**UNIT-01**

**Data warehouse**

## What is Data?

Data is a collection of a distinct small unit of information. It can be used in a variety of forms like text, numbers, media, bytes, etc. it can be stored in pieces of paper or electronic memory, etc.

Word 'Data' is originated from the word 'datum' that means 'single piece of information.' It is plural of the word datum.

In computing, Data is information that can be translated into a form for efficient movement and processing. Data is interchangeable.

## What is Database?

A **database** is an organized collection of data, so that it can be easily accessed and managed.

You can organize data into tables, rows, columns, and index it to make it easier to find relevant information.

**INTRODUCTION ABOUT DATA WARE HOUSE**:

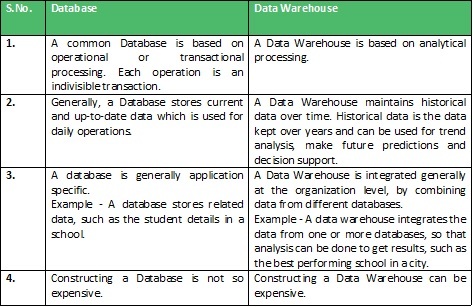
A **Data Warehouse**is separate from DBMS, it stores a huge amount of data, which is typically collected from multiple heterogeneous sources like files, DBMS, etc. The goal is to produce statistical results that may help in decision makings. For example, a college might want to see quick different results, like how the placement of CS students has improved over the last 10 years, in terms of salaries, counts, etc.

**Need for Data Warehouse**  
An ordinary Database can store MBs to GBs of data and that too for a specific purpose. For storing data of TB size, the storage shifted to Data Warehouse. Besides this, a transactional database doesn’t offer itself to analytics. To effectively perform analytics, an organization keeps a central Data Warehouse to closely study its business by organizing, understanding, and using its historic data for taking strategic decisions and analyzing trends.

**Benefits of Data Warehouse:**

1. **Better business analytics:**Data warehouse plays an important role in every business to store and analysis of all the past data and records of the company. which can further increase the understanding or analysis of data to the company.
2. **Faster Queries:**Data warehouse is designed to handle large queries that’s why it runs queries faster than the database.
3. **Improved data Quality:**In the data warehouse the data you gathered from different sources is being stored and analyzed it does not interfere with or add data by itself so your quality of data is maintained and if you get any issue regarding data quality then the data warehouse team will solve this.
4. **Historical Insight:**The warehouse stores all your historical data which contains details about the business so that one can analyze it at any time and extract insights from it.

**Data Warehouse vs DBMS**



**Example Applications of Data Warehousing** 

Data Warehousing can be applied anywhere where we have a huge amount of data and we want to see statistical results that help in decision making. 

* **Social Media Websites:** The social networking websites like Facebook, Twitter, LinkedIn, etc. are based on analyzing large data sets. These sites gather data related to members, groups, locations, etc., and store it in a single central repository. Being a large amount of data, Data Warehouse is needed for implementing the same.
* **Banking:**Most of the banks these days use warehouses to see the spending patterns of account/cardholders. They use this to provide them with special offers, deals, etc.
* **Government:** Government uses a data warehouse to store and analyze tax payments which are used to detect tax thefts.

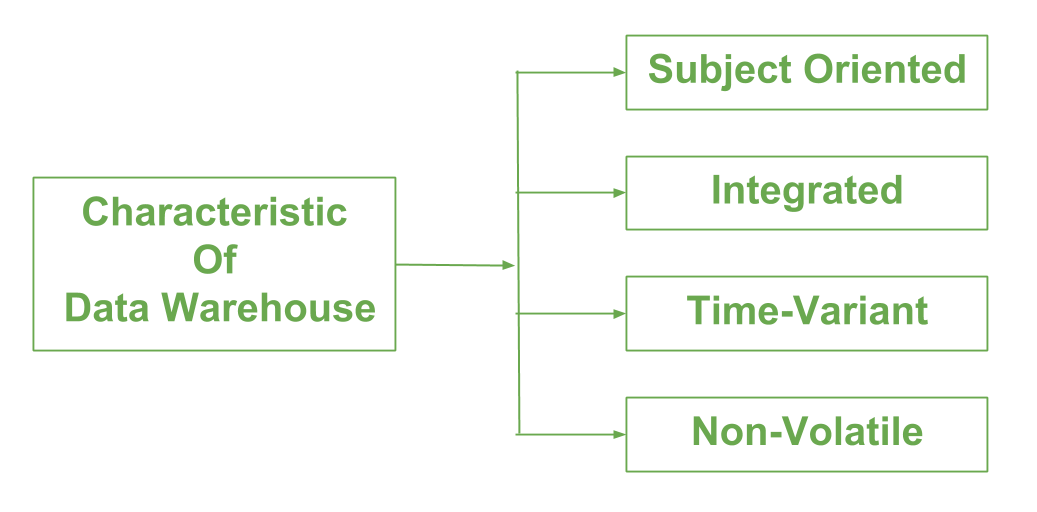
There can be many more applications in different sectors like E-Commerce, telecommunications, Transportation Services, Marketing and Distribution, Healthcare, and Retail.

**Difference between Data Warehouse and Operational Database**

The following are the important differences between a data warehouse and an operational database −

|  |  |  |
| --- | --- | --- |
| **Key** | **Data Warehouse** | **Operational Database** |
| Basic | A data warehouse is a repository for structured, filtered data that has already been processed for a specific purpose. | Operational Database are those databases where data changes frequently. |
| Data Structure | Data warehouse has de-normalized schema. | It has normalized schema. |
| Performance | It is fast for analysis queries. | It is slow for analytics queries. |
| Type of Data | It focuses on historical data. | It focuses on current transactional data. |
| Use Case | It is used for OLAP. | It is used for OLT |

Characteristics of a Data Warehouse

**Integrated Data**

One of the key characteristics of a data warehouse is that it contains integrated data. This means that the data is collected from various sources, such as transactional systems, and then cleaned, transformed, and consolidated into a single, unified view. This allows for easy access and analysis of the data, as well as the ability to track data over time.

**Subject-Oriented**

A data warehouse is also subject-oriented, which means that the data is organized around specific subjects, such as customers, products, or sales. This allows for easy access to the data relevant to a specific subject, as well as the ability to track the data over time.

**Non-Volatile**

Another characteristic of a data warehouse is that it is non-volatile. This means that the data in the warehouse is never updated or deleted, only added to. This is important because it allows for the preservation of historical data, making it possible to track trends and patterns over time.

**Time-Variant**

A data warehouse is also time-variant, which means that the data is stored with a time dimension. This allows for easy access to data for specific time periods, such as last quarter or last year. This makes it possible to track trends and patterns over time.

## Data Warehouse Architecture:

A data warehouse architecture is a method of defining the overall architecture of data communication processing and presentation that exist for end-clients computing within the enterprise. Each data warehouse is different, but all are characterized by standard vital components.

Production applications such as payroll accounts payable product purchasing and inventory control are designed for online transaction processing **(OLTP)**. Such applications gather detailed data from day-to-day operations.

Data Warehouse applications are designed to support the user ad-hoc data requirements, an activity recently dubbed online analytical processing (OLAP). These include applications such as forecasting, profiling, summary reporting, and trend analysis.

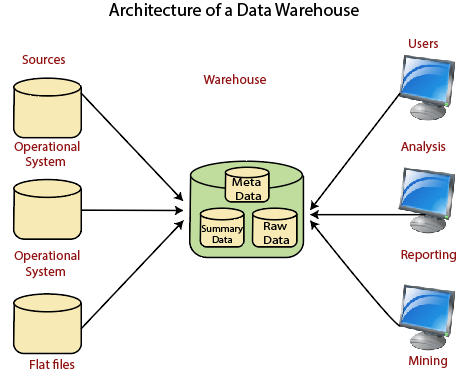
Production databases are updated continuously by either by hand or via OLTP applications. In contrast, a warehouse database is updated from operational systems periodically, usually during off-hours. As OLTP data accumulates in production databases, it is regularly extracted, filtered, and then loaded into a dedicated warehouse server that is accessible to users. As the warehouse is populated, it must be restructured tables de-normalized, data cleansed of errors and redundancies and new fields and keys added to reflect the needs to the user for sorting, combining, and summarizing data.

