Data warehousing concepts

Data warehousing concepts

What is Data:

Data is collection of raw material in unorganized format, which refers an object.

Data:

- ✓ Items that are the most elementary descriptions of things, events, activities, and transactions
- ✓ May be internal or external

Information:

- ✓ Organized data that has meaning and value

Knowledge:

- ✓ Processed data or information that conveys understanding or learning applicable to a problem or activity

In Data Warehouse data is converted in to information format to get knowledge for making decisions.

Data---→Information------→Knowledge

Why DWH implemented:

Enterprise: Enterprise is integration of various departments which is working for business. Different departments will work on different transactions, we need to store all those transactions. Those all transactions are stored in a database called as Operational Data Store (ODS).

The main characteristic of ODS is data is volatile means data is changed randomly, to kept historical data is not possible in ODS. For that reason DWH is implemented.

DWH is also called as **Decision Support System** (DSS) because we use DWH to make decisions and also called as **Historical Database** because DWH stores historical data and **Read Only Database** because of we can just read and analyze the data.

Who required DWH:

Business Analyst: He is working for organization to make decisions. To make decision definitely he required historical data because while making decisions he should compare present data and historical data. But ODS is unable to store historical data, so a container is developed called as DWH. Main users for DWH are Business analysts,CEO's,High and Middle level management in the company.

Why we need DWH:

- ➢ *To Integrate Data from multiple diverse sources*
- ➢ *Allows for analysis of data over time*
- ➢ *Provides ad-hoc querying, reporting and analysis capabilities to decisions makers*

By using DWH users can resolve below questions and based on results he will take decisions.

- ➢ Which are our lowest/highest margin customers?
- ➢ What is the most effective distribution channel?
- ➢ Who are my customers and what products are they buying?
- ➢ What product promotions have the biggest impact on revenue?
- ➢ Which customers are most likely to go to the competition?

What is DWH:

- A DWH is integration of data from different sources, that data is used for analysis to make managerial decisions.
- A data warehouse is a relational database that is designed for query and analysis rather than for transaction processing. It usually contains historical data derived from transaction data.
- Data Warehouse is a subject-oriented, integrated, nonvolatile and time-variant collection of data in support of management's decisions.

Characteristics of DWH:

- ***Subject-oriented*** — Data warehouse data are organized around major subject areas such as sales, claims, shipments, and enrollments. For example, a data warehouse for sales contains historical records of sales over specific time intervals.
- ***Integrated*** — A data warehouse provides the facility for integration in a heterogeneous, fragmented environment of independent application systems, where the data is stored in multiple, incompatible formats. For example, a department store may have information about the same customers stored in several databases using different format representations. The data warehouse brings the data together into a single representation.
- ***Time-variant*** — The data warehouse organizes and stores the data needed for informational and analytical processing over an extended historical time range. For example, a marketing analyst can analyze the sales history of five years from the information that was collected at the end of each year.
- ***Non-volatile*** — Changes to the data warehouse environment occur in a controlled and scheduled manner, unlike the more volatile OLTP environment in which updates continually occur. A similar query run in five minute intervals in an OLTP environment may yield different results, while the same query run within the data warehouse should remain stable and consistent. For example, an airline may capture frequent flyer information in its data warehouse. During check-in for a flight, the additional mileage for a specific passenger is immediately updated in the OLTP system, but is not yet reflected in the data warehouse until its next scheduled load.

Difference between ODS & DWH:

OLTP (ODS)	OLAP (DWH)
Application Oriented	Subject Oriented
Used to run business	Used to analyze business
Detailed data	Summarized and refined
Current up to date	Snapshot data
Isolated Data	Integrated Data
Repetitive access	Ad-hoc access
Clerical User	Knowledge User (Manager)
Performance Sensitive	Performance relaxed

Few Records accessed at a time (tens)	Large volumes accessed at a time(millions)
Read/Update Access	Mostly Read (Batch Update)
No data redundancy	Redundancy present
Database Size 100MB -100 GB	Database Size 100 GB - few terabytes

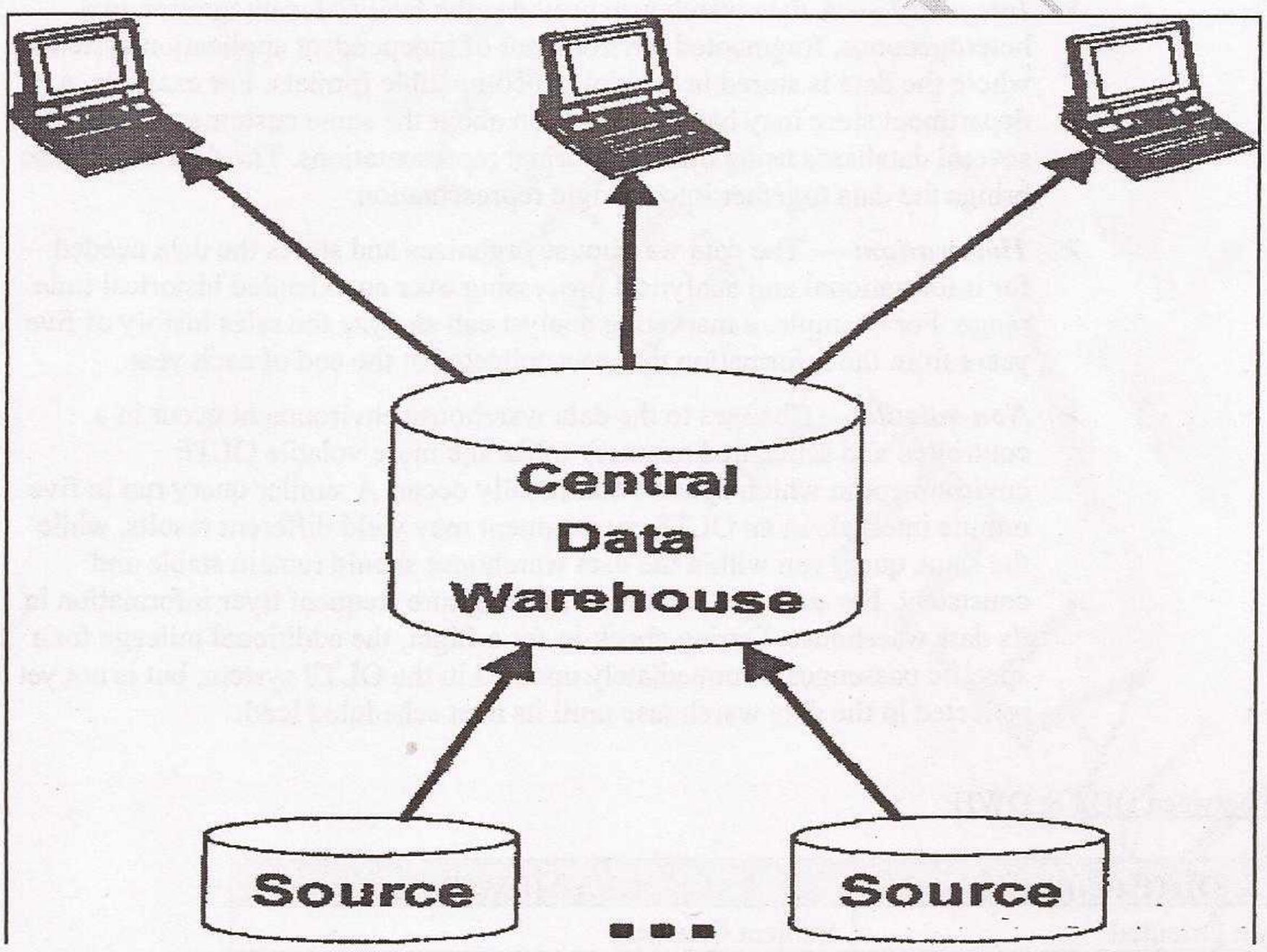
DWH architecture:

We are having 3 kinds of architecture for DWH.

