them. For example the place someone finishes in a race first, second, third, and so on. If we know the place that they finished we know how they did relative to others. So, the first place person did better than second, second did better than third, and so on.

ii Numeric Attributes

Numeric attributes are represented by numbers and have most of the properties of numbers. Numeric attributes are also called quantitative attributes. These attributes can be either integer-valued or continuous.

a. Interval Scaled attribute

Interval scaled attribute are the values which are measured in a scale of equal size units between adjacent categories. For example temperature in Celsius scale. So if we take temperature in degrees Celsius the difference between 78 degrees and 79 degrees or that one degree difference is the same as the difference between 45 degrees and 46 degrees.

b. Ratio Scaled attributes

The ratio scaled attributes have meaningful intervals and also a true zero point. True zero point means the absence of the property like 0 inches means there is no length. For example weight in KG and here 10 KG is twice as much as five KG (i.e. $10 / 5$ is 2 or 2 times as much). So that's a meaningful ratio and in addition to this zero pounds means no weight or an absence of weight so there's a true zero point.

| Attribute type | | Description | Example | Properties |
|------------------------------|---------|---|---|------------------------|
| Categorical (Qualitative) | Nominal | The values of nominal attributes are just different symbol or name of things | zip codes, employee ID numbers, gender | Distinctness |
| | Ordinal | The values of ordinal attributes are possible values that have meaningful order or ranking among them | order of finishing a race, grades in exam | Distinctness and order |

| | | | | |
|---------------------------|----------|--|--|--|
| Numeric (Quantitative) | Interval | Interval scaled attributes are measured on a scale of equal size units | calendar dates, temperature in Celsius or Fahrenheit | Distinctness, order and Addition |
| | Ratio | The values of ratio scaled attributes has a true zero point | frequency of words in a document | Distinctness, order, addition and multiplication |

2.2 Data Pre-processing

Data in the real world is dirty due to their typically huge size and possible origin from multiple, heterogeneous sources. In other words, the data we wish to analyze by data mining techniques are incomplete (lacking attribute values or certain attributes of interest, or containing only aggregate data), noisy (containing errors, or outlier values that deviate from the expected), and inconsistent (e.g., containing discrepancies in the department codes used to categorize items). Data pre-processing is required to handle these problems about real world data.

Major Tasks in Data Preprocessing

1. Data cleaning
2. Data integration
3. Data transformation
4. Data reduction
5. Data discretization

1. Data cleaning

Data cleaning routines work to “clean” the data by filling in missing values, smoothing noisy data, identifying or removing outliers, and resolving inconsistencies. If users believe the data are dirty, they are unlikely to trust the results of any data mining that has been applied to it. Furthermore, dirty data can cause confusion for the mining procedure, resulting in unreliable output. Although most mining routines have some procedures for dealing with incomplete or noisy data, they are not always strong. Therefore, a useful preprocessing step is to run the data through some data cleaning routines.

❖ Missing Data

Many tuples have no recorded value for several attributes. The values may be missing due to:

- Equipment malfunctioning
- Inconsistent with other data and hence deleted
- Not entered due to misunderstanding
- Certain data may be considered unimportant while entering.

❖ Ways to handle missing data

- *Ignore the tuple*: This is usually done when the class label is. This method is not very effective, unless the tuple contains several attributes with missing values.
- *Fill in the missing value manually*: In general, this approach is time-consuming and may not be feasible given a large data set with many missing values.
- **Fill missing values automatically with:**
 - *Global constant*: Replace all missing attribute values by the same constant, such as a label like “Unknown”.
 - *Attribute mean*: For example, suppose that the average income of customers is Rs.16,000. Use this value to replace the missing value for income.
 - *Most probable value*: use inference-based techniques such as Bayesian formula or decision tree.

❖ Noisy data

Noise is a random error or variance in a measured variable. For a numerical attribute such as, price, we can “smooth” out the data to remove the noise by following techniques:

a. Binning

Binning methods smooth a sorted data value by consulting its “neighborhood,” that is, the values around it. The sorted values are distributed into a number of “buckets,” or bins. Because binning methods consult the neighborhood of values, they perform local smoothing.

For example,

Let us consider following data for price: 4, 15, 28, 34, 8, 21, 9, 21, 29, 26, 24, 25

First, Sort the data for price (in dollars) as: 4, 8, 9, 15, 21, 21, 24, 25, 26, 28, 29, 34

Partition into equal-frequency bins:

Here, number of Bins=3, so, each Bin has $12/3=4$ items.

Bin 1: 4, 8, 9, 15

Bin 2: 21, 21, 24, 25

Bin 3: 26, 28, 29, 34

i Smoothing by bin means:

Bin 1: 9, 9, 9, 9

Bin 2: 23, 23, 23, 23

Bin 3: 29, 29, 29, 29

ii Smoothing by bin boundaries:

Bin 1: 4, 4, 4, 15

Bin 2: 21, 21, 25, 25

Bin 3: 26, 26, 26, 34

b. Regression:

Data can be smoothed by fitting the data to a function, such as with regression. Linear regression involves finding the “best” line to fit two attributes (or variables), so that one attribute can be used to predict the other. Multiple linear regression is an extension of linear regression, where more than two attributes are involved and the data are fit to a multidimensional surface.

c. Clustering:

Outliers may be detected by clustering, where similar values are organized into groups, or “clusters.” Intuitively, values that fall outside of the set of clusters may be considered outliers.

2. Data Integration

Data integration combines data from multiple sources into a coherent data store, as in data warehousing. These sources may include multiple databases, data cubes, or flat files.

There are a number of issues to consider during data integration.

