Data Warehousing - Quick Guide

Data Warehousing - Overview

The term "Data Warehouse" was first coined by Bill Inmon in 1990. According to Inmon, a data warehouse is a subject oriented, integrated, time-variant, and non-volatile collection of data. This data helps analysts to take informed decisions in an organization.

An operational database undergoes frequent changes on a daily basis on account of the transactions that take place. Suppose a business executive wants to analyze previous feedback on any data such as a product, a supplier, or any consumer data, then the executive will have no data available to analyze because the previous data has been updated due to transactions.

A data warehouses provides us generalized and consolidated data in multidimensional view. Along with generalized and consolidated view of data, a data warehouses also provides us Online Analytical Processing (OLAP) tools. These tools help us in interactive and effective analysis of data in a multidimensional space. This analysis results in data generalization and data mining.

Data mining functions such as association, clustering, classification, prediction can be integrated with OLAP operations to enhance the interactive mining of knowledge at multiple level of abstraction. That's why data warehouse has now become an important platform for data analysis and online analytical processing.

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.
- It possesses consolidated historical data, which helps the organization to analyze its business.
- A data warehouse helps executives to organize, understand, and use their data to take strategic decisions.
- Data warehouse systems help in the integration of diversity of application systems.
- A data warehouse system helps in consolidated historical data analysis.

Why a Data Warehouse is Separated from Operational Databases

A data warehouses is kept separate from operational databases due to the following reasons −

- An operational database is constructed for well-known tasks and workloads such as searching particular records, indexing, etc. In contract, data warehouse queries are

often complex and they present a general form of data.

- Operational databases support concurrent processing of multiple transactions. Concurrency control and recovery mechanisms are required for operational databases to ensure robustness and consistency of the database.
- An operational database query allows to read and modify operations, while an OLAP query needs only **read only** access of stored data.
- An operational database maintains current data. On the other hand, a data warehouse maintains historical data.

Data Warehouse Features

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 modelling 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 is not reflected in the data warehouse.

Note − A data warehouse does not require transaction processing, recovery, and concurrency controls, because it is physically stored and separate from the operational database.

Data Warehouse Applications

As discussed before, a data warehouse helps business executives to organize, analyze, and use their data for decision making. A data warehouse serves as a sole part of a plan-execute-assess "closed-loop" feedback system for the enterprise management. Data warehouses are widely used in the following fields −

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

Types of Data Warehouse

Information processing, analytical processing, and data mining are the three types of data warehouse applications that are discussed below −

- **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.

Sr.No.	Data Warehouse (OLAP)	Operational Database(OLTP)
1	It involves historical processing of information.	It 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	It is used to analyze the business.	It is used to run the business.
4	It focuses on Information out.	It focuses on Data in.
5	It is based on Star Schema, Snowflake Schema, and Fact Constellation Schema.	It is based on Entity Relationship Model.
6	It focuses on Information out.	It is application oriented.
7	It contains historical data.	It contains current data.
8	It provides summarized and consolidated data.	It provides primitive and highly detailed data.
9	It provides summarized and multidimensional view of data.	It provides detailed and flat relational view of data.
10	The number of users is in hundreds.	The number of users is in thousands.
11	The number of records accessed is in millions.	The number of records accessed is in tens.
12	The database size is from 100GB to 100 TB.	The database size is from 100 MB to 100 GB.
13	These are highly flexible.	It provides high performance.

Data Warehousing - Concepts

What is Data Warehousing?

Data warehousing is the process of constructing and using a data warehouse. A data warehouse is constructed by integrating data from multiple heterogeneous sources that support analytical reporting, structured and/or ad hoc queries, and decision making. Data warehousing involves data cleaning, data integration, and data consolidations.

Using Data Warehouse Information

There are decision support technologies that help utilize the data available in a data warehouse. These technologies help executives to use the warehouse quickly and effectively. They can gather data, analyze it, and take decisions based on the information present in the warehouse. The information gathered in a warehouse can be used in any of the following domains −

- **Tuning Production Strategies** − The product strategies can be well tuned by repositioning the products and managing the product portfolios by comparing the sales quarterly or yearly.
- **Customer Analysis** − Customer analysis is done by analyzing the customer's buying preferences, buying time, budget cycles, etc.
- **Operations Analysis** − Data warehousing also helps in customer relationship management, and making environmental corrections. The information also allows us to analyze business operations.

Integrating Heterogeneous Databases

To integrate heterogeneous databases, we have two approaches −

- Query-driven Approach
- Update-driven Approach

Query-Driven Approach

This is the traditional approach to integrate heterogeneous databases. This approach was used to build wrappers and integrators on top of multiple heterogeneous databases. These integrators are also known as mediators.

Process of Query-Driven Approach

- When a query is issued to a client side, a metadata dictionary translates the query into an appropriate form for individual heterogeneous sites involved.
- Now these queries are mapped and sent to the local query processor.
- The results from heterogeneous sites are integrated into a global answer set.

Disadvantages

- Query-driven approach needs complex integration and filtering processes.
- This approach is very inefficient.
- It is very expensive for frequent queries.
- This approach is also very expensive for queries that require aggregations.

Update-Driven Approach

This is an alternative to the traditional approach. Today's data warehouse systems follow update-driven approach rather than the traditional approach discussed earlier. In update-driven

approach, the information from multiple heterogeneous sources are integrated in advance and are stored in a warehouse. This information is available for direct querying and analysis.

Advantages

This approach has the following advantages −

- This approach provide high performance.
- The data is copied, processed, integrated, annotated, summarized and restructured in semantic data store in advance.
- Query processing does not require an interface to process data at local sources.

Functions of Data Warehouse Tools and Utilities

The following are the functions of data warehouse tools and utilities −

- **Data Extraction** − Involves gathering data from multiple heterogeneous sources.
- **Data Cleaning** − Involves finding and correcting the errors in data.
- **Data Transformation** − Involves converting the data from legacy format to warehouse format.
- **Data Loading** − Involves sorting, summarizing, consolidating, checking integrity, and building indices and partitions.
- **Refreshing** − Involves updating from data sources to warehouse.

Note − Data cleaning and data transformation are important steps in improving the quality of data and data mining results.

Data Warehousing - Terminologies

In this chapter, we will discuss some of the most commonly used terms in data warehousing.

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.

In terms of data warehouse, we can define metadata as following −

- Metadata is a road-map to data warehouse.
- Metadata in data warehouse defines the warehouse objects.
- Metadata acts as a directory. This directory helps the decision support system to locate the contents of a data warehouse.

Metadata Repository

Metadata repository is an integral part of a data warehouse system. It contains the following metadata −

- **Business metadata** − It contains the data ownership information, business definition, and changing policies.
- **Operational metadata** − It includes currency of data and data lineage. Currency of data refers to the data being active, archived, or purged. Lineage of data means history of data migrated and transformation applied on it.
- **Data for mapping from operational environment to data warehouse** − It metadata includes source databases and their contents, data extraction, data partition, cleaning, transformation rules, data refresh and purging rules.
- **The algorithms for summarization** − It includes dimension algorithms, data on granularity, aggregation, summarizing, etc.

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.

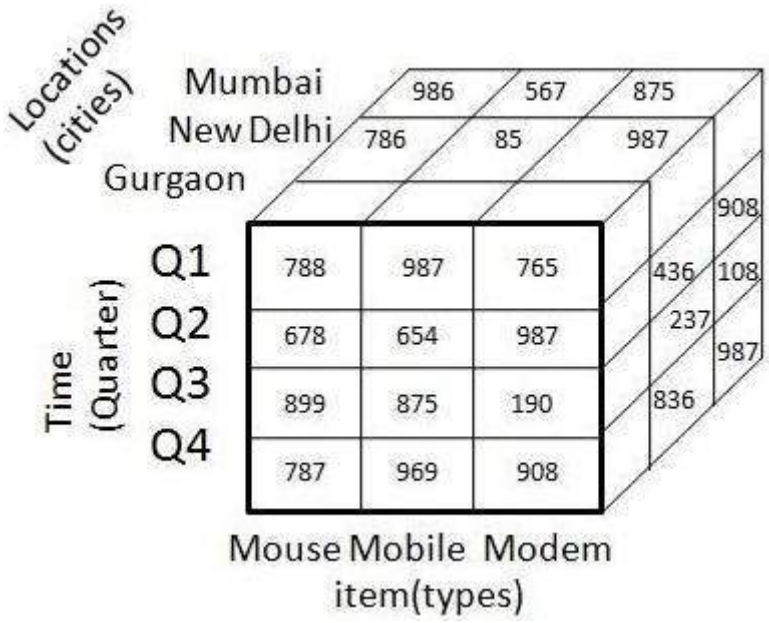
Location="New Delhi"				
Time(quarter)	Item(type)			
	Entertainment	Keyboard	Mobile	Locks
Q1	500	700	10	300
Q2	769	765	30	476
Q3	987	489	18	659
Q4	666	976	40	539

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 −

Time	Location="Gurgaon"			Location="New Delhi"			Location="Mumbai"		
	Item			Item			Item		
	Mouse	Mobile	Modem	Mouse	Mobile	Modem	Mouse	Mobile	Modem
Q1	788	987	765	786	85	987	986	567	875
Q2	678	654	987	659	786	436	980	876	908
Q3	899	875	190	983	909	237	987	100	1089
Q4	787	969	908	537	567	836	837	926	987

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



Data Mart

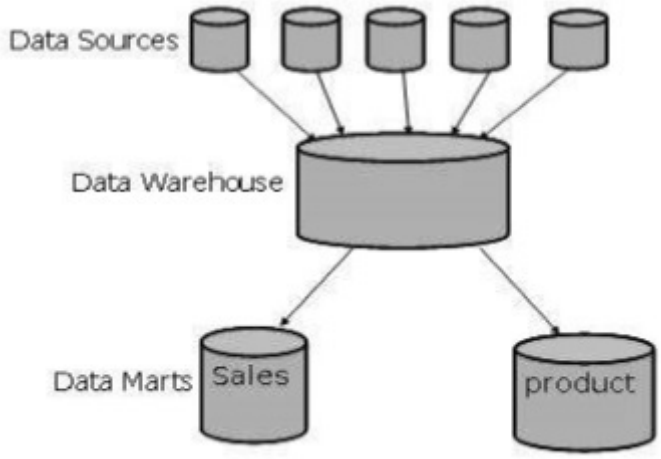
Data marts contain a subset of organization-wide data that is valuable to specific groups of people in an organization. In other words, a data mart contains only those data that is specific to a particular group. For example, the marketing data mart may contain only data related to items, customers,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

Points to Remember About Data Marts

- Windows-based or Unix/Linux-based servers are used to implement data marts. They are implemented on low-cost servers.

- The implementation cycle of a data mart is measured in short periods of time, i.e., in weeks rather than months or years.
- The life cycle of data marts may be complex in the long run, if their planning and design are not organization-wide.
- 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.



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.

Data Warehousing - Delivery Process

A data warehouse is never static; it evolves as the business expands. As the business evolves, its requirements keep changing and therefore a data warehouse must be designed to ride with these changes. Hence a data warehouse system needs to be flexible.

Ideally there should be a delivery process to deliver a data warehouse. However data warehouse projects normally suffer from various issues that make it difficult to complete tasks and deliverables in the strict and ordered fashion demanded by the waterfall method. Most of the times, the requirements are not understood completely. The architectures, designs, and build components can be completed only after gathering and studying all the requirements.

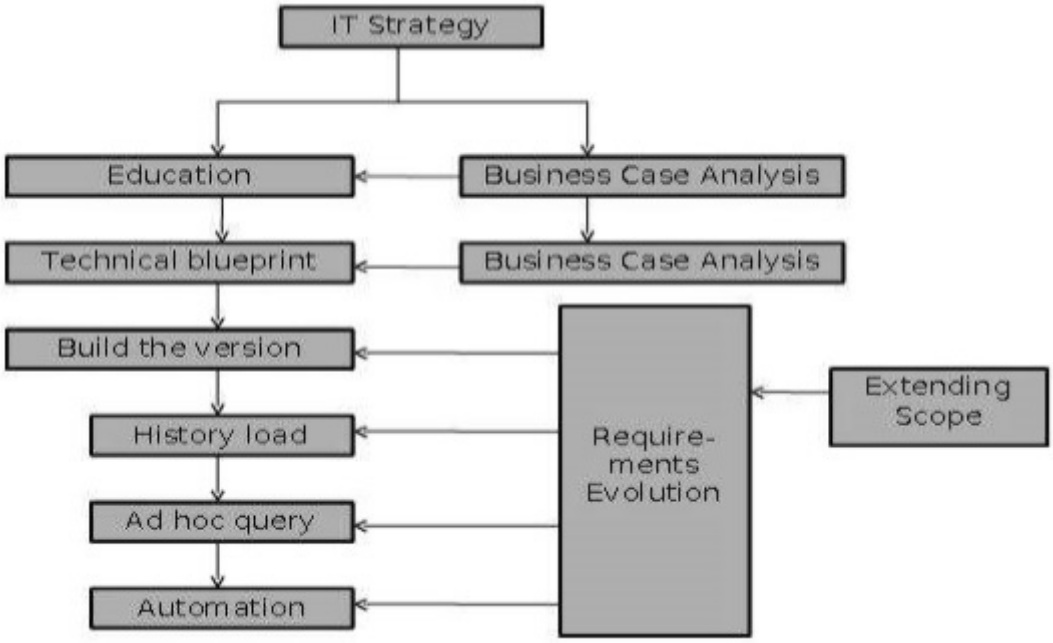
Delivery Method

The delivery method is a variant of the joint application development approach adopted for the delivery of a data warehouse. We have staged the data warehouse delivery process to minimize

risks. The approach that we will discuss here does not reduce the overall delivery time-scales but ensures the business benefits are delivered incrementally through the development process.

Note − The delivery process is broken into phases to reduce the project and delivery risk.

The following diagram explains the stages in the delivery process −



IT Strategy

Data warehouse are strategic investments that require a business process to generate benefits. IT Strategy is required to procure and retain funding for the project.

Business Case

The objective of business case is to estimate business benefits that should be derived from using a data warehouse. These benefits may not be quantifiable but the projected benefits need to be clearly stated. If a data warehouse does not have a clear business case, then the business tends to suffer from credibility problems at some stage during the delivery process. Therefore in data warehouse projects, we need to understand the business case for investment.

Education and Prototyping

Organizations experiment with the concept of data analysis and educate themselves on the value of having a data warehouse before settling for a solution. This is addressed by prototyping. It helps in understanding the feasibility and benefits of a data warehouse. The prototyping activity on a small scale can promote educational process as long as −

- The prototype addresses a defined technical objective.
- The prototype can be thrown away after the feasibility concept has been shown.
- The activity addresses a small subset of eventual data content of the data warehouse.
- The activity timescale is non-critical.

The following points are to be kept in mind to produce an early release and deliver business benefits.

- Identify the architecture that is capable of evolving.
- Focus on business requirements and technical blueprint phases.
- Limit the scope of the first build phase to the minimum that delivers business benefits.
- Understand the short-term and medium-term requirements of the data warehouse.

Business Requirements

To provide quality deliverables, we should make sure the overall requirements are understood. If we understand the business requirements for both short-term and medium-term, then we can design a solution to fulfil short-term requirements. The short-term solution can then be grown to a full solution.

The following aspects are determined in this stage −

- The business rule to be applied on data.
- The logical model for information within the data warehouse.
- The query profiles for the immediate requirement.
- The source systems that provide this data.

Technical Blueprint

This phase need to deliver an overall architecture satisfying the long term requirements. This phase also deliver the components that must be implemented in a short term to derive any business benefit. The blueprint need to identify the followings.

- The overall system architecture.
- The data retention policy.
- The backup and recovery strategy.
- The server and data mart architecture.
- The capacity plan for hardware and infrastructure.
- The components of database design.

Building the Version

In this stage, the first production deliverable is produced. This production deliverable is the smallest component of a data warehouse. This smallest component adds business benefit.

History Load

This is the phase where the remainder of the required history is loaded into the data warehouse. In this phase, we do not add new entities, but additional physical tables would probably be created to store increased data volumes.