1. Centralized architecture
2. Federated architecture
3. Tiered architecture

Centralized architecture:

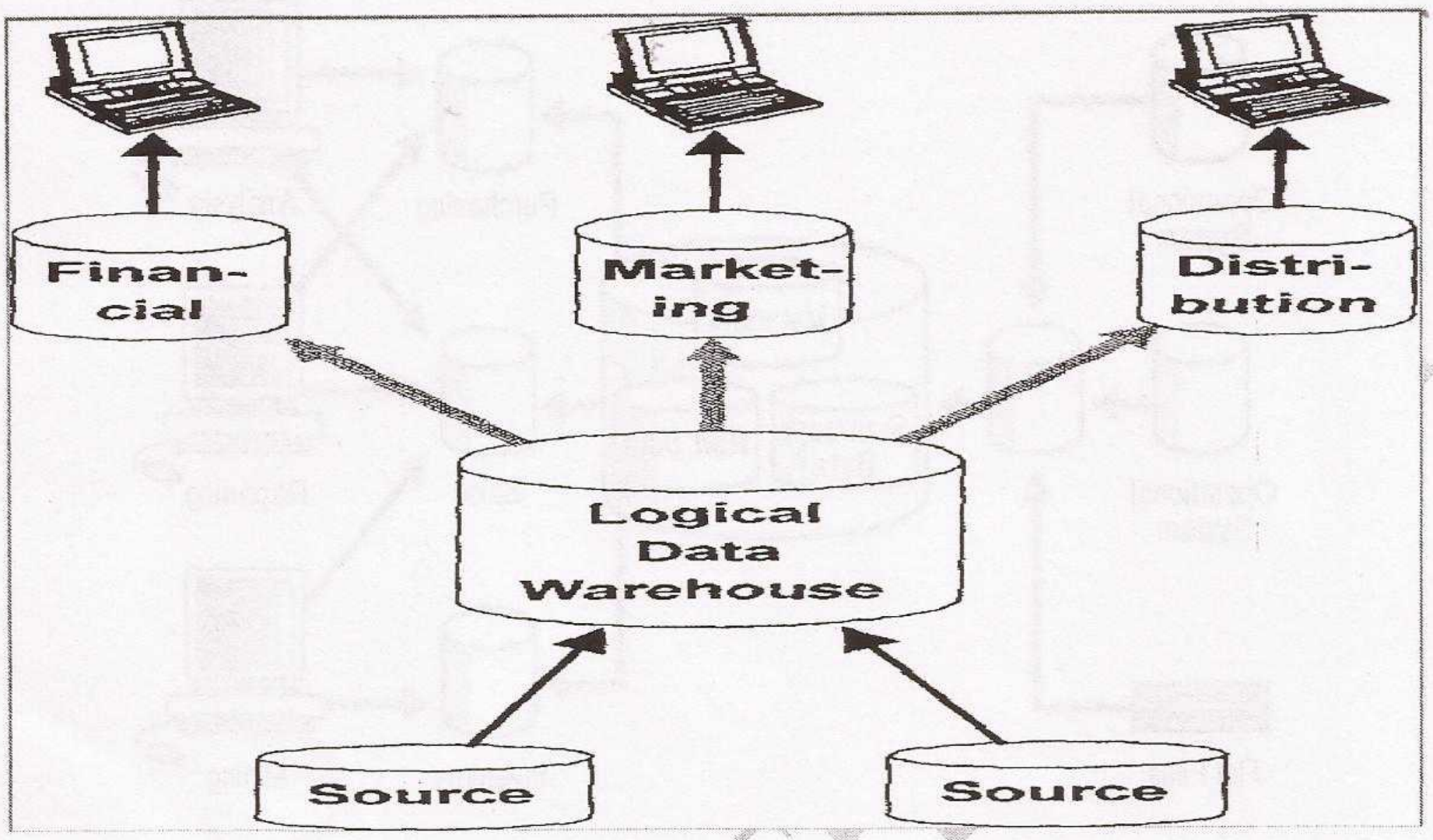
In a centralized architecture, there exists only one data warehouse which stores all data necessary for business analysis.



Federated architecture:

In a federated architecture the data is logically consolidated but stored in separate physical databases, at the same or at different physical sites. The local data marts store only the relevant information for a department.

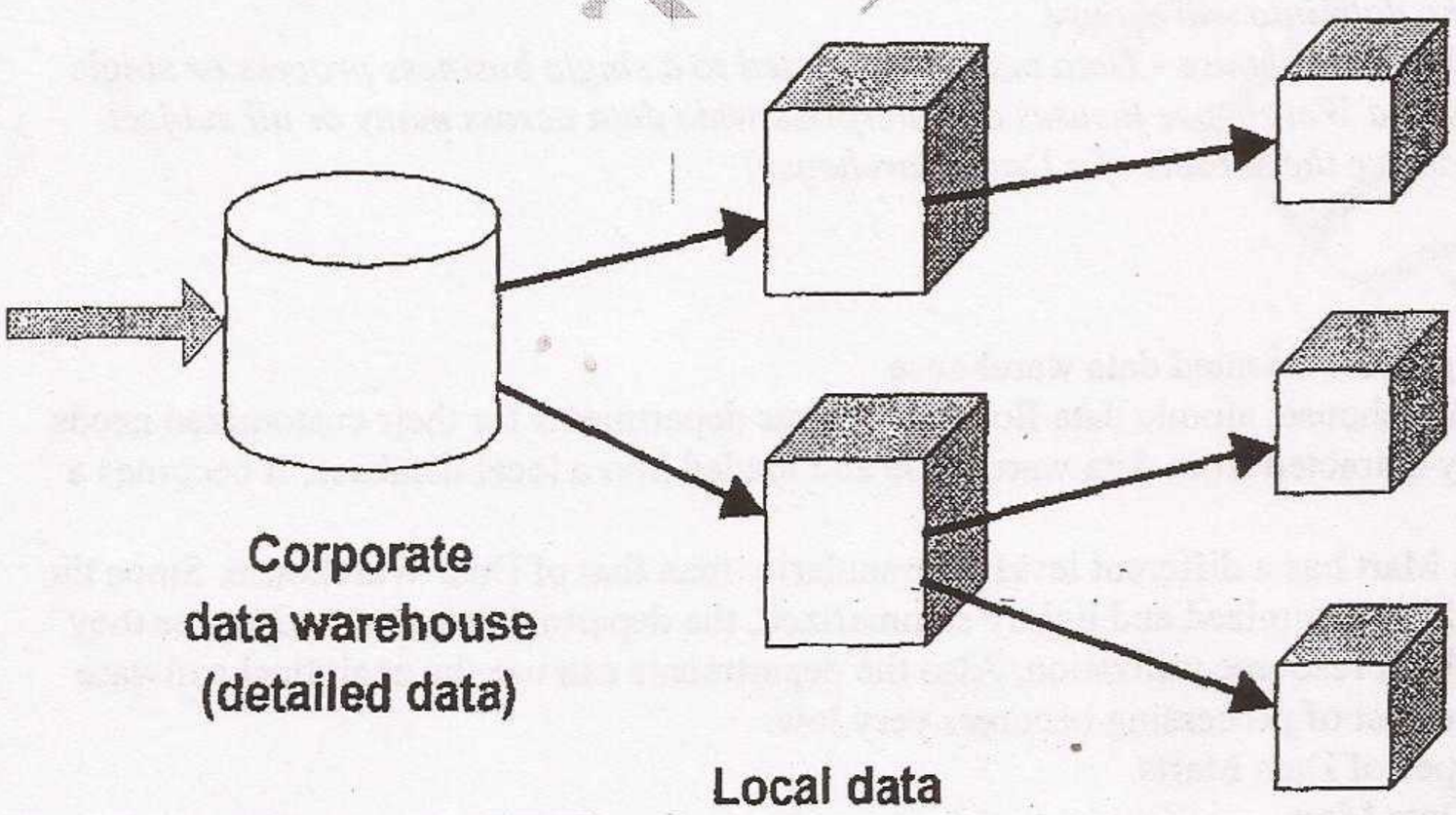
The amount of data is reduced in contrast to a central data warehouse. The level of detail is enhanced.



