# UNIT I

# DATA WAREHOUSING

Data warehousing Components –Building a Data warehouse – Mapping the Data Warehouse to a Multiprocessor Architecture – DBMS Schemas for Decision Support – Data Extraction, Cleanup, and Transformation Tools –Metadata.

## Data Warehouse Introduction

A data warehouse is a collection of data marts representing historical data from different operations in the company. This data is stored in a structure optimized for querying and data analysis as a data warehouse. Table design, dimensions and organization should be consistent throughout a data warehouse so that reports or queries across the data warehouse are consistent.

A data warehouse can also be viewed as a database for historical data from different functions within a company. The term Data Warehouse was coined by Bill Inmon in 1990, which he defined in the following way: "A warehouse is a subject-oriented, integrated, time-variant and non-volatile collection of data in support of management's decision making process". He defined the terms in the sentence as follows:

- *Subject Oriented:* Data that gives information about a particular subject instead of about a company's ongoing operations.
- *Integrated:* Data that is gathered into the data warehouse from a variety of sources and merged into a coherent whole.
- *Time-variant:* All data in the data warehouse is identified with a particular time period.
- *Non-volatile:* Data is stable in a data warehouse. More data is added but data is never removed. This enables management to gain a consistent picture of the business. It is a single, complete and consistent store of data obtained from a variety of different sources made available to end users in what they can understand and use in a business context. It can be Used for decision Support, Used to manage and control business, Used by managers and end-users to understand the business and make judgments.

Data Warehousing is an architectural construct of information systems that provides users with current and historical decision support information that is hard to access or present in traditional operational data stores

**Other important terminology**

- *Enterprise Data warehouse:* It collects all information about subjects (*customers, products, sales, assets, personnel*) that span the entire organization
- Data Mart: Departmental subsets that focus on selected subjects. A data mart is a segment of a data warehouse that can provide data for reporting and analysis on a section, unit, department or operation in the company, e.g. sales, payroll, production. Data marts are sometimes complete individual data warehouses which are usually smaller than the corporate data warehouse.
- *Decision Support System (DSS):* Information technology to help the knowledge worker (executive, manager, and analyst) makes faster & better decisions
- *Drill-down:* Traversing the summarization levels from highly summarized data to the underlying current or old detail
- *Metadata:* Data about data. Containing location and description of warehouse system components: names, definition, structure…

**Benefits of data warehousing**

• Data warehouses are designed to perform well with aggregate queries running on large amounts of data.

• The structure of data warehouses is easier for end users to navigate, understand and query against unlike the relational databases primarily designed to handle lots of transactions.

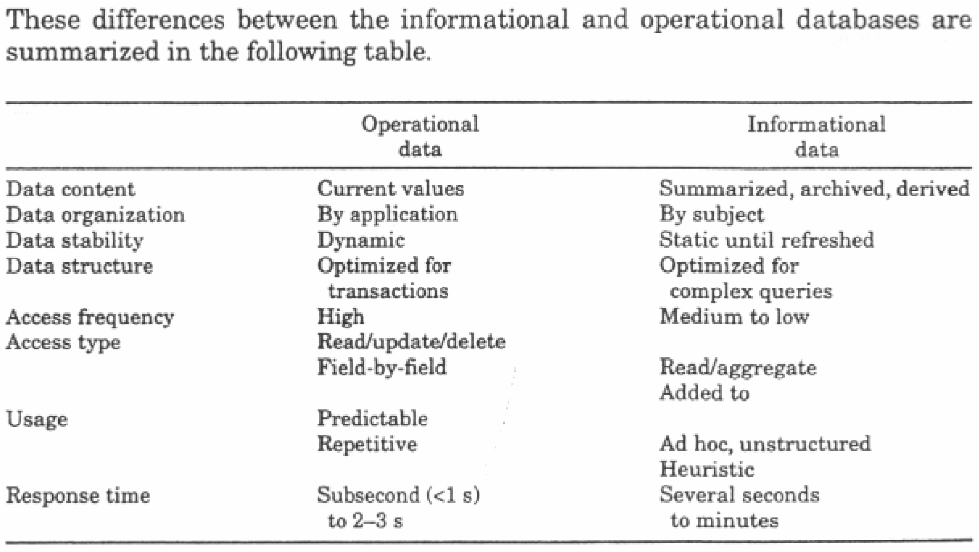
• Data warehouses enable queries that cut across different segments of a company's operation. E.g. production data could be compared against inventory data even if they were originally stored in different databases with different structures.

- Queries that would be complex in very normalized databases could be easier to build and maintain in data warehouses, decreasing the workload on transaction systems.
- Data warehousing is an efficient way to manage and report on data that is from a variety of sources, non uniform and scattered throughout a company.
- Data warehousing is an efficient way to manage demand for lots of information from lots of users.
- Data warehousing provides the capability to analyze large amounts of historical data for nuggets of wisdom that can provide an organization with competitive advantage.

**Operational and informational Data**

**Operational Data:**

- Focusing on transactional function such as bank card withdrawals and deposits
- Detailed
- Updateable
- Reflects current data



These differences between the informational and operational databases are summarized in the following table.

| | Operational data | Informational data |
|---|---|---|
| Data content | Current values | Summarized, archived, derived |
| Data organization | By application | By subject |
| Data stability | Dynamic | Static until refreshed |
| Data structure | Optimized for transactions | Optimized for complex queries |
| Access frequency | High | Medium to low |
| Access type | Read/update/delete Field-by-field | Read/aggregate Added to |
| Usage | Predictable Repetitive | Ad hoc, unstructured Heuristic |
| Response time | Subsecond (<1 s) to 2–3 s | Several seconds to minutes |

**Informational Data:**

- Focusing on providing answers to problems posed by decision makers
- Summarized
- Non updateable

**Data Warehouse Characteristics**

• A data warehouse can be viewed as an information system with the following attributes:

– It is a database designed for analytical tasks

– It's content is periodically updated

– It contains current and historical data to provide a historical perspective of information

**Operational data store (ODS)**

• ODS is an architecture concept to support day-to-day operational decision support and contains current value data propagated from operational applications

