Chapter 1: Introduction to Data Warehousing and Dimensional Modeling

Data Warehouse:

A data warehouse is a subject-oriented, integrated, time-variant, and nonvolatile collection of data in support of management's decision making process The four keywords, subject-oriented, integrated, time-variant, and nonvolatile, distinguish data warehouses from other data repository systems, such as relational database systems, transaction processing systems, and file systems.

Subject-oriented: A data warehouse is organized around major subjects, such as customer, supplier, product, and sales. Rather than concentrating on the day-to-day operations and transaction processing of an organization, a data warehouse focuses on the modeling and analysis of data for decision makers. Hence, data warehouses typically provide a simple and concise view around particular subject issues by excluding data that are not useful in the decision support process.

Integrated: A data warehouse is usually constructed by integrating multiple heterogeneous sources, such as relational databases, flat files, and on-line transaction records. Data cleaning and data integration techniques are applied to ensure consistency in naming conventions, encoding structures, attribute measures.

Time-variant: Data are stored to provide information from a historical perspective (e.g., the past 5–10 years). Every key structure in the data warehouse contains, either implicitly or explicitly, an element of time.

Nonvolatile: A data warehouse is always a physically separate store of data transformed from the application data found in the operational environment. Due to this separation, a data warehouse does not require transaction processing, recovery, and concurrency control mechanisms. It usually requires only two operations in data accessing: initial loading of data and access of data.

How are organizations using the information from data warehouses

Many organizations use this information to support business decision-making activities, including (1) increasing customer focus, which includes the analysis of customer buying pat- terns (such as buying preference, buying time, budget cycles, and appetites for spending)

(2) repositioning products and managing product portfolios by comparing the performance of sales by quarter, by year, and by geographic regions in order to fine- tune production strategies;

(3) analyzing operations and looking for sources of profit; and

(4) managing the customer relationships, making environmental corrections, and managing the cost

Differences between Operational Database Systems and Data Warehouses

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

The major distinguishing features between OLTP and OLAP are summarized as follows:

Users and system orientation: An OLTP system is customer-oriented and is used for transaction and query processing by clerks, clients, and information technology professionals. An OLAP system is market-oriented and is used for data analysis by knowledge workers, including managers, executives, and analysts.

Data contents: An OLTP system manages current data that, typically, are too detailed to be easily used for decision making. An OLAP system manages large amounts of historical data, provides facilities for summarization and aggregation, and stores and manages information at different levels of granularity. These features make the data easier to use in informed decision making.

Database design: An OLTP system usually adopts an entity-relationship (ER) data model and an application-oriented database design. An OLAP system typically adopts either a star or snowflake model (to be discussed in Section 3.2.2) and a subject- oriented database design.

View: An OLTP system focuses mainly on the current data within an enterprise or department, without referring to historical data or data in different organizations. In contrast, an OLAP system often spans multiple versions of a database schema, due to the evolutionary process of an organization. OLAP systems also deal with information that originates from different organizations, integrating information from many data stores. Because of their huge volume, OLAP data are stored on multiple storage media.

Access patterns: The access patterns of an OLTP system consist mainly of short, atomic transactions. Such a system requires concurrency control and recovery mechanisms. However, accesses to OLAP systems are mostly read-only operations (because most
of corporate assets. data warehouses store historical rather than up-to-date information), although many could be complex queries.

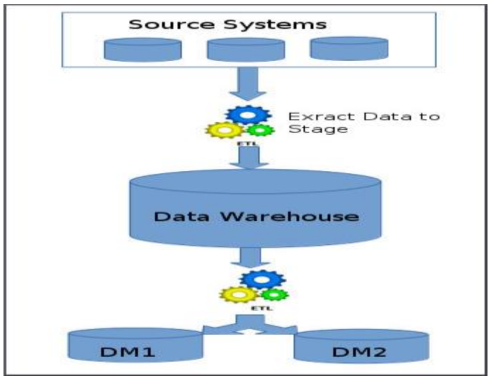
Data Warehouse Design Process:

A data warehouse can be built using a top-down approach, a bottom-up approach, or a combination of both. These methodologies are a result of research from Bill Inmon and Ralph Kimball.