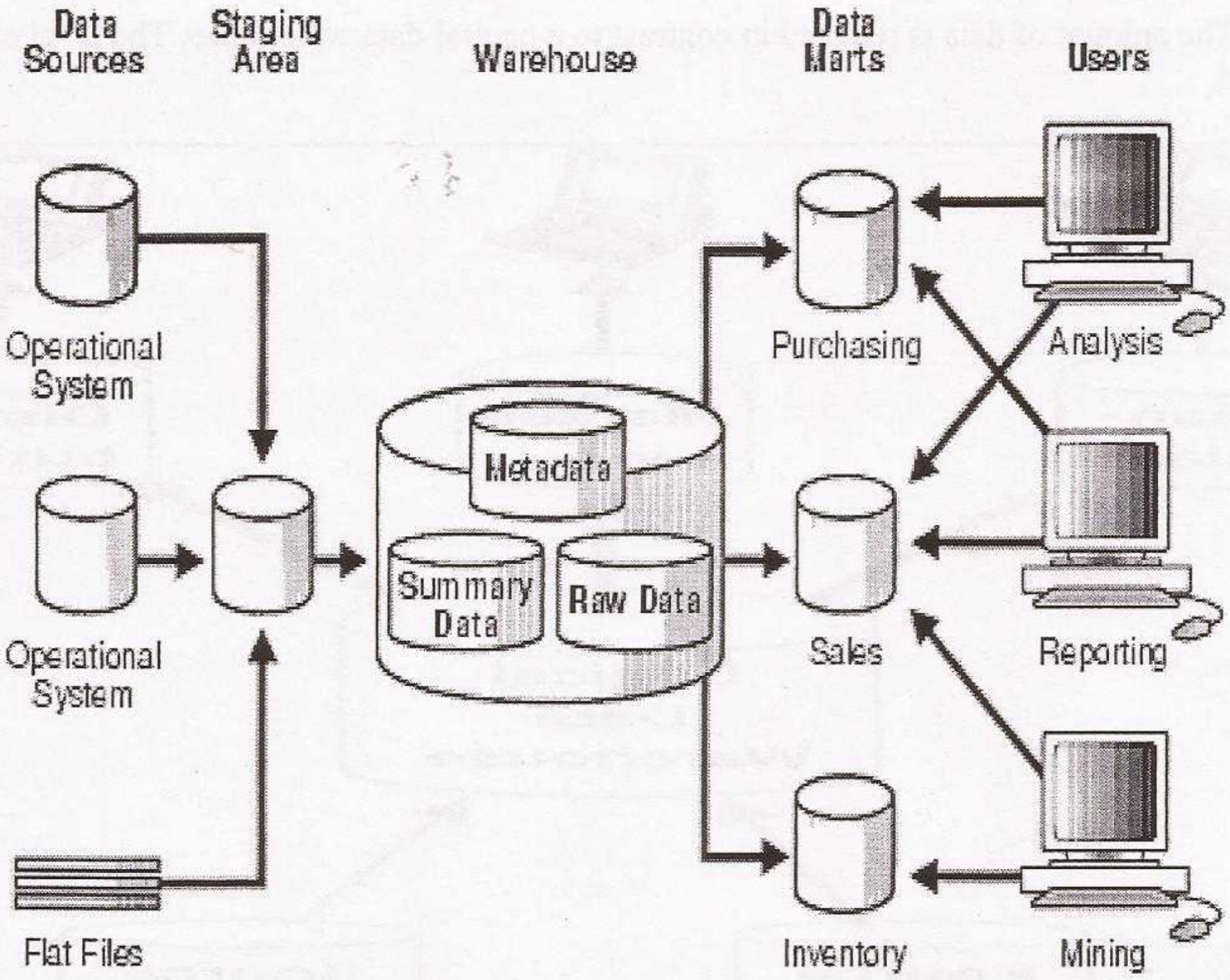
Tiered Architecture:

A tiered architecture is a distributed data approach. This process cannot be done in one step because many sources have to be integrated into the warehouse.

On a first level, the data of all branches in one region is collected, in the second level the data from the regions is integrated into one data warehouse.Eg: World→Countries→Cities→Regions



Generic Architecture for DWH:



As for above architecture data is integrated from different sources and that data stored in Staging DB. In staging Database all requirement related transformations performed. After that data will be loaded in to DWH form there data will be loaded in to different Data Marts for the purpose of analyzing data to make decisions by users.

- *Staging Area - Staging area is a place where you hold temporary tables on data warehouse server. We basically need staging area to hold the data and perform data cleansing and merging before loading the data into warehouse.*
- *Data marts Vs Data Warehouse - Data mart is restricted to a single business process or single business group. Data Warehouse focuses on enterprise wide data across many or all subject areas .Data Marts are the subsets of a Data Warehouse.*

Data Mart:

A data mart is a subject oriented data warehouse.

From the Data Warehouse, atomic data flows to various departments for their customized needs. If this data is periodically extracted from data warehouse and loaded into a local database, it becomes a data mart.

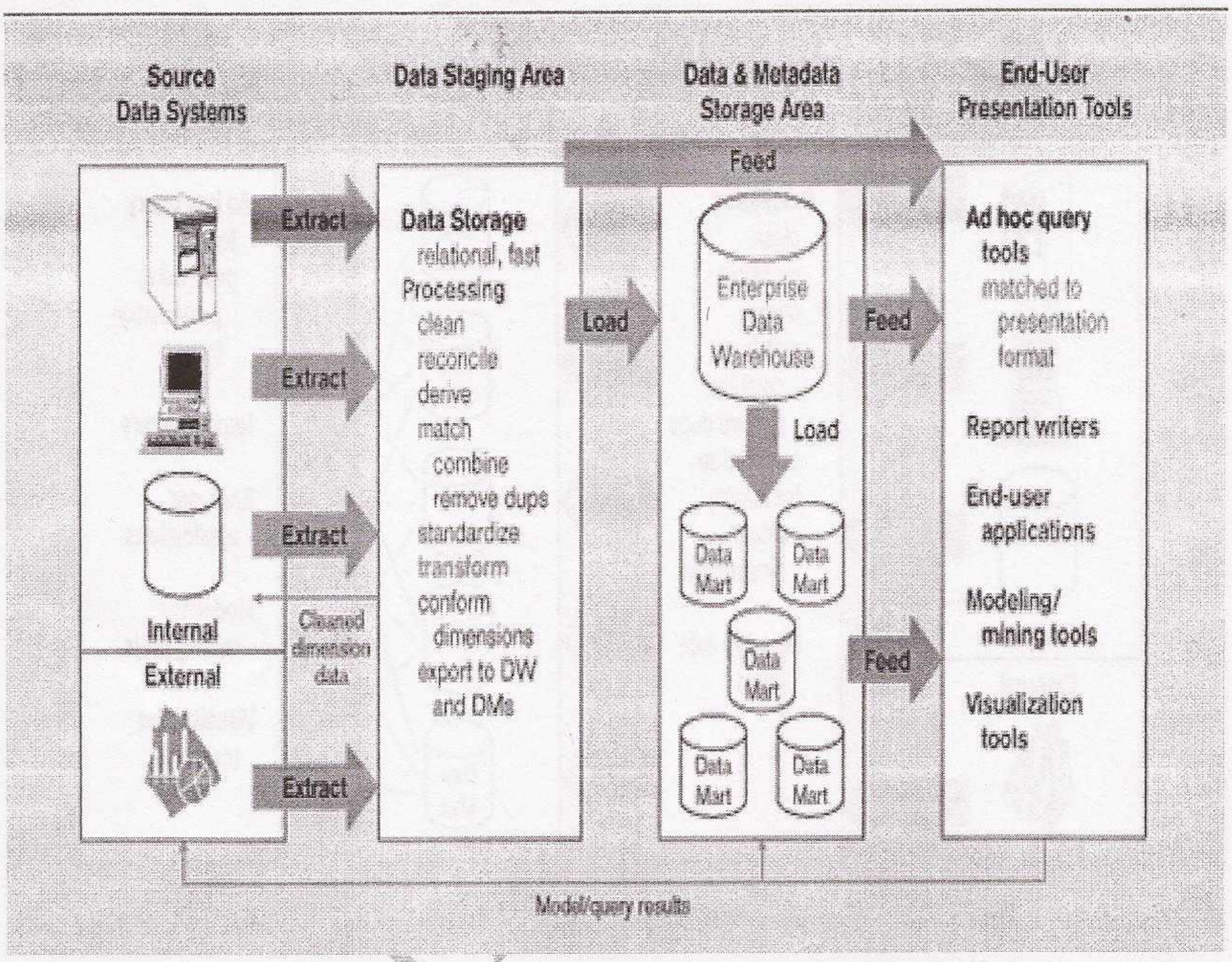
The data in Data Mart has a different level of granularity than that of Data Warehouse. Since the data in Data Marts is highly customized and lightly summarized, the departments can do whatever they want without worrying about resource utilization. Also the departments can use the analytical software they find convenient. The cost of processing becomes very low.

There are two types of Data Marts.

1. Dependent Data Mart
2. Independent Data Mart

Dependent Data Mart:

Here Data Marts are developed by using DWH.