Data warehouses and their architectures very depending upon the elements of an organization's situation.

Three common architectures are:

* Data Warehouse Architecture: Basic
* Data Warehouse Architecture: With Staging Area
* Data Warehouse Architecture: With Staging Area and Data Marts

**Data Warehouse Architecture: Basic**



**Operational System**

An **operational system** is a method used in data warehousing to refer to a **system** that is used to process the day-to-day transactions of an organization.

**Flat Files**

A **Flat file** system is a system of files in which transactional data is stored, and every file in the system must have a different name.

**Meta Data**

A set of data that defines and gives information about other data.

Meta Data used in Data Warehouse for a variety of purpose, including:

Meta Data summarizes necessary information about data, which can make finding and work with particular instances of data more accessible. For example, author, data build, and data changed, and file size are examples of very basic document metadata.

Metadata is used to direct a query to the most appropriate data source.

**Lightly and highly summarized data**

The area of the data warehouse saves all the predefined lightly and highly summarized (aggregated) data generated by the warehouse manager.

The goals of the summarized information are to speed up query performance. The summarized record is updated continuously as new information is loaded into the warehouse.

**End-User access Tools**

The principal purpose of a data warehouse is to provide information to the business managers for strategic decision-making. These customers interact with the warehouse using end-client access tools.

The examples of some of the end-user access tools can be:

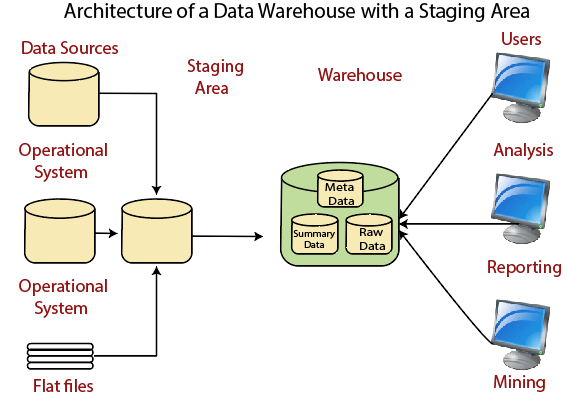
* Reporting and Query Tools
* Application Development Tools
* Executive Information Systems Tools
* Online Analytical Processing Tools
* Data Mining Tools

**Data Warehouse Architecture: With Staging Area**

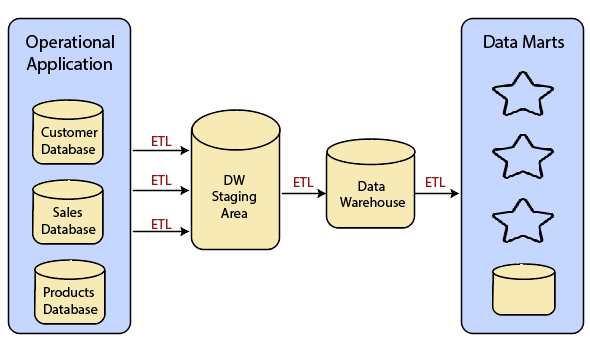
We must clean and process your operational information before put it into the warehouse.

We can do this programmatically, although data warehouses uses a **staging area** (A place where data is processed before entering the warehouse).

A staging area simplifies data cleansing and consolidation for operational method coming from multiple source systems, especially for enterprise data warehouses where all relevant data of an enterprise is consolidated.



**Data Warehouse Staging Area** is a temporary location where a record from source systems is copied.

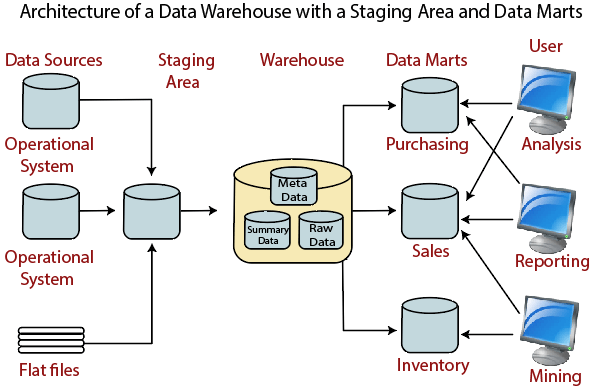


**Data Warehouse Architecture: With Staging Area and Data Marts**

We may want to customize our warehouse's architecture for multiple groups within our organization.

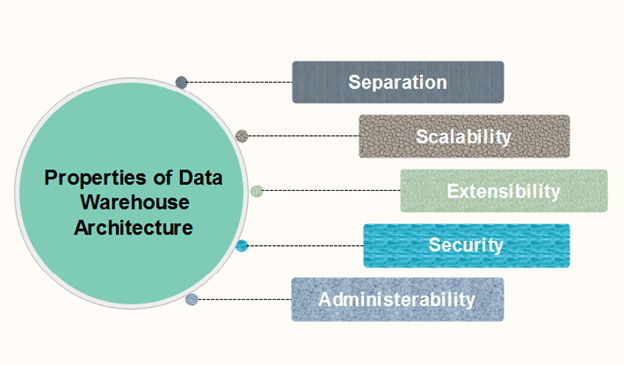
We can do this by adding **data marts**. A data mart is a segment of a data warehouses that can provided information for reporting and analysis on a section, unit, department or operation in the company, e.g., sales, payroll, production, etc.

The figure illustrates an example where purchasing, sales, and stocks are separated. In this example, a financial analyst wants to analyze historical data for purchases and sales or mine historical information to make predictions about customer behavior.



## Properties of Data Warehouse Architectures

The following architecture properties are necessary for a data warehouse system:



**1. Separation:** Analytical and transactional processing should be keep apart as much as possible.

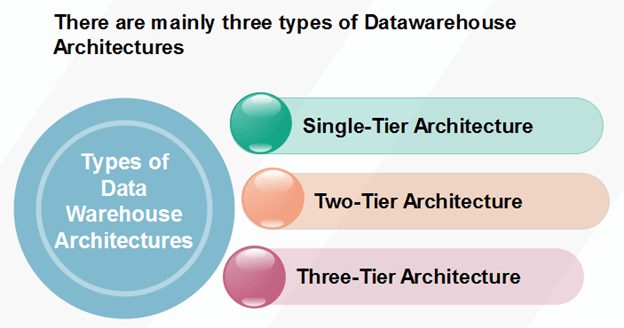
**2. Scalability:** Hardware and software architectures should be simple to upgrade the data volume, which has to be managed and processed, and the number of user's requirements, which have to be met, progressively increase.

**3. Extensibility:** The architecture should be able to perform new operations and technologies without redesigning the whole system.

**4. Security:** Monitoring accesses are necessary because of the strategic data stored in the data warehouses.

**5. Administrability:** Data Warehouse management should not be complicated.

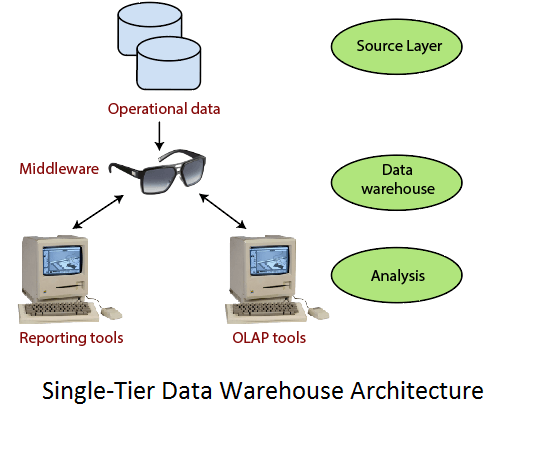
**Types of Data Warehouse Architectures**



**Single-Tier Architecture**

Single-Tier architecture is not periodically used in practice. Its purpose is to minimize the amount of data stored to reach this goal; it removes data redundancies.