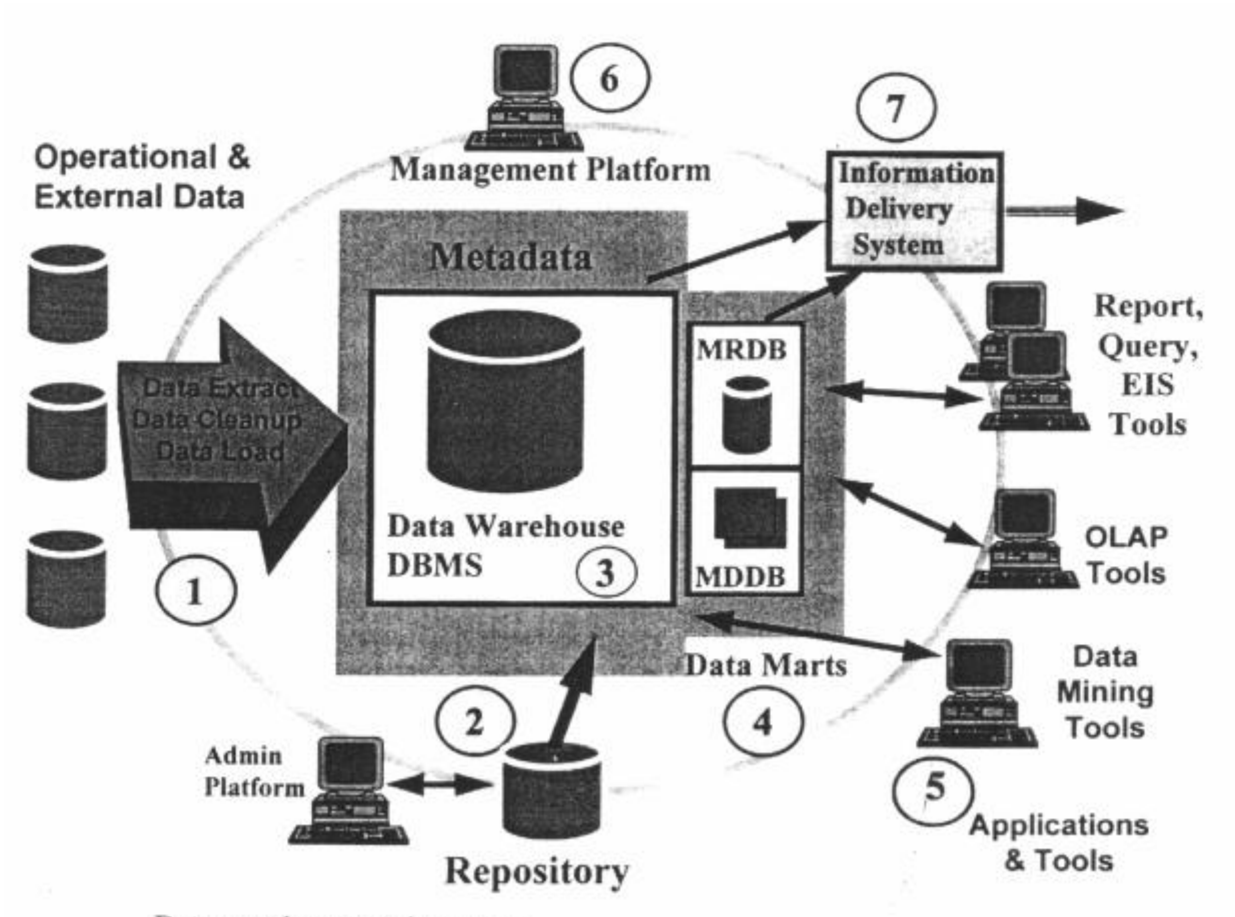
• ODS is subject-oriented, similar to a classic definition of a Data warehouse

• ODS is integrated

## Data warehouse Architecture and its seven components

1. Data sourcing, cleanup, transformation, and migration tools
2. Metadata repository
3. Warehouse/database technology
4. Data marts
5. Data query, reporting, analysis, and mining tools
6. Data warehouse administration and management
7. Information delivery system

Data warehouse is an environment, not a product which is based on relational database management system that functions as the central repository for informational data. The central repository information is surrounded by number of key components designed to make the environment is functional, manageable and accessible.



Data warehouse environment.

The data source for data warehouse is coming from operational applications. The data entered into the data warehouse transformed into an integrated structure and format. The transformation process involves conversion, summarization, filtering and condensation. The data warehouse must be capable of holding and managing large volumes of data as well as different structure of data structures over the time.

1. **Data warehouse database**

This is the central part of the data warehousing environment. This is the item number 2 in the above arch. diagram. This is implemented based on RDBMS technology.

2. **Sourcing, Acquisition, Clean up, and Transformation Tools**

This is item number 1 in the above arch diagram. They perform conversions, summarization, key changes, structural changes and condensation. The data transformation is required so that the information can by used by decision support tools. The transformation produces programs, control statements, JCL code, COBOL code, UNIX scripts, and SQL DDL code etc., to move the data into data warehouse from multiple operational systems.

**The functionalities of these tools are listed below:**

- To remove unwanted data from operational db
- Converting to common data names and attributes
- Calculating summaries and derived data
- Establishing defaults for missing data
- Accommodating source data definition change.

*Issues to be considered while data sourcing, cleanup, extract and transformation:*

Data heterogeneity: It refers to DBMS different nature such as it may be in different data modules, it may have different access languages, it may have data navigation methods, operations, concurrency, integrity and recovery processes etc.,

## 3. Meta data

It is data about data. It is used for maintaining, managing and using the data warehouse. It is classified into two:

1.*Technical Meta data*: It contains information about data warehouse data used by warehouse designer, administrator to carry out development and management tasks. It includes,

- Info about data stores
- Transformation descriptions. That is mapping methods from operational db to warehouse db
- Warehouse Object and data structure definitions for target data
- The rules used to perform clean up, and data enhancement
- Data mapping operations
- Access authorization, backup history, archive history, info delivery history, data acquisition history, data access etc.

2.*Business Meta data:* It contains info that gives info stored in data warehouse to users. It includes,

- Subject areas, and info object type including queries, reports, images, video, audio clips etc.
- Internet home pages
- Info related to info delivery system
- Data warehouse operational info such as ownerships, audit trails etc.,

Meta data helps the users to understand content and find the data. Meta data are stored in a separate data stores which is known as informational directory or Meta data repository which helps to integrate, maintain and view the contents of the data warehouse.

**The following lists the characteristics of info directory/ Meta data:**

- It is the gateway to the data warehouse environment
- It supports easy distribution and replication of content for high performance and availability
- It should be searchable by business oriented key words
- It should act as a launch platform for end user to access data and analysis tools
- It should support the sharing of info
- It should support scheduling options for request
- IT should support and provide interface to other applications
- It should support end user monitoring of the status of the data warehouse environment

**4. Access tools**

Its purpose is to provide info to business users for decision making. There are five categories:

- Data query and reporting tools
- Application development tools
- Executive info system tools (EIS)
- OLAP tools
- Data mining tools

Query and reporting tools are used to generate query and report. There are two types of reporting tools. They are:

- Production reporting tool used to generate regular operational reports
- Desktop report writer are inexpensive desktop tools designed for end users.

***Managed Query tools:*** used to generate SQL query. It uses Meta layer software in between users and databases which offers a point-and-click creation of SQL statement. This tool is a preferred choice of users to perform segment identification, demographic analysis, territory management and preparation of customer mailing lists etc.

***Application development tools:*** This is a graphical data access environment which integrates OLAP tools with data warehouse and can be used to access all db systems

***OLAP Tools:*** are used to analyze the data in multi dimensional and complex views. To enable multidimensional properties it uses MDDB and MRDB where MDDB refers multi dimensional data base and MRDB refers multi relational data bases.

***Data mining tools:*** are used to discover knowledge from the data warehouse data also can be used for data visualization and data correction purposes.