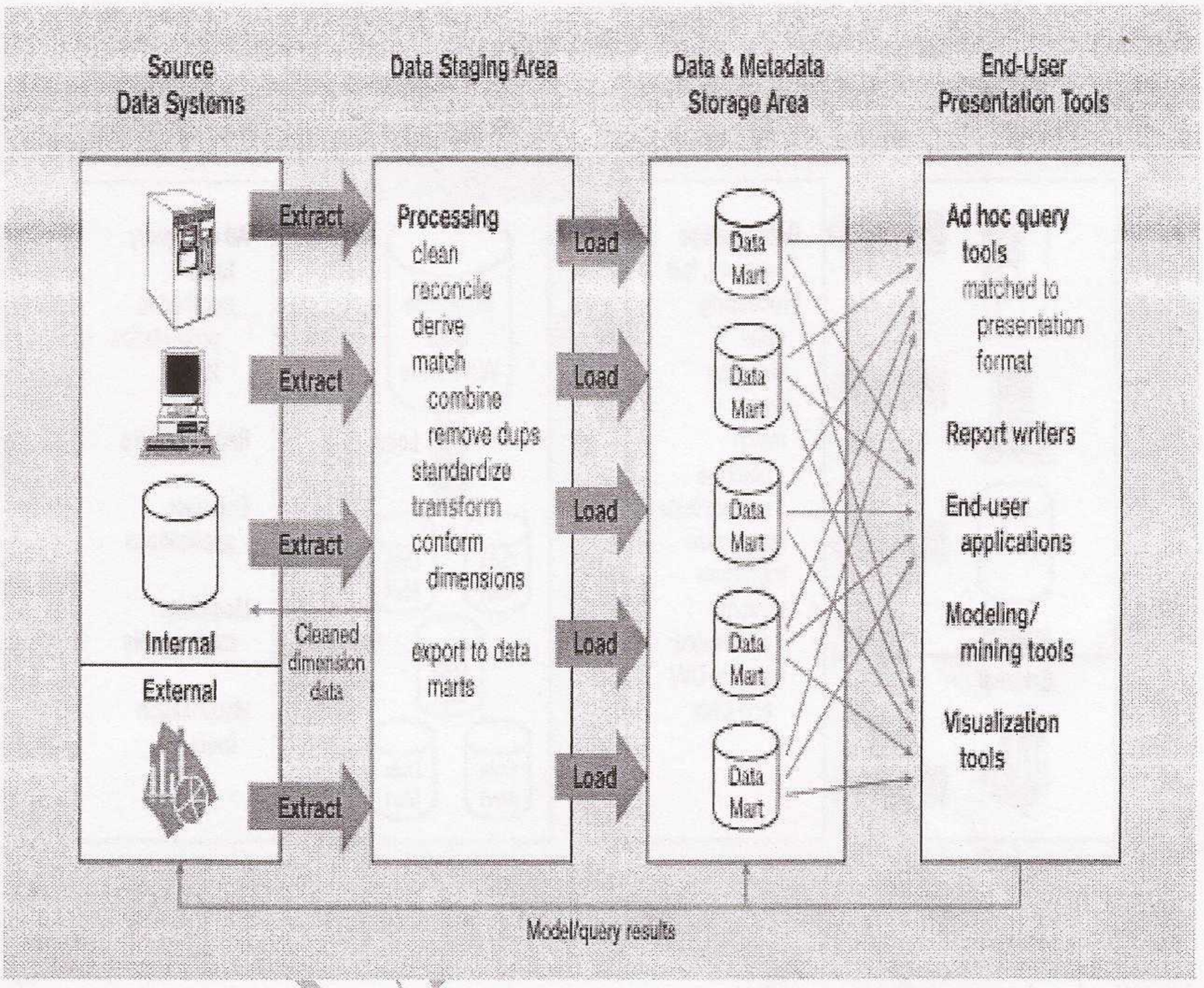


Top down Approach (*Inmon Approach)*

- *The data flow begins with data extraction from the operational data sources. This data is loaded into the staging area and validated and consolidated for ensuring a level of accuracy and then transferred to the Data warehouse.*
- *A new set of transformations in done on the data in the data warehouse to help organize the data in particular structures required by data marts. Then the data marts can be loaded with the data and the OLAP environment becomes available to the users.*

Independent DATAMART:

Here from Data Marts DWH will populate.



Bottom Up Approach (*Kimball Approach)*

- *The bottom-up approach reverses the positions of the Data warehouse and the Data marts. Data marts are directly loaded with the data from the operational systems through the staging area.*
- *The data flow in the bottom up approach starts with extraction of data from operational databases into the staging area where it is processed and consolidated and then loaded into the Data mart.*
- *The data from the Data Mart, then is extracted to the staging area aggregated, summarized and - so on and loaded into the Data Warehouse and made available to the end user for analysis.*

Data Modeling:

Data Modeling is a technique aimed at optimizing the way that information is stored and used within an organization. It begins with the identification of the main data groups, for example the invoice, and continues by defining the detailed content of each of these groups.

Commonly Used Data Modeling Notations:

- ***Entity*** – Denotes the principal data object about which information is to be collected. E.g.: Student data, Employee data etc.
- ***Attributes*** – Attributes are characteristics of an Entity.
  E.g.: Student number, student name…e.t.c
- ***Relationship*** - A natural business association that exists between one or more entities. The relationship may represent an event that links the entities or merely a logical affinity that exists between the entities

Why should a Tester Know Data Modeling?

- *Data Models provides the functional and technical aspects of the database design.*
- *Data Models help in ensuring that the design is complete for the defined business rules.*
- *Understanding the data models gives an understanding of the functionality to be tested.*
- *To carry out DB Testing like constraint testing, null value testing etc.*
- *In case multiple source systems, understanding the data model helps in validating the quality of data.*

We need to understand the two important concepts that actually drive the design of these data models for OLTP and OLAP systems. They are,

1. ER Data Model
2. Dimensional Data Model

ER Data Model:

- Entity – Relationship Data Model is a data model that views the real world as entities and relationships. Entities are concepts, real or abstract about which information is collected. Entities are associated with each other by relationship and attributes are properties of entities. Business rules would determine the relationship between each of entities in a data model.
- The goal of OLTP data model is to *normalize (avoid redundancy) data* and to present it in a good normal form. While working with OLTP data modelling, a data modeller has to understand 1st normal form thru 5th normal form to design a good data model.
- The OLTP data model is popularly called as the *OLTP Model or Relational Model*

