**Data Warehousing Concepts:**

**Data warehouse**

A data warehouse is constructed by integrating data from multiple heterogeneous sources. It supports analytical reporting, structured and/or ad hoc queries and decision making

**Understanding a Data Warehouse**

* A data warehouse is a database, which is kept separate from the organization's operational database.
* There is no frequent updating done in a data warehouse.
* A data warehouse system helps in consolidated historical data analysis.

**Data Warehouse Features**

* Subject Oriented
* Integrated
* Time Variant
* Non-volatile

**Data Warehouse Applications**

* Financial services
* Banking services
* Consumer goods
* Retail sectors
* Controlled manufacturing

**Types of Data Warehouse**

* **Information Processing** - A data warehouse allows to process the data stored in it. The data can be processed by means of querying, basic statistical analysis, reporting using crosstabs, tables, charts, or graphs.
* **Analytical Processing** - A data warehouse supports analytical processing of the information stored in it. The data can be analyzed by means of basic OLAP operations, including slice-and-dice, drill down, drill up, and pivoting.
* **Data Mining** - Data mining supports knowledge discovery by finding hidden patterns and associations, constructing analytical models, performing classification and prediction. These mining results can be presented using the visualization tools.

**Data warehousing**

Data warehousing is the process of constructing and using a data warehouse.

**Metadata**

Metadata is simply defined as data about data. The data that are used to represent other data is known as metadata. For example, the index of a book serves as a metadata for the contents in the book. In other words, we can say that metadata is the summarized data that leads us to the detailed data.

**Dimension**:

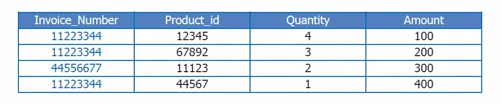
Dimensions describe the objects involved in a business intelligence effort. While facts correspond to events, dimensions correspond to people, items, or other objects. For example, in the retail scenario, we discussed that purchases, returns, and calls are facts. On the other hand, customers, employees, items and stores are dimensions and should be contained in dimension tables.

**Conformed Dimension-**This is used in multiple locations. It helps in creating consistency so that the same can be maintained across the fact tables. Different tables can use the dimension table across the fact table and it can help in creating different reports.

For example, there are two fact tables. Fact table 1 is to determine the number of products sold by geography. This table will calculate just the number of products by geography and fact table 2 will determine the revenue generated by customer. Both are dependent on the product which contains product Id, name and source.

There is the geography dimension and customer dimension which are being shared by two fact tables. The revenue fact gives the revenue generated by both the geography and the customer, while the product units fact gives number of units sold in the geography to a customer.

**Degenerate Dimension**– A degenerate dimension is when the dimension attribute is stored as part of the fact table and not in a separate dimension table. Product id comes from product dimension table. Invoice number is a standalone attribute and has no other attributes associated with it. An invoice number can be crucial since the business would want to know the quantity of the products.



**Junk Dimension**– A junk dimension is a single table with a combination of different and unrelated attributes to avoid having a large number of foreign keys in the fact table. They are often created to manage the foreign keys created by rapidly changing dimensions.

**Role play dimension**– It is a dimension table that has multiple valid relationships with a fact table. For example, a fact table may include foreign keys for both ship date and delivery date. But the same dimension attributes apply to each foreign key so the same dimension tables can be joined to the foreign keys.

**Types of Facts**

There are three types of facts:

* **Additive**: Additive facts are facts that can be summed up through all of the dimensions in the fact table.
* **Semi-Additive**: Semi-additive facts are facts that can be summed up for some of the dimensions in the fact table, but not the others.
* **Non-Additive**: Non-additive facts are facts that cannot be summed up for any of the dimensions present in the fact table.

Let us use examples to illustrate each of the three types of facts. The first example assumes that we are a retailer, and we have a fact table with the following columns:

|  |
| --- |
| Date |
| Store |
| Product |
| Sales\_Amount |

The purpose of this table is to record the sales amount for each product in each store on a daily basis.Sales\_Amount is the fact. In this case, Sales\_Amount is an additive fact, because you can sum up this fact along any of the three dimensions present in the fact table -- date, store, and product. For example, the sum of Sales\_Amount for all 7 days in a week represents the total sales amount for that week.