Let us take an example. Suppose the build version phase has delivered a retail sales analysis data warehouse with 2 months' worth of history. This information will allow the user to analyze only the recent trends and address the short-term issues. The user in this case cannot identify annual and seasonal trends. To help him do so, last 2 years' sales history could be loaded from the archive. Now the 40GB data is extended to 400GB.

Note − The backup and recovery procedures may become complex, therefore it is recommended to perform this activity within a separate phase.

Ad hoc Query

In this phase, we configure an ad hoc query tool that is used to operate a data warehouse. These tools can generate the database query.

Note − It is recommended not to use these access tools when the database is being substantially modified.

Automation

In this phase, operational management processes are fully automated. These would include −

- Transforming the data into a form suitable for analysis.
- Monitoring query profiles and determining appropriate aggregations to maintain system performance.
- Extracting and loading data from different source systems.
- Generating aggregations from predefined definitions within the data warehouse.
- Backing up, restoring, and archiving the data.

Extending Scope

In this phase, the data warehouse is extended to address a new set of business requirements. The scope can be extended in two ways −

- By loading additional data into the data warehouse.
- By introducing new data marts using the existing information.

Note − This phase should be performed separately, since it involves substantial efforts and complexity.

Requirements Evolution

From the perspective of delivery process, the requirements are always changeable. They are not static. The delivery process must support this and allow these changes to be reflected within the system.

This issue is addressed by designing the data warehouse around the use of data within business processes, as opposed to the data requirements of existing queries.

The architecture is designed to change and grow to match the business needs, the process operates as a pseudo-application development process, where the new requirements are

continually fed into the development activities and the partial deliverables are produced. These partial deliverables are fed back to the users and then reworked ensuring that the overall system is continually updated to meet the business needs.

Data Warehousing - System Processes

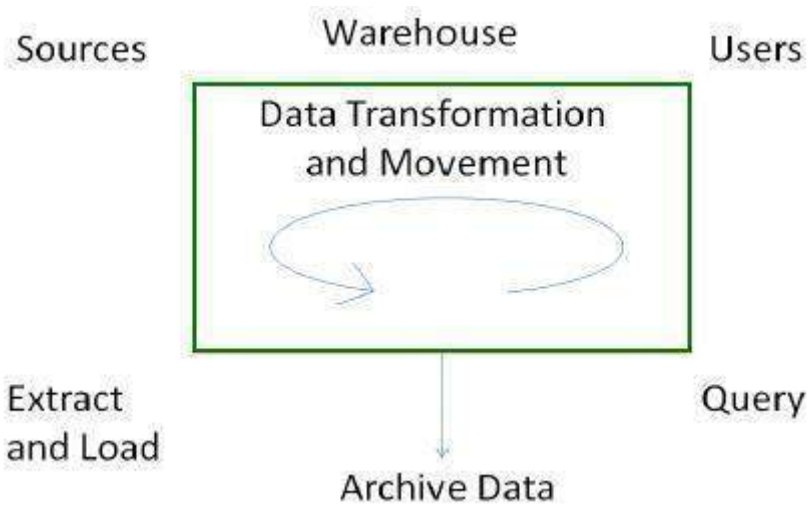
We have a fixed number of operations to be applied on the operational databases and we have well-defined techniques such as **use normalized data**, **keep table small**, etc. These techniques are suitable for delivering a solution. But in case of decision-support systems, we do not know what query and operation needs to be executed in future. Therefore techniques applied on operational databases are not suitable for data warehouses.

In this chapter, we will discuss how to build data warehousing solutions on top open-system technologies like Unix and relational databases.

Process Flow in Data Warehouse

There are four major processes that contribute to a data warehouse −

- Extract and load the data.
- Cleaning and transforming the data.
- Backup and archive the data.
- Managing queries and directing them to the appropriate data sources.



Extract and Load Process

Data extraction takes data from the source systems. Data load takes the extracted data and loads it into the data warehouse.

Note − Before loading the data into the data warehouse, the information extracted from the external sources must be reconstructed.

Controlling the Process

Controlling the process involves determining when to start data extraction and the consistency check on data. Controlling process ensures that the tools, the logic modules, and the programs are executed in correct sequence and at correct time.

When to Initiate Extract

Data needs to be in a consistent state when it is extracted, i.e., the data warehouse should represent a single, consistent version of the information to the user.

For example, in a customer profiling data warehouse in telecommunication sector, it is illogical to merge the list of customers at 8 pm on Wednesday from a customer database with the customer subscription events up to 8 pm on Tuesday. This would mean that we are finding the customers for whom there are no associated subscriptions.

Loading the Data

After extracting the data, it is loaded into a temporary data store where it is cleaned up and made consistent.

Note − Consistency checks are executed only when all the data sources have been loaded into the temporary data store.

Clean and Transform Process

Once the data is extracted and loaded into the temporary data store, it is time to perform Cleaning and Transforming. Here is the list of steps involved in Cleaning and Transforming −

- Clean and transform the loaded data into a structure
- Partition the data
- Aggregation

Clean and Transform the Loaded Data into a Structure

Cleaning and transforming the loaded data helps speed up the queries. It can be done by making the data consistent −

- within itself.
- with other data within the same data source.
- with the data in other source systems.
- with the existing data present in the warehouse.

Transforming involves converting the source data into a structure. Structuring the data increases the query performance and decreases the operational cost. The data contained in a data warehouse must be transformed to support performance requirements and control the ongoing operational costs.

Partition the Data

It will optimize the hardware performance and simplify the management of data warehouse. Here we partition each fact table into multiple separate partitions.

Aggregation

Aggregation is required to speed up common queries. Aggregation relies on the fact that most common queries will analyze a subset or an aggregation of the detailed data.

Backup and Archive the Data

In order to recover the data in the event of data loss, software failure, or hardware failure, it is necessary to keep regular back ups. Archiving involves removing the old data from the system in a format that allow it to be quickly restored whenever required.

For example, in a retail sales analysis data warehouse, it may be required to keep data for 3 years with the latest 6 months data being kept online. In such as scenario, there is often a requirement to be able to do month-on-month comparisons for this year and last year. In this case, we require some data to be restored from the archive.

Query Management Process

This process performs the following functions −

- manages the queries.
- helps speed up the execution time of queris.
- directs the queries to their most effective data sources.
- ensures that all the system sources are used in the most effective way.
- monitors actual query profiles.

The information generated in this process is used by the warehouse management process to determine which aggregations to generate. This process does not generally operate during the regular load of information into data warehouse.

Data Warehousing - Architecture

In this chapter, we will discuss the business analysis framework for the data warehouse design and architecture of a data warehouse.

Business Analysis Framework

The business analyst get the information from the data warehouses to measure the performance and make critical adjustments in order to win over other business holders in the market. Having a data warehouse offers the following advantages −

- Since a data warehouse can gather information quickly and efficiently, it can enhance business productivity.
- A data warehouse provides us a consistent view of customers and items, hence, it helps us manage customer relationship.

- A data warehouse also helps in bringing down the costs by tracking trends, patterns over a long period in a consistent and reliable manner.

To design an effective and efficient data warehouse, we need to understand and analyze the business needs and construct a **business analysis framework**. Each person has different views regarding the design of a data warehouse. These views are as follows −

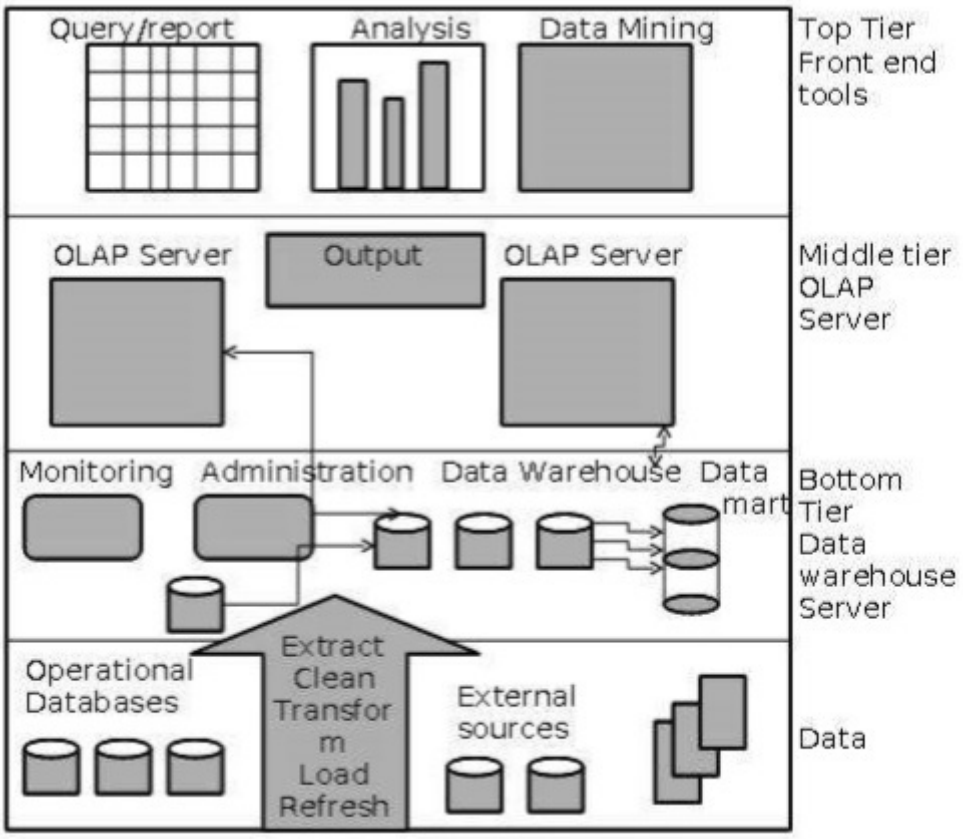
- **The top-down view** − This view allows the selection of relevant information needed for a data warehouse.
- **The data source view** − This view presents the information being captured, stored, and managed by the operational system.
- **The data warehouse view** − This view includes the fact tables and dimension tables. It represents the information stored inside the data warehouse.
- **The business query view** − It is the view of the data from the viewpoint of the end-user.

Three-Tier Data Warehouse Architecture

Generally a data warehouses adopts a three-tier architecture. Following are the three tiers of the data warehouse architecture.

- **Bottom Tier** − The bottom tier of the architecture is the data warehouse database server. It is the relational database system. We use the back end tools and utilities to feed data into the bottom tier. These back end tools and utilities perform the Extract, Clean, Load, and refresh functions.
- **Middle Tier** − In the middle tier, we have the OLAP Server that can be implemented in either of the following ways.
    - By Relational OLAP (ROLAP), which is an extended relational database management system. The ROLAP maps the operations on multidimensional data to standard relational operations.
    - By Multidimensional OLAP (MOLAP) model, which directly implements the multidimensional data and operations.
- **Top-Tier** − This tier is the front-end client layer. This layer holds the query tools and reporting tools, analysis tools and data mining tools.

The following diagram depicts the three-tier architecture of data warehouse −



Data Warehouse Models

From the perspective of data warehouse architecture, we have the following data warehouse models −

- Virtual Warehouse
- Data mart
- Enterprise Warehouse

Virtual Warehouse

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

Data Mart

Data mart contains a subset of organization-wide data. This subset of data is valuable to specific groups of an organization.

In other words, we can claim that data marts contain data specific to a particular group. For example, the marketing data mart may contain data related to items, customers, and sales. Data marts are confined to subjects.

Points to remember about data marts −

- Window-based or Unix/Linux-based servers are used to implement data marts. They are implemented on low-cost servers.

- The implementation data mart cycles is measured in short periods of time, i.e., in weeks rather than months or years.
- The life cycle of a data mart may be complex in long run, if its planning and design are not organization-wide.
- Data marts are small in size.
- Data marts are customized by department.
- The source of a data mart is departmentally structured data warehouse.
- Data mart are flexible.

Enterprise Warehouse

- An enterprise warehouse collects all the information and the subjects spanning an entire organization
- It provides us enterprise-wide data integration.
- The data is integrated from operational systems and external information providers.
- This information can vary from a few gigabytes to hundreds of gigabytes, terabytes or beyond.

Load Manager

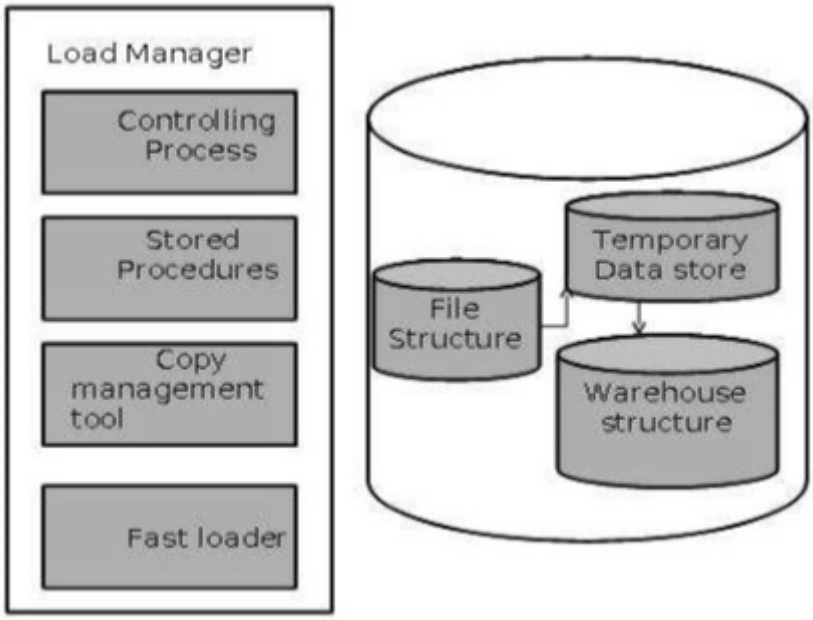
This component performs the operations required to extract and load process.

The size and complexity of the load manager varies between specific solutions from one data warehouse to other.

Load Manager Architecture

The load manager performs the following functions −

- Extract the data from source system.
- Fast Load the extracted data into temporary data store.
- Perform simple transformations into structure similar to the one in the data warehouse.



Extract Data from Source

The data is extracted from the operational databases or the external information providers. Gateways is the application programs that are used to extract data. It is supported by underlying DBMS and allows client program to generate SQL to be executed at a server. Open Database Connection(ODBC), Java Database Connection (JDBC), are examples of gateway.

Fast Load

- In order to minimize the total load window the data need to be loaded into the warehouse in the fastest possible time.
- The transformations affects the speed of data processing.
- It is more effective to load the data into relational database prior to applying transformations and checks.
- Gateway technology proves to be not suitable, since they tend not be performant when large data volumes are involved.

Simple Transformations

While loading it may be required to perform simple transformations. After this has been completed we are in position to do the complex checks. Suppose we are loading the EPOS sales transaction we need to perform the following checks:

- Strip out all the columns that are not required within the warehouse.
- Convert all the values to required data types.

Warehouse Manager

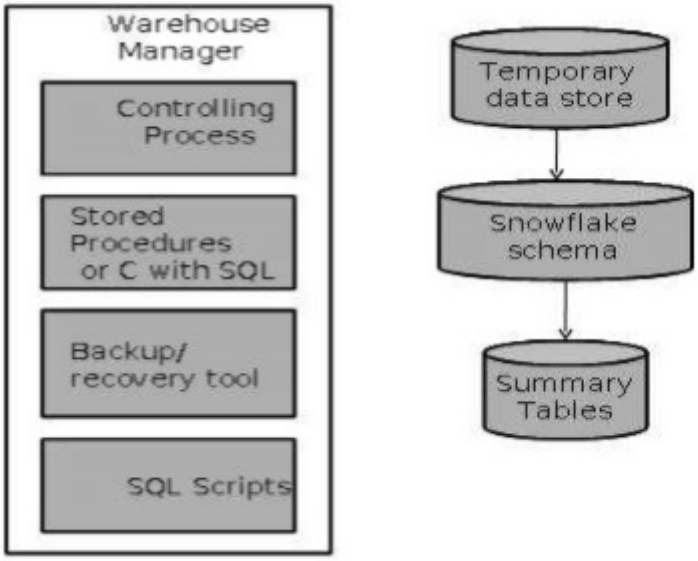
A warehouse manager is responsible for the warehouse management process. It consists of third-party system software, C programs, and shell scripts.