**5. Data marts**

Departmental subsets that focus on selected subjects. They are independent used by dedicated user group. They are used for rapid delivery of enhanced decision support functionality to end users. Data mart is used in the following situation:

- Extremely urgent user requirement
- The absence of a budget for a full scale data warehouse strategy
- The decentralization of business needs
- The attraction of easy to use tools and mind sized project

Data mart presents two problems:

1. Scalability: A small data mart can grow quickly in multi dimensions. So that while designing it, the organization has to pay more attention on system scalability, consistency and manageability issues
2. Data integration

**6. Data warehouse admin and management**

The management of data warehouse includes,

- Security and priority management
- Monitoring updates from multiple sources
- Data quality checks
- Managing and updating meta data
- Auditing and reporting data warehouse usage and status
- Purging data
- Replicating, sub setting and distributing data
- Backup and recovery
- Data warehouse storage management which includes capacity planning, hierarchical storage management and purging of aged data etc.,

7. **Information delivery system**
   - It is used to enable the process of subscribing for data warehouse info.
   - Delivery to one or more destinations according to specified scheduling algorithm

## Building a Data warehouse

There are two reasons why organizations consider data warehousing a critical need. In other words, there are two factors that drive you to build and use data warehouse. They are:

***Business factors:***

- Business users want to make decision quickly and correctly using all available data.

***Technological factors:***

- To address the incompatibility of operational data stores
- IT infrastructure is changing rapidly. Its capacity is increasing and cost is decreasing so that building a data warehouse is easy

There are several things to be considered while building a successful data warehouse

**Business considerations:**

Organizations interested in development of a data warehouse can choose one of the following two approaches:

- Top - Down Approach (Suggested by Bill Inmon)
- Bottom - Up Approach (Suggested by Ralph Kimball)

**Top - Down Approach**

In the top down approach suggested by Bill Inmon, we build a centralized repository to house corporate wide business data. This repository is called Enterprise Data Warehouse (EDW). The data in the EDW is stored in a normalized form in order to avoid redundancy. The central repository for corporate wide data helps us maintain one version of truth of the data. The data in the EDW is stored at the most detail level. The reason to build the EDW on the most detail level is to leverage

1. Flexibility to be used by multiple departments.
2. Flexibility to cater for future requirements.

The disadvantages of storing data at the detail level are

1. The complexity of design increases with increasing level of detail.
2. It takes large amount of space to store data at detail level, hence increased cost.

Once the EDW is implemented we start building subject area specific data marts which contain data in a de normalized form also called star schema. The data in the marts are usually summarized based on the end users analytical requirements. The reason to de normalize the data in the mart is to provide faster access to the data for the end users analytics. If we were to have queried a normalized schema for the same analytics, we would end up in a complex multiple level joins that would be much slower as compared to the one on the de normalized schema.

We should implement the top-down approach when

1. The business has complete clarity on all or multiple subject areas data warehouse requirements.
2. The business is ready to invest considerable time and money.

The advantage of using the Top Down approach is that we build a centralized repository to cater for one version of truth for business data. This is very important for the data to be reliable, consistent across subject areas and for reconciliation in case of data related contention between subject areas.

The disadvantage of using the Top Down approach is that it requires more time and initial investment. The business has to wait for the EDW to be implemented followed by building the data marts before which they can access their reports.

**Bottom Up Approach**

The bottom up approach suggested by Ralph Kimball is an incremental approach to build a data warehouse. Here we build the data marts separately at different points of time as and when the specific subject area requirements are clear. The data marts are integrated or combined together to form a data warehouse. Separate data marts are combined through the use of conformed dimensions and conformed facts. A conformed dimension and a conformed fact is one that can be shared across data marts.

**A Conformed dimension** has consistent dimension keys, consistent attribute names and consistent values across separate data marts. The conformed dimension means exact same thing with every fact table it is joined.

**A Conformed fact** has the same definition of measures, same dimensions joined to it and at the same granularity across data marts.

The bottom up approach helps us incrementally build the warehouse by developing and integrating data marts as and when the requirements are clear. We don't have to wait for knowing the overall requirements of the warehouse.

We should implement the bottom up approach when

1. We have initial cost and time constraints.
2. The complete warehouse requirements are not clear. We have clarity to only one data mart.

The advantage of using the Bottom Up approach is that they do not require high initial costs and have a faster implementation time; hence the business can start using the marts much earlier as compared to the top-down approach.

The disadvantages of using the Bottom Up approach are that it stores data in the de normalized format; hence there would be high space usage for detailed data. We have a tendency of not keeping detailed data in this approach hence losing out on advantage of having detail data i.e. flexibility to easily cater to future requirements. Bottom up approach is more realistic but the complexity of the integration may become a serious obstacle.

**Design considerations**

To be a successful data warehouse designer must adopt a holistic approach that is considering all data warehouse components as parts of a single complex system, and take into account all possible data sources and all known usage requirements.

Most successful data warehouses that meet these requirements have these common characteristics:

- Are based on a dimensional model
- Contain historical and current data
- Include both detailed and summarized data
- Consolidate disparate data from multiple sources while retaining consistency

Data warehouse is difficult to build due to the following reason:

- Heterogeneity of data sources
- Use of historical data
- Growing nature of data base

Data warehouse design approach muse be business driven, continuous and iterative engineering approach. In addition to the general considerations there are following specific points relevant to the data warehouse design:

**Data content**

The content and structure of the data warehouse are reflected in its data model. The data model is the template that describes how information will be organized within the integrated warehouse framework. The data warehouse data must be a detailed data. It must be formatted, cleaned up and transformed to fit the warehouse data model.

**Meta data**

It defines the location and contents of data in the warehouse. Meta data is searchable by users to find definitions or subject areas. In other words, it must provide decision support oriented pointers to warehouse data and thus provides a logical link between warehouse data and decision support applications.

**Data distribution**

One of the biggest challenges when designing a data warehouse is the data placement and distribution strategy. Data volumes continue to grow in nature. Therefore, it becomes necessary to know how the data should be divided across multiple servers and which users should get access to which types of data. The data can be distributed based on the subject area, location (geographical region), or time (current, month, year).

**Tools**

A number of tools are available that are specifically designed to help in the implementation of the data warehouse. All selected tools must be compatible with the given data warehouse environment and with each other. All tools must be able to use a common Meta data repository.

## Design steps

The following nine-step method is followed in the design of a data warehouse:

1. Choosing the subject matter
2. Deciding what a fact table represents
3. Identifying and conforming the dimensions
4. Choosing the facts
5. Storing pre calculations in the fact table
6. Rounding out the dimension table
7. Choosing the duration of the db
8. The need to track slowly changing dimensions
9. Deciding the query priorities and query models

**Technical considerations**

A number of technical issues are to be considered when designing a data warehouse environment. These issues include:

- The hardware platform that would house the data warehouse
- The DBMS that supports the warehouse data

- The communication infrastructure that connects data marts, operational systems and end users
- The hardware and software to support meta data repository
- The systems management framework that enables admin of the entire environment

**Implementation considerations**

The following logical steps needed to implement a data warehouse:

- Collect and analyze business requirements
- Create a data model and a physical design
- Define data sources
- Choose the DB tech and platform
- Extract the data from operational DB, transform it, clean it up and load it into the warehouse
- Choose DB access and reporting tools
- Choose DB connectivity software
- Choose data analysis and presentation s/w
- Update the data warehouse

**Access tools**

Data warehouse implementation relies on selecting suitable data access tools. The best way to choose this is based on the type of data can be selected using this tool and the kind of access it permits for a particular user. The following lists the various types of data that can be accessed:

- Simple tabular form data
- Ranking data
- Multivariable data
- Time series data
- Graphing, charting and pivoting data
- Complex textual search data
- Statistical analysis data

- Data for testing of hypothesis, trends and patterns
- Predefined repeatable queries
- Ad hoc user specified queries
- Reporting and analysis data
- Complex queries with multiple joins, multi level sub queries and sophisticated search criteria

**Data extraction, clean up, transformation and migration**

A proper attention must be paid to data extraction which represents a success factor for a data warehouse architecture. When implementing data warehouse several the following selection criteria that affect the ability to transform, consolidate, integrate and repair the data should be considered:

- Timeliness of data delivery to the warehouse
- The tool must have the ability to identify the particular data and that can be read by conversion tool
- The tool must support flat files, indexed files since corporate data is still in this type
- The tool must have the capability to merge data from multiple data stores
- The tool should have specification interface to indicate the data to be extracted
- The tool should have the ability to read data from data dictionary
- The code generated by the tool should be completely maintainable
- The tool should permit the user to extract the required data
- The tool must have the facility to perform data type and character set translation
- The tool must have the capability to create summarization, aggregation and derivation of records
- The data warehouse database system must be able to perform loading data directly from these tools

**Data placement strategies**

– As a data warehouse grows, there are at least two options for data placement. One is to put some of the data in the data warehouse into another storage media.