- *Entity identification problem:* The entity identification problem arises when different sources have different values for same attributes. For example, customer id in one database and cust_number in another may refer to the same attribute.
- *Data redundancy:* Data is said to be redundant if same data appears in various locations. Inconsistencies in attribute or dimension naming can cause redundancies in the resulting data set.

The data redundancy problem can be handled by correlation analysis. Given two attributes, such analysis can measure how strongly one attribute implies the other, based on the available data.

3. Data transformation

In data transformation, the data are transformed or combined into forms appropriate for mining. Data transformation can involve the following:

- i **Smoothing:** which works to remove noise from the data. Such techniques include binning, regression, and clustering.
- ii **Aggregation:** summary or aggregation operations are applied to the data. For example, the daily sales data may be aggregated so as to compute monthly and annual total amounts.
- iii **Generalization:** low-level or “primitive” (raw) data are replaced by higher-level concepts through the use of concept hierarchies. For example, values for numerical attributes, like age, may be mapped to higher-level concepts, like youth, middle-aged, and senior.
- iv **Normalization:** the attribute data are scaled so as to fall within a small specified range, such as -1.0 to 1.0 or 0.0 to 1.0.

a. Min-max normalization:

Suppose that \min_A and \max_A are the minimum and maximum values of an attribute, A. Min-max normalization maps a value, v , of A to v' in the range $[\text{new_min}_A, \text{new_max}_A]$ by computing

$$v' = \frac{v - \min_A}{\max_A - \min_A} (\text{new_max}_A - \text{new_min}_A) + \text{new_min}_A$$

Eg. Let the income range is 12000 to 98000 be normalized to $[0.0, 1.0]$. Then 73600 is normalized to:

$$\frac{73600 - 12000}{98000 - 12000} (1.0 - 0) + 0 = 0.716$$

b. z-score normalization:

In z-score normalization (or zero-mean normalization), the values for an attribute, A, are normalized based on the mean and standard deviation of A. A value, v , of A is normalized to v' by computing

$$v' = \frac{v - \text{mean}_A}{\sigma_A}$$

Eg. Let $\text{mean} = 54000$ and $\text{Standard Deviation } (\sigma) = 16000$. Then 73600 is mapped to

$$\frac{73600 - 54000}{16000} = 1.225$$

c. Normalization by decimal scaling:

It normalizes the value by moving the decimal point of values of attribute A. The number of decimal points moved depends on the maximum absolute value of A. A value, v , of A is normalized to v' by computing

$$v' = \frac{v}{10^j}$$

Where j is the smallest integer such that $\text{Max}(|v'|) < 1$.

Eg. Suppose that the recorded values of A ranges from -986 to 917. The maximum absolute value of A is 986. To normalize by decimal scaling, we therefore divide each value by 1,000 (i.e., $j = 3$) so that -986 normalizes to -0.986 and 917 normalizes to 0.917.

4. Data reduction


Data reduction techniques can be applied to obtain a reduced representation of the data set that is much smaller in volume, yet closely maintains the integrity of the original data. That is, mining on the reduced data set should be more efficient yet produce the same (or almost the same) analytical results.

Strategies for data cube reduction:**a. Data cube aggregation:**

Aggregation operation is performed to combine two or more objects into a single object. For example the data consist of the sales per quarter, for the years 2002 to 2004. If the interest of user

is in the annual sales (total per year), rather than the total per quarter. The data can be aggregated so that the resulting data summarize the total sales per year instead of per quarter. The resulting data set is smaller in volume, without loss of information necessary for the analysis task.

| 2002 | | 2003 | | 2004 | |
|---------|--------|---------|--------|---------|--------|
| Quarter | Sales | Quarter | Sales | Quarter | Sales |
| Q1 | 50,000 | Q1 | 45,000 | Q1 | 50,000 |
| Q2 | 25,000 | Q2 | 40,000 | Q2 | 45,000 |
| Q3 | 15,000 | Q3 | 15,000 | Q3 | 40,000 |
| Q4 | 20,000 | Q4 | 25,000 | Q4 | 35,000 |



| year | Sales |
|------|---------|
| 2002 | 110,000 |
| 2003 | 125,000 |
| 2004 | 170,000 |

b. Dimensionality reduction:

In dimensionality reduction, data encoding or transformations are applied so as to obtain a reduced or “compressed” representation of the original data.

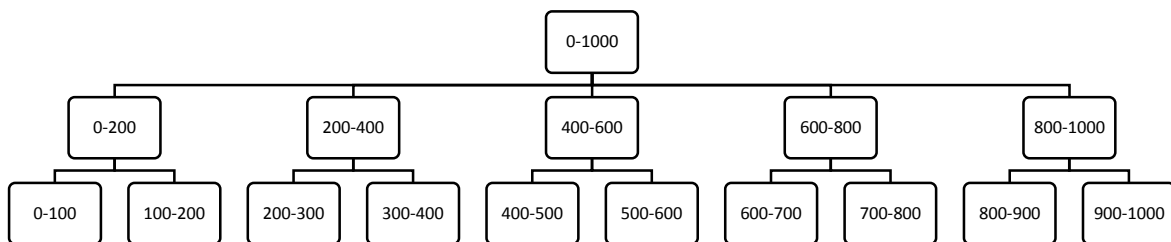
c. Data compression:

Generally done for space optimization on mostly multimedia data.

5. Data discretization

Raw data values for attributes are replaced by ranges or higher conceptual levels. Data discretization is a form of numerosity reduction that is very useful for the automatic generation of concept hierarchies.

Discretization and concept hierarchy generation are powerful tools for data mining, in that they allow the mining of data at multiple levels of abstraction.