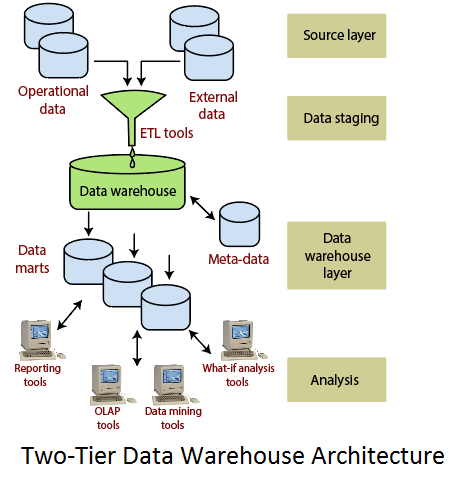
The figure shows the only layer physically available is the source layer. In this method, data warehouses are virtual. This means that the data warehouse is implemented as a multidimensional view of operational data created by specific middleware, or an intermediate processing layer.



The vulnerability of this architecture lies in its failure to meet the requirement for separation between analytical and transactional processing. Analysis queries are agreed to operational data after the middleware interprets them. In this way, queries affect transactional workloads.

**Two-Tier Architecture**

The requirement for separation plays an essential role in defining the two-tier architecture for a data warehouse system, as shown in fig:



Although it is typically called two-layer architecture to highlight a separation between physically available sources and data warehouses, in fact, consists of four subsequent data flow stages:

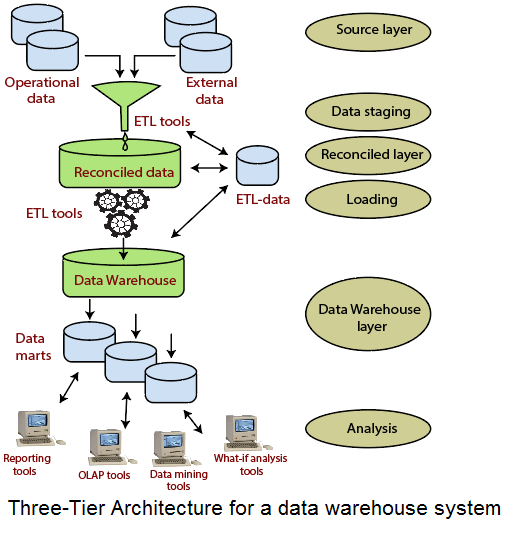
1. **Source layer:** A data warehouse system uses a heterogeneous source of data. That data is stored initially to corporate relational databases or legacy databases, or it may come from an information system outside the corporate walls.
2. **Data Staging:** The data stored to the source should be extracted, cleansed to remove inconsistencies and fill gaps, and integrated to merge heterogeneous sources into one standard schema. The so-named **Extraction**, **Transformation**, and **Loading Tools (ETL)** can combine heterogeneous schemata, extract, transform, cleanse, validate, filter, and load source data into a data warehouse.
3. **Data Warehouse layer:** Information is saved to one logically centralized individual repository: a data warehouse. The data warehouses can be directly accessed, but it can also be used as a source for creating data marts, which partially replicate data warehouse contents and are designed for specific enterprise departments. Meta-data repositories store information on sources, access procedures, data staging, users, data mart schema, and so on.
4. **Analysis:** In this layer, integrated data is efficiently, and flexible accessed to issue reports, dynamically analyze information, and simulate hypothetical business scenarios. It should feature aggregate information navigators, complex query optimizers, and customer-friendly GUIs.

**Three-Tier Architecture**

The three-tier architecture consists of the source layer (containing multiple source system), the reconciled layer and the data warehouse layer (containing both data warehouses and data marts). The reconciled layer sits between the source data and data warehouse.

The main advantage of the **reconciled layer** is that it creates a standard reference data model for a whole enterprise. At the same time, it separates the problems of source data extraction and integration from those of data warehouse population. In some cases, the **reconciled layer** is also directly used to accomplish better some operational tasks, such as producing daily reports that cannot be satisfactorily prepared using the corporate applications or generating data flows to feed external processes periodically to benefit from cleaning and integration.

This architecture is especially useful for the extensive, enterprise-wide systems. A disadvantage of this structure is the extra file storage space used through the extra redundant reconciled layer. It also makes the analytical tools a little further away from being real-time.



# **Three-Tier Data Warehouse Architecture**

Data Warehouses usually have a three-level (tier) architecture that includes:

1. Bottom Tier (Data Warehouse Server)
2. Middle Tier (OLAP Server)
3. Top Tier (Front end Tools).

A **bottom-tier** that consists of the **Data Warehouse server**, which is almost always an RDBMS. It may include several specialized data marts and a metadata repository.

Data from operational databases and external sources (such as user profile data provided by external consultants) are extracted using application program interfaces called a gateway. A gateway is provided by the underlying DBMS and allows customer programs to generate SQL code to be executed at a server.

**Examples** of gateways contain **ODBC** (Open Database Connection) and **OLE-DB** (Open-Linking and Embedding for Databases), by **Microsoft**, and **JDBC** (Java Database Connection).

A **middle-tier** which consists of an **OLAP server** for fast querying of the data warehouse.

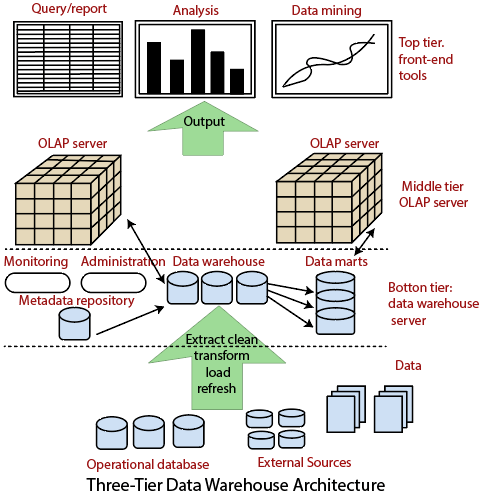
The OLAP server is implemented using either

**(1)** A **Relational OLAP (ROLAP) model**, i.e., an extended relational DBMS that maps functions on multidimensional data to standard relational operations.

**(2)** A **Multidimensional OLAP (MOLAP) model**, i.e., a particular purpose server that directly implements multidimensional information and operations.

A **top-tier** that contains **front-end tools** for displaying results provided by OLAP, as well as additional tools for data mining of the OLAP-generated data.

The overall Data Warehouse Architecture is shown in fig:

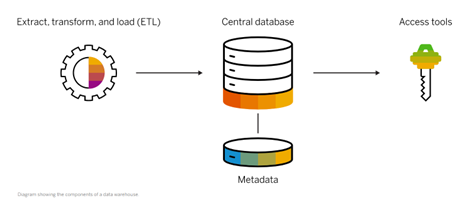


The **metadata repository** stores information that defines DW objects. It includes the following parameters and information for the middle and the top-tier applications:

1. A description of the DW structure, including the warehouse schema, dimension, hierarchies, data mart locations, and contents, etc.
2. Operational metadata, which usually describes the currency level of the stored data, i.e., active, archived or purged, and warehouse monitoring information, i.e., usage statistics, error reports, audit, etc.
3. System performance data, which includes indices, used to improve data access and retrieval performance.
4. Information about the mapping from operational databases, which provides source **RDBMSs** and their contents, cleaning and transformation rules, etc.
5. Summarization algorithms, predefined queries, and reports business data, which include business terms and definitions, ownership information, etc.

**Components of a data warehouse**

A typical data warehouse has four main components: a central database, ETL (extract, transform, load) tools, metadata, and access tools. All of these components are engineered for speed so that you can get results quickly and analyze data on the fly.



**Diagram showing the components of a data warehouse.**

**Central database:**A database serves as the foundation of your data warehouse. Traditionally, these have been standard relational databases running on premise or in the cloud. But because of Big Data, the need for true, real-time performance, and a drastic reduction in the cost of RAM, in-memory databases are rapidly gaining in popularity.

**Data integration:** Data is pulled from source systems and modified to align the information for rapid analytical consumption using a variety of data integration approaches such as ETL (extract, transform, load) and ELT as well as real-time data replication, bulk-load processing, data transformation, and data quality and enrichment services.