The size and complexity of warehouse managers varies between specific solutions.

Warehouse Manager Architecture

A warehouse manager includes the following −

- The controlling process
- Stored procedures or C with SQL
- Backup/Recovery tool
- SQL Scripts



Operations Performed by Warehouse Manager

- A warehouse manager analyzes the data to perform consistency and referential integrity checks.
- Creates indexes, business views, partition views against the base data.
- Generates new aggregations and updates existing aggregations. Generates normalizations.
- Transforms and merges the source data into the published data warehouse.
- Backup the data in the data warehouse.
- Archives the data that has reached the end of its captured life.

Note − A warehouse Manager also analyzes query profiles to determine index and aggregations are appropriate.

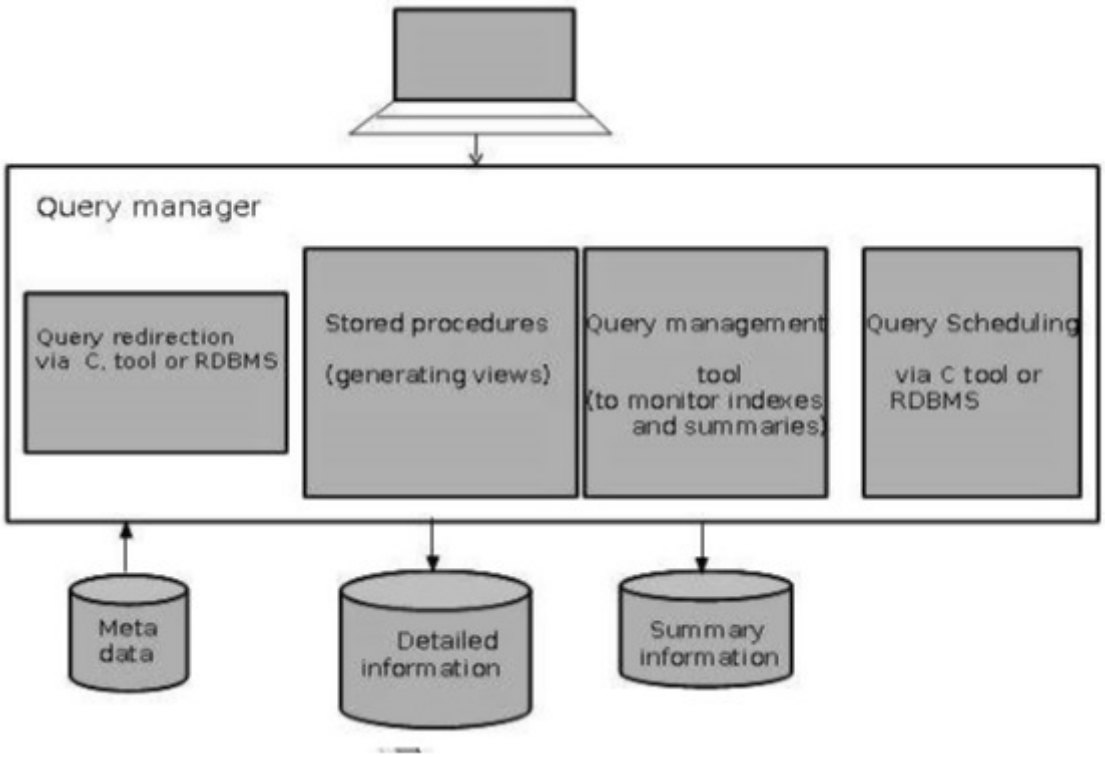
Query Manager

- Query manager is responsible for directing the queries to the suitable tables.
- By directing the queries to appropriate tables, the speed of querying and response generation can be increased.
- Query manager is responsible for scheduling the execution of the queries posed by the user.

Query Manager Architecture

The following screenshot shows the architecture of a query manager. It includes the following:

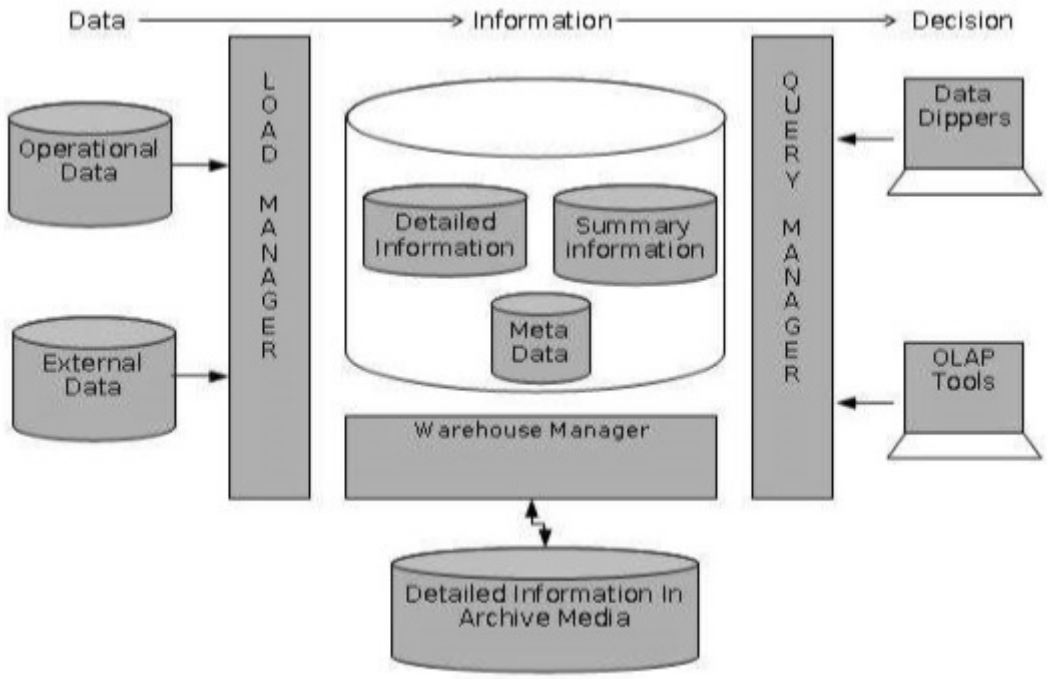
- Query redirection via C tool or RDBMS
- Stored procedures
- Query management tool
- Query scheduling via C tool or RDBMS
- Query scheduling via third-party software



Detailed Information

Detailed information is not kept online, rather it is aggregated to the next level of detail and then archived to tape. The detailed information part of data warehouse keeps the detailed information in the starflake schema. Detailed information is loaded into the data warehouse to supplement the aggregated data.

The following diagram shows a pictorial impression of where detailed information is stored and how it is used.



Note − If detailed information is held offline to minimize disk storage, we should make sure that the data has been extracted, cleaned up, and transformed into starflake schema before it is archived.

Summary Information

Summary Information is a part of data warehouse that stores predefined aggregations. These aggregations are generated by the warehouse manager. Summary Information must be treated as transient. It changes on-the-go in order to respond to the changing query profiles.

The points to note about summary information are as follows −

- Summary information speeds up the performance of common queries.
- It increases the operational cost.
- It needs to be updated whenever new data is loaded into the data warehouse.
- It may not have been backed up, since it can be generated fresh from the detailed information.

Data Warehousing - 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

Relational OLAP

ROLAP servers are placed between relational back-end server and client front-end tools. To store and manage warehouse data, ROLAP uses relational or extended-relational DBMS.

ROLAP includes the following −

- Implementation of aggregation navigation logic.
- Optimization for each DBMS back end.
- Additional tools and services.

Multidimensional OLAP

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. Therefore, many MOLAP server use two levels of data storage representation to handle dense and sparse data sets.

Hybrid OLAP

Hybrid OLAP is a combination of both ROLAP and MOLAP. It offers higher scalability of ROLAP and faster computation of MOLAP. HOLAP servers allows to store the large data volumes of detailed information. The aggregations are stored separately in MOLAP store.

Specialized SQL Servers

Specialized SQL servers provide advanced query language and query processing support for SQL queries over star and snowflake schemas in a read-only environment.

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 −

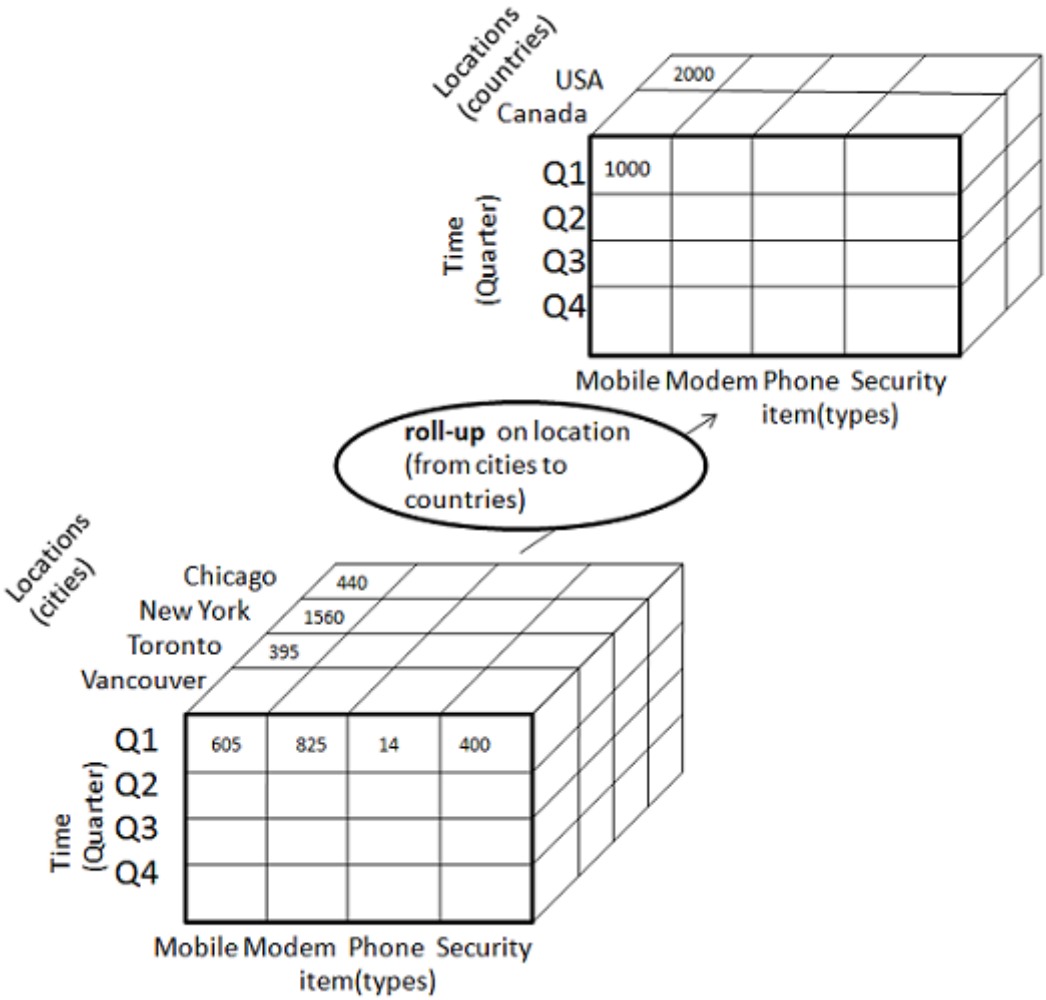
- Roll-up
- Drill-down
- Slice and dice
- Pivot (rotate)

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.



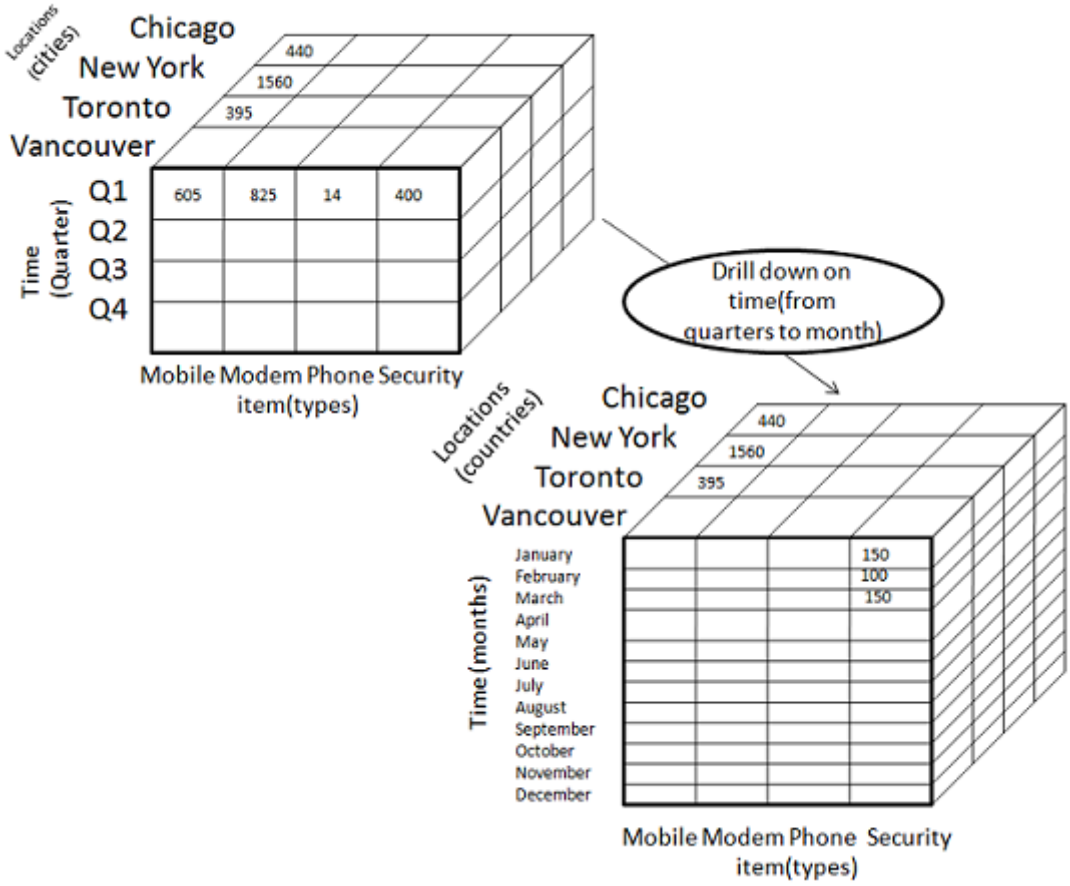
- Roll-up is performed by climbing up a concept hierarchy for the dimension location.
- Initially the concept hierarchy was "street < city < province < country".
- On rolling up, the data is aggregated by ascending the location hierarchy from the level of city to the level of country.
- The data is grouped 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.