– The second option is to distribute the data in the data warehouse across multiple servers.

**User levels**

The users of data warehouse data can be classified on the basis of their skill level in accessing the warehouse.

There are three classes of users:

***Casual users:*** are most comfortable in retrieving info from warehouse in pre defined formats and running pre existing queries and reports. These users do not need tools that allow for building standard and ad hoc reports

***Power Users:*** can use pre defined as well as user defined queries to create simple and ad hoc reports. These users can engage in drill down operations. These users may have the experience of using reporting and query tools.

***Expert users:*** These users tend to create their own complex queries and perform standard analysis on the info they retrieve. These users have the knowledge about the use of query and report tools

**Benefits of data warehousing**

Data warehouse usage includes,

– Locating the right info

– Presentation of info

– Testing of hypothesis

– Discovery of info

– Sharing the analysis

The benefits can be classified into two:

- **Tangible benefits (quantified / measureable):**It includes,
    - – Improvement in product inventory
    - – Decrement in production cost
    - – Improvement in selection of target markets
    - – Enhancement in asset and liability management
- **Intangible benefits (not easy to quantified):** It includes,
    - – Improvement in productivity by keeping all data in single location and eliminating rekeying of data
    - – Reduced redundant processing
    - – Enhanced customer relation

---

## Mapping the data warehouse architecture to Multiprocessor architecture

The functions of data warehouse are based on the relational data base technology. The relational data base technology is implemented in parallel manner. There are two advantages of having parallel relational data base technology for data warehouse:

- ***Linear Speed up****:* refers the ability to increase the number of processor to reduce response time
- ***Linear Scale up****:* refers the ability to provide same performance on the same requests as the database size increases

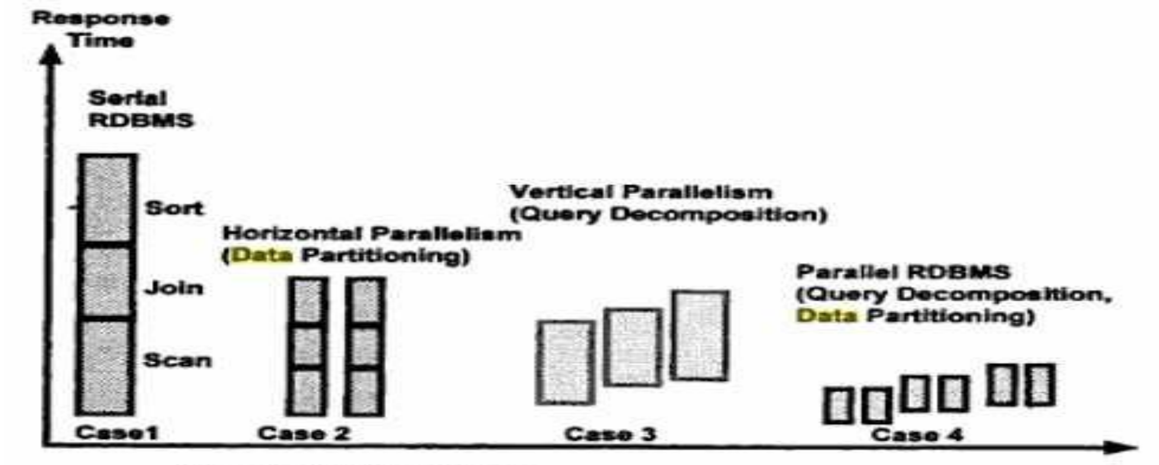
**Types of parallelism**

There are two types of parallelism:

- *Inter query Parallelism:* In which different server threads or processes handle multiple requests at the same time.
- *Intra query Parallelism:* This form of parallelism decomposes the serial SQL query into lower level operations such as scan, join, sort etc. Then these lower level operations are executed concurrently in parallel.

Intra query parallelism can be done in either of two ways:

- *Horizontal parallelism:* which means that the data base is partitioned across multiple disks and parallel processing occurs within a specific task that is performed concurrently on different processors against different set of data
- *Vertical parallelism:* This occurs among different tasks. All query components such as scan, join, sort etc are executed in parallel in a pipelined fashion. In other words, an output from one task becomes an input into another task.



Types of DBMS parallelism.

**Data partitioning:**

Data partitioning is the key component for effective parallel execution of data base operations. Partition can be done randomly or intelligently.

*Random portioning* includes random data striping across multiple disks on a single server. Another option for random portioning is round robin fashion partitioning in which each record is placed on the next disk assigned to the data base.

*Intelligent partitioning* assumes that DBMS knows where a specific record is located and does not waste time searching for it across all disks.

The various intelligent partitioning include:

*Hash partitioning:* A hash algorithm is used to calculate the partition number based on the value of the partitioning key for each row

*Key range partitioning:* Rows are placed and located in the partitions according to the value of the partitioning key. That is all the rows with the key value from A to K are in partition 1, L to T are in partition 2 and so on.

*Schema portioning:* an entire table is placed on one disk; another table is placed on different disk etc. This is useful for small reference tables.

*User defined portioning:* It allows a table to be partitioned on the basis of a user defined expression.

## Data base architectures of parallel processing

There are three DBMS software architecture styles for parallel processing:

1. Shared memory or shared everything Architecture
2. Shared disk architecture
3. Shred nothing architecture

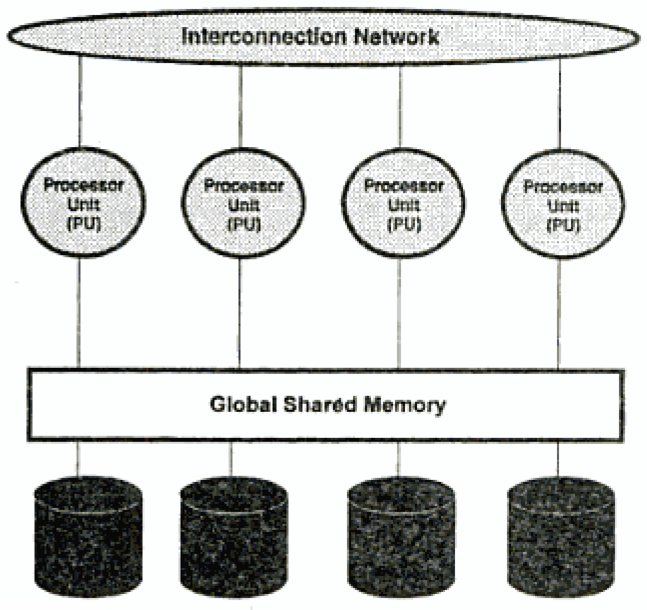
***Shared Memory Architecture***

Tightly coupled shared memory systems, illustrated in following figure have the following characteristics:

- Multiple PUs share memory.
- Each PU has full access to all shared memory through a common bus.
- Communication between nodes occurs via shared memory.
- Performance is limited by the bandwidth of the memory bus.