**Metadata:** Metadata is data about your data. It specifies the source, usage, values, and other features of the data sets in your data warehouse. There is business metadata, which adds context to your data, and technical metadata, which describes how to access data – including where it resides and how it is structured.

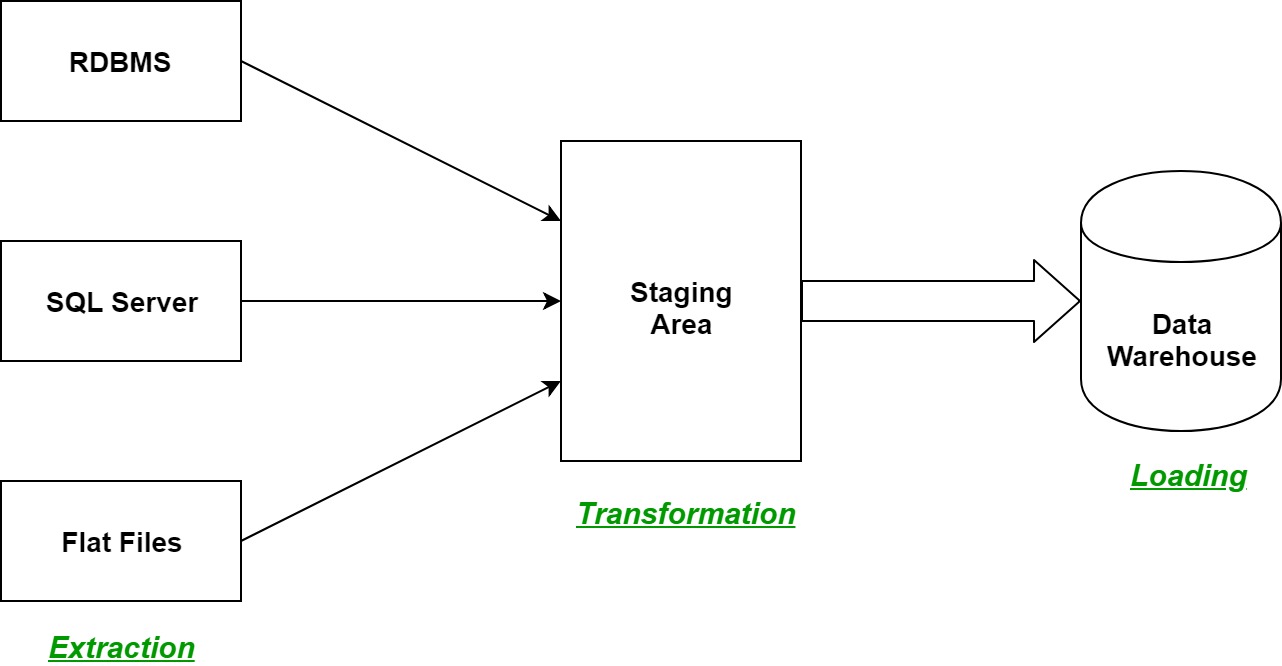
**Data warehouse access tools:** Access tools allow users to interact with the data in your data warehouse. Examples of access tools include: query and reporting tools, application development tools, data mining tools, and OLAP tools.

# **ETL Process in Data Warehouse**

1. ETL stands for Extract, Transform, Load and it is a process used in data warehousing to extract data from various sources, transform it into a format suitable for loading into a data warehouse, and then load it into the warehouse. The process of ETL can be broken down into the following three stages:
2. **Extract**: The first stage in the ETL process is to extract data from various sources such as transactional systems, spreadsheets, and flat files. This step involves reading data from the source systems and storing it in a staging area.
3. **Transform**: In this stage, the extracted data is transformed into a format that is suitable for loading into the data warehouse. This may involve cleaning and validating the data, converting data types, combining data from multiple sources, and creating new data fields.
4. **Load**: After the data is transformed, it is loaded into the data warehouse. This step involves creating the physical data structures and loading the data into the warehouse.
5. The ETL process is an iterative process that is repeated as new data is added to the warehouse. The process is important because it ensures that the data in the data warehouse is accurate, complete, and up-to-date. It also helps to ensure that the data is in the format required for data mining and reporting.

Additionally, there are many different ETL tools and technologies available, such as Informatica, Talend, DataStage, and others, that can automate and simplify the ETL process.

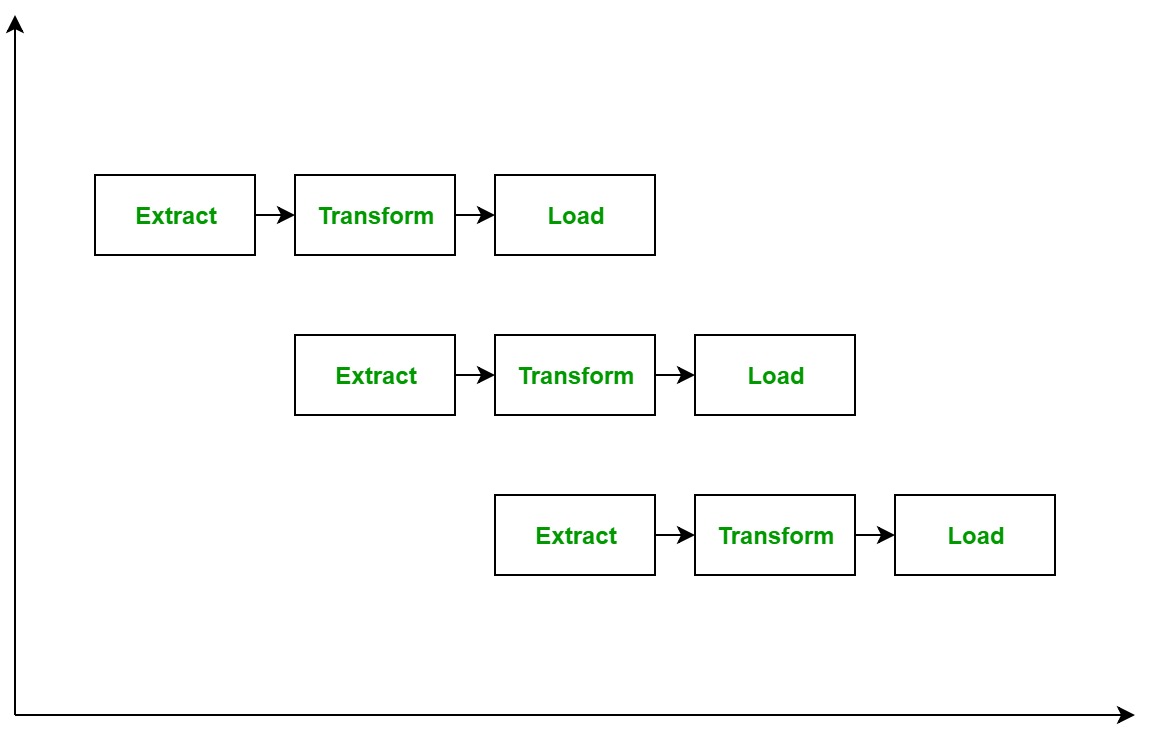
ETL is a process in Data Warehousing and it stands for **Extract**, **Transform**and **Load**. It is a process in which an ETL tool extracts the data from various data source systems, transforms it in the staging area, and then finally, loads it into the Data Warehouse system.



Let us understand each step of the ETL process in-depth:

1. **Extraction:**  
   The first step of the ETL process is extraction. In this step, data from various source systems is extracted which can be in various formats like relational databases, No SQL, XML, and flat files into the staging area. It is important to extract the data from various source systems and store it into the staging area first and not directly into the data warehouse because the extracted data is in various formats and can be corrupted also. Hence loading it directly into the data warehouse may damage it and rollback will be much more difficult. Therefore, this is one of the most important steps of ETL process.
2. **Transformation:**  
   The second step of the ETL process is transformation. In this step, a set of rules or functions are applied on the extracted data to convert it into a single standard format. It may involve following processes/tasks:
   * Filtering – loading only certain attributes into the data warehouse.
   * Cleaning – filling up the NULL values with some default values, mapping U.S.A, United States, and America into USA, etc.
   * Joining – joining multiple attributes into one.
   * Splitting – splitting a single attribute into multiple attributes.
   * Sorting – sorting tuples on the basis of some attribute (generally key-attribute).
3. **Loading:**  
   The third and final step of the ETL process is loading. In this step, the transformed data is finally loaded into the data warehouse. Sometimes the data is updated by loading into the data warehouse very frequently and sometimes it is done after longer but regular intervals. The rate and period of loading solely depends on the requirements and varies from system to system.