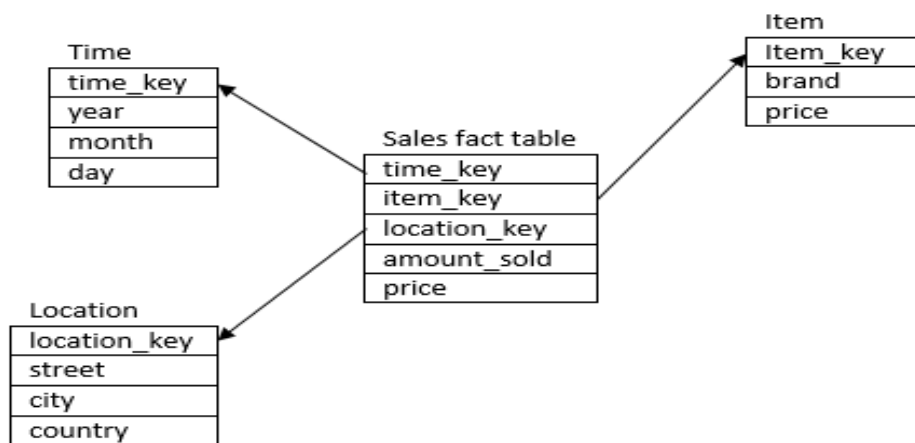


2.3 OLAP

Schemas for Multidimensional Databases: Stars, Snowflakes, and Fact Constellations

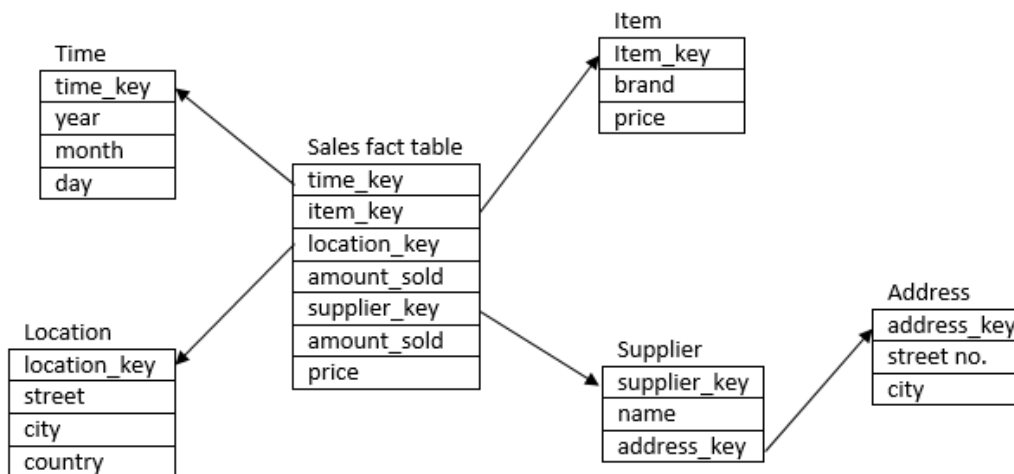
a. Star Schema

The most common modeling paradigm is the star schema, in which the data warehouse contains a large central table (fact table) containing the bulk of the data with no redundancy, and 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.



b. Snowflake Schema

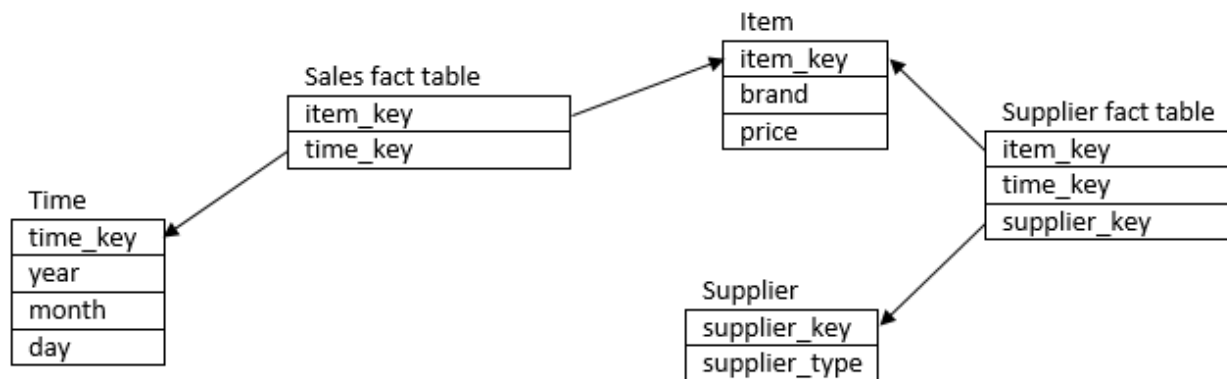
The snowflake schema is a variant of the star schema model, where some dimension tables are *normalized*, thereby further splitting the data into additional tables. The resulting schema graph forms a shape similar to a snowflake.



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 saving of space 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.

c. Fact constellation schema

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.



OLAP

OLAP (online analytical processing) is a term used to describe the analysis of complex data from the data warehouse. The analysis often requires resource intensive aggregations processing and therefore it becomes necessary to implement a special database (e.g. data warehouse) to improve OLAP response time. In the hands of skilled knowledge workers, OLAP tools use distributed computing capabilities for analyses that require more storage and processing power than can be economically and efficiently located on an individual desktop. OLAP has two immediate

consequences: online part requires the answers of queries to be fast, and the analytical part is a hint that the queries itself are complex. So, OLAP are **Complex questions with Fast Answers**.

Definition: *OLAP is the dynamic enterprise analysis required to create, manipulate, animate and synthesize information from exegetical, contemplative and formulaic data analysis models.*

Essentially what this definition means is that the information is manipulated from the point of view of a manager (exegetical), from the point of view of someone who has thought about it (contemplative) and according to some formula (formulaic).