Symmetric multiprocessor (SMP) machines are often nodes in a cluster. Multiple SMP nodes can be used with Oracle Parallel Server in a tightly coupled system, where memory is shared among the multiple PUs, and is accessible by all the PUs through a memory bus. Examples of tightly coupled systems include the Pyramid, Sequent, and Sun SparcServer.

Performance is potentially limited in a tightly coupled system by a number of factors. These include various system components such as the memory bandwidth, PU to PU communication bandwidth, the memory available on the system, the I/O bandwidth, and the bandwidth of the common bus.



Shared-memory architecture.

Parallel processing advantages of shared memory systems are these:

- Memory access is cheaper than inter-node communication. This means that internal synchronization is faster than using the Lock Manager.
- Shared memory systems are easier to administer than a cluster.

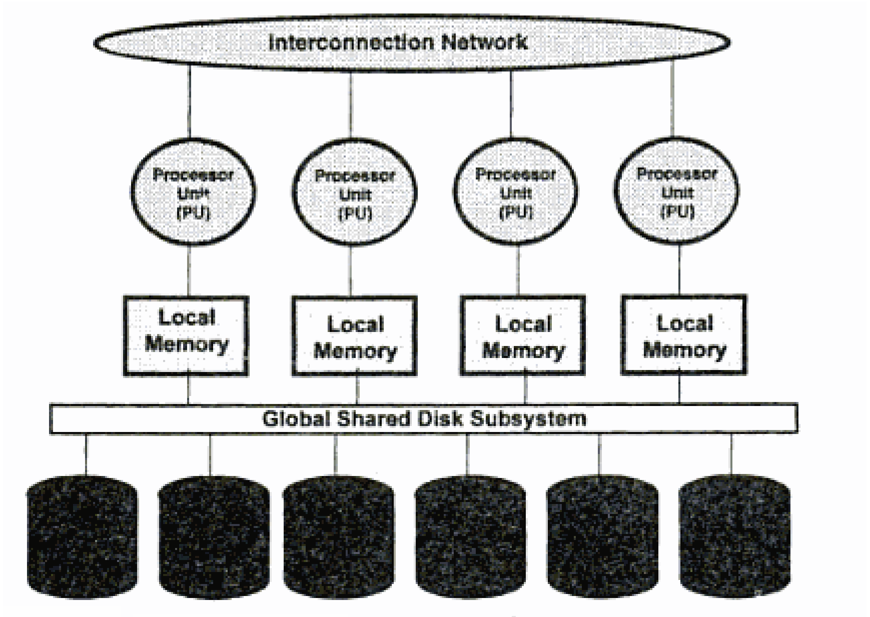
A disadvantage of shared memory systems for parallel processing is as follows:

- Scalability is limited by bus bandwidth and latency, and by available memory.

**Shared Disk Architecture**

Shared disk systems are typically loosely coupled. Such systems, illustrated in following figure, have the following characteristics:

- Each node consists of one or more PUs and associated memory.
- Memory is not shared between nodes.
- Communication occurs over a common high-speed bus.
- Each node has access to the same disks and other resources.
- A node can be an SMP if the hardware supports it.
- Bandwidth of the high-speed bus limits the number of nodes (scalability) of the system.



shared-disk architecture.

The cluster illustrated in figure is composed of multiple tightly coupled nodes. The Distributed Lock Manager (DLM ) is required. Examples of loosely coupled systems are VAX clusters or Sun clusters.

Since the memory is not shared among the nodes, each node has its own data cache. Cache consistency must be maintained across the nodes and a lock manager is needed to maintain the consistency. Additionally, instance locks using the DLM on the Oracle level must be maintained to ensure that all nodes in the cluster see identical data.

There is additional overhead in maintaining the locks and ensuring that the data caches are consistent. The performance impact is dependent on the hardware and software components, such as the bandwidth of the high-speed bus through which the nodes communicate, and DLM performance.

**Parallel processing advantages of shared disk systems are as follows:**

- Shared disk systems permit high availability. All data is accessible even if one node dies.
- These systems have the concept of one database, which is an advantage over shared nothing systems.

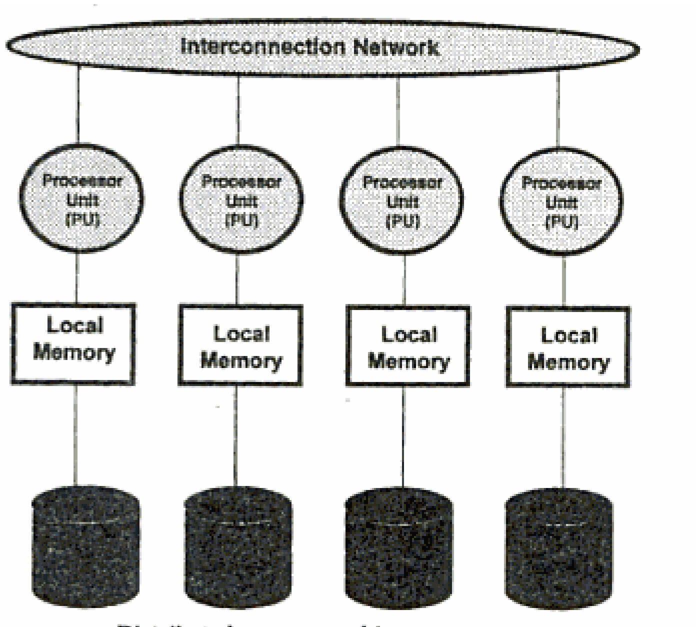
- Shared disk systems provide for incremental growth.

Parallel processing disadvantages of shared disk systems are these:

- Inter-node synchronization is required, involving DLM overhead and greater dependency on high-speed interconnect.
- If the workload is not partitioned well there may be high synchronization overhead.
- There is operating system overhead of running shared disk software.

**Shared Nothing Architecture**

Shared nothing systems are typically loosely coupled. In shared nothing systems only one CPU is connected to a given disk. If a table or database is located on that disk, access depends entirely on the PU which owns it. Shared nothing systems can be represented as follows:



Distributed memory architecture.

Shared nothing systems are concerned with access to disks, not access to memory. Nonetheless, adding more PUs and disks can improve scale up. Oracle Parallel Server can access the disks on a shared nothing system as long as the operating system provides transparent disk access, but this access is expensive in terms of latency.