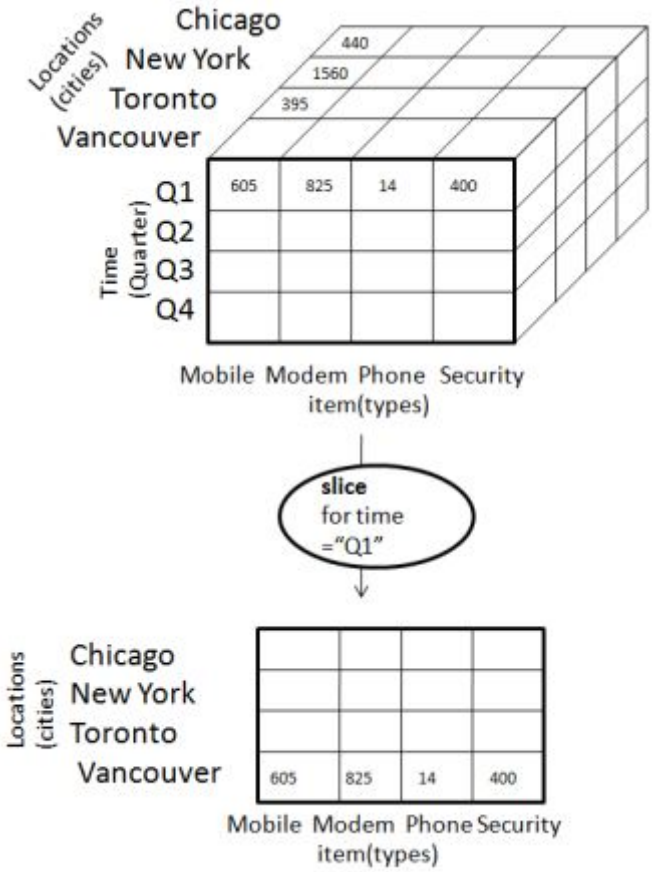
The following diagram illustrates how drill-down works −



- 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

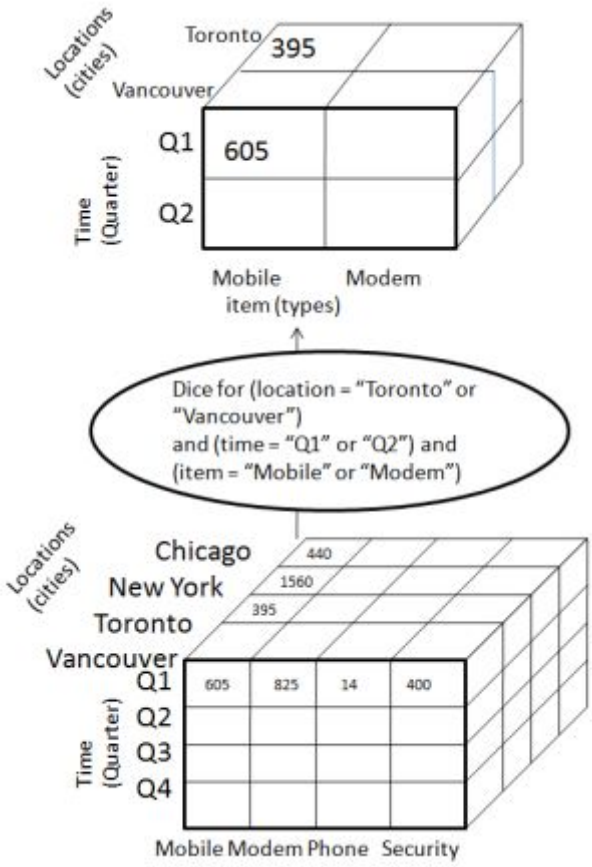
The slice operation selects one particular dimension from a given cube and provides a new sub-cube. Consider the following diagram that shows how slice works.



- 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.

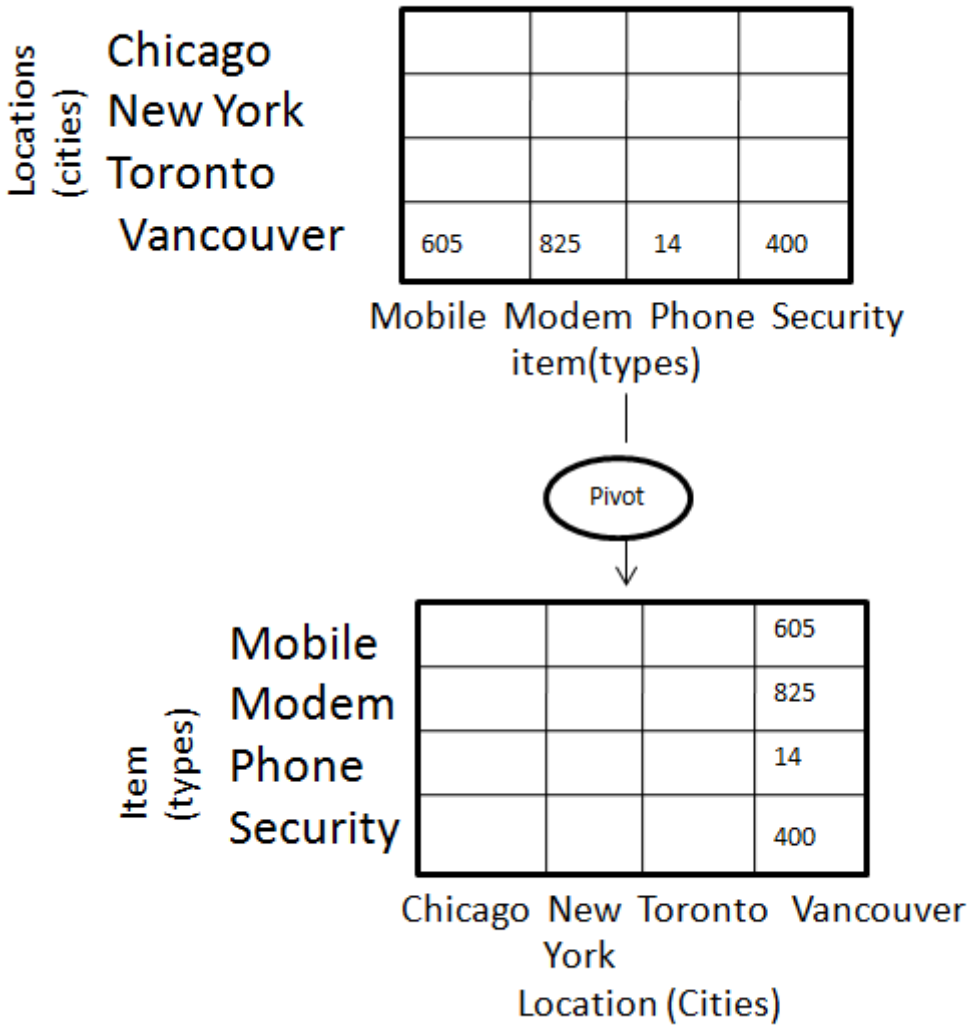


The dice operation on the cube based on the following selection criteria involves three dimensions.

- (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.


Locations (cities)
Chicago
New York
Toronto
Vancouver
605 825 14 400
Mobile Modem Phone Security
item(types)
Pivot
Item (types)
Mobile
Modem
Phone
Security
605
825
14
400
Chicago New York Toronto Vancouver
Location (Cities)

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 Warehousing - Relational OLAP

Relational OLAP servers are placed between relational back-end server and client front-end tools. To store and manage the warehouse data, the relational OLAP uses relational or extended-relational DBMS.

ROLAP includes the following −

- Implementation of aggregation navigation logic
- Optimization for each DBMS back-end
- Additional tools and services

Points to Remember

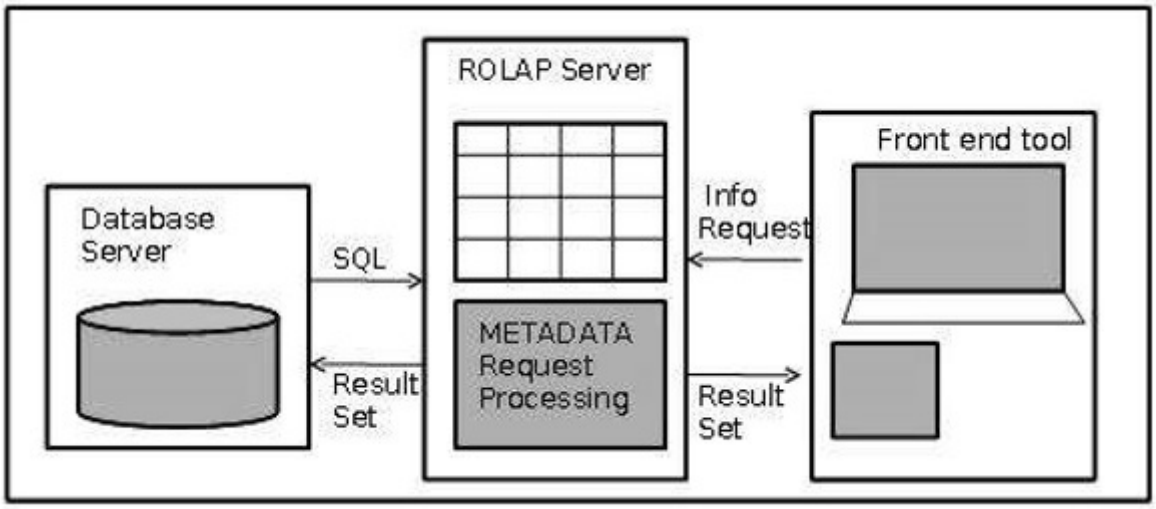
- ROLAP servers are highly scalable.

- ROLAP tools analyze large volumes of data across multiple dimensions.
- ROLAP tools store and analyze highly volatile and changeable data.

Relational OLAP Architecture

ROLAP includes the following components −

- Database server
- ROLAP server
- Front-end tool.



Advantages

- ROLAP servers can be easily used with existing RDBMS.
- Data can be stored efficiently, since no zero facts can be stored.
- ROLAP tools do not use pre-calculated data cubes.
- DSS server of micro-strategy adopts the ROLAP approach.

Disadvantages

- Poor query performance.
- Some limitations of scalability depending on the technology architecture that is utilized.

Data Warehousing - Multidimensional OLAP

Multidimensional OLAP (MOLAP) uses array-based multidimensional storage engines for multidimensional views of data. With multidimensional data stores, the storage utilization may be low if the dataset is sparse. Therefore, many MOLAP servers use two levels of data storage representation to handle dense and sparse datasets.

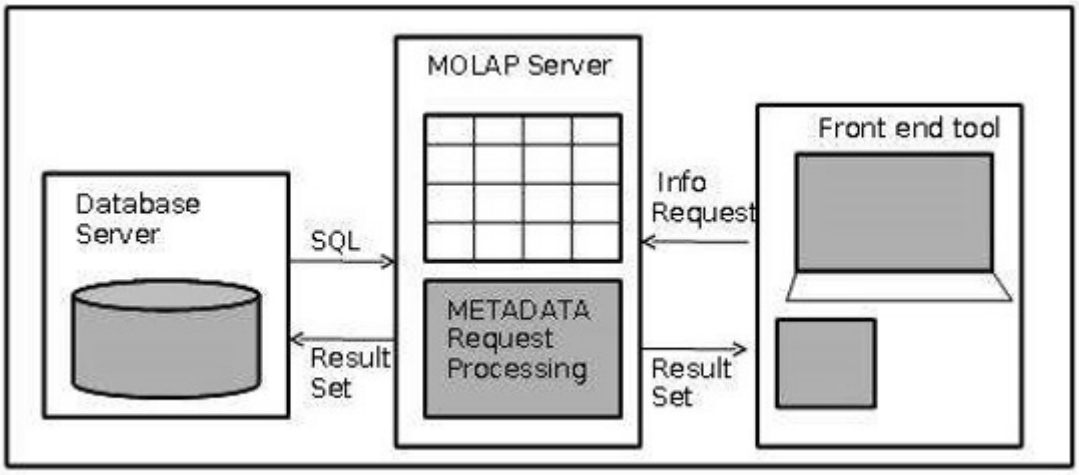
Points to Remember −

- MOLAP tools process information with consistent response time regardless of level of summarizing or calculations selected.
- MOLAP tools need to avoid many of the complexities of creating a relational database to store data for analysis.
- MOLAP tools need fastest possible performance.
- MOLAP server adopts two level of storage representation to handle dense and sparse data sets.
- Denser sub-cubes are identified and stored as array structure.
- Sparse sub-cubes employ compression technology.

MOLAP Architecture

MOLAP includes the following components −

- Database server.
- MOLAP server.
- Front-end tool.



Advantages

- MOLAP allows fastest indexing to the pre-computed summarized data.
- Helps the users connected to a network who need to analyze larger, less-defined data.
- Easier to use, therefore MOLAP is suitable for inexperienced users.

Disadvantages

- MOLAP are not capable of containing detailed data.
- The storage utilization may be low if the data set is sparse.

MOLAP vs ROLAP

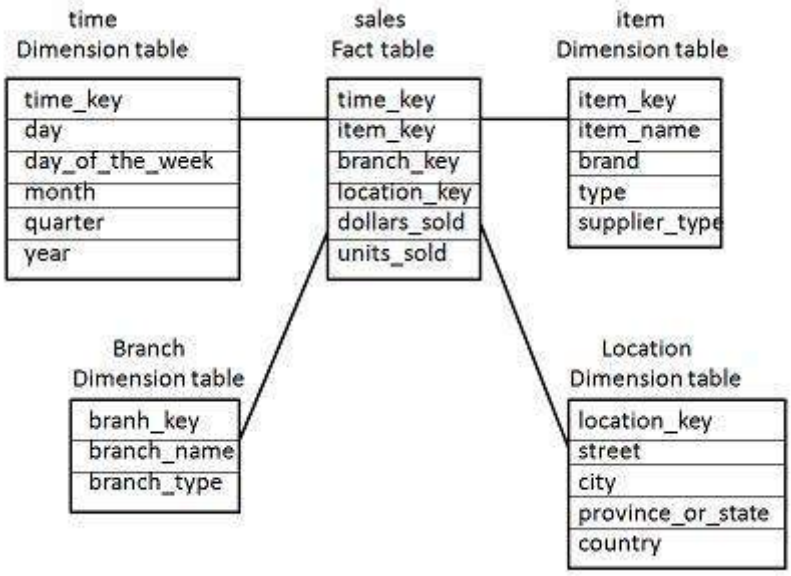
Sr.No.	MOLAP	ROLAP
1	Information retrieval is fast.	Information retrieval is comparatively slow.
2	Uses sparse array to store data-sets.	Uses relational table.
3	MOLAP is best suited for inexperienced users, since it is very easy to use.	ROLAP is best suited for experienced users.
4	Maintains a separate database for data cubes.	It may not require space other than available in the Data warehouse.
5	DBMS facility is weak.	DBMS facility is strong.

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.



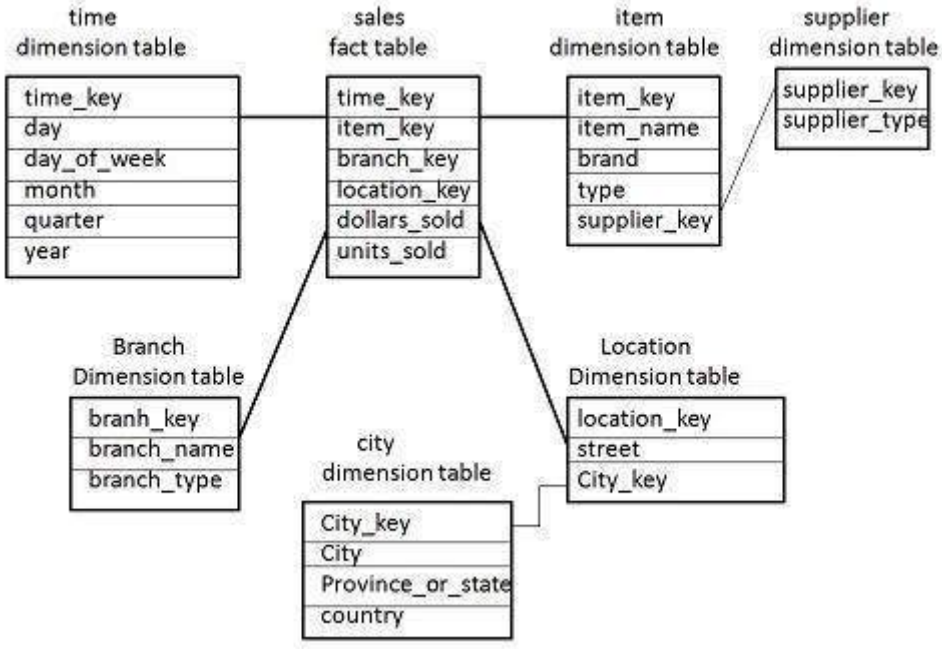
- There is a fact table at the center. It contains the keys to each of four dimensions.

- 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.