An even simpler definition is that OLAP is a fast analysis of shared multidimensional information for advanced analysis. Or FASMI.

Types of **OLAP**

i Relational OLAP (ROLAP)

These are the intermediate servers that stand in between a relational back-end server and client front-end tools. It provides functionality by using relational databases and relational query tools to store and analyze multidimensional data. This model permits multidimensional analysis of data as this enables users to perform a function equivalent to that of the traditional OLAP slicing and dicing feature.

One advantage of ROLAP over the other styles of OLAP analytic tools is that it is deemed to be more scalable in handling huge amounts of data. ROLAP sits on top of relational databases therefore enabling it to leverage several functionalities that a relational database is capable of. Another gain of a ROLAP tool is that it is efficient in managing both numeric and textual data. It also permits users to “drill down” to the leaf details or the lowest level of a hierarchy structure. However, ROLAP applications display a slower performance as compared to other style of OLAP tools since, oftentimes, calculations are performed inside the server. Another demerit of a ROLAP tool is that as it is dependent on use of SQL for data manipulation, it may not be ideal for performance of some calculations that are not easily translatable into an SQL query.

ii Multidimensional OLAP (MOLAP)

MOLAP uses array-based multidimensional storage engines for multidimensional views of data. With multidimensional data stores, the storage utilization may be low if the data set is sparse. One

of the major distinctions of MOLAP against a ROLAP tool is that data are pre-summarized and are stored in an optimized format in a multidimensional cube, instead of in a relational database. This is OLAP tool is efficient to use in making analysis reports since this enables users to easily reorganize or rotate the cube structure to view different aspects of data. This is done by way of slicing and dicing. MOLAP analytic tool are also capable of performing complex calculations. Since calculations are predefined upon cube creation, this results in the faster return of computed data. MOLAP systems also provide users the ability to quickly write back data into a data set. Moreover, in comparison to ROLAP, MOLAP is considerably less heavy on hardware due to compression techniques. In a nutshell, MOLAP is more optimized for fast query performance and retrieval of summarized information.

There are certain limitations to implementation of a MOLAP system, one primary weakness of which is that MOLAP tool is less scalable than a ROLAP tool as the former is capable of handling only a limited amount of data. The MOLAP approach also introduces data redundancy. There are also certain MOLAP products that encounter difficulty in updating models with dimensions with very high cardinality.

iii Hybrid OLAP (HOLAP)

HOLAP is the product of the attempt to incorporate the best features of MOLAP and ROLAP into a single architecture. This tool tried to bridge the technology gap of both products by enabling access or use to both multidimensional database (MDDb) and Relational Database Management System (RDBMS) data stores. HOLAP systems stores larger quantities of detailed data in the relational tables while the aggregations are stored in the pre-calculated cubes. HOLAP also has the capacity to “drill through” from the cube down to the relational tables for delineated data.

Some of the advantages of this system are better scalability, quick data processing and flexibility in accessing of data sources.

OLTP

OLTP stands for OnLine Transactional Processing. OLTP involves short, simple, frequent queries and/or modifications, each involving a small number of tuples. For example in a banking system cover most of the day-to-day operations, such as purchasing, inventory, manufacturing, banking, payroll, registration, and accounting.

The differences between OLTP and OLAP are as follows:

1. Users: OLTP systems are designed for office workers while the OLAP systems are designed for Decision makers. Therefore while an OLTP system may be accessed by hundreds or even thousands of users in a large enterprise, an OLAP system is likely to be accessed only by a select group of managers and may be used only by dozens of users.

2. Functions: OLTP systems are mission-critical. They support day-to-day operations of an enterprise and are mostly performance and availability driven. These systems carry out simple repetitive operations. OLAP systems are management-critical to support decision of an enterprise support functions using analytical investigations. They are more functionality driven. These are often much more complex operations.

3. Nature: Although SQL queries often return a set of records, OLTP systems are designed to process one record at a time, for example a record related to the customer who might be on the phone or in the store. OLAP systems are not designed to deal with individual customer records. Instead they involve queries that deal with many records at a time and provide summary or aggregate data to a manager. OLAP applications involve data stored in a data warehouse that has been extracted from many tables and perhaps from more than one enterprise database.

4. Design: OLTP database systems are designed to be application-oriented while OLAP systems are designed to be subject-oriented. OLTP systems view the enterprise data as a collection of tables (perhaps based on an entity-relationship model). OLAP systems view enterprise information as multidimensional).

5. Data: OLTP systems normally deal only with the current status of information. For example, information about an employee who left three years ago may not be available on the Human Resources System. The old information may have been achieved on some type of stable storage media and may not be accessible online. On the other hand, OLAP systems require historical data over several years since trends are often important in decision making.

6. Kind of use: OLTP systems are used for reading and writing operations while OLAP systems normally do not update the data.

2.4 Characteristics of OLAP Systems

FASMI Characteristics

In the FASMI characteristics of OLAP systems, the name derived from the first letters of the characteristics are:

Fast: As noted earlier, most OLAP queries should be answered very quickly, perhaps within seconds. The performance of an OLAP system has to be like that of a search engine. Achieving such performance is difficult. The data structures must be efficient. The hardware must be powerful enough for the amount of data and the number of users. Full pre-computation of aggregates helps but is often not practical due to the large number of aggregates. One approach is to pre-compute the most commonly queried aggregates and compute the remaining on-the-fly.

Analytic: An OLAP system must provide rich analytic functionality. The system should be able to cope with any relevant queries for the application and the user.