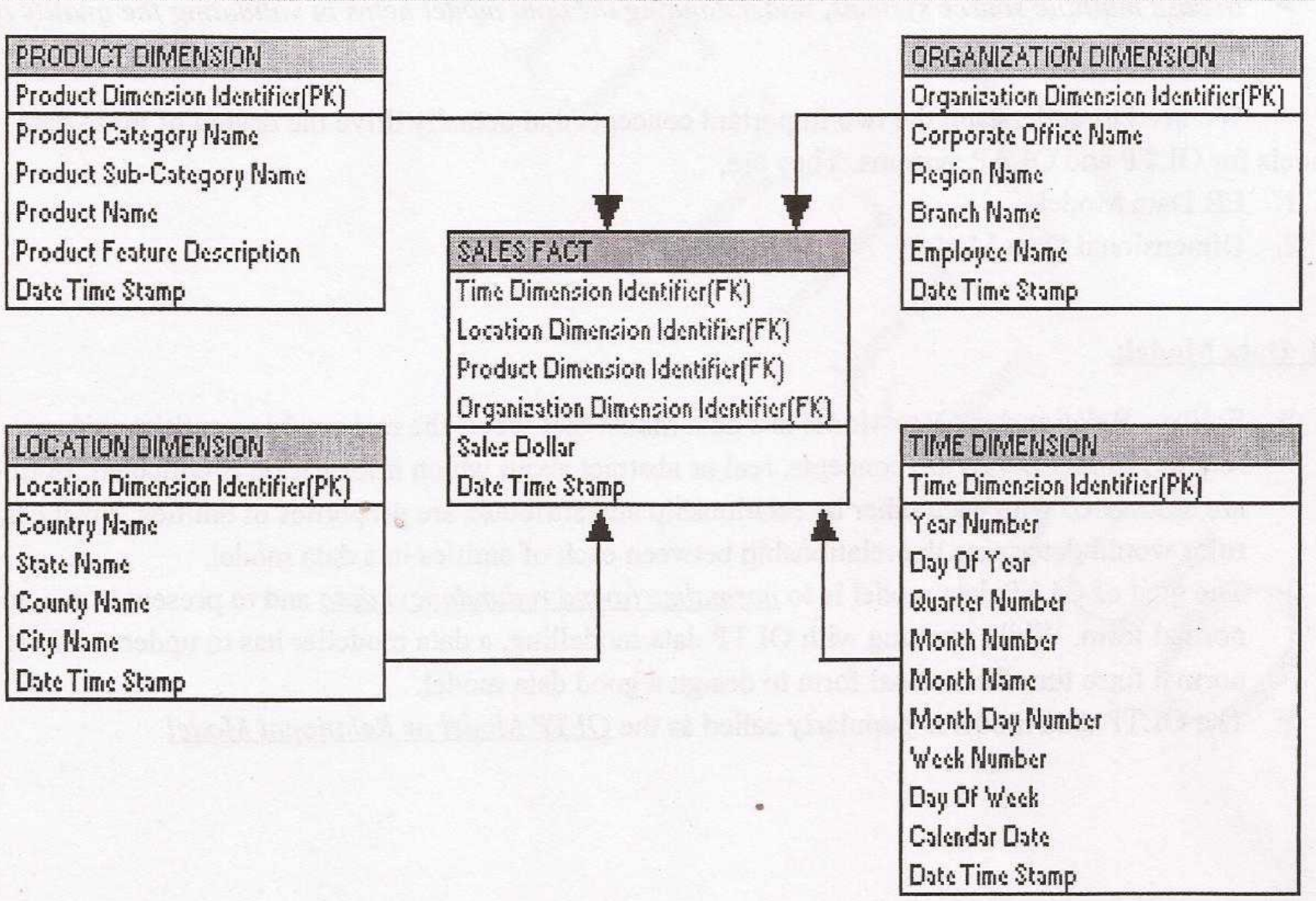
Dimensional Data Model

- Dimensional modelling is the design concept used by many data warehouse designers to build their data warehouse. It is a logical design technique that seeks to present the data in a standard, intuitive framework that allows for high-performance access.
- It adheres to a discipline that uses the relational model with some important restrictions.
    1. *Every dimensional model is composed of one table with a multi-part key, called the fact table, and a set of smaller tables called dimension tables.*
    2. *Good examples of dimensions are location, product, time, promotion, organization etc. Dimension tables store records related to that particular dimension and no facts (measures) are stored in these tables.*
    3. *A fact (measure) table contains measures (sales gross value, total units sold) and dimension columns. These dimension columns are actually foreign keys from the respective dimension tables.*
    4. *Since here we look at faster query process, the data is de-normalized.*

Dimensional Data Modeling – Example:

Sales fact table is connected to dimensions location, product, time and organization.

It shows that data can be sliced across all dimensions and again it is possible for the data to be aggregated across multiple dimensions. "Sales Dollar" in sales fact table can be calculated across all dimensions independently or in a combined manner.



Difference Between ER modeling & Dimensional Modeling:

ER Data Modeling	Dimensional Data Modeling
Data is stored in RDBMS	Data is stored in RDBMS or Multidimensional databases
Tables are units of storage	Cubes are units of storage
Data is normalized and used for OLTP. Optimized for OLTP processing	Data is demoralized and used in data warehouse and data mart. Optimized for OLAP
Several tables and chains of relationships among them	Few tables and fact tables are connected to dimensional tables
Volatile(several updates) and time variant	Non volatile and time invariant
SQL is used to manipulate data	ETL tools is used to manipulate data
Detailed level of transactional data	Summary of bulky transactional data(Aggregates and Measures) used in business decisions
Normal Reports	User friendly, interactive, drag and drop multidimensional OLAP Reports

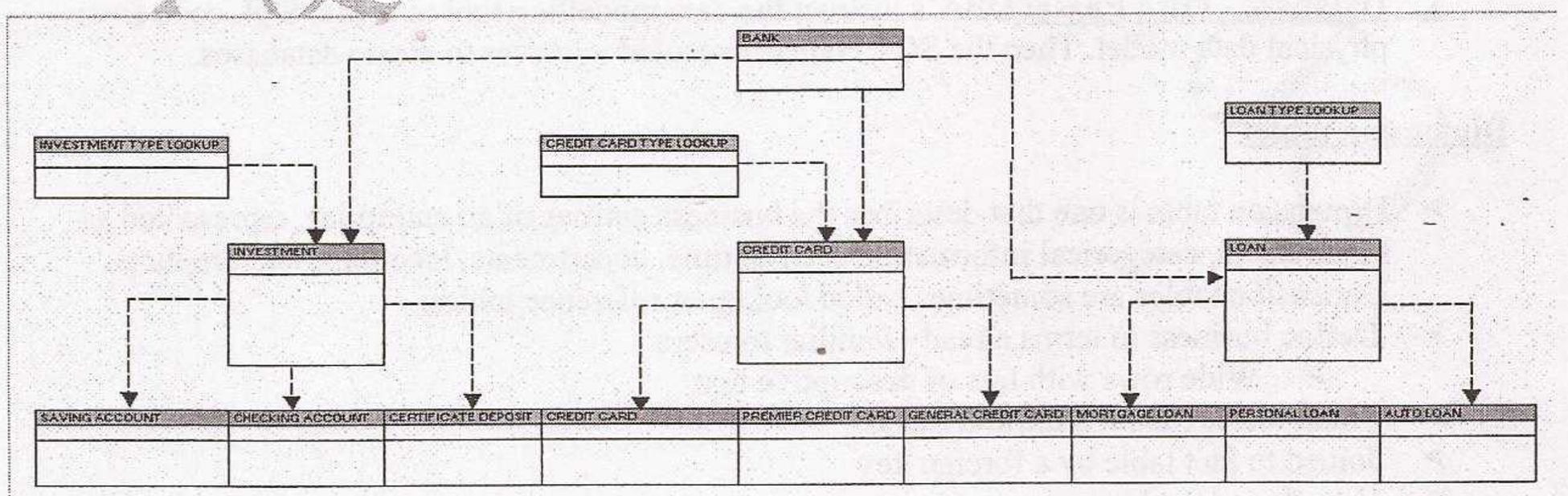
How to build Dimensional Modeling:

To build Dimensional modeling we need to follow five different phases.

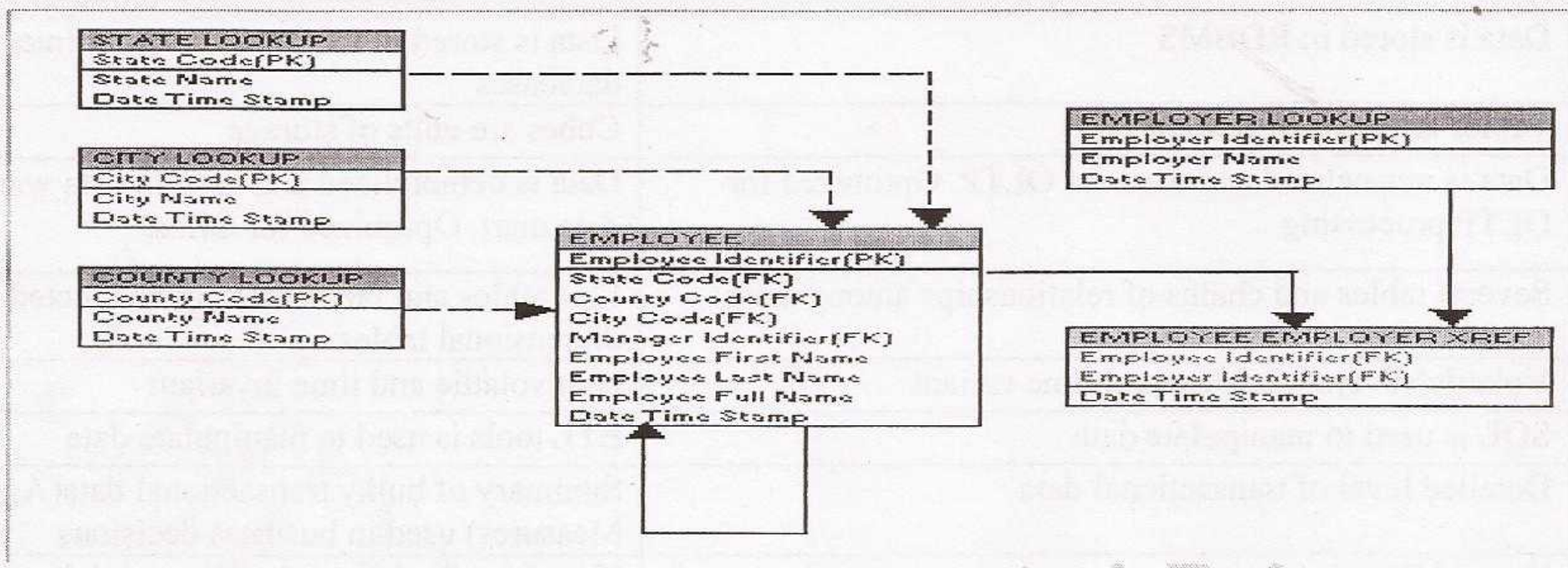
1. Gathering Business Requirements
2. Conceptual Data Modelling (CDM)
3. Logical Data Modelling (LDM)
4. Physical Data Modelling (PDM)
5. Generate Database

1. **Gathering Business Requirements - First Phase:** Data Modellers have to interact with business analysts to get the functional requirements and with end users to find out the reporting needs.