ETL process can also use the pipelining concept i.e. as soon as some data is extracted, it can transformed and during that period some new data can be extracted. And while the transformed data is being loaded into the data warehouse, the already extracted data can be transformed. The block diagram of the pipelining of ETL process is shown below:



**ETL Tools:** Most commonly used ETL tools are **Hevo**, Sybase, Oracle Warehouse builder, CloverETL, and Mark Logic.

**Data Warehouses:**Most commonly used Data Warehouses are **Snowflake**, Redshift, BigQuery, and Firebolt.

**ADVANTAGES OR DISADVANTAGES:**

**Advantages of ETL process in data warehousing:**

1. **Improved data quality:** ETL process ensures that the data in the data warehouse is accurate, complete, and up-to-date.
2. **Better data integration:**ETL process helps to integrate data from multiple sources and systems, making it more accessible and usable.
3. **Increased data security:** ETL process can help to improve data security by controlling access to the data warehouse and ensuring that only authorized users can access the data.
4. **Improved scalability: E**TL process can help to improve scalability by providing a way to manage and analyze large amounts of data.
5. **Increased automation:**ETL tools and technologies can automate and simplify the ETL process, reducing the time and effort required to load and update data in the warehouse.

**Disadvantages of ETL process in data warehousing:**

1. **High cost:** ETL process can be expensive to implement and maintain, especially for organizations with limited resources.
2. C**omplexity:**ETL process can be complex and difficult to implement, especially for organizations that lack the necessary expertise or resources.
3. **Limited flexibility:**ETL process can be limited in terms of flexibility, as it may not be able to handle unstructured data or real-time data streams.
4. **Limited scalability**: ETL process can be limited in terms of scalability, as it may not be able to handle very large amounts of data.
5. **Data privacy concerns**: ETL process can raise concerns about data privacy, as large amounts of data are collected, stored, and analyzed.

Overall, ETL process is an essential process in data warehousing that helps to ensure that the data in the data warehouse is accurate, complete, and up-to-date. However, it also comes with its own set of challenges and limitations, and organizations need to carefully consider the costs and benefits before implementing them.

What is data modelling?

Data modeling is the process of creating a visual representation of either a whole information system or parts of it to communicate connections between data points and structures. The goal is to illustrate the types of data used and stored within the system, the relationships among these data types, the ways the data can be grouped and organized and its formats and attributes.

Data models are built around business needs. Rules and requirements are defined upfront through feedback from business stakeholders so they can be incorporated into the design of a new system or adapted in the iteration of an existing one.

Data can be modelled at various levels of abstraction. The process begins by collecting information about business requirements from stakeholders and end users. These business rules are then translated into data structures to formulate a concrete database design. A data model can be compared to a roadmap, an architect’s blueprint or any formal diagram that facilitates a deeper understanding of what is being designed.

Data modeling employs standardized schemas and formal techniques. This provides a common, consistent, and predictable way of defining and managing data resources across an organization, or even beyond.

Ideally, data models are living documents that evolve along with changing business needs. They play an important role in supporting business processes and planning IT architecture and strategy. Data models can be shared with vendors, partners, and/or industry peers.

Types of data models

Like any design process, database and information system design begins at a high level of abstraction and becomes increasingly more concrete and specific. Data models can generally be divided into three categories, which vary according to their degree of abstraction. The process will start with a conceptual model, progress to a logical model and conclude with a physical model. Each type of data model is discussed in more detail below:

* **Conceptual data models.**They are also referred to as domain models and offer a big-picture view of what the system will contain, how it will be organized, and which business rules are involved. Conceptual models are usually created as part of the process of gathering initial project requirements. Typically, they include entity classes (defining the types of things that are important for the business to represent in the data model), their characteristics and constraints, the relationships between them and relevant security and data integrity requirements. Any notation is typically simple.

* **Logical data models.**They are less abstract and provide greater detail about the concepts and relationships in the domain under consideration. One of several formal data modeling notation systems is followed. These indicate data attributes, such as data types and their corresponding lengths, and show the relationships among entities. Logical data models don’t specify any technical system requirements. This stage is frequently omitted in agile or [DevOps](https://www.ibm.com/topics/devops) practices. Logical data models can be useful in highly procedural implementation environments, or for projects that are data-oriented by nature, such as [data warehouse](https://www.ibm.com/topics/data-warehouse) design or reporting system development.

* **Physical data models.**They provide a schema for how the data will be physically stored within a database. As such, they’re the least abstract of all. They offer a finalized design that can be implemented as a [relational database](https://www.ibm.com/topics/relational-databases), including associative tables that illustrate the relationships among entities as well as the primary keys and foreign keys that will be used to maintain those relationships. Physical data models can include database management system (DBMS)-specific properties, including performance tuning.

Data modeling process

As a discipline, data modeling invites stakeholders to evaluate data processing and storage in painstaking detail. Data modeling techniques have different conventions that dictate which symbols are used to represent the data, how models are laid out, and how business requirements are conveyed. All approaches provide formalized workflows that include a sequence of tasks to be performed in an iterative manner. Those workflows generally look like this:

1. **Identify the entities.** The process of data modeling begins with the identification of the things, events or concepts that are represented in the data set that is to be modeled. Each entity should be cohesive and logically discrete from all others.
2. **Identify key properties of each entity.** Each entity type can be differentiated from all others because it has one or more unique properties, called attributes. For instance, an entity called “customer” might possess such attributes as a first name, last name, telephone number and salutation, while an entity called “address” might include a street name and number, a city, state, country and zip code.
3. **Identify relationships among entities.**The earliest draft of a data model will specify the nature of the relationships each entity has with the others. In the above example, each customer “lives at” an address. If that model were expanded to include an entity called “orders,” each order would be shipped to and billed to an address as well. These relationships are usually documented via unified modeling language (UML).
4. **Map attributes to entities completely.**This will ensure the model reflects how the business will use the data. Several formal data modeling patterns are in widespread use. Object-oriented developers often apply analysis patterns or design patterns, while stakeholders from other business domains may turn to other patterns.
5. **Assign keys as needed, and decide on a degree of normalization that balances the need to reduce redundancy with performance requirements.** Normalization is a technique for organizing data models (and the databases they represent) in which numerical identifiers, called keys, are assigned to groups of data to represent relationships between them without repeating the data. For instance, if customers are each assigned a key, that key can be linked to both their address and their order history without having to repeat this information in the table of customer names. Normalization tends to reduce the amount of storage space a database will require, but it can at cost to query performance.
6. **Finalize and validate the data model.** Data modeling is an iterative process that should be repeated and refined as business needs change.

**Benefits of data modeling**

Data modeling makes it easier for developers, data architects, business analysts, and other stakeholders to view and understand relationships among the data in a database or data warehouse. In addition, it can:

* Reduce errors in software and database development.
* Increase consistency in documentation and system design across the enterprise.
* Improve application and database performance.
* Ease data mapping throughout the organization.
* Improve communication between developers and business intelligence teams.
* Ease and speed the process of database design at the conceptual, logical and physical levels.

Data modeling tools

Numerous commercial and open-source computer-aided software engineering (CASE) solutions are widely used today, including multiple data modeling, diagramming and visualization tools. Here are several examples:

* **erwin Data Modeler** is a data modeling tool based on the Integration DEFinition for information modeling (IDEF1X) data modeling language that now supports other notation methodologies, including a dimensional approach.
* **Enterprise Architect** is a visual modeling and design tool that supports the modeling of enterprise information systems and architectures as well as software applications and databases. It’s based on object-oriented languages and standards.
* **ER/Studio** is database design software that’s compatible with several of today’s most popular database management systems. It supports both relational and dimensional data modeling.
* **Free data modeling tools** include open-source solutions such as Open ModelSphere.