Say we are a bank with the following fact table:

|  |
| --- |
| Date |
| Account |
| Current\_Balance |
| Profit\_Margin |

The purpose of this table is to record the current balance for each account at the end of each day, as well as the profit margin for each account for each day. Current\_Balance and Profit\_Margin are the facts.Current\_Balance is a semi-additive fact, as it makes sense to add them up for all accounts (what's the total current balance for all accounts in the bank?), but it does not make sense to add them up through time (adding up all current balances for a given account for each day of the month does not give us any useful information). Profit Margin is a non-additive fact, for it does not make sense to add them up for the account level or the day level.

Types of Fact Tables

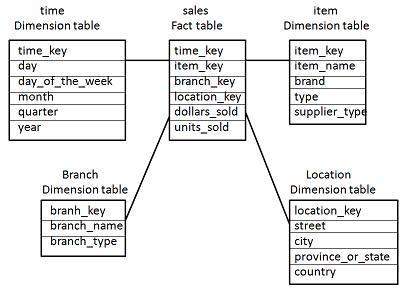
Based on the above classifications, there are two types of fact tables:

* **Cumulative**: This type of fact table describes what has happened over a period of time. For example, this fact table may describe the total sales by product by store by day. The facts for this type of fact tables are mostly additive facts. The first example presented here is a cumulative fact table.
* **Snapshot**: This type of fact table describes the state of things in a particular instance of time, and usually includes more semi-additive and non-additive facts. The second example presented here is a snapshot fact table.

**Factless Fact table**

**Star Schema**

* Each dimension in a star schema is represented with only one-dimension table.
* This dimension table contains the set of attributes.
* The following diagram shows the sales data of a company with respect to the four dimensions, namely time, item, branch, and location.

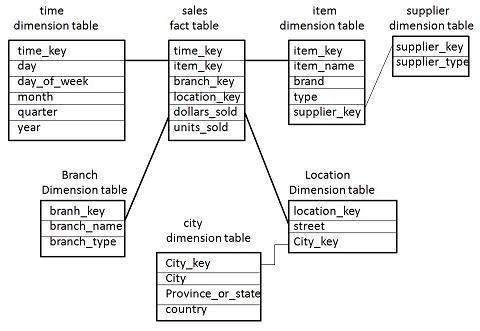


* There is a fact table at the center. It contains the keys to each of four dimensions.
* The fact table also contains the attributes, namely dollars sold and units sold.

**Note**: Each dimension has only one dimension table and each table holds a set of attributes. For example, the location dimension table contains the attribute set {location\_key, street, city, province\_or\_state,country}. This constraint may cause data redundancy. For example, "Vancouver" and "Victoria" both the cities are in the Canadian province of British Columbia. The entries for such cities may cause data redundancy along the attributes province\_or\_state and country.

**Snowflake Schema**

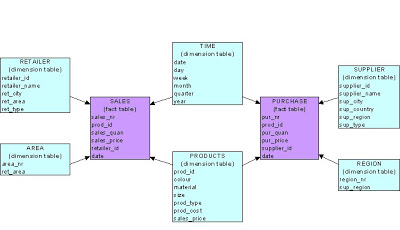
* Some dimension tables in the Snowflake schema are normalized.
* The normalization splits up the data into additional tables.
* Unlike Star schema, the dimensions table in a snowflake schema are normalized. For example, the item dimension table in star schema is normalized and split into two dimension tables, namely item and supplier table.



* Now the item dimension table contains the attributes item\_key, item\_name, type, brand, and supplier-key.
* The supplier key is linked to the supplier dimension table. The supplier dimension table contains the attributes supplier\_key and supplier\_type.

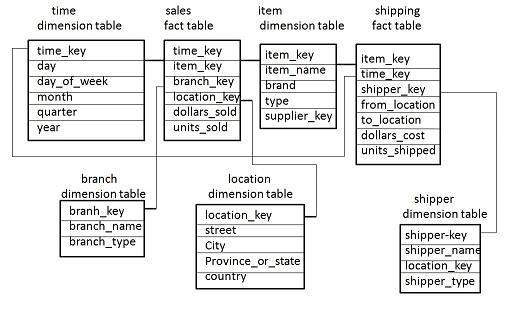
**Note**: Due to normalization in the Snowflake schema, the redundancy is reduced and therefore, it becomes easy to maintain and the save storage space.