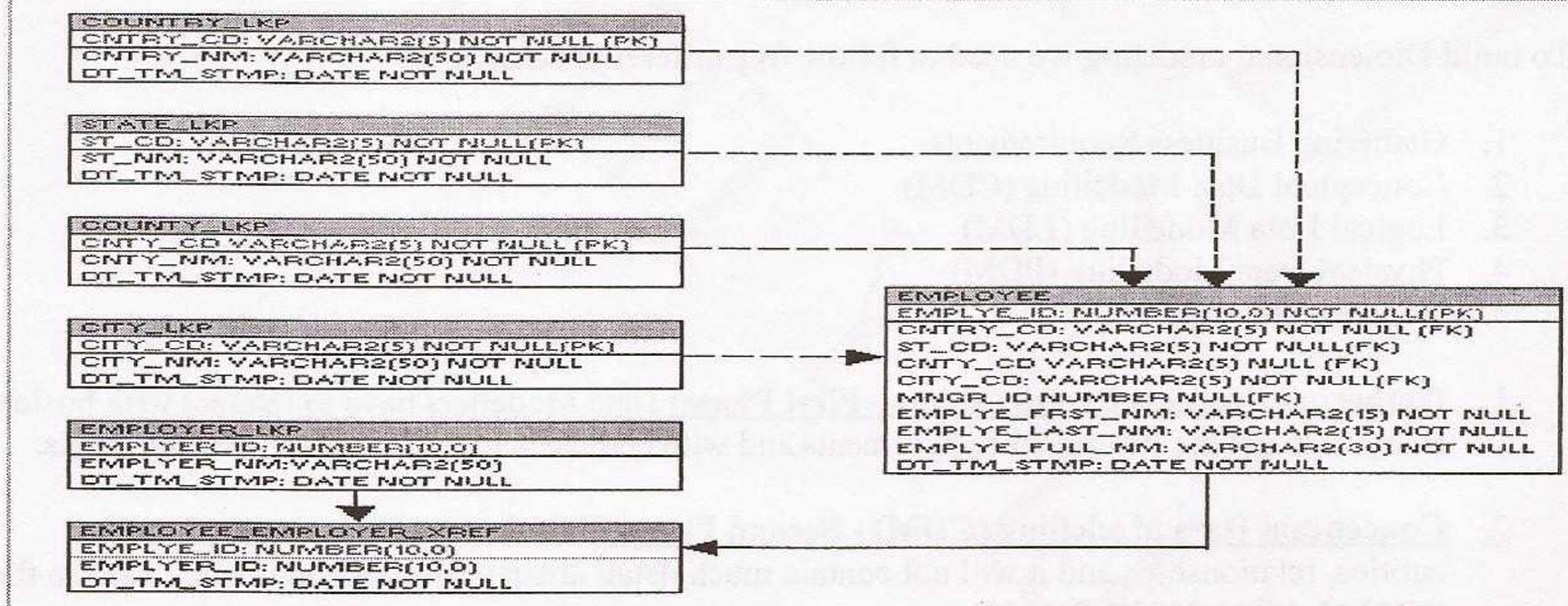
2. **Conceptual Data Modelling (CDM) - Second Phase:** This data model includes all major entities, relationships and it will not contain much detail about attributes and is often used in the INITIAL PLANNING PHASE.



3. **Logical Data Modelling (LDM) - Third Phase:** This is the actual implementation of a conceptual model in a logical data model. A logical data model is the version of the model that represents all of the business requirements of an organization.



4. **Physical Data Modelling (PDM) - Fourth Phase:** This is a complete model that includes all required tables, columns, relationship, database properties for the physical implementation of the database.



5. **Database - Fifth Phase:** DBA's instruct the data modelling tool to create SQL code from physical data model. Then the SQL code is executed in server to create databases.

Dimension table:

- Dimension table is one that describes the business entities of an enterprise, represented as hierarchical, categorical information such as time, departments, locations, and products. Dimension tables are sometimes called lookup or reference tables.
- Define business in terms already familiar to users
  - Wide rows with lots of descriptive text
- Small tables (about a million rows)
- Joined to fact table by a foreign key
- Heavily indexed

- Some Typical dimensions - time periods, geographic region (markets, cities), products, customers, salesperson, etc.

Fact Table:

- Represents a business process, i.e., models the business process as an artifact in the data model
- Contain the measurements or metrics or facts of business processes. Usually a numeric data.
- Some typical facts in a Fact Table are,
  - "monthly sales number" in the Sales business process
  - most are additive (sales this month), some are semi-additive (balance as of), some are not additive (unit price)

Fact tables contain foreign keys to the dimension tables

Measures: (Facts)

- A Fact table consists of measures.
- The *measures* are quantitative or factual data about the subject. The measures are generally numeric and correspond to the *how much* or *how many* aspects of a question.
- A measure can be based on a column in a table or it can be calculated.
- Examples - price, product sales, product inventory, revenue, sale amount, net profit margin and so forth.

Facts do not exist in a vacuum. They exist in the context of time, place, product, etc. For example, "units sold" means sales of a particular product, at a particular time, in a particular place. A fact stands at a cross-section of multiple dimensions. The combination of facts and the "who, what, where, and when" of these facts can be represented in a dimensional star schema.

Dimensions:

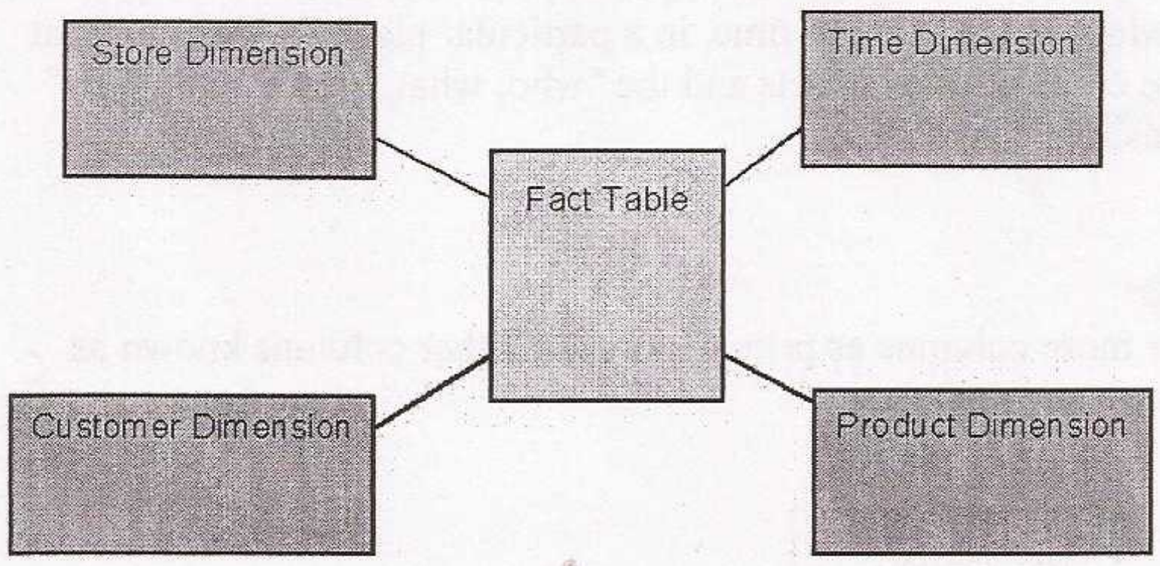
Dimension tables are a composite of one or more columns as primary key and other columns known as attributes. As an example, a time dimension is shown below.