Bill Inmon – Top-down Data Warehouse Design Approach

Below are the steps that are involved in Top down approach

In the top-down approach, the data warehouse is designed first and then data mart are built on top of data warehouse.

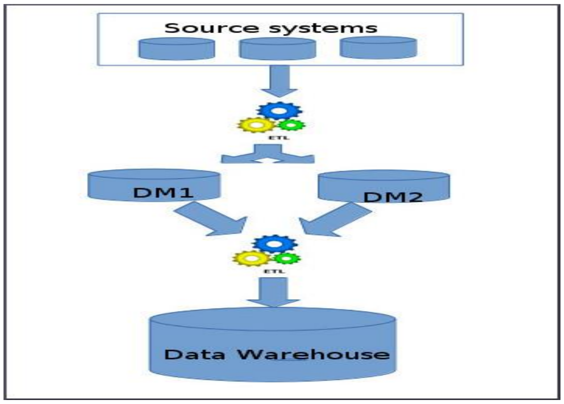


- Data is extracted from the various source systems. The extracts are loaded and validated in the stage area. Validation is required to make sure the extracted data is accurate and correct. You can use the ETL tools or approach to extract and push to the data warehouse.

- Data is extracted from the data warehouse in regular basis in stage area. At this step, you will apply various aggregation, summarization techniques on extracted data and loaded back to the data warehouse.
- Once the aggregation and summarization is completed, various data marts extract that data and apply the some more transformation to make the data structure as defined by the data marts.

Ralph Kimball – Bottom-up Data Warehouse Design Approach

Ralph Kimball is a renowned author on the subject of data warehousing. His data warehouse design approach is called dimensional modelling or the Kimball methodology. This methodology follows the bottom-up approach In this method, data marts are first created to provide the reporting and analytics capability for specific business process, later with these data marts enterprise data warehouse is created.



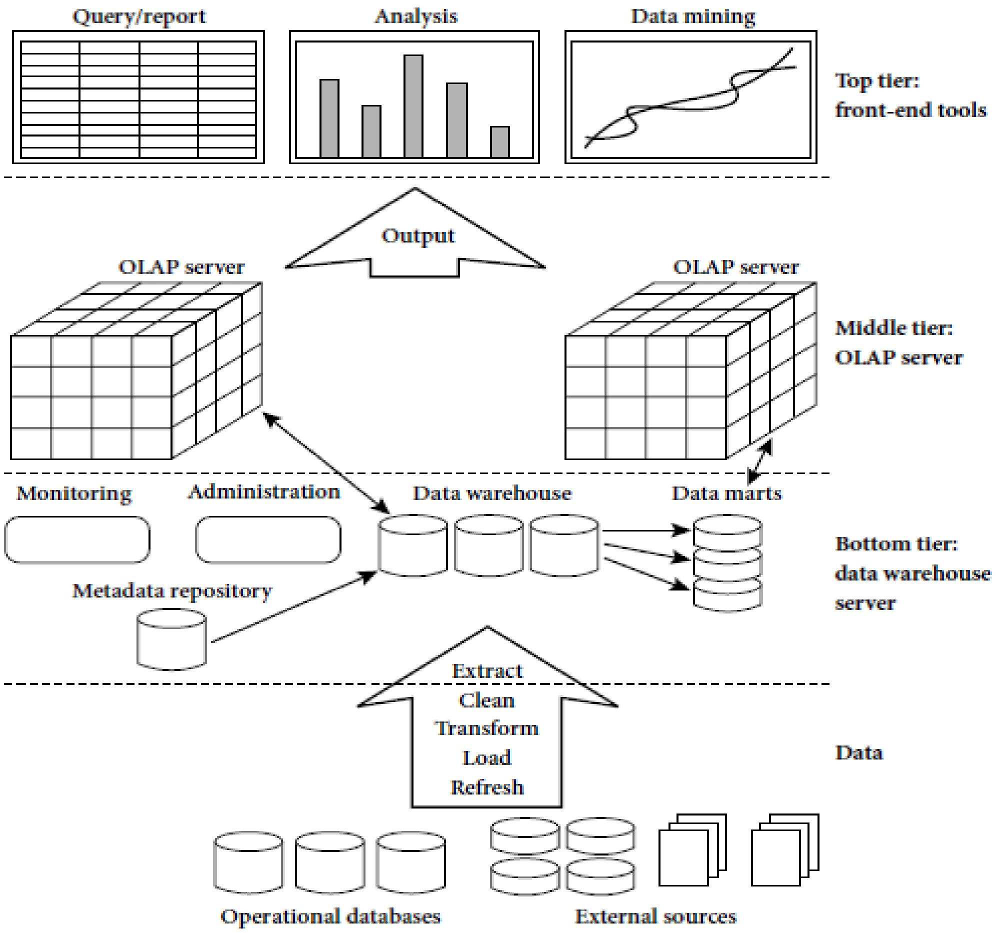
Basically, Kimball model reverses the Inmon model i.e. Data marts are directly loaded with the data from the source systems and then ETL process is used to load in to Data Warehouse.