**Galaxy Schema:**  
Galaxy schema contains many fact tables with some common dimensions (conformed dimensions). This schema is a combination of many data marts.

[](http://1.bp.blogspot.com/_pjSOGJIjDMo/S1w_SAaqJBI/AAAAAAAAADY/_kYiR3xwbCQ/s1600-h/galaxy.bmp)

**Fact Constellation Schema**

* A fact constellation has multiple fact tables. It is also known as galaxy schema.
* The following diagram shows two fact tables, namely sales and shipping.



* The sales fact table is same as that in the star schema.
* The shipping fact table has the five dimensions, namely item\_key, time\_key, shipper\_key, from\_location, to\_location.
* The shipping fact table also contains two measures, namely dollars sold and units sold.
* It is also possible to share dimension tables between fact tables. For example, time, item, and location dimension tables are shared between the sales and shipping fact table.

**OLAP**

Online Analytical Processing Server (OLAP) is based on the multidimensional data model. It allows managers, and analysts to get an insight of the information through fast, consistent, and interactive access to information. This chapter cover the types of OLAP, operations on OLAP, difference between OLAP, and statistical databases and OLTP.

**Types of OLAP Servers**

We have four types of OLAP servers:

* Relational OLAP (ROLAP)
* Multidimensional OLAP (MOLAP)
* Hybrid OLAP (HOLAP)
* Specialized SQL Servers

**OLAP Operations**

Since OLAP servers are based on multidimensional view of data, we will discuss OLAP operations in multidimensional data.

Here is the list of OLAP operations:

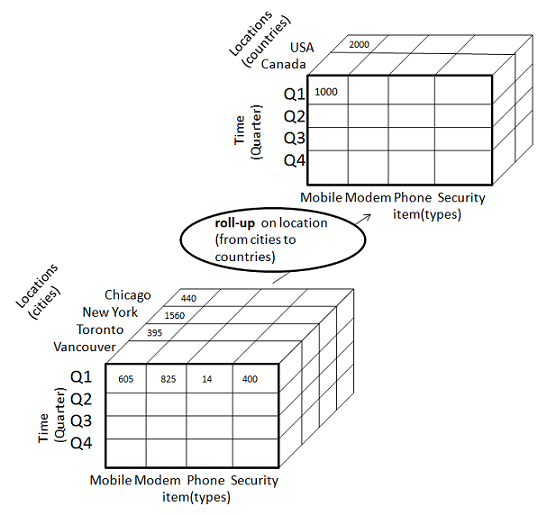
* Roll-up
* Drill-down
* Slice and dice
* Pivot (rotate)

**Roll-up**

Roll-up performs aggregation on a data cube in any of the following ways:

* By climbing up a concept hierarchy for a dimension
* By dimension reduction

The following diagram illustrates how roll-up works.



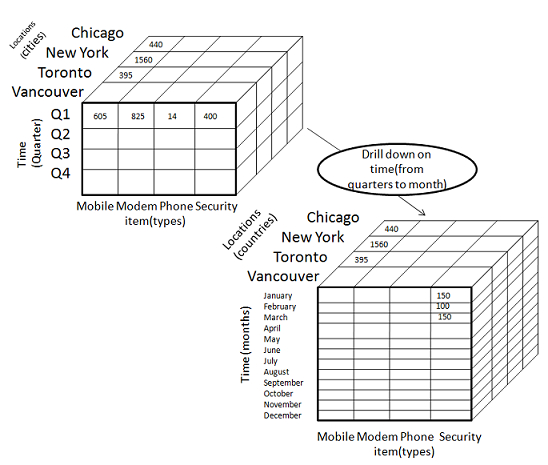
* Roll-up is performed by climbing up a concept hierarchy for the dimension location.
* Initially the concept hierarchy was "street < city < province < country".
* On rolling up, the data is aggregated by ascending the location hierarchy from the level of city to the level of country.
* The data is grouped into cities rather than countries.
* When roll-up is performed, one or more dimensions from the data cube are removed.

**Drill-down**

Drill-down is the reverse operation of roll-up. It is performed by either of the following ways:

* By stepping down a concept hierarchy for a dimension
* By introducing a new dimension.