Dimensions typically contain hierarchies. Hierarchies show the parent-child relationships between elements of the dimension. Hierarchies are used to logically group and analyze information within one dimension. In other words, dimensions contain hierarchies, which are natural structures within business dimensions, as shown in Table 4 below:

Description	Hierarchy
Geography Dimension - table is at the store level but can be rolled up through to region	Store within Zip Code Zip Code within City City within Country Country within State State within Region
Product Dimension Hierarchy	Universal Product Code within Product Line Product Line within Brand Brand within Category Category within Department
Time Dimension Hierarchy	Day within Week Week within Month Month within Quarter Quarter within Year

Star Schema:

In the star schema design, a single object (the fact table) sits in the middle and is radially connected to other surrounding objects (dimension lookup tables) like a star. Each dimension is represented as a single table. The primary key in each dimension table is related to a foreign key in the fact table.



http://www.1keydata.com/datawarehousing/star-schema.html

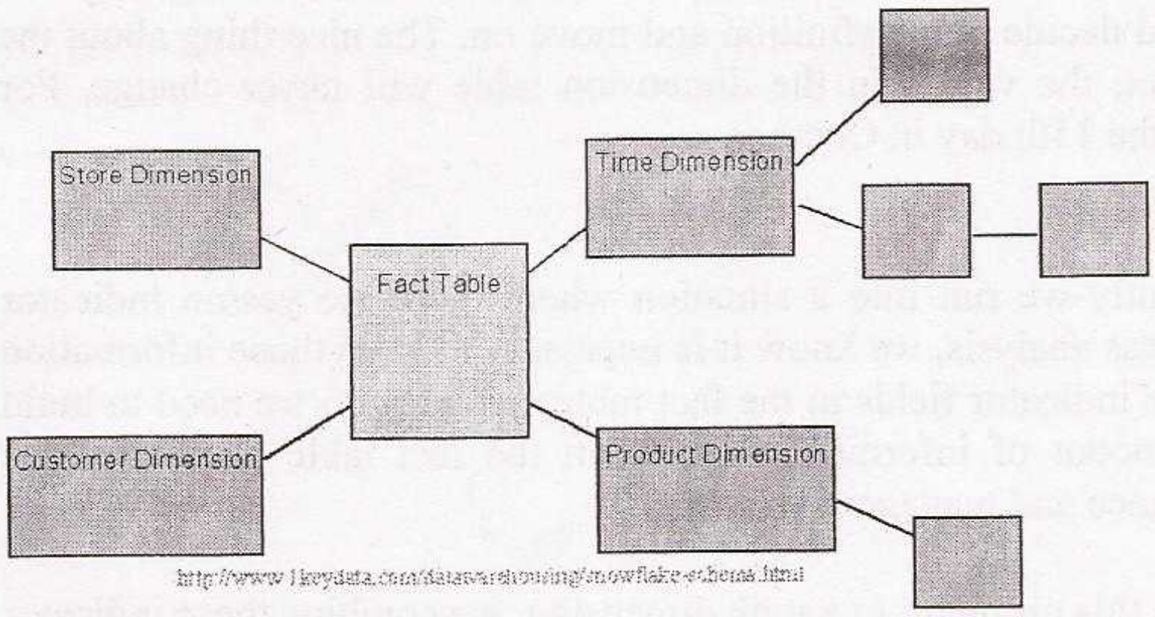
All measures in the fact table are related to all the dimensions that fact table is related to. In other words, they all have the same level of granularity.

A star schema can be simple or complex. A simple star consists of one fact table; a complex star can have more than one fact table.

Let's look at an example: Assume our data warehouse keeps store sales data, and the different dimensions are time, store, product, and customer. In this case, the figure on the left repesents our star schema. The lines between two tables indicate that there is a primary key / foreign key relationship between the two tables. Note that different dimensions are not related to one another.

Snow flake schema

The snowflake schema is an extension of the star schema, where each point of the star explodes into more points. In a star schema, each dimension is represented by a single dimensional table, whereas in a snowflake schema, that dimensional table is normalized into multiple lookup tables, each representing a level in the dimensional hierarchy.



For example, the Time Dimension that consists of 2 different hierarchies:

1. Year → Month → Day
2. Week → Day

We will have 4 lookup tables in a snowflake schema: A lookup table for year, a lookup table for month, a lookup table for week, and a lookup table for day. Year is connected to Month, which is then connected to Day. Week is only connected to Day. A sample snowflake schema illustrating the above relationships in the Time Dimension is shown to the right.

The main advantage of the snowflake schema is the improvement in query performance due to minimized disk storage requirements and joining smaller lookup tables. The main disadvantage of the snowflake schema is the additional maintenance efforts needed due to the increase number of lookup tables.

Types of Dimensions:

- Conformed Dimensions
- Degenerate Dimensions
- Junk Dimensions
- Slowly changing Dimensions

Conformed Dimension:

A conformed dimension is a dimension that has exactly the same meaning and content when being referred from different fact tables. A conformed dimension can refer to multiple tables in multiple data marts within the same organization. For two dimension tables to be considered as conformed, they must either be identical or one must be a subset of another. There cannot be any other type of difference

between the two tables. For example, two dimension tables that are exactly the same except for the primary key are not considered conformed dimensions.

The time dimension is a common conformed dimension in an organization. Usually the only rules to consider with the time dimension are whether there is a fiscal year in addition to the calendar year and the definition of a week. Fortunately, both are relatively easy to resolve. In the case of fiscal vs calendar year, one may go with either fiscal or calendar, or an alternative is to have two separate conformed dimensions, one for fiscal year and one for calendar year. The definition of a week is also something that can be different in large organizations: Finance may use Saturday to Friday, while marketing may use Sunday to Saturday. In this case, we should decide on a definition and move on. The nice thing about the time dimension is once these rules are set, the values in the dimension table will never change. For example, October 16th will never become the 15th day in October.

Junk Dimension:

In data warehouse design, frequently we run into a situation where there are yes/no indicator fields in the source system. Through business analysis, we know it is necessary to keep those information in the fact table. However, if keep all those indicator fields in the fact table, not only do we need to build many small dimension tables, but the amount of information stored in the fact table also increases tremendously, leading to possible performance and management issues.

Junk dimension is the way to solve this problem. In a junk dimension, we combine these indicator fields into a single dimension. This way, we'll only need to build a single dimension table, and the number of fields in the fact table, as well as the size of the fact table, can be decreased. The content in the junk dimension table is the combination of all possible values of the individual indicator fields.

Let's look at an example. Assuming that we have the following fact table:

FACT_TABLE

CUSTOMER_ID
PRODUCT_CD
TXN_ID
STORE_ID
TXN_CODE
COUPON_IND
PREPAY_IND
TXN_AMT

In this example, the last 3 fields are all indicator fields. In this existing format, each one of them is a dimension. Using the junk dimension principle, we can combine them into a single junk dimension, resulting in the following fact table:

FACT_TABLE

CUSTOMER_ID
PRODUCT_CD
TXN_ID
STORE_ID
JUNK_ID
TXN_AMT

Note that now the number of dimensions in the fact table went from 7 to 5.
The content of the junk dimension table would look like the following:

DIM_JUNK

JUNK_ID	TXN_CODE	COUPON_IND	PREPAY_IND
1	1	Y	Y
2	2	Y	Y
3	3	Y	Y
4	1	Y	N
5	2	Y	N
6	3	Y	N
7	1	N	Y
8	2	N	Y
9	3	N	Y
10	1	N	N
11	2	N	N
12	3	N	N

In this case, we have 3 possible values for the TXN_CODE field, 2 possible values for the COUPON_IND field, and 2 possible values for the PREPAY_IND field. This results in a total of 3 x 2 x 2 = 12 rows for the junk dimension table.

By using a junk dimension to replace the 3 indicator fields, we have decreased the number of dimensions by 2 and also decreased the number of fields in the fact table by 2. This will result in a data warehousing environment that offers better performance as well as being easier to manage.

Degenerate Dimensions:

A degenerate dimension is when the dimension attribute is stored as part of fact table, and not in a separate dimension table. These are essentially dimension keys for which there are no other attributes. In a data warehouse, these are often used as the result of a drill through query to analyze the source of an aggregated number in a report. You can use these values to trace back to transactions in the OLTP system.

Role Playing Dimensions:

A role-playing dimension is one where the same dimension key — along with its associated attributes — can be joined to more than one foreign key in the fact table. For example, a fact table may include foreign keys for both Ship Date and Delivery Date. But the same date dimension attributes apply to each foreign key, so you can join the same dimension table to both foreign keys. Here the date dimension is taking multiple roles to map ship date as well as delivery date, and hence the name of Role Playing dimension.

Slowly Changing Dimensions

Christina is a customer with ABC Inc. She first lived in Chicago, Illinois. So, the original entry in the customer lookup table has the following record

Customer Key	Name	State
1001	Christina	Illinois

At a later date, she moved to Los Angeles, California on January, 2003. How should ABC Inc. now modify its customer table to reflect this change? This is the "Slowly Changing Dimension"

Type 1: The new record replaces the original record. No trace of the old record exists.

Type 2: A new record is added into the customer dimension table. Therefore, the customer is treated essentially as two people.

Type 3: The original record is modified to reflect the change.

In Type 1 Slowly Changing Dimension, the new information simply overwrites the original information. In other words, no history is kept.