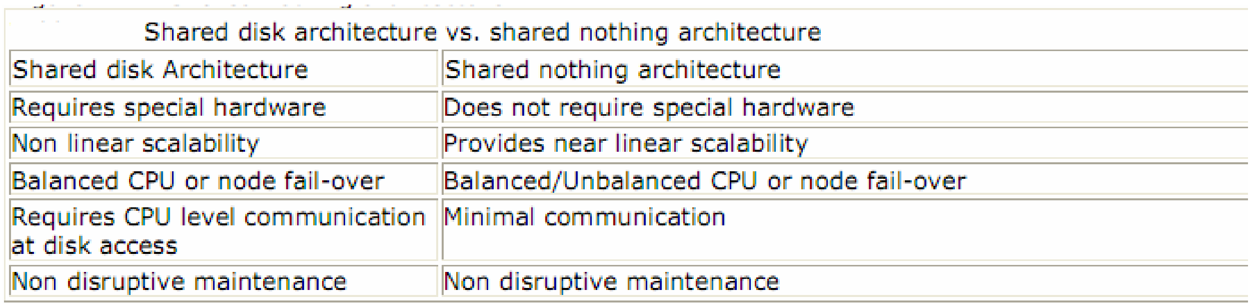
**Shared nothing systems have advantages and disadvantages for parallel processing:**

**Advantages**

- Shared nothing systems provide for incremental growth.
- System growth is practically unlimited.
- MPPs are good for read-only databases and decision support applications.
- Failure is local: if one node fails, the others stay up.

**Disadvantages**

- More coordination is required.
- More overhead is required for a process working on a disk belonging to another node.
- If there is a heavy workload of updates or inserts, as in an online transaction processing system, it may be worthwhile to consider data-dependent routing to alleviate contention.



Shared disk architecture vs. shared nothing architecture

| Shared disk Architecture | Shared nothing architecture |
|---|---|
| Requires special hardware | Does not require special hardware |
| Non linear scalability | Provides near linear scalability |
| Balanced CPU or node fail-over | Balanced/Unbalanced CPU or node fail-over |
| Requires CPU level communication at disk access | Minimal communication |
| Non disruptive maintenance | Non disruptive maintenance |

**Parallel DBMS features**

- Scope and techniques of parallel DBMS operations
- Optimizer implementation
- Application transparency
- Parallel environment which allows the DBMS server to take full advantage of the existing facilities on a very low level
- DBMS management tools help to configure, tune, admin and monitor a parallel RDBMS as effectively as if it were a serial RDBMS
- Price / Performance: The parallel RDBMS can demonstrate a non linear speed up and scale up at reasonable costs.

**Parallel DBMS vendors**

1. Oracle: Parallel Query Option (PQO)

Architecture: shared disk arch

Data partition: Key range, hash, round robin

Parallel operations: hash joins, scan and sort

2. Informix: eXtended Parallel Server (XPS)

Architecture: Shared memory, shared disk and shared nothing models

Data partition: round robin, hash, schema, key range and user defined

Parallel operations: INSERT, UPDATE, DELELTE

3. IBM: DB2 Parallel Edition (DB2 PE)

Architecture: Shared nothing models

Data partition: hash

Parallel operations: INSERT, UPDATE, DELELTE, load, recovery, index creation, backup, table reorganization

4. SYBASE: SYBASE MPP

Architecture: Shared nothing models

Data partition: hash, key range, Schema

Parallel operations: Horizontal and vertical parallelism

## DBMS schemas for decision support

The basic concepts of dimensional modeling are: facts, dimensions and measures. A fact is a collection of related data items, consisting of measures and context data. It typically represents business items or business transactions. A dimension is a collection of data that describe one business dimension. Dimensions determine the contextual background for the facts; they are the parameters over which we want to perform OLAP. A measure is a numeric attribute of a fact, representing the performance or behavior of the business relative to the dimensions. Considering Relational context, there are three basic schemas that are used in dimensional modeling:

1. Star schema
2. Snowflake schema
3. Fact constellation schema

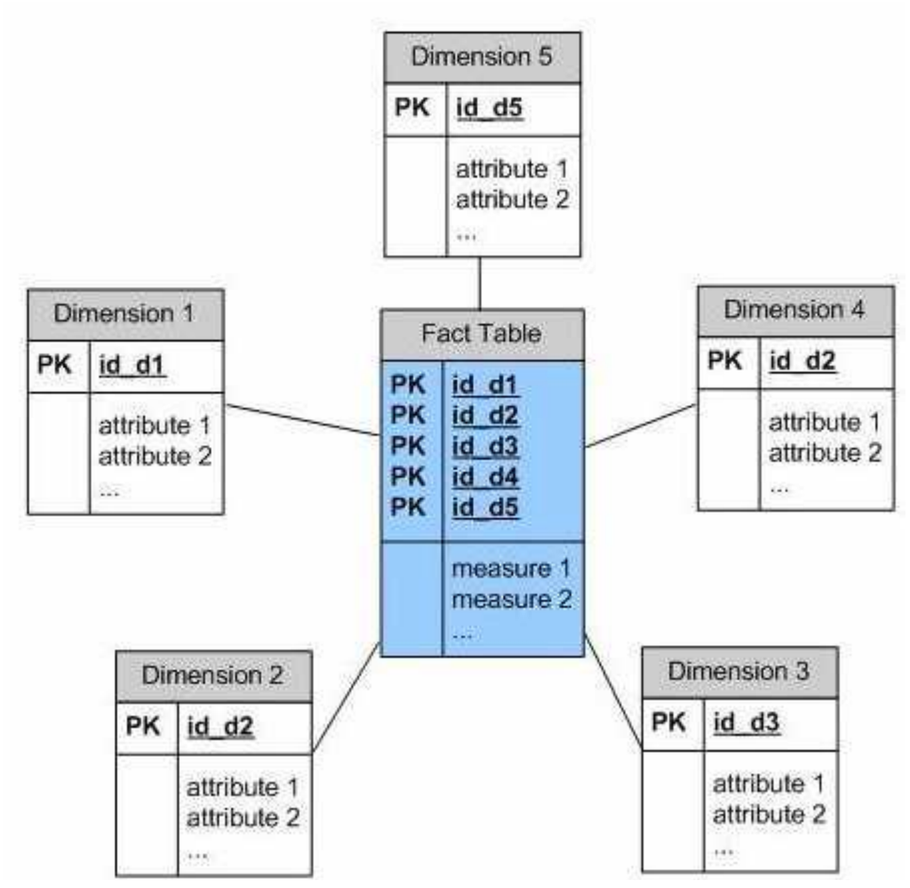
**Star schema**

The multidimensional view of data that is expressed using relational data base semantics is provided by the data base schema design called star schema. The basic of stat schema is that information can be classified into two groups:

- Facts
- Dimension

Star schema has one large central table (fact table) and a set of smaller tables (dimensions) arranged in a radial pattern around the central table.

Facts are core data element being analyzed while dimensions are attributes about the facts. The determination of which schema model should be used for a data warehouse should be based upon the analysis of project requirements, accessible tools and project team preferences.



The star schema architecture is the simplest data warehouse schema. It is called a star schema because the diagram resembles a star, with points radiating from a center. The center of the star consists of fact table and the points of the star are the dimension tables. Usually the fact tables in a star schema are in third normal form(3NF) whereas dimensional tables are de-normalized. Despite the fact that the star schema is the simplest architecture, it is most commonly used nowadays and is recommended by Oracle.

**Fact Tables**

A fact table is a table that contains summarized numerical and historical data (facts) and a multipart index composed of foreign keys from the primary keys of related dimension tables. A fact table typically has two types of columns: foreign keys to dimension tables and measures those that contain numeric facts. A fact table can contain fact's data on detail or aggregated level.

