**Unit 2**

**Key Features of Data Warehouse**

The key features of a data warehouse are discussed below:

* **Subject Oriented** - A data warehouse is subject oriented because it provides information around a subject rather than the organization's ongoing operations. These subjects can be product, customers, suppliers, sales, revenue, etc. A data warehouse does not focus on the ongoing operations; rather it focuses on modeling and analysis of data for decision making.
* **Integrated** - A data warehouse is constructed by integrating data from heterogeneous sources such as relational databases, flat files, etc. This integration enhances the effective analysis of data.
* **Time Variant** - The data collected in a data warehouse is identified with a particular time period. The data in a data warehouse provides information from the historical point of view.
* **Non-volatile** - Non-volatile means the previous data is not erased when new data is added to it. A data warehouse is kept separate from the operational database and therefore frequent changes in operational database are not reflected in the data warehouse.

**DBMS vs. Data Warehouse**

The major task of database systems is to perform on-line transaction and query processing. These systems are called on-line transaction processing (OLTP) systems. They cover most of the day-to-day operations of an organization, such as purchasing, inventory, manufacturing, banking, payroll, registration, and accounting. Data warehouse systems, on the other hand, serve users or knowledge workers in the role of data analysis and decision making. Such systems can organize and present data in various formats in order to accommodate the diverse needs of the different users. These systems are known as on-line analytical processing (OLAP) systems. The major distinguishing features between OLTP and OLAP are summarized as follows:

* **Users and System Orientation**: An OLTP system is *customer-oriented* and is used for transaction and query processing by clerks, clients, and information technology professionals. An OLAP system is *market-oriented* and is used for data analysis by knowledge workers, including managers, executives, and analysts.
* **Data Contents:** An OLTP system manages current data that, typically, are too detailed to be easily used for decision making. An OLAP system manages large amounts of historical data, provides facilities for summarization and aggregation, and stores and manages information at different levels of granularity. These features make the data easier to use in informed decision making.
* **Database Design**: An OLTP system usually adopts an entity-relationship (ER) data model and an application-oriented database design. An OLAP system typically adopts either a *star* or *snowflake* model (to be discussed in Section 3.2.2) and a subject oriented database design.
* **View:** An OLTP system focuses mainly on the current data within an enterprise or department, without referring to historical data or data in different organizations. In contrast, an OLAP system often spans multiple versions of a database schema, due to the evolutionary process of an organization. OLAP systems also deal with information that originates from different organizations, integrating information from many data stores. Because of their huge volume, OLAP data are stored on multiple storage media.
* **Access Patterns:** The access patterns of an OLTP system consist mainly of short, atomic transactions. Such a system requires concurrency control and recovery mechanisms. However, accesses to OLAP systems are mostly read-only operations (because most data warehouses store historical rather than up-to-date information)

**Why Separate Data Warehouse?**

Databases store huge amounts of data. Now the major question is *“why not perform on-line analytical processing directly on such databases instead of spending additional time and resources to construct a separate data warehouse?”* A major reason for such a separation is to help promote the *high performance of both systems*.

* An operational database is designed and tuned from known tasks and workloads, such as indexing and hashing using primary keys, searching for particular records, and optimizing canned queries. On the other hand, data warehouse queries are often complex. They involve the computation of large groups of data at summarized levels, and may require the use of special data organization, access, and implementation methods based on multidimensional views. Processing OLAP queries in operational databases would substantially degrade the performance of operational tasks.
* Concurrency control and recovery mechanisms, such as locking and logging, are required to ensure the consistency and robustness of transactions in database systems. An OLAP query often needs read-only access of data records for summarization and aggregation. Concurrency control and recovery mechanisms, if applied for such OLAP operations may jeopardize the execution of concurrent transactions and thus substantially reduce the throughput of an OLTP system.
* Finally, the separation of operational databases from data warehouses is based on the different structures, contents, and uses of the data in these two systems. Decision support requires historical data, whereas operational databases do not typically maintain historical data. In this context, the data in operational databases, though abundant, is usually far from complete for decision making.

**Multidimensional Data Model**

Data warehouses and OLAP tools are based on a multidimensional data model. This model views data in the form of a *data cube*.

**From Tables and Spreadsheets to Data Cubes**

A data cube allows data to be modeled and viewed in multiple dimensions. It is defined by *dimensions* and *facts*. Dimensions are the entities with respect to which an organization wants to keep records. For example, an organization 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*. Each dimension may have a table associated with it, called a *dimension table*. This table further describes the dimensions. For example, a dimension table for *item* may contain the attributes *item name, brand*, and *type*.

A multidimensional data model is typically organized around a central theme, like *sales*. This theme is represented by a fact table. Facts are numerical measures. Themes are the quantities by which we want to analyze relationships between dimensions. Examples of facts for a sales data warehouse include *dollars\_sold* (sales amount in dollars), *units\_sold* (number of units sold), and *amount budgeted*. The fact table contains the names of the *facts*, or measures, as well as keys to each of the related dimension tables.



***Figure:*** *Sales data for an organization according to the dimensions time, item, and location. The measure displayed is dollars\_sold.*

**

*Figure A 3-D data cube representation of the data in above table, according to the dimensions time, item, and location. The measure displayed is dollars\_sold (in thousands).*

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 Figure below.



*Figure 4-D data cube representation of sales data, according to the dimensions time, item, location, and supplier. The measure displayed is dollars\_sold (in thousands)*

If we continue in this way, we may display any *n*-dimensional data as a series of (*n*-1) dimensional cubes. The data cube is a metaphor for multidimensional data storage. The actual physical storage of such data may differ from its logical representation. The important thing to remember is that data cubes are *n*-dimensional and do not confine data to 3-D.

**Schemas for Multidimensional Database**