Below are the steps that are involved in bottom-up approach:

- The data flow in the bottom up approach starts from extraction of data from various source system into the stage area where it is processed and loaded into the data marts that are handling specific business process.
- After data marts are refreshed the current data is once again extracted in stage area and transformations are applied to create data into the data mart structure. The data is the extracted from Data Mart to the staging area is aggregated, summarized and so on loaded into EDW and then made available for the end user for analysis and enables critical business decisions.

In the combined approach, an organization can exploit the planned and strategic nature of the top-down approach while retaining the rapid implementation and opportunistic application of the bottom-up approach.

A Three Tier Data Warehouse Architecture:



Tier-1:

The bottom tier is a warehouse database server that is almost always a relational database system. Back-end tools and utilities are used to feed data into the bottom tier from operational databases or other external sources (such as customer profile information provided by external consultants). These tools and utilities perform data extraction, cleaning, and transformation (e.g., to merge similar data from different sources into a unified format), as well as load and refresh functions to update the data warehouse . The data are extracted using application program interfaces known as gateways. A gateway is supported by the underlying DBMS and allows client programs to generate SQL code to be executed at a server. Examples of gateways include ODBC (Open Database Connection) and OLEDB (Open Linking and Embedding for Databases) by Microsoft and JDBC (Java Database Connection).

This tier also contains a metadata repository, which stores information about the data warehouse and its contents.

Tier-2:

The middle tier is an OLAP server that is typically implemented using either a relational OLAP (ROLAP) model or a multidimensional OLAP.

- OLAP model is an extended relational DBMS that maps operations on multidimensional data to standard relational operations.
- A multidimensional OLAP (MOLAP) model, that is, a special-purpose server that directly implements multidimensional data and operations.

Tier-3:

The top tier is a front-end client layer, which contains query and reporting tools, analysis tools, and/or data mining tools (e.g., trend analysis, prediction, and so on).

Data Warehouse Back-End Tools and Utilities Data warehouse systems use back-end tools and utilities to populate and refresh their data (Figure above) These tools and utilities include the following functions:

Data extraction, which typically gathers data from multiple, heterogeneous, and external sources

Data cleaning, which detects errors in the data and rectifies them when possible

Data transformation, which converts data from legacy or host format to warehouse format **Load**, which sorts, summarizes, consolidates, computes views, checks integrity, and builds indices and partitions

Refresh, which propagates the updates from the data sources to the warehouse

Besides cleaning, loading, refreshing, and metadata definition tools, data warehouse systems usually provide a good set of data warehouse management tools. Data cleaning and data transformation are important steps in improving the quality of the data and, subsequently, of the data mining results.

Data Warehouse Models:

There are three data warehouse models.

1. Enterprise warehouse:

- An enterprise warehouse collects all of the information about subjects spanning the entire organization.
- It provides corporate-wide data integration, usually from one or more operational systems or external information providers, and is cross-functional in scope.
- It typically contains detailed data as well as summarized data, and can range in size from a few gigabytes to hundreds of gigabytes, terabytes, or beyond.
- An enterprise data warehouse may be implemented on traditional mainframes, computer super servers, or parallel architecture platforms. It requires extensive business modeling and may take years to design and build.