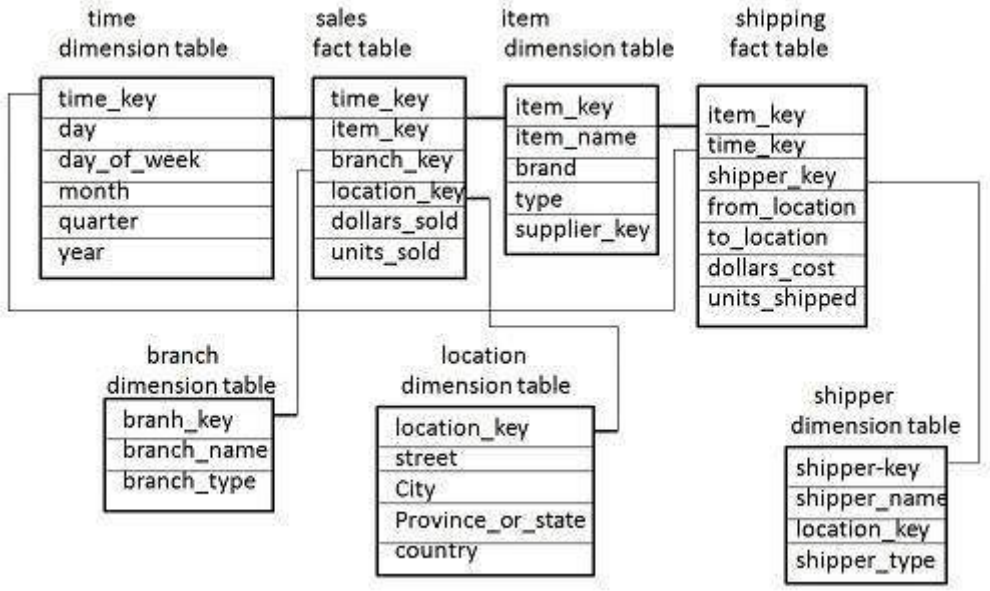


- Now the item dimension table contains the attributes item_key, item_name, type, brand, and supplier-key.
- The supplier key is linked to the supplier dimension table. The supplier dimension table contains the attributes supplier_key and supplier_type.

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.

Schema Definition

Multidimensional schema is defined using Data Mining Query Language (DMQL). The two primitives, cube definition and dimension definition, can be used for defining the data warehouses and data marts.

Syntax for Cube Definition

```
define cube < cube_name > [ < dimension-list > }: < measure_list >
```

Syntax for Dimension Definition

```
define dimension < dimension_name > as ( < attribute_or_dimension_list > )
```

Star Schema Definition

The star schema that we have discussed can be defined using Data Mining Query Language (DMQL) as follows −

```
define cube sales star [time, item, branch, location]:

dollars sold = sum(sales in dollars), units sold = count(*)
```

```
define dimension time as (time key, day, day of week, month, quarter, year)
define dimension item as (item key, item name, brand, type, supplier type)

define dimension branch as (branch key, branch name, branch type)
define dimension location as (location key, street, city, province or state, coun
```

Snowflake Schema Definition

Snowflake schema can be defined using DMQL as follows −

```
define cube sales snowflake [time, item, branch, location]:

dollars sold = sum(sales in dollars), units sold = count(*)

define dimension time as (time key, day, day of week, month, quarter, year)
define dimension item as (item key, item name, brand, type, supplier (supplier ke
define dimension branch as (branch key, branch name, branch type)
define dimension location as (location key, street, city (city key, city, provinc
```

Fact Constellation Schema Definition

Fact constellation schema can be defined using DMQL as follows −

```
define cube sales [time, item, branch, location]:

dollars sold = sum(sales in dollars), units sold = count(*)

define dimension time as (time key, day, day of week, month, quarter, year)
define dimension item as (item key, item name, brand, type, supplier type)
define dimension branch as (branch key, branch name, branch type)
define dimension location as (location key, street, city, province or state,count
define cube shipping [time, item, shipper, from location, to location]:

dollars cost = sum(cost in dollars), units shipped = count(*)

define dimension time as time in cube sales
define dimension item as item in cube sales
define dimension shipper as (shipper key, shipper name, location as location in c
define dimension from location as location in cube sales
define dimension to location as location in cube sales
```

Data Warehousing - Partitioning Strategy

Partitioning is done to enhance performance and facilitate easy management of data. Partitioning also helps in balancing the various requirements of the system. It optimizes the hardware performance and simplifies the management of data warehouse by partitioning each fact table into multiple separate partitions. In this chapter, we will discuss different partitioning strategies.

Why is it Necessary to Partition?

Partitioning is important for the following reasons −

- For easy management,
- To assist backup/recovery,
- To enhance performance.

For Easy Management

The fact table in a data warehouse can grow up to hundreds of gigabytes in size. This huge size of fact table is very hard to manage as a single entity. Therefore it needs partitioning.

To Assist Backup/Recovery

If we do not partition the fact table, then we have to load the complete fact table with all the data. Partitioning allows us to load only as much data as is required on a regular basis. It reduces the time to load and also enhances the performance of the system.

Note − To cut down on the backup size, all partitions other than the current partition can be marked as read-only. We can then put these partitions into a state where they cannot be modified. Then they can be backed up. It means only the current partition is to be backed up.

To Enhance Performance

By partitioning the fact table into sets of data, the query procedures can be enhanced. Query performance is enhanced because now the query scans only those partitions that are relevant. It does not have to scan the whole data.

Horizontal Partitioning

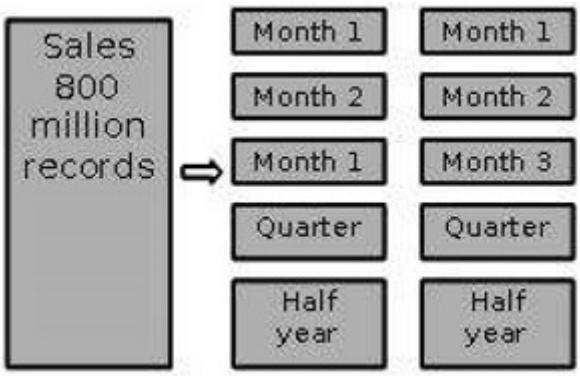
There are various ways in which a fact table can be partitioned. In horizontal partitioning, we have to keep in mind the requirements for manageability of the data warehouse.

Partitioning by Time into Equal Segments

In this partitioning strategy, the fact table is partitioned on the basis of time period. Here each time period represents a significant retention period within the business. For example, if the user queries for **month to date data** then it is appropriate to partition the data into monthly segments. We can reuse the partitioned tables by removing the data in them.

Partition by Time into Different-sized Segments

This kind of partition is done where the aged data is accessed infrequently. It is implemented as a set of small partitions for relatively current data, larger partition for inactive data.



Points to Note

- The detailed information remains available online.
- The number of physical tables is kept relatively small, which reduces the operating cost.
- This technique is suitable where a mix of data dipping recent history and data mining through entire history is required.
- This technique is not useful where the partitioning profile changes on a regular basis, because repartitioning will increase the operation cost of data warehouse.

Partition on a Different Dimension

The fact table can also be partitioned on the basis of dimensions other than time such as product group, region, supplier, or any other dimension. Let's have an example.

Suppose a market function has been structured into distinct regional departments like on a **state by state** basis. If each region wants to query on information captured within its region, it would prove to be more effective to partition the fact table into regional partitions. This will cause the queries to speed up because it does not require to scan information that is not relevant.

Points to Note

- The query does not have to scan irrelevant data which speeds up the query process.
- This technique is not appropriate where the dimensions are unlikely to change in future. So, it is worth determining that the dimension does not change in future.
- If the dimension changes, then the entire fact table would have to be repartitioned.

Note − We recommend to perform the partition only on the basis of time dimension, unless you are certain that the suggested dimension grouping will not change within the life of the data warehouse.

Partition by Size of Table

When there are no clear basis for partitioning the fact table on any dimension, then we should **partition the fact table on the basis of their size.** We can set the predetermined size as a critical point. When the table exceeds the predetermined size, a new table partition is created.

Points to Note

- This partitioning is complex to manage.
- It requires metadata to identify what data is stored in each partition.

Partitioning Dimensions

If a dimension contains large number of entries, then it is required to partition the dimensions. Here we have to check the size of a dimension.

Consider a large design that changes over time. If we need to store all the variations in order to apply comparisons, that dimension may be very large. This would definitely affect the response time.

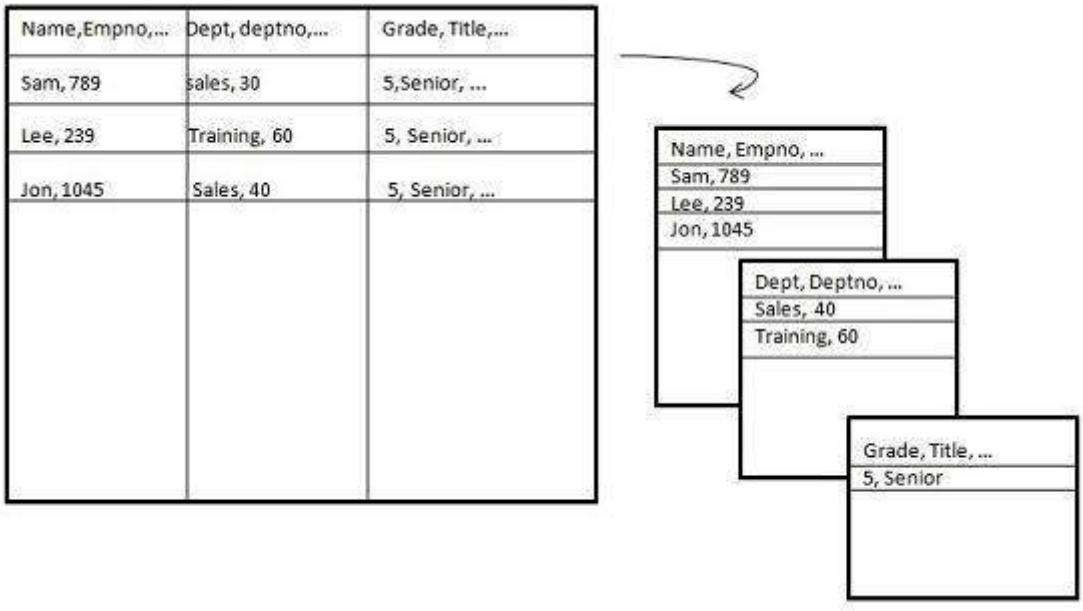
Round Robin Partitions

In the round robin technique, when a new partition is needed, the old one is archived. It uses metadata to allow user access tool to refer to the correct table partition.

This technique makes it easy to automate table management facilities within the data warehouse.

Vertical Partition

Vertical partitioning, splits the data vertically. The following images depicts how vertical partitioning is done.



Vertical partitioning can be performed in the following two ways −

- Normalization

- Row Splitting

Normalization

Normalization is the standard relational method of database organization. In this method, the rows are collapsed into a single row, hence it reduce space. Take a look at the following tables that show how normalization is performed.

Table before Normalization

Product_id	Qty	Value	sales_date	Store_id	Store_name	Location	Region
30	5	3.67	3-Aug-13	16	sunny	Bangalore	S
35	4	5.33	3-Sep-13	16	sunny	Bangalore	S
40	5	2.50	3-Sep-13	64	san	Mumbai	W
45	7	5.66	3-Sep-13	16	sunny	Bangalore	S

Table after Normalization

Store_id	Store_name	Location	Region
16	sunny	Bangalore	W
64	san	Mumbai	S

Product_id	Quantity	Value	sales_date	Store_id
30	5	3.67	3-Aug-13	16
35	4	5.33	3-Sep-13	16
40	5	2.50	3-Sep-13	64
45	7	5.66	3-Sep-13	16

Row Splitting

Row splitting tends to leave a one-to-one map between partitions. The motive of row splitting is to speed up the access to large table by reducing its size.

Note − While using vertical partitioning, make sure that there is no requirement to perform a major join operation between two partitions.

Identify Key to Partition

It is very crucial to choose the right partition key. Choosing a wrong partition key will lead to reorganizing the fact table. Let's have an example. Suppose we want to partition the following table.

```
Account_Txn_Table
transaction_id
account_id
transaction_type
value
transaction_date
region
branch_name
```

We can choose to partition on any key. The two possible keys could be

- region
- transaction_date

Suppose the business is organized in 30 geographical regions and each region has different number of branches. That will give us 30 partitions, which is reasonable. This partitioning is good enough because our requirements capture has shown that a vast majority of queries are restricted to the user's own business region.

If we partition by transaction_date instead of region, then the latest transaction from every region will be in one partition. Now the user who wants to look at data within his own region has to query across multiple partitions.

Hence it is worth determining the right partitioning key.

Data Warehousing - Metadata Concepts

What is Metadata?

Metadata is simply defined as data about data. The data that is 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 detailed data. In terms of data warehouse, we can define metadata as follows.

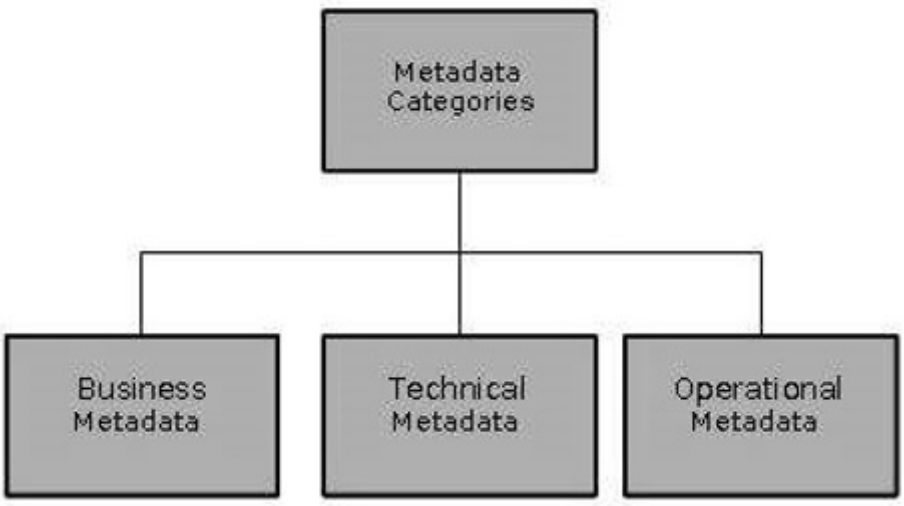
- Metadata is the road-map to a data warehouse.
- Metadata in a data warehouse defines the warehouse objects.
- Metadata acts as a directory. This directory helps the decision support system to locate the contents of a data warehouse.

Note − In a data warehouse, we create metadata for the data names and definitions of a given data warehouse. Along with this metadata, additional metadata is also created for time-stamping any extracted data, the source of extracted data.

Categories of Metadata

Metadata can be broadly categorized into three categories −

- **Business Metadata** − It has the data ownership information, business definition, and changing policies.
- **Technical Metadata** − It includes database system names, table and column names and sizes, data types and allowed values. Technical metadata also includes structural information such as primary and foreign key attributes and indices.
- **Operational Metadata** − It includes currency of data and data lineage. Currency of data means whether the data is active, archived, or purged. Lineage of data means the history of data migrated and transformation applied on it.

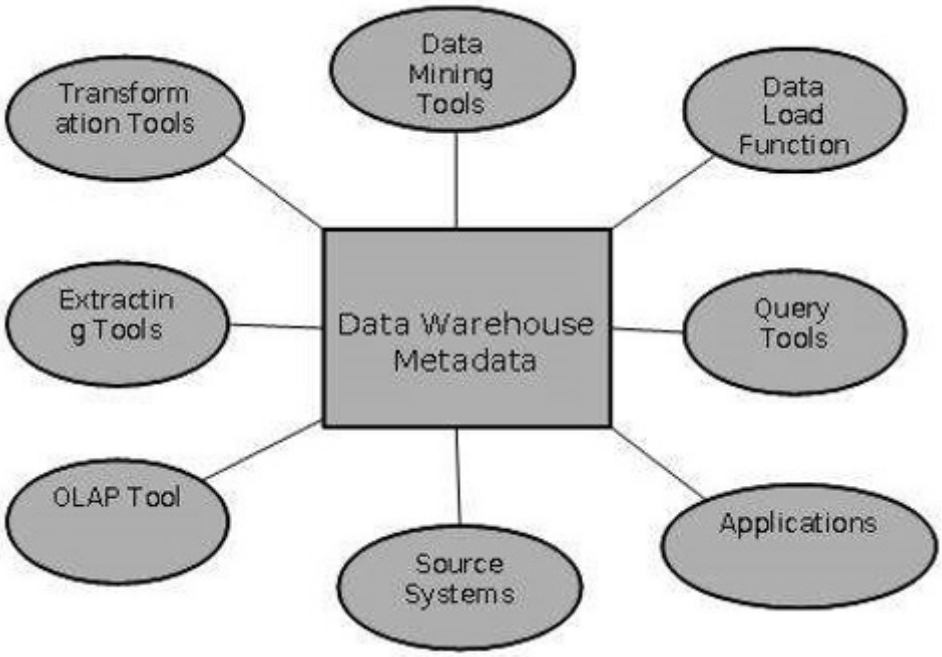


Role of Metadata

Metadata has a very important role in a data warehouse. The role of metadata in a warehouse is different from the warehouse data, yet it plays an important role. The various roles of metadata are explained below.