**Dimension Tables**

Dimensions are categories by which summarized data can be viewed. E.g. a profit summary in a fact table can be viewed by a Time dimension (profit by month, quarter, year), Region dimension
(profit by country, state, city), Product dimension (profit for product1, product2). A dimension is a structure usually composed of one or more hierarchies that categorizes data. If a dimension hasn't got a hierarchies and levels it is called flat dimension or list. The primary keys of each of the dimension tables are part of the composite primary key of the fact table.

Dimensional attributes help to describe the dimensional value. They are normally descriptive, textual values. Dimension tables are generally small in size then fact table. Typical fact tables store data about sales while dimension tables data about geographic region (markets, cities), clients, products, times, channels.

**Measures**

Measures are numeric data based on columns in a fact table. They are the primary data which end users are interested in. E.g. a sales fact table may contain a profit measure which represents profit on each sale.

Aggregations are pre calculated numeric data. By calculating and storing the answers to a query before users ask for it, the query processing time can be reduced. This is key in providing fast query performance in OLAP.

Cubes are data processing units composed of fact tables and dimensions from the data warehouse. They provide multidimensional views of data, querying and analytical capabilities to clients.

The main characteristics of star schema:

- Simple structure -> easy to understand schema
- Great query effectives -> small number of tables to join
- Relatively long time of loading data into dimension tables -> de-normalization, redundancy data caused that size of the table could be large.
- The most commonly used in the data warehouse implementations -> widely supported by a large number of business intelligence tools

**Snowflake schema**: is the result of decomposing one or more of the dimensions. The many-to one relationship among sets of attributes of a dimension can separate new dimension tables, forming a hierarchy. The decomposed snowflake structure visualizes the hierarchical structure of dimensions very well.

**Fact constellation schema**: For each star schema it is possible to construct fact constellation schema (for example by splitting the original star schema into more star schemes each of them describes facts on another level of dimension hierarchies). The fact constellation architecture contains multiple fact tables that share many dimension tables.

The main shortcoming of the fact constellation schema is a more complicated design because many variants for particular kinds of aggregation must be considered and selected. Moreover, dimension tables are still large.

**Multi relational Database:**

The relational implementation of multidimensional data base systems is referred to as multi relational database systems.

## Data Extraction, Cleanup and Transformation Tools

- The task of capturing data from a source data system, cleaning and transforming it and then loading the results into a target data system can be carried out either by separate products, or by a single integrated solution. More contemporary integrated solutions can fall into one of the categories described below:
  - Code Generators
  - Database data Replications
  - Rule-driven Dynamic Transformation Engines (Data Mart Builders)

Code Generator:

- It creates 3GL/4GL transformation programs based on source and target data definitions, and data transformation and enhancement rules defined by the developer.
- This approach reduces the need for an organization to write its own data capture, transformation, and load programs. These products employ DML Statements to capture a set of the data from source system.
- These are used for data conversion projects, and for building an enterprise-wide data warehouse, when there is a significant amount of data transformation to be done involving a variety of different flat files, non-relational, and relational data sources.

Database Data Replication Tools:

- These tools employ database triggers or a recovery log to capture changes to a single data source on one system and apply the changes to a copy of the data source data located on a different system.
- Most replication products do not support the capture of changes to non-relational files and databases, and often do not provide facilities for significant data transformation and enhancement.
- These point-to-point tools are used for disaster recovery and to build an operational data store, a data warehouse, or a data mart when the number of data sources

involved are small and a limited amount of data transformation and enhancement is required.

Rule-driven Dynamic Transformation Engines (Data Mart Builders):

- They are also known as Data Mart Builders and capture data from a source system at User-defined intervals, transform data, and then send and load the results into a target environment, typically a data mart.
- To date most of the products of this category support only relational data sources, though now this trend have started changing.
- Data to be captured from source system is usually defined using query language statements, and data transformation and enhancement is done on a script or a function logic defined to the tool.
- With most tools in this category, data flows from source systems to target systems through one or more servers, which perform the data transformation and enhancement. These transformation servers can usually be controlled from a single location, making the job of such environment much easier.

**Meta Data:**

Meta Data Definitions:

- Metadata – additional data warehouse used to understand what information is in the warehouse, and what it means
- Metadata Repository – specialized database designed to maintain metadata, together with the tools and interfaces that allow a company to collect and distribute its metadata.
- Operational Data – elements from operation systems, external data (or other sources) mapped to the warehouse structures.

Industry trend:

Why were early Data Warehouses that did not include significant amounts of metadata collection able to succeed?

- Usually a subset of data was targeted, making it easier to understand content, organization, ownership.
- Usually targeted a subset of (technically inclined) end users

Early choices were made to ensure the success of initial data warehouse efforts.

Meta Data Transitions:

- Usually, metadata repositories are already in existence. Traditionally, metadata was aimed at overall systems management, such as aiding in the maintenance of legacy systems through impact analysis, and determining the appropriate reuse of legacy data structures.
- Repositories can now aide in tracking metadata to help all data warehouse users understand what information is in the warehouse and what it means. Tools are now being positioned to help manage and maintain metadata.

Meta Data Lifecycle:

1. **Collection**: Identify metadata and capture it in a central repository.
2. **Maintenance**: Put in place processes to synchronize metadata automatically with the changing data architecture.
3. **Deployment**: Provide metadata to users in the right form and with the right tools.

The key to ensuring a high level of collection and maintenance accuracy is to incorporate as much automation as possible. The key to a successful metadata deployment is to correctly match the metadata offered to the specific needs of each audience.

Meta Data Collection:

- Collecting the right metadata at the right time is the basis for a success. If the user does not already have an idea about what information would answer a question, the user will not find anything helpful in the warehouse.
- Metadata spans many domains from physical structure data, to logical model data, to business usage and rules.
- Typically the metadata that should be collected is already generated and processed by the development team anyway. Metadata collection preserves the analysis performed by the team.

Meta Data Categories:

Warehouse Data Sources:

- Information about the potential sources of data for a data warehouse (existing operational systems, external data, manually maintained information). The intent is to understand both the physical structure of the data and the meaning of the data. Typically the physical structure is easier to collect as it may exist in a repository that can be parsed automatically.

Data Models:

Correlate the enterprise model to the warehouse model.

- Map entities in the enterprise model to their representation in the warehouse model. This will provide the basis for further change impact analysis and end user content analysis.
- Ensure the entity, element definition, business rules, valid values, and usage guidelines are transposed properly from the enterprise model to the warehouse model.

Warehouse Mappings:

Map the operational data into the warehouse data structures

- Each time a data element is mapped to the warehouse, the logical connection between the data elements, as well as any transformations should be recorded.
- Along with being able to determine that an element in the warehouse is populated from specific sources of data, the metadata should also discern exactly what happens to those elements as they are extracted from the data sources, moved, transformed, and loaded into the warehouse.

Warehouse Usage Information:

Usage information can be used to:

- Understand what tables are being accessed, by whom, and how often. This can be used to fine tune the physical structure of the data warehouse.
- Improve query reuse by identifying existing queries (catalog queries, identify query authors, descriptions).
- Understand how data is being used to solve business problems.

This information is captured after the warehouse has been deployed. Typically, this information is not easy to collect.