2. Data mart:

- A data mart contains a subset of corporate-wide data that is of value to a specific group of users. The scope is confined to specific selected subjects. For example ,a marketing data mart may confine its subjects to customer, item, and sales. The data contained in data marts tend to be summarized.
- Data marts are usually implemented on low-cost departmental servers that are UNIX/LINUX- or Windows-based. The implementation cycle of a data mart is more likely to be measured in weeks rather than months or years. However, it may involve complex integration in the long run if its design and planning were not enterprise-wide.
- Depending on the source of data, data marts can be categorized as independent or dependent. Independent data marts are sourced from data captured from one or more operational systems or external information providers, or from data generated locally within a particular department or geographic area. Dependent data marts are sourced directly from enterprise data warehouses.

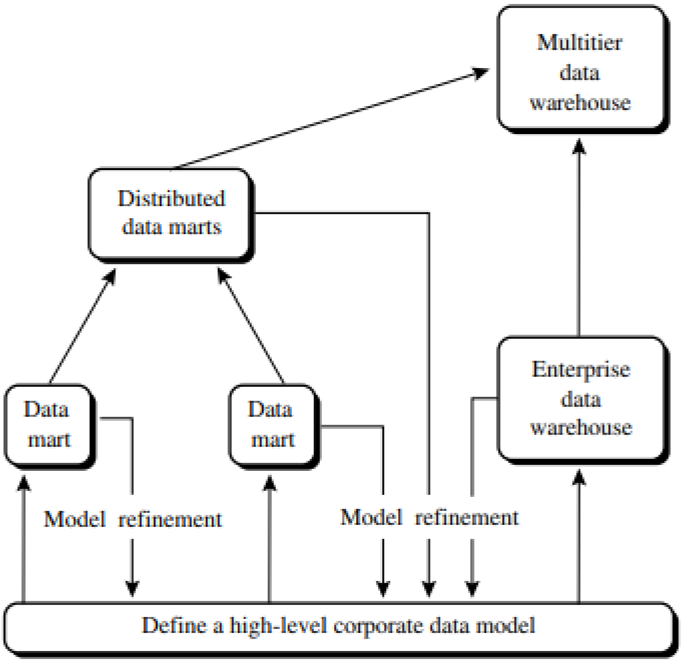
3. Virtual warehouse:

- A virtual warehouse is a set of views over operational databases. For efficient query processing, only some of the possible summary views may be materialized.

- A virtual warehouse is easy to build but requires excess capacity on operational database servers.

What are the pros and cons of the top-down and bottom-up approaches to data warehouse development?

The top-down development of an enterprise warehouse serves as a systematic solution and minimizes integration problems. However, it is expensive, takes a long time to develop, and lacks flexibility due to the difficulty in achieving consistency and consensus for a common data model for the entire organization. The bottom-up approach to the design, development, and deployment of independent data marts provides flexibility, low cost, and rapid return of investment. It, however, can lead to problems when integrating various disparate data marts into a consistent enterprise data warehouse. A recommended method for the development of data warehouse systems is to implement the warehouse in an incremental and evolutionary manner, as shown in Figure.



First, a high-level corporate data model is defined within a reasonably short period (such as one or two months) that provides a corporate-wide, consistent, integrated view of data among different subjects and potential usages. This high-level model, although it will need to be refined in the further development of enterprise data warehouses and departmental data marts, will greatly reduce future integration problems. Second, independent data marts can be implemented in parallel with the enterprise warehouse based on the same corporate data model set as above. Third, distributed data marts can be constructed to integrate different data marts via hub servers. Finally, a multitier data warehouse is constructed where the enterprise warehouse is the sole custodian of all warehouse data, which is then distributed to the various dependent data marts.

Meta Data Repository:

Metadata are data about data. When used in a data warehouse, metadata are the data that define warehouse objects. Metadata are created for the data names and definitions of the given warehouse. Additional metadata are created and captured for timestamping any extracted data, the source of the extracted data, and missing fields that have been added by data cleaning or integration processes.