Shared: An OLAP system is shared resource likely to be shared by a few business analyst and hundreds of users. Being a shared system, an OLAP system should be provide adequate security for confidentiality as well as integrity.

Multidimensional: This is the basic requirement. Whatever OLAP software is being used, it must provide a multidimensional conceptual view of the data. It is because of the multidimensional view of data that we often refer to the data as a cube. A dimension often has hierarchies that show parent-child relationships between the members of a dimension. The multidimensional structure should allow such hierarchies.

Information: OLAP systems usually obtain information from a data warehouse. The system should be able to handle a large amount of input data. The capacity of an OLAP system to handle information and its integration with the data warehouse may be critical.

Codd's OLAP Characteristics

Codd et al's 1993 paper listed 12 characteristics (or rules) OLAP systems. Another six in 1995 followed these. Codd restructured the 18 rules into four groups.

Here we discuss 10 characteristics that are most important.

1. Multidimensional conceptual view: This is central characteristic of an OLAP system. By requiring a multidimensional view, it is possible to carry out operations like slice and dice.

- 2. Accessibility (OLAP as a mediator):** The OLAP software should be sitting between data sources (e.g. data warehouse) and an OLAP front-end.
- 3. Batch extraction vs. interpretive:** An OLAP system should provide multidimensional data staging plus precomputation of aggregates in large multidimensional databases.
- 4. Multi-user support:** Since the OLAP system is shared, the OLAP software should provide many normal database operations including retrieval, concurrency control, integrity and security.
- 5. Storing OLAP results:** OLAP results data should be kept separate from source data. Read-write OLAP applications should not be implemented directly on live transaction data if OLAP source systems are supplying information to the OLAP system directly.
- 6. Extraction of missing values:** The OLAP system should distinguish missing values from zero values. A large data cube may have a large number of zeros as well as some missing values. If a distinction is not made between zero values and missing values, the aggregates are likely to be computed incorrectly.
- 7. Treatment of missing values:** An OLAP system should ignore all missing values regardless of their source. Correct aggregate values will be computed once the missing values are ignored.
- 8. Uniform reporting performance:** Increasing the number of dimensions or database size should not significantly degrade the reporting performance of the OLAP system. This is a good objective although it may be difficult to achieve in practice.
- 9. Generic dimensionality:** An OLAP system should treat each dimension as equivalent in both structure and operational capabilities. Additional operational capabilities may be granted to selected dimensions but such additional functions should be grantable to any dimension.
- 10. Unlimited dimensions and aggregation levels:** An OLAP system should allow unlimited dimensions and aggregation levels. In practice, the number of dimensions is rarely more than 10 and the number of hierarchies rarely more than six.

2.5 Multidimensional view and data cube

“What is a data cube?” A data cube allows data to be modeled and viewed in multiple dimensions. It is defined by dimensions and facts.

In general terms, **dimensions** are the perspectives or entities with respect to which an organization wants to keep records. For example, an organization, *AllElectronics* may create a *sales* data warehouse in order to keep records of the store’s sales with respect to the dimensions *time*, *item*,

branch, and *location*. 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. For example, a dimension table for *item* may contain the attributes *item_name*, *brand*, and *type*.

The multidimensional storage model involves two types of tables: dimension tables and fact tables. A **dimension table** consists of tuples of attributes of the dimension. **Fact table** is usually a tabular representation of the data. The **fact table** contains the names of the *facts*, or measures, as well as keys to each of the related dimension tables.

Examples of facts for a sales data warehouse include *dollars sold* (sales amount in dollars), *units sold* (number of units sold), and *amount budgeted*.

Although we usually think of cubes as 3-D geometric structures, in data warehousing the data cube is *n*-dimensional.

To understand what a 3-D data cube, let's start by looking at a simple 2-D data cube that is, in fact, a table or spreadsheet for sales data from *abcDrinks* enterprise. In particular, we will look at the *AbcDrinks* sales data for items sold per month in the city of Lalitpur. These data are shown in following table:

| Location= Lalitpur | | | | |
|--------------------|-------|------|-------|--------|
| | Items | | | |
| Time(quarter) | | Coke | Fanta | Sprite |
| | Q1 | 15 | 20 | 10 |
| | Q2 | 20 | 15 | 5 |
| | Q3 | 15 | 10 | 20 |
| | Q4 | 5 | 20 | 15 |

In this 2-D representation, the sales for Lalitpur are shown with respect to the *time* dimension (organized organized to quarters) and the *item* dimension (organized according to the product of items sold). The fact or measure displayed is *number of items*. Now, suppose that we would like to view the sales data with a third dimension. For instance, suppose we would like to view the data according to *time* and *item*, as well as *location* for the cities Kathmandu, Lalitpur, and Kavre. These 3-D data are shown in following table:

| | Location= Lalitpur | | | | Location= Kathmandu | | | Location=Kavre | | |
|---------------|--------------------|------|-------|--------|---------------------|-------|--------|----------------|-------|--------|
| | Items | | | | Items | | | Items | | |
| Time(Quarter) | | Coke | Fanta | Sprite | Coke | Fanta | Sprite | Coke | Fanta | Sprite |
| | Q1 | 15 | 20 | 10 | 7 | 10 | 20 | 5 | 20 | 10 |
| | Q2 | 20 | 15 | 5 | 20 | 15 | 25 | 25 | 15 | 10 |
| | Q3 | 15 | 10 | 20 | 3 | 10 | 20 | 7 | 5 | 5 |
| | Q4 | 5 | 20 | 15 | 15 | 10 | 20 | 5 | 25 | 10 |

Conceptually, we may also represent the same data in the form of a 3-D data cube, as in following figure:

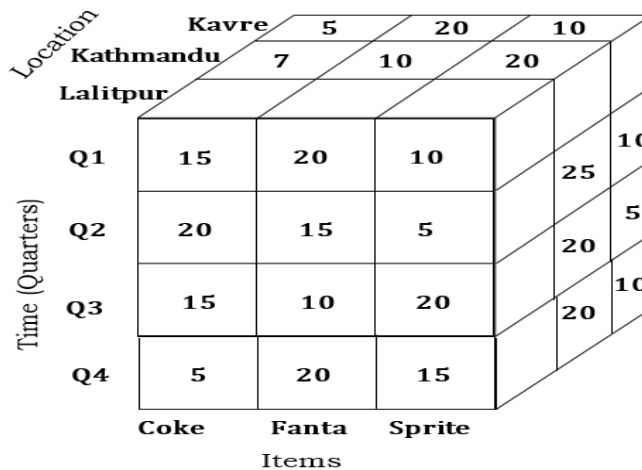


Figure 1. 3-D data cube representation of the data according to the dimensions time, items and location

In a cube with n dimensions, there are 2^n possible aggregation operations. So, in the three-dimensional cube, the following eight types of aggregations or queries are possible:

1. null [ALL, ALL, ALL]
e.g. How many items have been sold in total?
One possible query
2. time [a, ALL, ALL]
e.g. How many item have been sold in Q1?
Four possible query
3. items [ALL, b, ALL]
e.g. How many sprite have been sold?
three possible query
4. location [ALL, ALL, c]
e.g. How many items have been sold in Kathmandu?
three possible query

5. time, item [a, b, ALL]
e.g. How many sprite sold in Q4?
12 possible queries. (4*3*1).
6. Location, time [a, ALL, c]
e.g. How many items sold in Lalitpur in Q3?
12 possible queries
7. Item, location [ALL, b, c]
e.g. How many sprite sold in Lalitpur?
9 possible queries.
8. all [a, b, c]
e.g. How many Fanta sold in Kavre in Q3?
36 possible queries.

The aggregate operations can be represented in a lattice of cuboids. 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.

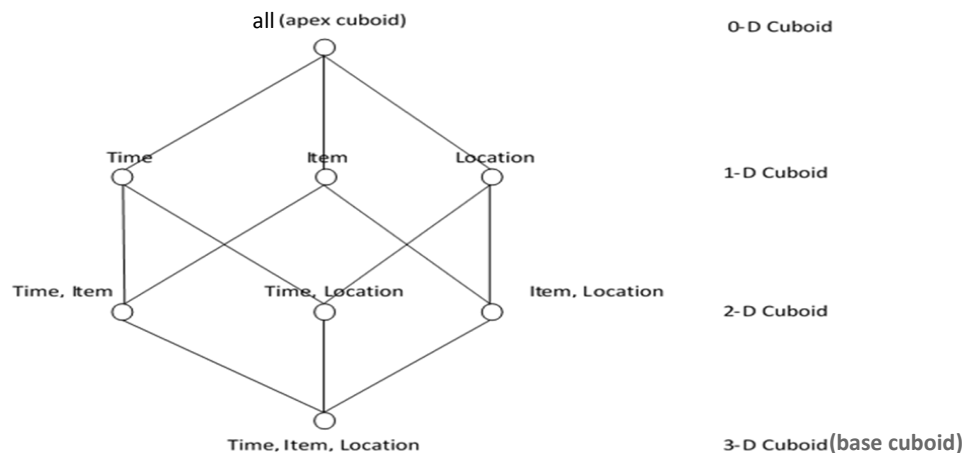


Figure 2. Lattice of cuboids, making up a 3-D data cube. Each cuboid represents a different group-by.

Figure 2 is a 3-D cuboid for *month*, *item*, and *location*, summarized for all suppliers. The 0-D cuboid, which holds the highest level of summarization, is called the apex cuboid. The cuboid that holds the lowest level of summarization is called the base cuboid. The apex cuboid is typically denoted by all.

Suppose that we would now like to view our sales data with an additional fourth dimension, such as *supplier*. Viewing things in 4-D becomes tricky. However, we can think of a 4-D cube as being

a series of 3-D cubes, as shown in following figure. If we continue in this way, we may display any n -D data as a series of $(n-1)$ -D “cubes.” The data cube is a metaphor for multidimensional data storage.

The important thing to remember is that data cubes are n -dimensional and do not confine data to 3-D.

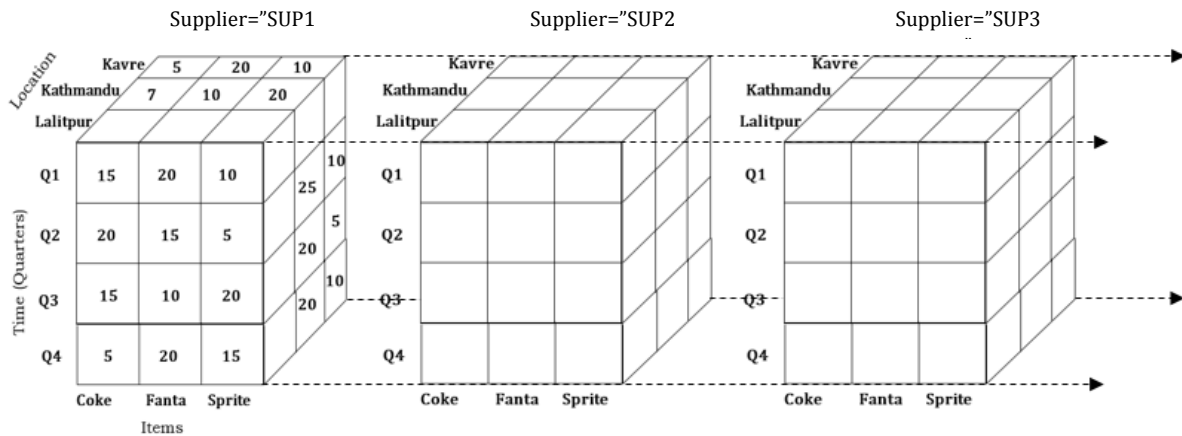


Figure 3. A 4-D data cube representation of sales data, according to the dimensions time, item, location and supplier