Numerous commercial and open-source computer-aided software engineering (CASE) solutions are widely used today, including multiple data modeling, diagramming and visualization tools. Here are several examples:

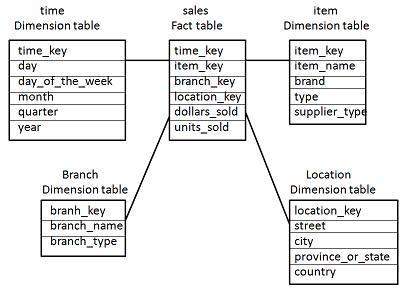
* **erwin Data Modeler** is a data modeling tool based on the Integration DEFinition for information modeling (IDEF1X) data modeling language that now supports other notation methodologies, including a dimensional approach.
* **Enterprise Architect** is a visual modeling and design tool that supports the modeling of enterprise information systems and architectures as well as software applications and databases. It’s based on object-oriented languages and standards.
* **ER/Studio** is database design software that’s compatible with several of today’s most popular database management systems. It supports both relational and dimensional data modeling.
* **Free data modeling tools** include open-source solutions such as Open ModelSphere.

# **Data Warehousing – Schemas**

Schema is a logical description of the entire database. It includes the name and description of records of all record types including all associated data-items and aggregates. Much like a database, a data warehouse also requires to maintain a schema. A database uses relational model, while a data warehouse uses Star, Snowflake, and Fact Constellation schema. In this chapter, we will discuss the schemas used in a data warehouse.

**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.

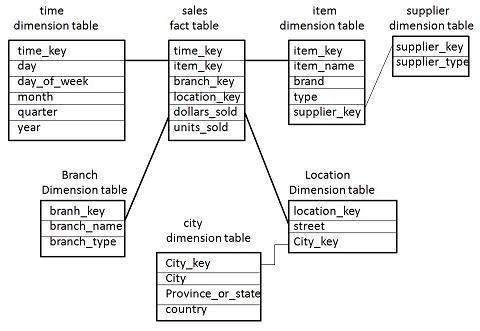


* There is a fact table at the center. It contains the keys to each of four dimensions.
* The fact table also contains the attributes, namely dollars sold and units sold.

**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.

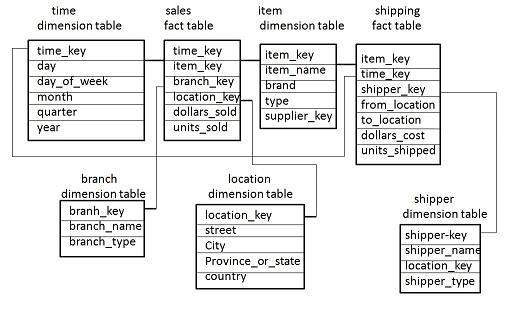


* Now the item dimension table contains the attributes item\_key, item\_name, type, brand, and supplier-key.
* The supplier key is linked to the supplier dimension table. The supplier dimension table contains the attributes supplier\_key and supplier\_type.

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

**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.

**Fact Table:**

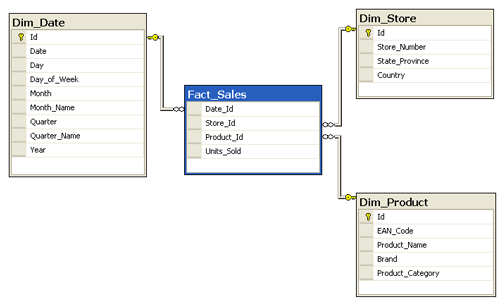
A fact table is used in the [dimensional model](https://www.zentut.com/data-warehouse/dimensional-modeling/) in [data warehouse](https://www.zentut.com/data-warehouse/) design. A fact table is found at the center of a [star schema](https://www.zentut.com/data-warehouse/star-schema/) or [snowflake schema](https://www.zentut.com/data-warehouse/snowflake-schema/) surrounded by [dimension tables](https://www.zentut.com/data-warehouse/dimension-table/).

A fact table consists of facts of a particular business process e.g., sales revenue by month by product. Facts are also known as measurements or metrics. A fact table record captures a measurement or a metric.

An example of a fact table

In the schema below, we have a fact table FACT\_SALES that has a grain that gives us the number of units sold by date, by store, and product.

All other tables such as DIM\_DATE*,*DIM\_STORE and DIM\_PRODUCT are dimensions tables. This schema is known as the star schema.

[](https://www.zentut.com/wp-content/uploads/2012/10/fact-table-example1.png)

Measure types

A fact table can store different types of measures such as additive, non-additive, semi-additive.

* **Additive** – As its name implied, additive measures are measures that can be added to all dimensions.
* **Non-additive** – different from additive measures, non-additive measures are measures that cannot be added to all dimensions.
* **Semi-additive** – semi-additive measures are the measure that can be added to only some dimensions and not across other.

Types of fact tables

All fact tables are categorized by the three most basic measurement events:

* **Transactional** – Transactional fact table is the most basic one that each grain associated with it indicated as “one row per line in a transaction”, e.g., every line item appears on an invoice. Transaction fact table stores data of the most detailed level, therefore, it has a high number of dimensions associated with it.
* **Periodic snapshots** – Periodic snapshots fact table stores the data that is a snapshot in a period of time. The source data of the periodic snapshots fact table is data from a transaction fact table where you choose a period to get the output.
* **Accumulating snapshots** – The accumulating snapshots fact table describes the activity of a business process that has a clear beginning and end. This type of fact table, therefore, has multiple date columns to represent milestones in the process. A good example of accumulating snapshots fact table is the processing of a material. As steps towards handling the material are finished, the corresponding record in the accumulating snapshots fact table gets updated.

# Dimensions vs Measures

**Measures are the facts**, the numbers of an event. For example the total profit is a measure, the total sales too.

**The dimensions are the details that explains your fact**. The time is a dimension (when?), the location is a dimension (where?) and all attributes of a fact (who?, What product?, …).

**Additive measures**

The numeric value in a fact table that is more flexible is an additive measure. For each dimension you can even sum up. If you want to know the total sales of your company you can easily sum up all the sales.

**Semi-additive measures**

With these measures you have to pay attention. Semi-additive facts are facts that can be summed up for some of the dimensions in the fact table, but not the others. For example, if you have the number of items in the warehouse for each day, you can sum up the items for each day (total warehouse of the day), but it makes no senso to sum up in the year.

**Non-additive measures**

With these facts you can never make a sum. If you are dealing with the discount percentage you easily understand what I’m talking about. It’s a little crazy to say “in my shop the percentage of total discount is 12000%”, nobody would believe you and would do well!

# **Fully Additive and Semi-Additive Measures**

Fully additive measures are summable with any dimension in the dataset and are always the same. They can be further aggregated on a subset of dimensions.

Semi-additive measures are summable with some dimensions, but the inclusion of some dimensions can result in unexpected sums that can skew your data analysis.

In Tableau, you can only aggregate semi-additive Commerce Cloud measures with certain dimensions.

How Measures Affect Your Metrics

This scenario shows how fully additive and semi-additive measures affect your metrics. Using the tools in Tableau, we imported a simplified version of the productOrderItems subject area.

The productOrderItems subject area allows export of the orderPlacedDate and productId dimensions and the ordersPlaced, unitsSold, and GMV measures. To have all the data available to build a dashboard, we import all the fields (dimensions and measures) for the subject area.

**Semi-Additive Example**

Table 1 shows a simplified version of data from the productOrderItems subject area. The ordersPlaced and GMV measures are aggregated by the orderPlacedDate dimension. The sum of all orders placed on 9/10/2021 is 7. An order can contain multiple products.

Table 1

| Table 1. Table 1 | | |
| --- | --- | --- |
| **orderPlacedDate** | **ordersPlaced** | **GMV** |
| 9/10/2021 | 7 | $2,943 |

Table 2 shows the same simplified data from the productOrderItems subject area. The ordersPlaced and GMV measures are aggregated by the orderPlacedDate and productId dimensions. The sum of all orders placed on 9/10/21 is 14, one count for each productId ordered. However, 14 orders is an error.