A metadata repository should contain the following:

- A description of the structure of the data warehouse, which includes the warehouse schema, view, dimensions, hierarchies, and derived data definitions, as well as data mart locations and contents.
- **Operational metadata**, which include data lineage (history of migrated data and the sequence of transformations applied to it), currency of data (active, archived, or purged), and monitoring information (warehouse usage statistics, error reports, and audit trails).
- The algorithms used for summarization, which include measure and dimension definition algorithms, data on granularity, partitions, subject areas, aggregation, summarization, and predefined queries and reports.
- The mapping from the operational environment to the data warehouse, which includes source databases and their contents, gateway descriptions, data partitions, data extraction, cleaning, transformation rules and defaults, data refresh and purging rules, and security

(user authorization and access control).

- Data related to system performance, which include indices and profiles that improve data access and retrieval performance, in addition to rules for the timing and scheduling of refresh, update, and replication cycles.
- **Business metadata**, which include business terms and definitions, data ownership information, and charging policies.

A data warehouse contains different levels of summarization, of which metadata is one type. Other types include current detailed data (which are almost always on disk), older detailed data (which are usually on tertiary storage), lightly summarized data and highly summarized data (which may or may not be physically housed).

Role of Metadata

Metadata play a very different role than other data warehouse data and are important for many reasons. For example, metadata are used as a directory to help the decision support system analyst locate the contents of the data warehouse, as a guide to the mapping of data when the data are transformed from the operational environment to the data warehouse environment, and as a guide to the algorithms used for summarization between the current detailed data and the lightly summarized data, and between the lightly summarized data and the highly summarized data. Metadata should be stored and managed persistently (i.e., on disk).

Dimensional Data Modeling

Is one of the data modeling techniques used in data warehouse design.The concept of Dimensional Modeling was developed by Ralph Kimball which is comprised of facts and dimension tables. Since the main goal of this modeling is to improve the data retrieval so it is optimized for SELECT OPERATION. The advantage of using this model is that we can store data in such a way that it is easier to store and retrieve the data once stored in a data warehouse. Dimensional model is the data model used by many OLAP systems.

E-R modelling versus Dimensional Modelling

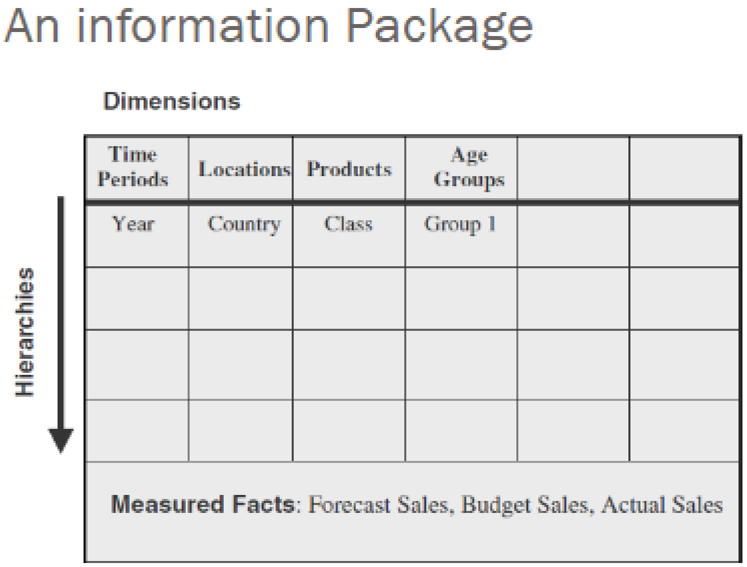
ER modelling:The main goal of ER modelling is to remove redundancy from data. To remove redundancy, designers must use hundreds of entities and relations between entities, which makes ER model complex. There is no easy way to enable end users navigate through the data in ER models. ER modelling aims to optimize performance for transaction processing. It is also hard to query ER models because of the complexity; many tables should be joined to obtain a result set. Therefore ER models are not suitable for high performance retrieval of data.