Maintaining Meta Data:

- As with any maintenance process, automation is key to maintaining current high-quality information. The data warehouse tools can play an important role in how the metadata is maintained.
- Most proposed database changes already go through appropriate verification and authorization, so adding a metadata maintenance requirement should not be significant.
- Capturing incremental changes is encouraged since metadata (particularly structure information) is usually very large.

Maintaining the Warehouse:

The warehouse team must have comprehensive impact analysis capabilities to respond to change that may affect:

- Data extraction\movement\transformation routines
- Table structures
- Data marts and summary data structures
- Stored user queries
- Users who require new training (due to query or other changes)

What business problems are addressed in part using the element that is changing (help understand the significance of the change, and how it may impact decision making).

Meta Data Deployment:

Supply the right metadata to the right audience

- Warehouse developers will primarily need the physical structure information for data sources. Further analysis on that metadata leads to the development of more metadata (mappings).
- Warehouse maintainers typically require direct access to the metadata as well.
- End Users require an easy-to-access format. They should not be burdened with technical names or cryptic commands. Training, documentation and other forms of help, should be readily available.

End Users:

Users of the warehouse are primarily concerned with two types of metadata.

1. A high-level topic inventory of the warehouse (what is in the warehouse and where it came from).
2. Existing queries that are pertinent to their search (reuse).

The important goal is that the user is easily able to correctly find and interpret the data they need.

Integration with Data Access Tools:

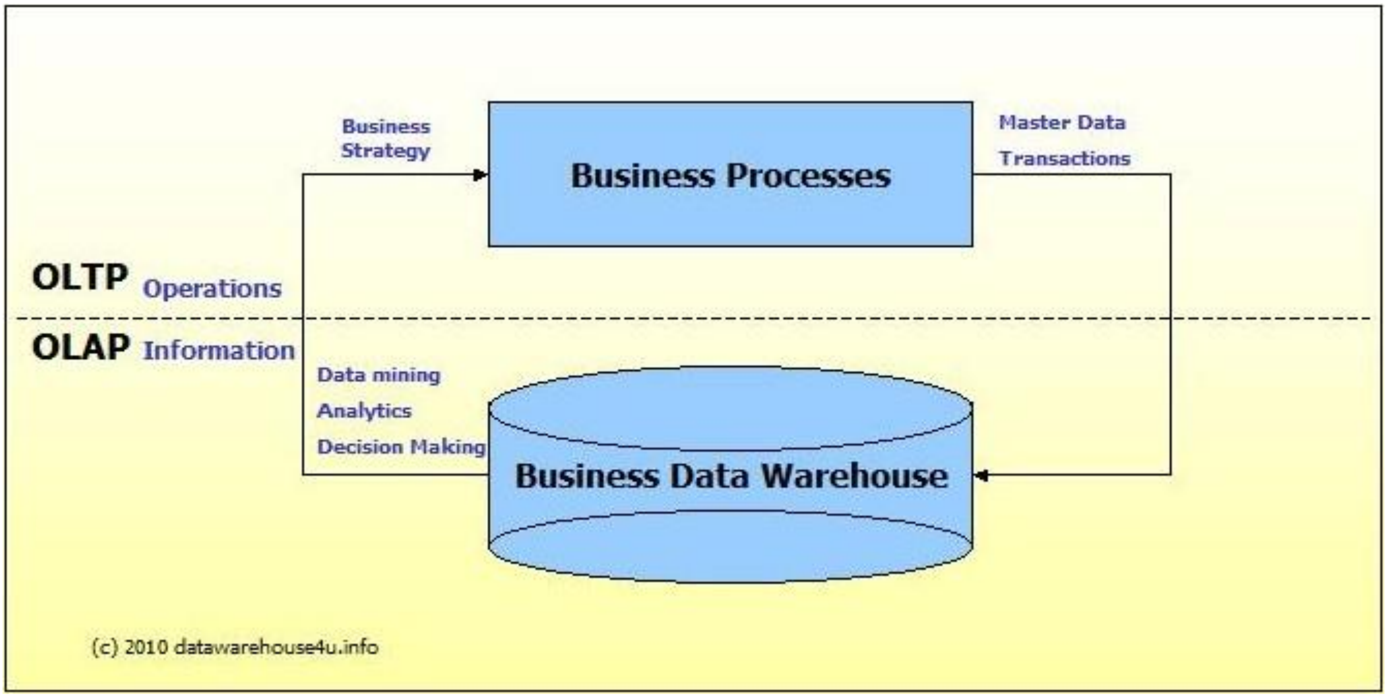
1. Side by Side access to metadata and to real data. The user can browse metadata and write queries against the real data.
2. Populate query tool help text with metadata exported from the repository. The tool can now provide the user with context sensitive help at the expense of needing updating whenever metadata changes and the user may be using outdated metadata.
3. Provide query tools that access the metadata directly to provide context sensitive help. This eliminates the refresh issue, and ensures the user always sees current metadata.
4. Full interconnectivity between query tool and metadata tool (transparent transactions between tools).

## OLTP vs. OLAP

We can divide IT systems into transactional (OLTP) and analytical (OLAP). In general we can assume that OLTP systems provide source data to data warehouses, whereas OLAP systems help to analyze it.

- **OLTP (On-line Transaction Processing)** is characterized by a large number of short on-line transactions (INSERT, UPDATE, DELETE). The main emphasis for OLTP systems is put on very fast query processing, maintaining data integrity in multi-access environments and an effectiveness measured by number of transactions per second. In OLTP database there is detailed and current data, and schema used to store transactional databases is the entity model (usually 3NF).

- **OLAP (On-line Analytical Processing)** is characterized by relatively low volume of transactions. Queries are often very complex and involve aggregations. For OLAP systems a response time is an effectiveness measure. OLAP applications are widely used by Data Mining techniques. In OLAP database there is aggregated, historical data, stored in multi-dimensional schemas (usually star schema).



The following table summarizes the major differences between OLTP and OLAP system design.

| | **OLTP System**<br>**Online Transaction Processing**<br>**(Operational System)** | **OLAP System**<br>**Online Analytical Processing**<br>**(Data Warehouse)** |
|---|---|---|
| Source of data | Operational data; OLTPs are the original source of the data. | Consolidation data; OLAP data comes from the various OLTP Databases |
| Purpose of data | To control and run fundamental business tasks | To help with planning, problem solving, and decision support |
| What the data | Reveals a snapshot of ongoing business processes | Multi-dimensional views of various kinds of business activities |
| Inserts and Updates | Short and fast inserts and updates initiated by end users | Periodic long-running batch jobs refresh the data |
| Queries | Relatively standardized and simple queries Returning relatively few records | Often complex queries involving aggregations |
| Processing Speed | Typically very fast | Depends on the amount of data involved; batch data refreshes and complex queries may take many hours; query speed can be improved by creating indexes |
| Space Requirements | Can be relatively small if historical data is archived | Larger due to the existence of aggregation structures and history data; requires more indexes than OLTP |
| DatabaseDesign | Highly normalized with many tables | Typically de-normalized with fewer tables; use of star and/or snowflake schemas |
| Backup and Recovery | Backup religiously; operational data is critical to run the business, data loss is likely to entail significant monetary loss and legal liability | Instead of regular backups, some environments may consider simply reloading the OLTP data as a recovery method |