Figure 4 shows a lattice of cuboids forming a data cube for the dimensions *month*, *item*, *location*, and *supplier*

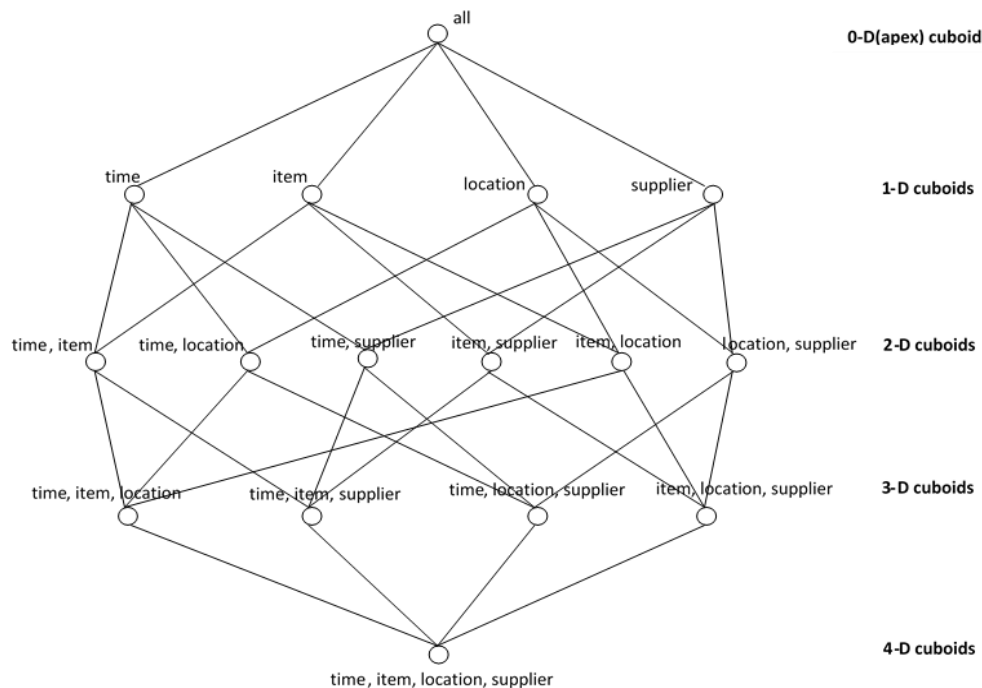


Figure 4. Lattice of cuboids, making up a 4-D data cube for the dimensions time, item, location, and supplier

2.6 Data Cube implementation

i. Full Materialization (Pre compute and store all)

This means that millions of aggregates will need to be computed and stored. Although this is the best solution as far as query response time is concerned, the solution is impractical since resources required to compute the aggregates and to store them will be prohibitively large for a large data cube.

ii. No Materialization (Pre compute none)

This means that the aggregates are computed on the fly using the raw data whenever a query is posed. This approach does not require additional space for storing the cube but the query response time is likely to be very poor for large data cubes.

iii. Partial Materialization (Pre compute and store some)

This means that we pre-compute and store the most frequently queried aggregates and compute others as the need arises. We may also be able to derive some of the remaining aggregates using the aggregates that have already been computed. It may therefore be worthwhile also to pre compute some aggregates that are not most frequently queried but help in deriving many other aggregates. It will of course not be possible to derive all the aggregates from the pre-computed aggregates and it will be necessary to access the database to compute the remaining aggregates. The more aggregates we are able to pre-compute the better the query performance.

2.7 Data Cube Operations

Hierarchies:

A concept hierarchy defines a sequence of mappings from a set of low-level concepts to higher-level, more general concepts. For example Street→City→State→Country or Day < Month < Quarter < Year. Climbing up a hierarchy means reducing the dimension.

“How are concept hierarchies useful in OLAP?” In the multidimensional model, data are organized into multiple dimensions, and each dimension contains multiple levels of abstraction defined by concept hierarchies. This organization provides users with the flexibility to view data from different perspectives.

OLAP Operations

Following are the major OLAP operations or data cube operations:

i. Roll up

The roll-up operation (also called the *drill-up* operation by some vendors) performs aggregation on a data cube, either by *climbing up a concept hierarchy* for a dimension or by *dimension reduction*. Figure 5 shows the result of a roll-up operation performed on the central cube by climbing up the concept hierarchy for *location*.

ii. Drill down

Drill-down is the reverse of roll-up. It 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 in figure 5 occurs by descending the *time* hierarchy from the level of *quarter* to the more detailed level of *month*.

iii. Slice

The *slice* operation performs a selection on one dimension of the given cube, resulting in a subcube. Figure 5 shows a slice operation where the sales data are selected from the central cube for the dimension *time* using the criterion *time* = “Q1”.

iv. Dice

The *dice* operation defines a subcube by performing a selection on two or more dimensions. Figure 5 shows a 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”).

v. Pivot (rotate):

Pivot (also called *rotate*) is a visualization operation that rotates the data axes in view in order to provide an alternative presentation of the data. Figure 5 shows a pivot operation where the *item* and *location* axes in a 2-D slice are rotated. We can also perform the rotation in 3-D cube.