- Metadata acts as a directory.
- This directory helps the decision support system to locate the contents of the data warehouse.
- Metadata helps in decision support system for mapping of data when data is transformed from operational environment to data warehouse environment.
- Metadata helps in summarization between current detailed data and highly summarized data.
- Metadata also helps in summarization between lightly detailed data and highly summarized data.
- Metadata is used for query tools.
- Metadata is used in extraction and cleansing tools.
- Metadata is used in reporting tools.
- Metadata is used in transformation tools.
- Metadata plays an important role in loading functions.

The following diagram shows the roles of metadata.



Metadata Repository

Metadata repository is an integral part of a data warehouse system. It has the following metadata −

- **Definition of data warehouse** − It includes the description of structure of data warehouse. The description is defined by schema, view, hierarchies, derived data definitions, and data mart locations and contents.
- **Business metadata** − It contains has the data ownership information, business definition, and changing policies.
- **Operational Metadata** − It includes currency of data and data lineage. Currency of data means whether the data is active, archived, or purged. Lineage of data means the history of data migrated and transformation applied on it.
- **Data for mapping from operational environment to data warehouse** − It includes the source databases and their contents, data extraction, data partition cleaning, transformation rules, data refresh and purging rules.
- **Algorithms for summarization** − It includes dimension algorithms, data on granularity, aggregation, summarizing, etc.

Challenges for Metadata Management

The importance of metadata can not be overstated. Metadata helps in driving the accuracy of reports, validates data transformation, and ensures the accuracy of calculations. Metadata also enforces the definition of business terms to business end-users. With all these uses of metadata, it also has its challenges. Some of the challenges are discussed below.

- Metadata in a big organization is scattered across the organization. This metadata is spread in spreadsheets, databases, and applications.

- Metadata could be present in text files or multimedia files. To use this data for information management solutions, it has to be correctly defined.
- There are no industry-wide accepted standards. Data management solution vendors have narrow focus.
- There are no easy and accepted methods of passing metadata.

Data Warehousing - Data Marting

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-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

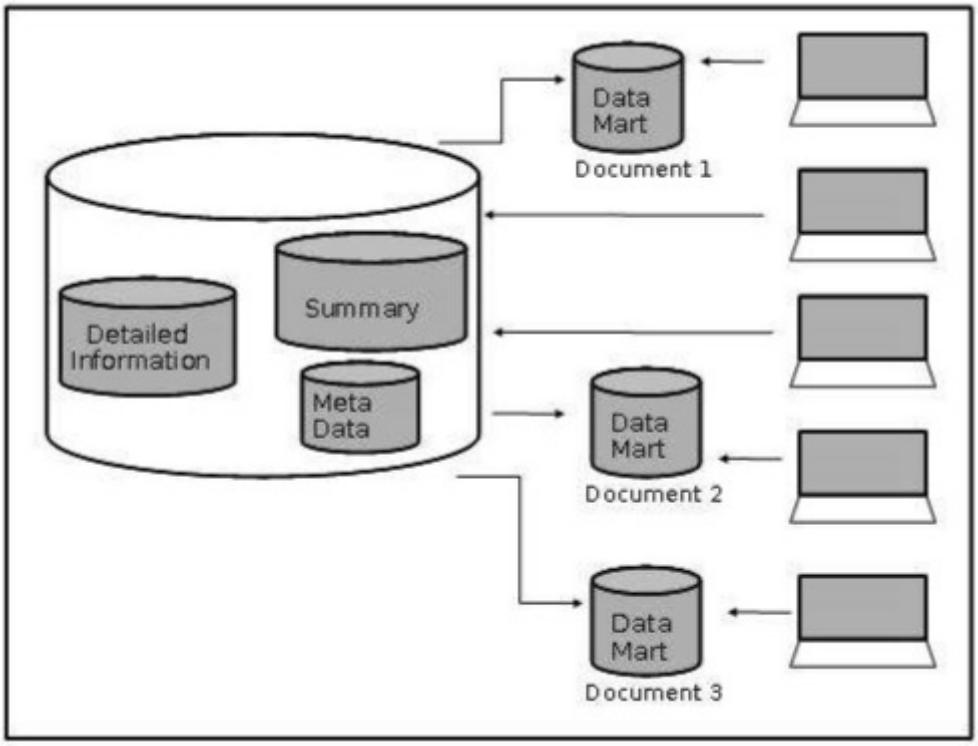
Identify the Functional Splits

In this step, we determine if the organization has natural functional splits. We look for departmental splits, and we determine whether the way in which departments use information tend to be in isolation from the rest of the organization. Let's have an example.

Consider a retail organization, where each merchant is accountable for maximizing the sales of a group of products. For this, the following are the valuable information −

- sales transaction on a daily basis
- sales forecast on a weekly basis
- stock position on a daily basis
- stock movements on a daily basis

As the merchant is not interested in the products they are not dealing with, the data marting is a subset of the data dealing which the product group of interest. The following diagram shows data marting for different users.



Given below are the issues to be taken into account while determining the functional split −

- The structure of the department may change.
- The products might switch from one department to other.
- The merchant could query the sales trend of other products to analyze what is happening to the sales.

Note − We need to determine the business benefits and technical feasibility of using a data mart.

Identify User Access Tool Requirements

We need data marts to support **user access tools** that require internal data structures. The data in such structures are outside the control of data warehouse but need to be populated and updated on a regular basis.

There are some tools that populate directly from the source system but some cannot. Therefore additional requirements outside the scope of the tool are needed to be identified for future.

Note − In order to ensure consistency of data across all access tools, the data should not be directly populated from the data warehouse, rather each tool must have its own data mart.

Identify Access Control Issues

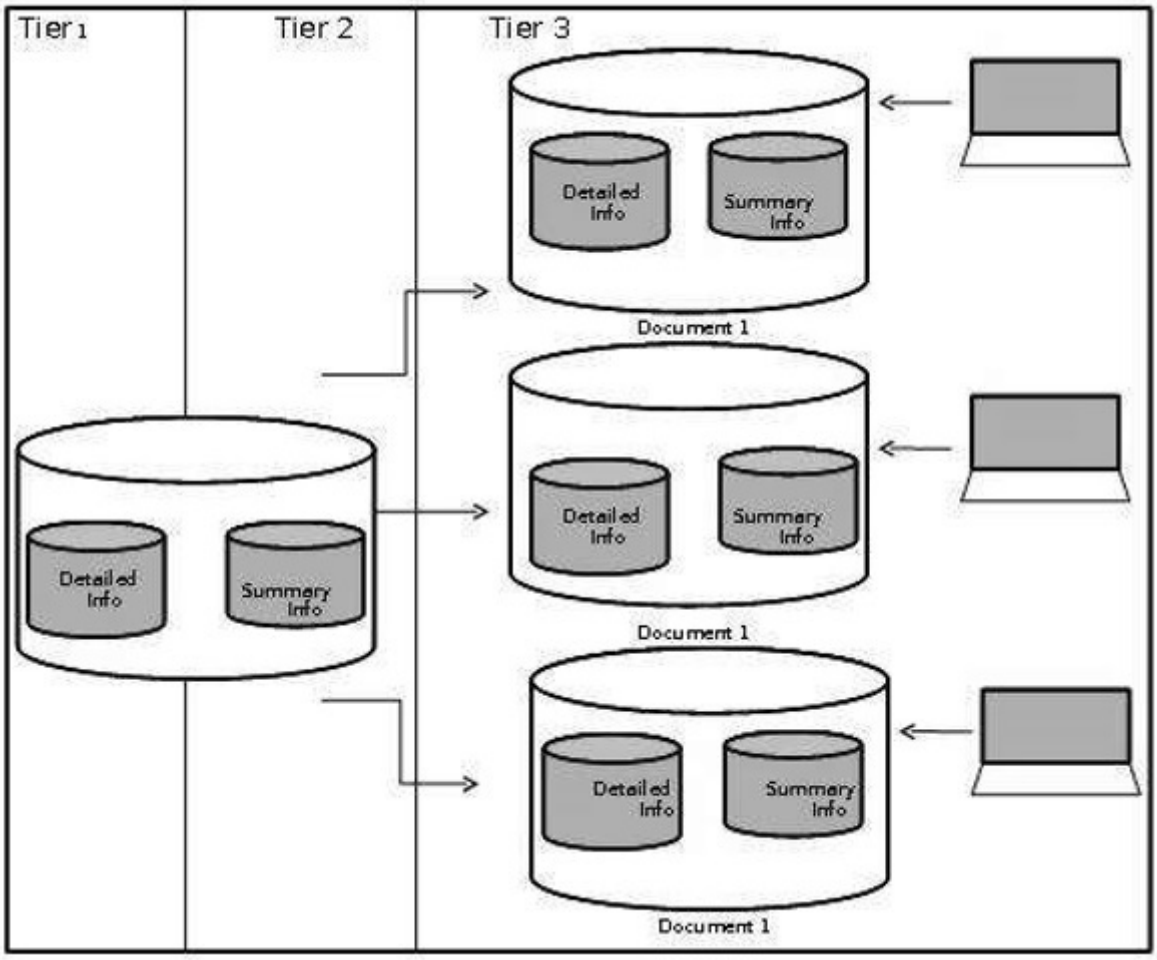
There should to be privacy rules to ensure the data is accessed by authorized users only. For example a data warehouse for retail banking institution ensures that all the accounts belong to the same legal entity. Privacy laws can force you to totally prevent access to information that is not owned by the specific bank.

Data marts allow us to build a complete wall by physically separating data segments within the data warehouse. To avoid possible privacy problems, the detailed data can be removed from the

data warehouse. We can create data mart for each legal entity and load it via data warehouse, with detailed account data.

Designing Data Marts

Data marts should be designed as a smaller version of starflake schema within the data warehouse and should match with the database design of the data warehouse. It helps in maintaining control over database instances.



The summaries are data marted in the same way as they would have been designed within the data warehouse. Summary tables help to utilize all dimension data in the starflake schema.

Cost of Data Marting

The cost measures for data marting are as follows −

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

Hardware and Software Cost

Although data marts are created on the same hardware, they require some additional hardware and software. To handle user queries, it requires additional processing power and disk storage.

If detailed data and the data mart exist within the data warehouse, then we would face additional cost to store and manage replicated data.

Note − Data marting is more expensive than aggregations, therefore it should be used as an additional strategy and not as an alternative strategy.

Network Access

A data mart could be on a different location from the data warehouse, so we should ensure that the LAN or WAN has the capacity to handle the data volumes being transferred within the **data mart load process.**

Time Window Constraints

The extent to which a data mart loading process will eat into the available time window depends on the complexity of the transformations and the data volumes being shipped. The determination of how many data marts are possible depends on −

- Network capacity.
- Time window available
- Volume of data being transferred
- Mechanisms being used to insert data into a data mart

Data Warehousing - System Managers

System management is mandatory for the successful implementation of a data warehouse. The most important system managers are −

- System configuration manager
- System scheduling manager
- System event manager
- System database manager
- System backup recovery manager

System Configuration Manager

- The system configuration manager is responsible for the management of the setup and configuration of data warehouse.
- The structure of configuration manager varies from one operating system to another.
- In Unix structure of configuration, the manager varies from vendor to vendor.
- Configuration managers have single user interface.
- The interface of configuration manager allows us to control all aspects of the system.

Note − The most important configuration tool is the I/O manager.

System Scheduling Manager

System Scheduling Manager is responsible for the successful implementation of the data warehouse. Its purpose is to schedule ad hoc queries. Every operating system has its own scheduler with some form of batch control mechanism. The list of features a system scheduling manager must have is as follows −

- Work across cluster or MPP boundaries
- Deal with international time differences
- Handle job failure
- Handle multiple queries
- Support job priorities
- Restart or re-queue the failed jobs
- Notify the user or a process when job is completed
- Maintain the job schedules across system outages
- Re-queue jobs to other queues
- Support the stopping and starting of queues
- Log Queued jobs
- Deal with inter-queue processing

Note − The above list can be used as evaluation parameters for the evaluation of a good scheduler.

Some important jobs that a scheduler must be able to handle are as follows −

- Daily and ad hoc query scheduling
- Execution of regular report requirements
- Data load
- Data processing
- Index creation
- Backup
- Aggregation creation
- Data transformation

Note − If the data warehouse is running on a cluster or MPP architecture, then the system scheduling manager must be capable of running across the architecture.

System Event Manager

The event manager is a kind of a software. The event manager manages the events that are defined on the data warehouse system. We cannot manage the data warehouse manually because the structure of data warehouse is very complex. Therefore we need a tool that automatically handles all the events without any intervention of the user.

Note − The Event manager monitors the events occurrences and deals with them. The event manager also tracks the myriad of things that can go wrong on this complex data warehouse system.

Events

Events are the actions that are generated by the user or the system itself. It may be noted that the event is a measurable, observable, occurrence of a defined action.

Given below is a list of common events that are required to be tracked.

- Hardware failure
- Running out of space on certain key disks
- A process dying
- A process returning an error
- CPU usage exceeding an 805 threshold
- Internal contention on database serialization points
- Buffer cache hit ratios exceeding or failure below threshold
- A table reaching to maximum of its size
- Excessive memory swapping
- A table failing to extend due to lack of space
- Disk exhibiting I/O bottlenecks
- Usage of temporary or sort area reaching a certain thresholds
- Any other database shared memory usage

The most important thing about events is that they should be capable of executing on their own. Event packages define the procedures for the predefined events. The code associated with each event is known as event handler. This code is executed whenever an event occurs.

System and Database Manager

System and database manager may be two separate pieces of software, but they do the same job. The objective of these tools is to automate certain processes and to simplify the execution of others. The criteria for choosing a system and the database manager are as follows −

- increase user's quota.
- assign and de-assign roles to the users
- assign and de-assign the profiles to the users
- perform database space management
- monitor and report on space usage
- tidy up fragmented and unused space
- add and expand the space
- add and remove users
- manage user password
- manage summary or temporary tables
- assign or deassign temporary space to and from the user
- reclaim the space form old or out-of-date temporary tables
- manage error and trace logs

- to browse log and trace files
- redirect error or trace information
- switch on and off error and trace logging
- perform system space management
- monitor and report on space usage
- clean up old and unused file directories
- add or expand space.

System Backup Recovery Manager

The backup and recovery tool makes it easy for operations and management staff to back-up the data. Note that the system backup manager must be integrated with the schedule manager software being used. The important features that are required for the management of backups are as follows −

- Scheduling
- Backup data tracking
- Database awareness

Backups are taken only to protect against data loss. Following are the important points to remember −

- The backup software will keep some form of database of where and when the piece of data was backed up.
- The backup recovery manager must have a good front-end to that database.
- The backup recovery software should be database aware.
- Being aware of the database, the software then can be addressed in database terms, and will not perform backups that would not be viable.