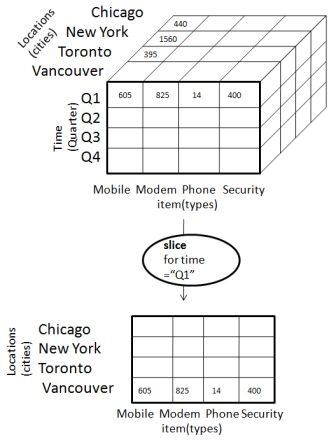
The following diagram illustrates how drill-down works:



* Drill-down is performed by stepping down a concept hierarchy for the dimension time.
* Initially the concept hierarchy was "day < month < quarter < year."
* On drilling down, the time dimension is descended from the level of quarter to the level of month.
* When drill-down is performed, one or more dimensions from the data cube are added.
* It navigates the data from less detailed data to highly detailed data.

**Slice**

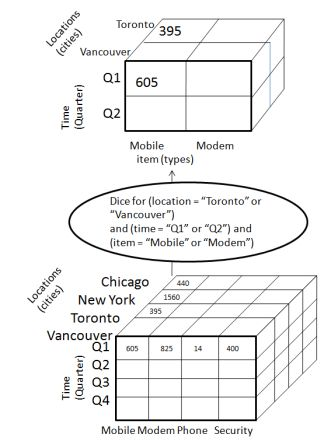
The slice operation selects one particular dimension from a given cube and provides a new sub-cube. Consider the following diagram that shows how slice works.



* Here Slice is performed for the dimension "time" using the criterion time = "Q1".
* It will form a new sub-cube by selecting one or more dimensions.

**Dice**

Dice selects two or more dimensions from a given cube and provides a new sub-cube. Consider the following diagram that shows the dice operation.

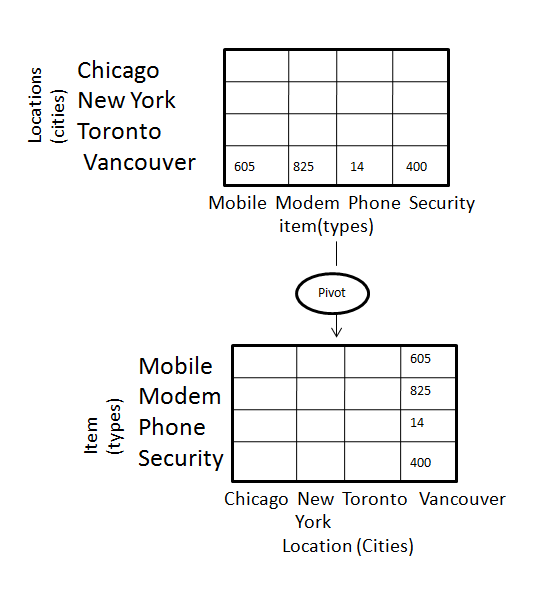


The dice operation on the cube based on the following selection criteria involves three dimensions.

* (location = "Toronto" or "Vancouver")
* (time = "Q1" or "Q2")
* (item =" Mobile" or "Modem")

**Pivot**

The pivot operation is also known as rotation. It rotates the data axes in view in order to provide an alternative presentation of data. Consider the following diagram that shows the pivot operation.



In this the item and location axes in 2-D slice are rotated.

**OLAP vs OLTP**

|  |  |  |
| --- | --- | --- |
| **Sr.No.** | **Data Warehouse (OLAP)** | **Operational Database (OLTP)** |
| 1 | Involves historical processing of information. | Involves day-to-day processing. |
| 2 | OLAP systems are used by knowledge workers such as executives, managers and analysts. | OLTP systems are used by clerks, DBAs, or database professionals. |
| 3 | Useful in analyzing the business. | Useful in running the business. |
| 4 | It focuses on Information out. | It focuses on Data in. |
| 5 | Based on Star Schema, Snowflake, Schema and Fact Constellation Schema. | Based on Entity Relationship Model. |
| 6 | Contains historical data. | Contains current data. |
| 7 | Provides summarized and consolidated data. | Provides primitive and highly detailed data. |
| 8 | Provides summarized and multidimensional view of data. | Provides detailed and flat relational view of data. |
| 9 | Number or users is in hundreds. | Number of users is in thousands. |
| 10 | Number of records accessed is in millions. | Number of records accessed is in tens. |
| 11 | Database size is from 100 GB to 1 TB | Database size is from 100 MB to 1 GB. |
| 12 | Highly flexible. | Provides high performance. |