In our example, recall we originally have the following table:

Customer Key	Name	State
1001	Christina	Illinois

After Christina moved from Illinois to California, the new information replaces the new record, and we have the following table:

Customer Key	Name	State
1001	Christina	California

- **Advantages**: - This is the easiest way to handle the Slowly Changing Dimension problem, since there is no need to keep track of the old information.
- **Disadvantages**: - All history is lost. By applying this methodology, it is not possible to trace back in history. For example, in this case, the company would not be able to know that Christina lived in Illinois before.
- **Usage**: About 50% of the time.
- **When to use Type** 1: Type 1 slowly changing dimension should be used when it is not necessary for the data warehouse to keep track of historical changes.

In Type 2 Slowly Changing Dimension, a new record is added to the table to represent the new information. Therefore, both the original and the new record will be present. The newe record gets its own primary key.

In our example, recall we originally have the following table:

Customer Key	Name	State
1001	Christina	Illinois

After Christina moved from Illinois to California, we add the new information as a new row into the table:

Customer Key	Name	State
1001	Christina	Illinois
1005	Christina	California

- **Advantages**: - This allows us to accurately keep all historical information.
- **Disadvantages**: - This will cause the size of the table to grow fast. In cases where the number of rows for the table is very high to start with, storage and performance can become a concern. - This necessarily complicates the ETL process.
- **Usage**: About 50% of the time.

- **When to use Type 2**: Type 2 slowly changing dimension should be used when it is necessary for the data warehouse to track historical changes.

In Type 3 Slowly Changing Dimension, there will be two columns to indicate the particular attribute of interest, one indicating the original value, and one indicating the current value. There will also be a column that indicates when the current value becomes active.

In our example, recall we originally have the following table:

Customer Key	Name	State
1001	Christina	Illinois

- To accommodate Type 3 Slowly Changing Dimension, we will now have the following columns:

  - Customer Key
  - Name
  - Original State
  - Current State
  - Effective Date

After Christina moved from Illinois to California, the original information gets updated, and we have the following table (assuming the effective date of change is January 15, 2003):

Customer Key	Name	Original State	Current State	Effective Date
1001	Christina	Illinois	California	15-JAN-2003

- **Advantages**: - This does not increase the size of the table, since new information is updated. This allows us to keep some part of history.
- **Disadvantages**: - Type 3 will not be able to keep all history where an attribute is changed more than once. For example, if Christina later moves to Texas on December 15, 2003, the California information will be lost.
- **Usage**: Type 3 is rarely used in actual practice.
- When to use Type 3: Type III slowly changing dimension should only be used when it is necessary for the data warehouse to track historical changes, and when such changes will only occur for a finite number of time.

1)SCD-1:

- HERE THE PREVIOUS DATA OVERWRITE BY CURRENT DATA
- MEANS HERE ONLY MAINTAIN CURRENT DATA.

2)SCD-2:

- HERE JUST ADD THE ADDITIONAL RECORDS
  - VERSIONING: MEANS HERE JUST SEND THE UPDATED RECORDS TO THE TARGET ALONG WITH VERSION NUMBER.NEW RECORDS WILL BE SENDING TO THE TARGET ALONG WITH PRIMARY KEY

- FLAGVALUE:HERE UPDATED RECORDS SEND TO THE TARGET ALONG WITH 0 AND RECENT RECORDS SEND TO THE TARGET ALONG WITH 1
- EFFECTIVE DATE RANGE:MEANS HERE TRACKS THE BOTH PREVIOUS AND CURRENT DATA

Method1 for Type 2: Versioning

Supplier_Key	Supplier_Code	Supplier_Name	Supplier_State	Version.
123	ABC	Acme Supply Co	CA	0
124	ABC	Acme Supply Co	IL	1

Method2: for Type To Add the Effective date Columns

Supplier_Key	Supplier_Code	Supplier_Name	Supplier_State	Start_Date	End_Date
123	ABC	Acme Supply Co	CA	01-Jan-2000	21-Dec-2004
124	ABC	Acme Supply Co	IL	22-Dec-2004	

The null End_Date in row two indicates the current tuple version. In some cases, a standardized surrogate high date (e.g. 9999-12-31) may be used as an end date, so that the field can be included in an index, and so that null-value substitution is not required when querying.

3)SCD-3:

- HERE MAINTAINS JUST PREVIOUS AND CURRENT DATA

Types of Facts -

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

Example of Additive Fact

Date
Store
Product
Sales_Amount

Sales_Amount is an additive fact, because you can sum up this fact along any of the three dimensions present in the fact table -- date, store, and product

Example of Semi-Additive or Non Additive Fact

Date
Account
Current_Balance
Profit_Margin

Current_Balance is a semi-additive fact, as it makes sense to add them up for all accounts (what's the total current balance for all accounts in the bank?), but it does not make sense to add them up through time

Profit_Margin is a non-additive fact, for it does not make sense to add them up for the account level or the day level.

Fact Less Fact tables:

A factless fact table is a fact table that does not have any measures. It is essentially an intersection of dimensions. On the surface, a factless fact table does not make sense, since a fact table is, after all, about facts. However, there are situations where having this kind of relationship makes sense in data warehousing.

For example, think about a record of student attendance in classes. In this case, the fact table would consist of 3 dimensions: the student dimension, the time dimension, and the class dimension. This factless fact table would look like the following:

FACT_ATTENDANCE

STUDENT_ID
CLASS_ID
TIME_ID

The only measure that you can possibly attach to each combination is "1" to show the presence of that particular combination. However, adding a fact that always shows 1 is redundant because we can simply use the COUNT function in SQL to answer the same questions.

Fact less fact tables offers the most flexibility in data warehouse design. For example, one can easily answer the following questions with this fact less fact table:

- How many students attended a particular class on a particular day?
- How many classes on average does a student attend on a given day?

Without using a fact less fact table, we will need two separate fact tables to answer the above two questions. With the above fact less fact table, it becomes the only fact table that's needed.

Types of Fact Tables

- **Cumulative**: This type of fact table describes what has happened over a period of time. For example, this fact table may describe the **total sales by product by store by day**. The facts for this type of fact tables are mostly additive facts. The first example presented here is a cumulative fact table.

- **Snapshot**: This type of fact table describes **the state of things in a particular instance of time,** and usually includes more semi-additive and non-additive facts. The second example presented here is a snapshot fact table

Database versus Data warehouse testing

Data Base Testing	Data Warehouse/ETL Testing
Smaller in Scale	Large Scale of data
Usually used to test the data at the source instead of testing using GUI	Includes several Facets , Extraction Transformation and Loading mechanisms
Usually Homogenous Data	Heterogeneous Data Involved
Transactional Operations	Usually Read Only Operations
Consistent data	Temporal Data inconsistency

1. *Based on which flow DWH is implemented?*
   a) Data→Information→Knowledge
   b) Information→Data→Knowledge
   c) Knowledge→Information→Data
   d) None
2. *Which technology will be used by DWH Project?*
   a) OLTP b) OLCP c) OLAP d) All
3. *In DWH data will present in _____ format?* Normalized
   a) Normalized b) Denormalized d)Unorganized d)None
4. *Who are end users for DWH?*
   a) Managers b) CEO d) Middle level management d) All
5. *Who will not involve in DWH project?*
   a) Business Analyst b) Data modeler d) Tester d) None
6. *DWH is also called as…?*
   a) Decision Support System b) History Database d)Read-only Database d)All
7. *By using DWH which questions can be resolved by DWH users?*
   a) What is the most effective distribution channel?
   b) Who are my customers and what products are they buying?
   c) What product promotions have the biggest impact on revenue?
   d) All
8. *Pick characteristics of DWH..?*
   a) Time Variant b) Integrated c) Non Volatile b) Subject – Oriented E) Normalized
9. *DWH is used for _____ business..?*
   a) Running b) Analyzing c)observing d) All
10. *In DWH is data is ___ ?*
    a) Redundant b) Non redundant c) Volatile d) None
11. *Different types of architectures in DWH…?*
    a) Centralized b) Federated C)Tiered d) All
12. *Data mart is..?*
    a) Subset of DWH b) Subject oriented DWH c) Both a&b d) None
13. *In Bottom-Up approach which data marts will be used..?*
    a) Dependent Data marts b) Independent Data marts c) Active data marts d)None
14. *Top Down approach is suggested by..?*
    a) Inmon b) Kimball c) Both d) None
15. *In DWH which modeling technique will be used..?*
    a) E-R modeling b) Dimensional Modeling C) Hybrid Modeling D) All
16. *Dimension & Fact contains….?*
    a) Descriptive & Numeric data respectively
    b) Numeric & Descriptive data respectively
    c) Both contains descriptive data
    d) Both contains numeric data
17. *Which schema will provide good query performance.*
    a) Star Schema b) Snowflake Schema C)Both
18. *To maintain partial history which type of SCD will use*
    a) SCD-1 b) SCD-2 d)SCD-3 d)None