OLAP can generate summarizations, aggregations, and hierarchies at each granularity level and at every dimension intersection. OLAP also supports functional models for forecasting, trend analysis, and statistical analysis. In this context, an OLAP engine is a powerful data analysis tool.

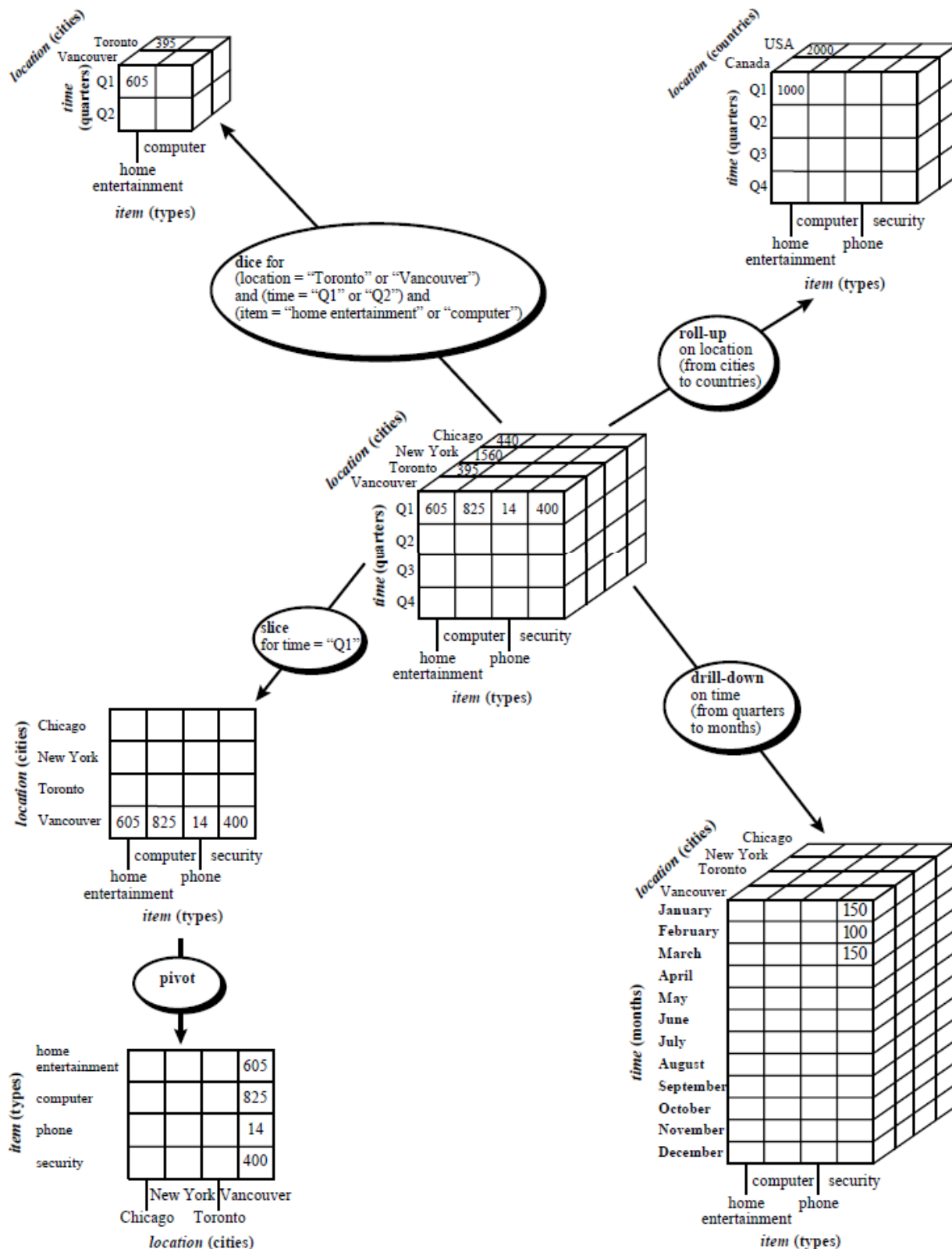


Figure 5 Examples of typical OLAP operations on multidimensional data.

2.8 Implementation guidelines for OLAP operations

Following are a number of guidelines for successful implementation of OLAP. The guidelines are, somewhat similar to those presented for data warehouse implementation.

1. ***Vision:*** The OLAP team must, in consultation with the users, develop a clear vision for the OLAP system. This vision including the business objectives should be clearly defined, understood, and shared by the stakeholders.
2. ***Senior management support:*** The OLAP project should be fully supported by the senior managers. Since a data warehouse may have been developed already, this should not be difficult.
3. ***Selecting an OLAP tool:*** The OLAP team should familiarize themselves with the ROLAP and MOLAP tools available in the market. Since tools are quite different, careful planning may be required in selecting a tool that is appropriate for the enterprise. In some situations, a combination of ROLAP and MOLAP may be most effective.
4. ***Corporate strategy:*** The OLAP strategy should fit in with the enterprise strategy and business objectives. A good fit will result in the OLAP tools being used more widely.
5. ***Focus on the users:*** The OLAP project should be focused on the users. Users should, in consultation with the technical professional, decide what tasks will be done first and what will be done later. Attempts should be made to provide each user with a tool suitable for that person's skill level and information needs. A good GUI user interface should be provided to non-technical users. The project can only be successful with the full support of the users.
6. ***Joint management:*** The OLAP project must be managed by both the IT and business professionals. Many other people should be involved in supplying ideas. An appropriate committee structure may be necessary to channel these ideas.
7. ***Review and adapt:*** Organizations evolve and so must the OLAP systems. Regular reviews of the project may be required to ensure that the project is meeting the current needs of the enterprise.