Data Warehousing - Process Managers

Process managers are responsible for maintaining the flow of data both into and out of the data warehouse. There are three different types of process managers −

- Load manager
- Warehouse manager
- Query manager

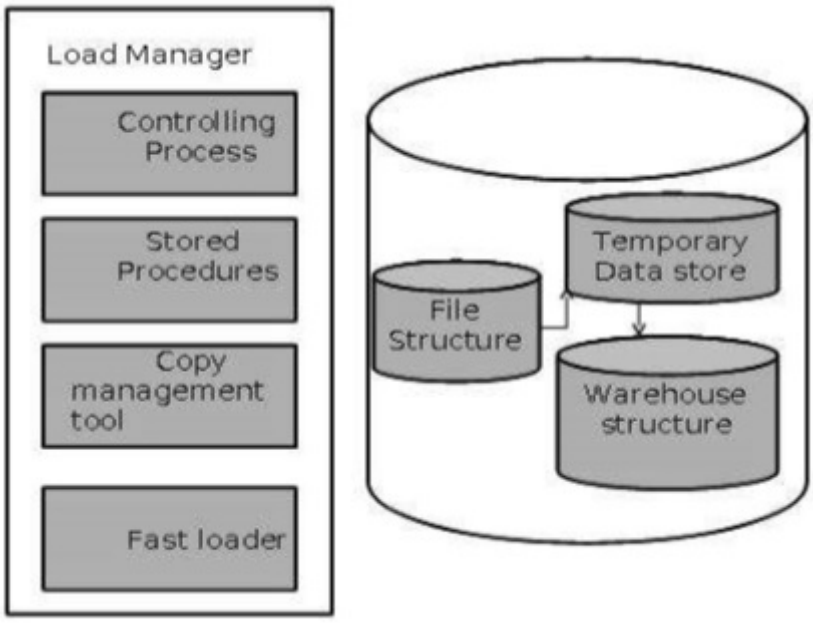
Data Warehouse Load Manager

Load manager performs the operations required to extract and load the data into the database. The size and complexity of a load manager varies between specific solutions from one data warehouse to another.

Load Manager Architecture

The load manager does performs the following functions −

- Extract data from the source system.
- Fast load the extracted data into temporary data store.
- Perform simple transformations into structure similar to the one in the data warehouse.



Extract Data from Source

The data is extracted from the operational databases or the external information providers. Gateways are the application programs that are used to extract data. It is supported by underlying DBMS and allows the client program to generate SQL to be executed at a server. Open Database Connection (ODBC) and Java Database Connection (JDBC) are examples of gateway.

Fast Load

- In order to minimize the total load window, the data needs to be loaded into the warehouse in the fastest possible time.
- Transformations affect the speed of data processing.
- It is more effective to load the data into a relational database prior to applying transformations and checks.
- Gateway technology is not suitable, since they are inefficient when large data volumes are involved.

Simple Transformations

While loading, it may be required to perform simple transformations. After completing simple transformations, we can do complex checks. Suppose we are loading the EPOS sales transaction, we need to perform the following checks −

- Strip out all the columns that are not required within the warehouse.

- Convert all the values to required data types.

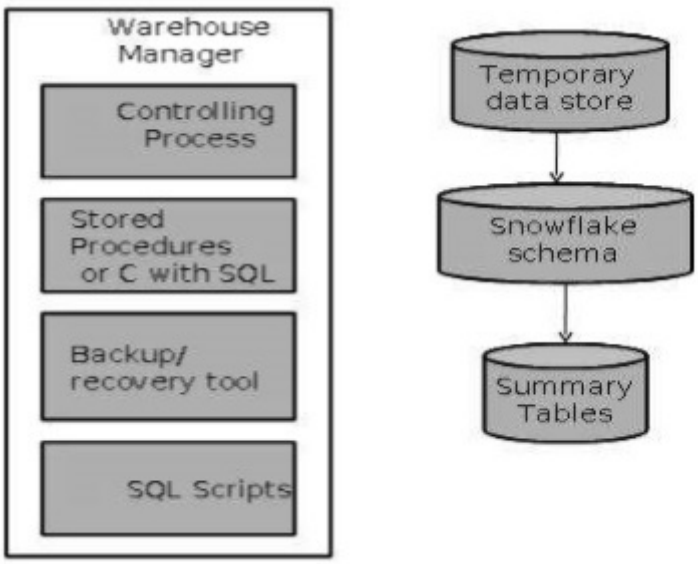
Warehouse Manager

The warehouse manager is responsible for the warehouse management process. It consists of a third-party system software, C programs, and shell scripts. The size and complexity of a warehouse manager varies between specific solutions.

Warehouse Manager Architecture

A warehouse manager includes the following −

- The controlling process
- Stored procedures or C with SQL
- Backup/Recovery tool
- SQL scripts



Functions of Warehouse Manager

A warehouse manager performs the following functions −

- Analyzes the data to perform consistency and referential integrity checks.
- Creates indexes, business views, partition views against the base data.
- Generates new aggregations and updates the existing aggregations.
- Generates normalizations.
- Transforms and merges the source data of the temporary store into the published data warehouse.
- Backs up the data in the data warehouse.
- Archives the data that has reached the end of its captured life.

Note − A warehouse Manager analyzes query profiles to determine whether the index and aggregations are appropriate.

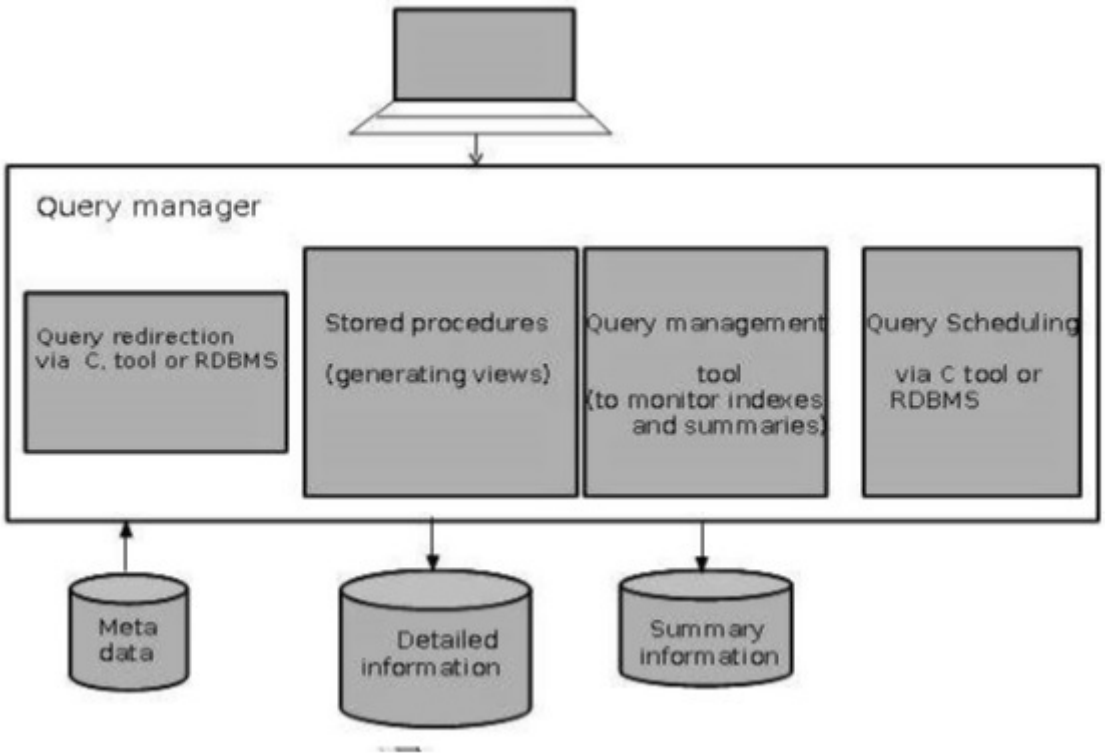
Query Manager

The query manager is responsible for directing the queries to suitable tables. By directing the queries to appropriate tables, it speeds up the query request and response process. In addition, the query manager is responsible for scheduling the execution of the queries posted by the user.

Query Manager Architecture

A query manager includes the following components −

- Query redirection via C tool or RDBMS
- Stored procedures
- Query management tool
- Query scheduling via C tool or RDBMS
- Query scheduling via third-party software



Functions of Query Manager

- It presents the data to the user in a form they understand.
- It schedules the execution of the queries posted by the end-user.
- It stores query profiles to allow the warehouse manager to determine which indexes and aggregations are appropriate.

Data Warehousing - Security

The objective of a data warehouse is to make large amounts of data easily accessible to the users, hence allowing the users to extract information about the business as a whole. But we know that there could be some security restrictions applied on the data that can be an obstacle for accessing the information. If the analyst has a restricted view of data, then it is impossible to capture a complete picture of the trends within the business.

The data from each analyst can be summarized and passed on to management where the different summaries can be aggregated. As the aggregations of summaries cannot be the same as that of the aggregation as a whole, it is possible to miss some information trends in the data unless someone is analyzing the data as a whole.

Security Requirements

Adding security features affect the performance of the data warehouse, therefore it is important to determine the security requirements as early as possible. It is difficult to add security features after the data warehouse has gone live.

During the design phase of the data warehouse, we should keep in mind what data sources may be added later and what would be the impact of adding those data sources. We should consider the following possibilities during the design phase.

- Whether the new data sources will require new security and/or audit restrictions to be implemented?
- Whether the new users added who have restricted access to data that is already generally available?

This situation arises when the future users and the data sources are not well known. In such a situation, we need to use the knowledge of business and the objective of data warehouse to know likely requirements.

The following activities get affected by security measures −

- User access
- Data load
- Data movement
- Query generation

User Access

We need to first classify the data and then classify the users on the basis of the data they can access. In other words, the users are classified according to the data they can access.

Data Classification

The following two approaches can be used to classify the data −

- Data can be classified according to its sensitivity. Highly-sensitive data is classified as highly restricted and less-sensitive data is classified as less restrictive.
- Data can also be classified according to the job function. This restriction allows only specific users to view particular data. Here we restrict the users to view only that part of the data in which they are interested and are responsible for.

There are some issues in the second approach. To understand, let's have an example. Suppose you are building the data warehouse for a bank. Consider that the data being stored in the data warehouse is the transaction data for all the accounts. The question here is, who is allowed to see the transaction data. The solution lies in classifying the data according to the function.

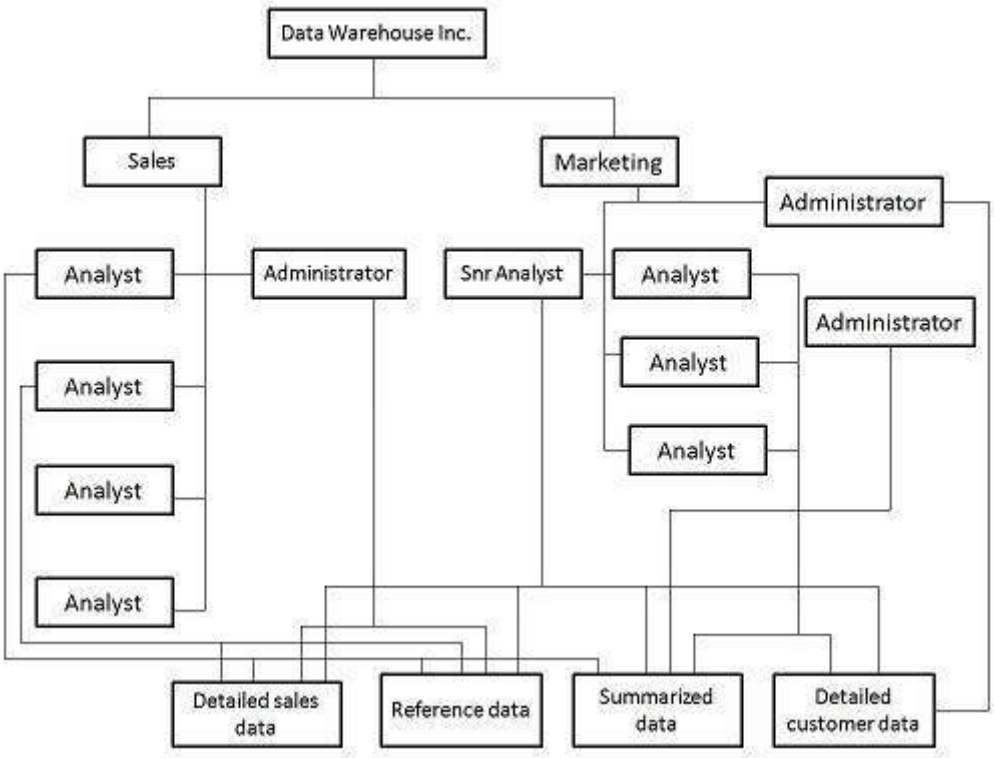
User classification

The following approaches can be used to classify the users −

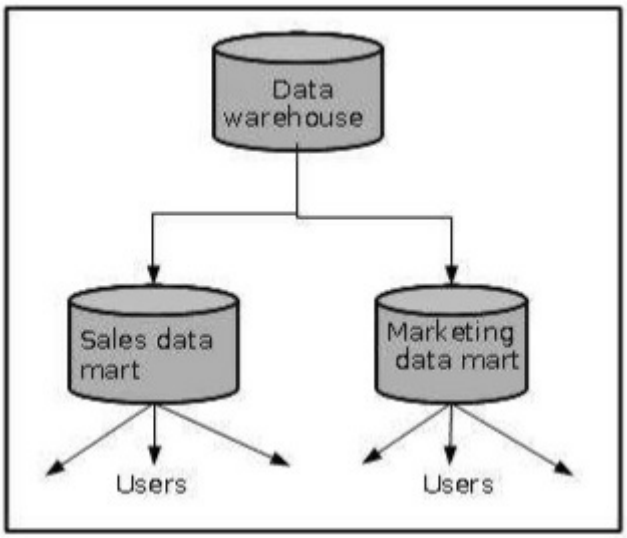
- Users can be classified as per the hierarchy of users in an organization, i.e., users can be classified by departments, sections, groups, and so on.
- Users can also be classified according to their role, with people grouped across departments based on their role.

Classification on basis of Department

Let's have an example of a data warehouse where the users are from sales and marketing department. We can have security by top-to-down company view, with access centered on the different departments. But there could be some restrictions on users at different levels. This structure is shown in the following diagram.

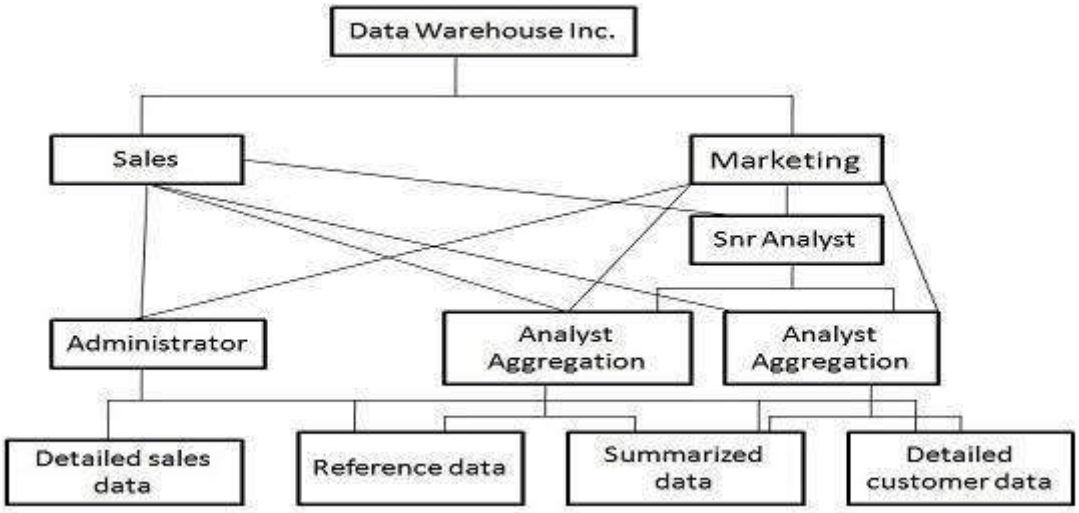


But if each department accesses different data, then we should design the security access for each department separately. This can be achieved by departmental data marts. Since these data marts are separated from the data warehouse, we can enforce separate security restrictions on each data mart. This approach is shown in the following figure.



Classification Based on Role

If the data is generally available to all the departments, then it is useful to follow the role access hierarchy. In other words, if the data is generally accessed by all the departments, then apply security restrictions as per the role of the user. The role access hierarchy is shown in the following figure.