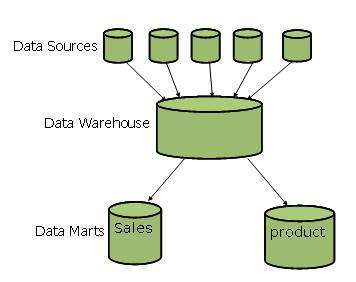
**Data Mart**

Data marts contain a subset of organization-wide data that is valuable to specific groups of people in an organization. In other words, a data mart contains only those data that is specific to a particular group. For example, the marketing data mart may contain only data related to items, customers, and sales. Data marts are confined to subjects.

**Points to Remember About Data Marts**

* Data marts are small in size.
* Data marts are customized by department.
* The source of a data mart is departmentally structured data warehouse.
* Data marts are flexible.

The following figure shows a graphical representation of data marts.



**Why Do We Need a Data Mart?**

Listed below are the reasons to create a data mart:

* To partition data in order to impose **access control strategies.**
* To speed up the queries by reducing the volume of data to be scanned.
* To segment data into different hardware platforms.
* To structure data in a form suitable for a user access tool.

**Note**: Do not data mart for any other reason since the operation cost of data marting could be very high. Before data marting, make sure that data marting strategy is appropriate for your particular solution.

**Cost of Data Marting**

The cost measures for data marting are as follows:

* Hardware and Software Cost
* Network Access
* Time Window Constraints

**Cost-effective Data Marting**

Follow the steps given below to make data marting cost-effective:

* Identify the Functional Splits
* Identify User Access Tool Requirements
* Identify Access Control Issues

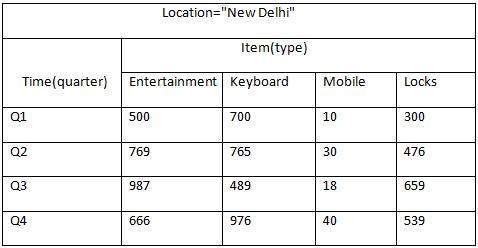
**Data Cube**

A data cube helps us represent data in multiple dimensions. It is defined by dimensions and facts. The dimensions are the entities with respect to which an enterprise preserves the records.

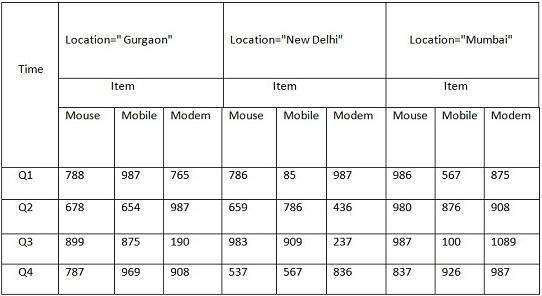
**Illustration of Data Cube**

Suppose a company wants to keep track of sales records with the help of sales data warehouse with respect to time, item, branch, and location. These dimensions allow to keep track of monthly sales and at which branch the items were sold. There is a table associated with each dimension. This table is known as dimension table. For example, "item" dimension table may have attributes such as item\_name, item\_type, and item\_brand.

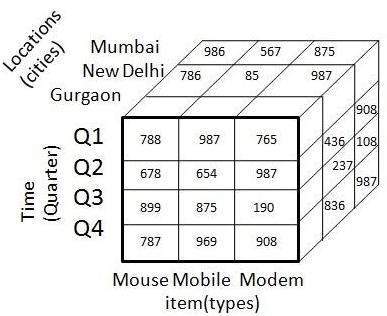
The following table represents the 2-D view of Sales Data for a company with respect to time, item, and location dimensions.



But here in this 2-D table, we have records with respect to time and item only. The sales for New Delhi are shown with respect to time, and item dimensions according to type of items sold. If we want to view the sales data with one more dimension, say, the location dimension, then the 3-D view would be useful. The 3-D view of the sales data with respect to time, item, and location is shown in the table below:



The above 3-D table can be represented as 3-D data cube as shown in the following figure:

Slice and Dice

**Virtual Warehouse**

The view over an operational data warehouse is known as virtual warehouse. It is easy to build a virtual warehouse. Building a virtual warehouse requires excess capacity on operational database servers.