The ordersPlaced measure is semi-additive and isn’t summable by the productId dimension. The productId dimension caused the sum of ordersPlaced to count all the productId line items in each customer order rather than just the customer orders. The result in table 2 can’t be used as a metric for orders placed.

Table 2

| Table 2. Table 2 | | | |
| --- | --- | --- | --- |
| **orderPlacedDate** | **productId** | **ordersPlaced** | **GMV** |
| 9/10/2021 | product\_1 | 2 | $247 |
| 9/10/2021 | product\_2 | 1 | $83 |
| 9/10/2021 | product\_3 | 3 | $1,087 |
| 9/10/2021 | product\_4 | 1 | $298 |
| 9/10/2021 | product\_5 | 4 | $975 |
| 9/10/2021 | product\_7 | 1 | $148 |
| 9/10/2021 | product\_9 | 2 | $105 |

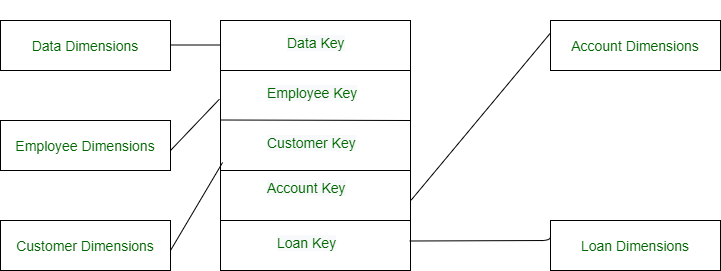
**Fully Additive Example**

GMV is a fully additive measure. The sum of GMV in both Table 1 and Table 2 is $2,943. Because GMV is a fully additive measure, no matter which dimensions aggregate GMV, the sum is always the same.

# Factless Fact Table

Factless tables simply mean the key available in the fact that no remedies are available. Factless fact tables are only used to establish relationships between elements of different dimensions. And are also useful for describing events and coverage, meaning tables contain information that nothing has happened. It often represents many-to-many relationships.

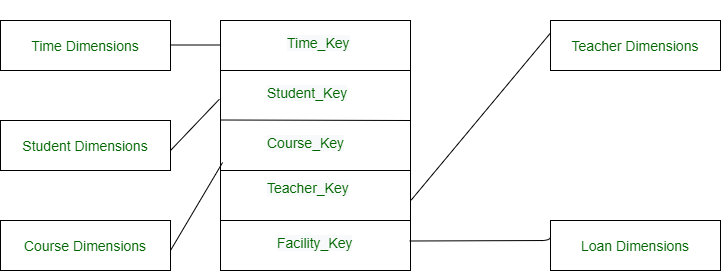
The only thing they have is an abbreviated key. They still represent a focal phenomenon that is identified by the combination referenced in the dimension tables.



*A Factless Fact Table*

There are two types of Factless table:

**1. Event Tracking Tables –**  
Use a factless fact table to track events of interest to the organization. For example, attendance at a cultural event can be tracked by creating a fact table that contains the following foreign keys (i.e. links to dimension tables) event identifier speaker/entertainment identifier, participant identifier, event type; Date. This table can then be searched for information, such as the most popular ones. Which cultural program or program type. The following example shows a factless fact table that records each time a student attends a course or which class has the maximum attendance? Or what is the average number of attendances of a given course?  All questions are based on COUNT () with group BY questions. So we can first count and then implement other aggregate functions like Aggress, Max, Min.



*Attendance fact*

**2. Coverage Tables –**  
The second type of factless fact table is called a coverage table by *Ralph*. It is used to support negative analysis reports. For example, to create a report that a store did not sell a product for a certain period of time, you should have a fact table to capture all possible combinations. Then you can find out what is missing.  
Common examples of factless fact table:  
Ex-Visitors to the office.  
List of people for the web click.  
Tracking student attendance or registration events.

## ****Dimension table****

A Dimension Table is a table in a star schema of a data warehouse. Data warehouses are built using dimensional data models which consist of fact and dimension tables. Dimension tables are used to describe dimensions; they contain dimension keys, values and attributes.

For example, the time dimension would contain every hour, day, week, month, quarter and year that has occurred since you started your business operations. Product dimension could contain a name and description of products you sell, their unit price, color, weight and other attributes as applicable. Attributes would be a customer’s first and last name, age, gender etc.

Dimension tables are typically small, ranging from a few to several thousand rows. Occasionally dimensions can grow fairly large, however. For example, a large credit card company could have a customer dimension with millions of rows. Dividing a data warehouse project into dimensions, provides structured information for reporting purpose.

When you create a dimension, you logically create a structure for your projects. This dimension table can be utilized across for reports and it’s about re-usability. If there are any changes to be made, it is evident that only a particular table will get affected. When a company wants to create a report, they can read the data from the dimension table since the table consists of necessary information.

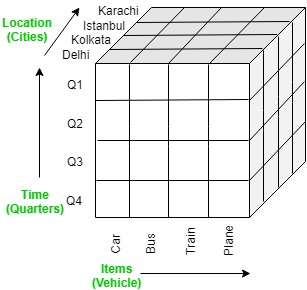
For example, an e-commerce company can create a dimension table with various columns depending on different subjects they would like to gain information from like the name of the person, address, date of order, shipment etc. This kind of an information becomes very crucial because in case of any clarification, the company can refer to this dimension table. Business users who generate these reports fire queries on these dimension tables as they contain descriptive information.

For an in-depth understanding of how Dimension Table works, here’s a video tutorial recorded by our[***Data warehousing and business intelligence***](https://www.edureka.co/data-warehousing-and-bi) Expert to explain the topic in a crisp manner with elaborate examples

.

**OLAP:**

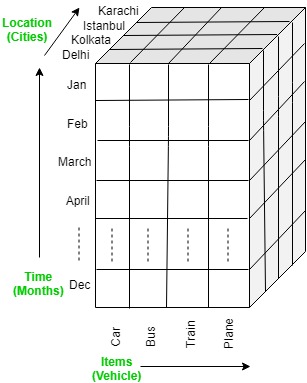
**OLAP** stands for ***Online Analytical Processing*** Server. It is a software technology that allows users to analyze information from multiple database systems at the same time. It is based on multidimensional data model and allows the user to query on multi-dimensional data (eg. Delhi -> 2018 -> Sales data). OLAP databases are divided into one or more cubes and these cubes are known as *Hyper-cubes*.



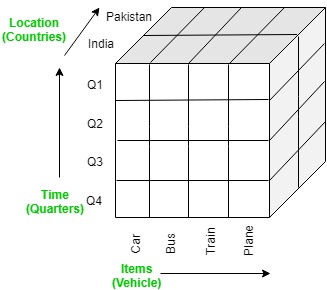
**OLAP operations:**

There are five basic analytical operations that can be performed on an OLAP cube:

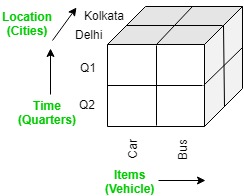
1. **Drill down:**In drill-down operation, the less detailed data is converted into highly detailed data. It can be done by:
   * Moving down in the concept hierarchy
   * Adding a new dimension

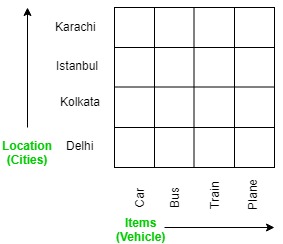
In the cube given in overview section, the drill down operation is performed by moving down in the concept hierarchy of *Time*dimension (Quarter -> Month).  


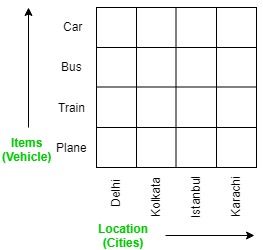
1. **Roll up:**It is just opposite of the drill-down operation. It performs aggregation on the OLAP cube. It can be done by:
   * Climbing up in the concept hierarchy
   * Reducing the dimensions

In the cube given in the overview section, the roll-up operation is performed by climbing up in the concept hierarchy of *Location*dimension (City -> Country).  