Audit Requirements

Auditing is a subset of security, a costly activity. Auditing can cause heavy overheads on the system. To complete an audit in time, we require more hardware and therefore, it is recommended that wherever possible, auditing should be switched off. Audit requirements can be categorized as follows −

- Connections
- Disconnections
- Data access
- Data change

Note − For each of the above-mentioned categories, it is necessary to audit success, failure, or both. From the perspective of security reasons, the auditing of failures are very important.

Auditing of failure is important because they can highlight unauthorized or fraudulent access.

Network Requirements

Network security is as important as other securities. We cannot ignore the network security requirement. We need to consider the following issues −

- Is it necessary to encrypt data before transferring it to the data warehouse?
- Are there restrictions on which network routes the data can take?

These restrictions need to be considered carefully. Following are the points to remember −

- The process of encryption and decryption will increase overheads. It would require more processing power and processing time.
- The cost of encryption can be high if the system is already a loaded system because the encryption is borne by the source system.

Data Movement

There exist potential security implications while moving the data. Suppose we need to transfer some restricted data as a flat file to be loaded. When the data is loaded into the data warehouse, the following questions are raised −

- Where is the flat file stored?
- Who has access to that disk space?

If we talk about the backup of these flat files, the following questions are raised −

- Do you backup encrypted or decrypted versions?
- Do these backups need to be made to special tapes that are stored separately?
- Who has access to these tapes?

Some other forms of data movement like query result sets also need to be considered. The questions raised while creating the temporary table are as follows −

- Where is that temporary table to be held?
- How do you make such table visible?

We should avoid the accidental flouting of security restrictions. If a user with access to the restricted data can generate accessible temporary tables, data can be visible to non-authorized users. We can overcome this problem by having a separate temporary area for users with access to restricted data.

Documentation

The audit and security requirements need to be properly documented. This will be treated as a part of justification. This document can contain all the information gathered from −

- Data classification
- User classification
- Network requirements

- Data movement and storage requirements
- All auditable actions

Impact of Security on Design

Security affects the application code and the development timescales. Security affects the following area −

- Application development
- Database design
- Testing

Application Development

Security affects the overall application development and it also affects the design of the important components of the data warehouse such as load manager, warehouse manager, and query manager. The load manager may require checking code to filter record and place them in different locations. More transformation rules may also be required to hide certain data. Also there may be requirements of extra metadata to handle any extra objects.

To create and maintain extra views, the warehouse manager may require extra codes to enforce security. Extra checks may have to be coded into the data warehouse to prevent it from being fooled into moving data into a location where it should not be available. The query manager requires the changes to handle any access restrictions. The query manager will need to be aware of all extra views and aggregations.

Database design

The database layout is also affected because when security measures are implemented, there is an increase in the number of views and tables. Adding security increases the size of the database and hence increases the complexity of the database design and management. It will also add complexity to the backup management and recovery plan.

Testing

Testing the data warehouse is a complex and lengthy process. Adding security to the data warehouse also affects the testing time complexity. It affects the testing in the following two ways −

- It will increase the time required for integration and system testing.
- There is added functionality to be tested which will increase the size of the testing suite.

Data Warehousing - Backup

A data warehouse is a complex system and it contains a huge volume of data. Therefore it is important to back up all the data so that it becomes available for recovery in future as per requirement. In this chapter, we will discuss the issues in designing the backup strategy.

Backup Terminologies

Before proceeding further, you should know some of the backup terminologies discussed below.

- **Complete backup** − It backs up the entire database at the same time. This backup includes all the database files, control files, and journal files.
- **Partial backup** − As the name suggests, it does not create a complete backup of the database. Partial backup is very useful in large databases because they allow a strategy whereby various parts of the database are backed up in a round-robin fashion on a day-to-day basis, so that the whole database is backed up effectively once a week.
- **Cold backup** − Cold backup is taken while the database is completely shut down. In multi-instance environment, all the instances should be shut down.
- **Hot backup** − Hot backup is taken when the database engine is up and running. The requirements of hot backup varies from RDBMS to RDBMS.
- **Online backup** − It is quite similar to hot backup.

Hardware Backup

It is important to decide which hardware to use for the backup. The speed of processing the backup and restore depends on the hardware being used, how the hardware is connected, bandwidth of the network, backup software, and the speed of server's I/O system. Here we will discuss some of the hardware choices that are available and their pros and cons. These choices are as follows −

- Tape Technology
- Disk Backups

Tape Technology

The tape choice can be categorized as follows −

- Tape media
- Standalone tape drives
- Tape stackers
- Tape silos

Tape Media

There exists several varieties of tape media. Some tape media standards are listed in the table below −

Tape Media	Capacity	I/O rates
DLT	40 GB	3 MB/s
3490e	1.6 GB	3 MB/s
8 mm	14 GB	1 MB/s

Other factors that need to be considered are as follows −

- Reliability of the tape medium
- Cost of tape medium per unit
- Scalability
- Cost of upgrades to tape system
- Cost of tape medium per unit
- Shelf life of tape medium

Standalone Tape Drives

The tape drives can be connected in the following ways −

- Direct to the server
- As network available devices
- Remotely to other machine

There could be issues in connecting the tape drives to a data warehouse.

- Consider the server is a 48node MPP machine. We do not know the node to connect the tape drive and we do not know how to spread them over the server nodes to get the optimal performance with least disruption of the server and least internal I/O latency.
- Connecting the tape drive as a network available device requires the network to be up to the job of the huge data transfer rates. Make sure that sufficient bandwidth is available during the time you require it.
- Connecting the tape drives remotely also require high bandwidth.

Tape Stackers

The method of loading multiple tapes into a single tape drive is known as tape stackers. The stacker dismounts the current tape when it has finished with it and loads the next tape, hence only one tape is available at a time to be accessed. The price and the capabilities may vary, but the common ability is that they can perform unattended backups.

Tape Silos

Tape silos provide large store capacities. Tape silos can store and manage thousands of tapes. They can integrate multiple tape drives. They have the software and hardware to label and store the tapes they store. It is very common for the silo to be connected remotely over a network or a dedicated link. We should ensure that the bandwidth of the connection is up to the job.

Disk Backups

Methods of disk backups are −

- Disk-to-disk backups
- Mirror breaking

These methods are used in the OLTP system. These methods minimize the database downtime and maximize the availability.

Disk-to-Disk Backups

Here backup is taken on the disk rather on the tape. Disk-to-disk backups are done for the following reasons −

- Speed of initial backups
- Speed of restore

Backing up the data from disk to disk is much faster than to the tape. However it is the intermediate step of backup. Later the data is backed up on the tape. The other advantage of disk-to-disk backups is that it gives you an online copy of the latest backup.

Mirror Breaking

The idea is to have disks mirrored for resilience during the working day. When backup is required, one of the mirror sets can be broken out. This technique is a variant of disk-to-disk backups.

Note − The database may need to be shutdown to guarantee consistency of the backup.

Optical Jukeboxes

Optical jukeboxes allow the data to be stored near line. This technique allows a large number of optical disks to be managed in the same way as a tape stacker or a tape silo. The drawback of this technique is that it has slow write speed than disks. But the optical media provides long-life and reliability that makes them a good choice of medium for archiving.

Software Backups

There are software tools available that help in the backup process. These software tools come as a package. These tools not only take backup, they can effectively manage and control the backup strategies. There are many software packages available in the market. Some of them are listed in the following table −

Package Name	Vendor
Networker	Legato
ADSM	IBM
Epoch	Epoch Systems
Omniback II	HP
Alexandria	Sequent

Criteria for Choosing Software Packages

The criteria for choosing the best software package are listed below −

- How scalable is the product as tape drives are added?
- Does the package have client-server option, or must it run on the database server itself?

- Will it work in cluster and MPP environments?
- What degree of parallelism is required?
- What platforms are supported by the package?
- Does the package support easy access to information about tape contents?
- Is the package database aware?
- What tape drive and tape media are supported by the package?

Data Warehousing - Tuning

A data warehouse keeps evolving and it is unpredictable what query the user is going to post in the future. Therefore it becomes more difficult to tune a data warehouse system. In this chapter, we will discuss how to tune the different aspects of a data warehouse such as performance, data load, queries, etc.

Difficulties in Data Warehouse Tuning

Tuning a data warehouse is a difficult procedure due to following reasons −

- Data warehouse is dynamic; it never remains constant.
- It is very difficult to predict what query the user is going to post in the future.
- Business requirements change with time.
- Users and their profiles keep changing.
- The user can switch from one group to another.
- The data load on the warehouse also changes with time.

Note − It is very important to have a complete knowledge of data warehouse.

Performance Assessment

Here is a list of objective measures of performance −

- Average query response time
- Scan rates
- Time used per day query
- Memory usage per process
- I/O throughput rates

Following are the points to remember.

- It is necessary to specify the measures in service level agreement (SLA).
- It is of no use trying to tune response time, if they are already better than those required.
- It is essential to have realistic expectations while making performance assessment.
- It is also essential that the users have feasible expectations.

- To hide the complexity of the system from the user, aggregations and views should be used.
- It is also possible that the user can write a query you had not tuned for.

Data Load Tuning

Data load is a critical part of overnight processing. Nothing else can run until data load is complete. This is the entry point into the system.

Note − If there is a delay in transferring the data, or in arrival of data then the entire system is affected badly. Therefore it is very important to tune the data load first.

There are various approaches of tuning data load that are discussed below −

- The very common approach is to insert data using the **SQL Layer**. In this approach, normal checks and constraints need to be performed. When the data is inserted into the table, the code will run to check for enough space to insert the data. If sufficient space is not available, then more space may have to be allocated to these tables. These checks take time to perform and are costly to CPU.
- The second approach is to bypass all these checks and constraints and place the data directly into the preformatted blocks. These blocks are later written to the database. It is faster than the first approach, but it can work only with whole blocks of data. This can lead to some space wastage.
- The third approach is that while loading the data into the table that already contains the table, we can maintain indexes.
- The fourth approach says that to load the data in tables that already contain data, **drop the indexes & recreate them** when the data load is complete. The choice between the third and the fourth approach depends on how much data is already loaded and how many indexes need to be rebuilt.

Integrity Checks

Integrity checking highly affects the performance of the load. Following are the points to remember −

- Integrity checks need to be limited because they require heavy processing power.
- Integrity checks should be applied on the source system to avoid performance degrade of data load.

Tuning Queries

We have two kinds of queries in data warehouse −

- Fixed queries
- Ad hoc queries

Fixed Queries

Fixed queries are well defined. Following are the examples of fixed queries −

- regular reports
- Canned queries
- Common aggregations

Tuning the fixed queries in a data warehouse is same as in a relational database system. The only difference is that the amount of data to be queried may be different. It is good to store the most successful execution plan while testing fixed queries. Storing these executing plan will allow us to spot changing data size and data skew, as it will cause the execution plan to change.

Note − We cannot do more on fact table but while dealing with dimension tables or the aggregations, the usual collection of SQL tweaking, storage mechanism, and access methods can be used to tune these queries.

Ad hoc Queries

To understand ad hoc queries, it is important to know the ad hoc users of the data warehouse. For each user or group of users, you need to know the following −

- The number of users in the group
- Whether they use ad hoc queries at regular intervals of time
- Whether they use ad hoc queries frequently
- Whether they use ad hoc queries occasionally at unknown intervals.
- The maximum size of query they tend to run
- The average size of query they tend to run
- Whether they require drill-down access to the base data
- The elapsed login time per day
- The peak time of daily usage
- The number of queries they run per peak hour

Points to Note

- It is important to track the user's profiles and identify the queries that are run on a regular basis.
- It is also important that the tuning performed does not affect the performance.
- Identify similar and ad hoc queries that are frequently run.
- If these queries are identified, then the database will change and new indexes can be added for those queries.
- If these queries are identified, then new aggregations can be created specifically for those queries that would result in their efficient execution.

Data Warehousing - Testing

Testing is very important for data warehouse systems to make them work correctly and efficiently. There are three basic levels of testing performed on a data warehouse −

- Unit testing
- Integration testing
- System testing

Unit Testing

- In unit testing, each component is separately tested.
- Each module, i.e., procedure, program, SQL Script, Unix shell is tested.
- This test is performed by the developer.

Integration Testing

- In integration testing, the various modules of the application are brought together and then tested against the number of inputs.
- It is performed to test whether the various components do well after integration.

System Testing

- In system testing, the whole data warehouse application is tested together.
- The purpose of system testing is to check whether the entire system works correctly together or not.
- System testing is performed by the testing team.
- Since the size of the whole data warehouse is very large, it is usually possible to perform minimal system testing before the test plan can be enacted.

Test Schedule

First of all, the test schedule is created in the process of developing the test plan. In this schedule, we predict the estimated time required for the testing of the entire data warehouse system.

There are different methodologies available to create a test schedule, but none of them are perfect because the data warehouse is very complex and large. Also the data warehouse system is evolving in nature. One may face the following issues while creating a test schedule −

- A simple problem may have a large size of query that can take a day or more to complete, i.e., the query does not complete in a desired time scale.
- There may be hardware failures such as losing a disk or human errors such as accidentally deleting a table or overwriting a large table.

Note − Due to the above-mentioned difficulties, it is recommended to always double the amount of time you would normally allow for testing.

Testing Backup Recovery

Testing the backup recovery strategy is extremely important. Here is the list of scenarios for which this testing is needed −

- Media failure
- Loss or damage of table space or data file
- Loss or damage of redo log file
- Loss or damage of control file
- Instance failure
- Loss or damage of archive file
- Loss or damage of table
- Failure during data failure

Testing Operational Environment

There are a number of aspects that need to be tested. These aspects are listed below.

- **Security** − A separate security document is required for security testing. This document contains a list of disallowed operations and devising tests for each.
- **Scheduler** − Scheduling software is required to control the daily operations of a data warehouse. It needs to be tested during system testing. The scheduling software requires an interface with the data warehouse, which will need the scheduler to control overnight processing and the management of aggregations.
- **Disk Configuration.** − Disk configuration also needs to be tested to identify I/O bottlenecks. The test should be performed with multiple times with different settings.
- **Management Tools.** − It is required to test all the management tools during system testing. Here is the list of tools that need to be tested.
    - Event manager
    - System manager
    - Database manager
    - Configuration manager
    - Backup recovery manager

Testing the Database

The database is tested in the following three ways −

- **Testing the database manager and monitoring tools** − To test the database manager and the monitoring tools, they should be used in the creation, running, and management of test database.
- **Testing database features** − Here is the list of features that we have to test −
    - Querying in parallel
    - Create index in parallel
    - Data load in parallel

- **Testing database performance** − Query execution plays a very important role in data warehouse performance measures. There are sets of fixed queries that need to be run regularly and they should be tested. To test ad hoc queries, one should go through the user requirement document and understand the business completely. Take time to test the most awkward queries that the business is likely to ask against different index and aggregation strategies.

Testing the Application

- All the managers should be integrated correctly and work in order to ensure that the end-to-end load, index, aggregate and queries work as per the expectations.
- Each function of each manager should work correctly
- It is also necessary to test the application over a period of time.
- Week end and month-end tasks should also be tested.

Logistic of the Test

The aim of system test is to test all of the following areas −

- Scheduling software
- Day-to-day operational procedures
- Backup recovery strategy
- Management and scheduling tools
- Overnight processing
- Query performance

Note − The most important point is to test the scalability. Failure to do so will leave us a system design that does not work when the system grows.

Data Warehousing - Future Aspects

Following are the future aspects of data warehousing.

- As we have seen that the size of the open database has grown approximately double its magnitude in the last few years, it shows the significant value that it contains.
- As the size of the databases grow, the estimates of what constitutes a very large database continues to grow.
- The hardware and software that are available today do not allow to keep a large amount of data online. For example, a Telco call record requires 10TB of data to be kept online, which is just a size of one month’s record. If it requires to keep records of sales, marketing customer, employees, etc., then the size will be more than 100 TB.
- The record contains textual information and some multimedia data. Multimedia data cannot be easily manipulated as text data. Searching the multimedia data is not an

easy task, whereas textual information can be retrieved by the relational software available today.

- Apart from size planning, it is complex to build and run data warehouse systems that are ever increasing in size. As the number of users increases, the size of the data warehouse also increases. These users will also require to access the system.
- With the growth of the Internet, there is a requirement of users to access data online.

Hence the future shape of data warehouse will be very different from what is being created today.