1. **Dice:**It selects a sub-cube from the OLAP cube by selecting two or more dimensions. In the cube given in the overview section, a sub-cube is selected by selecting following dimensions with criteria:
   * Location = “Delhi” or “Kolkata”
   * Time = “Q1” or “Q2”
   * Item = “Car” or “Bus”



1. **Slice:**It selects a single dimension from the OLAP cube which results in a new sub-cube creation. In the cube given in the overview section, Slice is performed on the dimension Time = “Q1”.  
   
2. **Pivot:**It is also known as *rotation* operation as it rotates the current view to get a new view of the representation. In the sub-cube obtained after the slice operation, performing pivot operation givesa new view of it.



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. |

# **OLAP Servers**

Online Analytical Processing (OLAP) refers to a set of software tools used for data analysis in order to make business decisions. OLAP provides a platform for gaining insights from databases retrieved from multiple database systems at the same time. It is based on a multidimensional data model, which enables users to extract and view data from various perspectives. A multidimensional database is used to store OLAP data. Many Business Intelligence (BI) applications rely on OLAP technology.

**Type of OLAP servers:**

The three major types of OLAP servers are as follows:

* **ROLAP**
* **MOLAP**
* **HOLAP**

**Relational OLAP (ROLAP):**

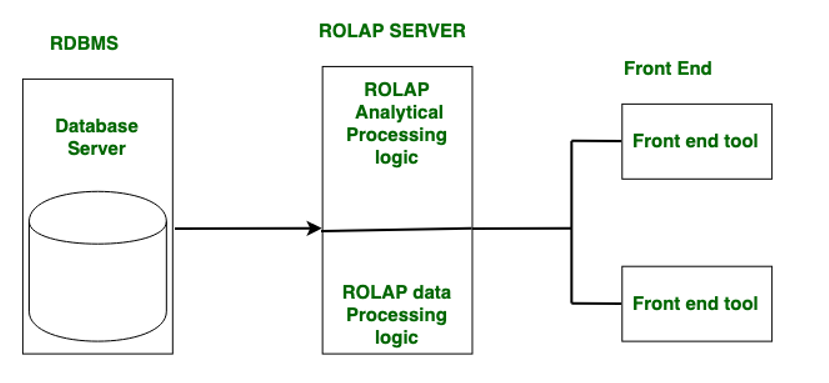
Relational On-Line Analytical Processing (ROLAP) is primarily used for data stored in a relational database, where both the base data and dimension tables are stored as relational tables. ROLAP servers are used to bridge the gap between the relational back-end server and the client’s front-end tools. ROLAP servers store and manage warehouse data using RDBMS, and OLAP middleware fills in the gaps.

**Benefits:**

* It is compatible with data warehouses and OLTP systems.
* The data size limitation of ROLAP technology is determined by the underlying RDBMS. As a result, ROLAP does not limit the amount of data that can be stored.

**Limitations:**

* SQL functionality is constrained.
* It’s difficult to keep aggregate tables up to date.



**Multidimensional OLAP (MOLAP):**

Through array-based multidimensional storage engines, Multidimensional On-Line Analytical Processing (MOLAP) supports multidimensional views of data. Storage utilization in multidimensional data stores may be low if the data set is sparse.

MOLAP stores data on discs in the form of a specialized multidimensional array structure. It is used for OLAP, which is based on the arrays’ random access capability. Dimension instances determine array elements, and the data or measured value associated with each cell is typically stored in the corresponding array element. The multidimensional array is typically stored in MOLAP in a linear allocation based on nested traversal of the axes in some predetermined order.

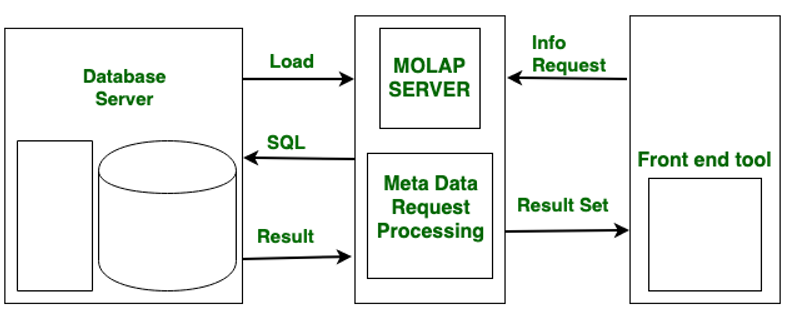
However, unlike ROLAP, which stores only records with non-zero facts, all array elements are defined in MOLAP, and as a result, the arrays tend to be sparse, with empty elements occupying a larger portion of them. MOLAP systems typically include provisions such as advanced indexing and hashing to locate data while performing queries for handling sparse arrays, because both storage and retrieval costs are important when evaluating online performance. MOLAP cubes are ideal for slicing and dicing data and can perform complex calculations. When the cube is created, all calculations are pre-generated.

**Benefits:**

* Suitable for slicing and dicing operations.
* Outperforms ROLAP when data is dense.
* Capable of performing complex calculations.

**Limitations:**

* It is difficult to change the dimensions without re-aggregating.
* Since all calculations are performed when the cube is built, a large amount of data cannot be stored in the cube itself.



**Hybrid OLAP (HOLAP):**

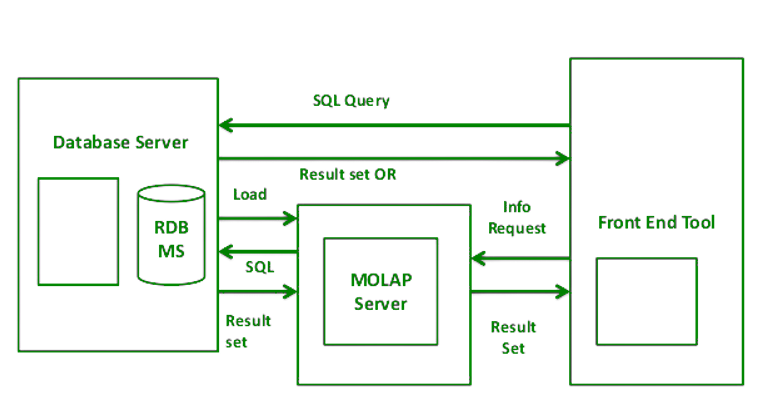
ROLAP and MOLAP are combined in Hybrid On-Line Analytical Processing (HOLAP). HOLAP offers greater scalability than ROLAP and faster computation than MOLAP.HOLAP is a hybrid of ROLAP and MOLAP. HOLAP servers are capable of storing large amounts of detailed data. On the one hand, HOLAP benefits from ROLAP’s greater scalability. HOLAP, on the other hand, makes use of cube technology for faster performance and summary-type information. Because detailed data is stored in a relational database, cubes are smaller than MOLAP.

**Benefits:**

* HOLAP combines the benefits of MOLAP and ROLAP.
* Provide quick access at all aggregation levels.

**Limitations**

* Because it supports both MOLAP and ROLAP servers, HOLAP architecture is extremely complex.
* There is a greater likelihood of overlap, particularly in their functionalities.



**Other types of OLAP include:**

* **Web OLAP (WOLAP):**WOLAP refers to an OLAP application that can be accessed through a web browser. WOLAP, in contrast to traditional client/server OLAP applications, is thought to have a three-tiered architecture consisting of three components: a client, middleware, and a database server.
* **Desktop OLAP (DOLAP):**DOLAP is an abbreviation for desktop analytical processing. In that case, the user can download the data from the source and work with it on their desktop or laptop. In comparison to other OLAP applications, functionality is limited. It is less expensive.
* **Mobile OLAP (MOLAP):**Wireless functionality or mobile devices are examples of MOLAP. The user is working and accessing data via mobile devices.
* **Spatial OLAP (SOLAP):**SOLAP egress combines the capabilities of Geographic Information Systems (GIS) and OLAP into a single user interface. SOLAP is created because the data can be alphanumeric, image, or vector. This allows for the quick and easy exploration of data stored in a spatial database.

6