Dimensional model :The dimensional model is a standard framework. End user tools can make strong assumptions about the dimensional model. It helps, to make user interfaces more user friendly and processing more efficient [20]. Dimensional model is more useful to random changes in user behaviour and requirements. The logical design can be made independent of expected query patterns. All dimensions can be thought as symmetrically equal entry points into the fact table. Dimensional model is extensible to new design decisions and data elements. All existing fact and dimension tables can be changed in place without having to reload data. End user query and reporting tools are not affected by the change.

Dimensional model involves business rules but ER modelling does not involve business rules, it involves data rules.

Information Packages Diagram:

- Novel idea for determining and recording information requirements for a data warehouse.
- Determining requirements for a data warehouse is based on business dimensions The relevant dimension and measurements in that dimension are captured and kept in a data warehouse
- This creates an information package for a specific subject



An information Package

Dimensions

Time Periods	Locations	Products	Age Groups		
Year	Country	Class	Group 1		

Hierarchies

Measured Facts: Forecast Sales, Budget Sales, Actual Sales

Multidimensional Data Model

Data warehouses and OLAP tools are based on a multidimensional data model. This model views data in the form of a data cube. A data cube allows data to be modeled and viewed in multiple dimensions. It is defined by dimensions and facts. dimensions are the perspectives or entities with respect to which an organization wants to keep records. For example, All Electronics may create a sales data warehouse in order to keep records of the store's sales with respect to the dimensions time, item, branch, and location. These dimensions allow the store to keep track of things like monthly sales of items and the branches and locations at which the items were sold. Each dimension may have a table associated with it, called a **dimension table**, which further describes the dimension. For example, a dimension table for item may contain the attributes item name, brand, and type. Dimension tables can be specified by users or experts, or automatically generated and adjusted based on data distributions.

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

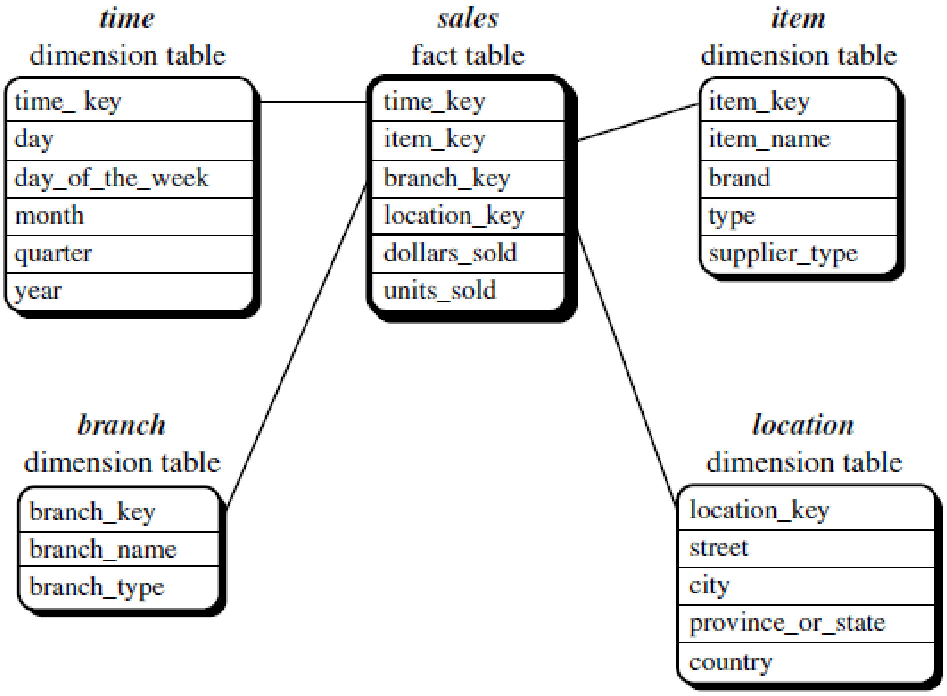
Stars, Snowflakes, and Fact Constellations: Schemas for Multidimensional Databases

The entity-relationship data model is commonly used in the design of relational databases, where a database schema consists of a set of entities and the relationships between them. Such a data model is appropriate for on-line transaction processing. A data warehouse, however, requires a concise, subject-oriented schema that facilitates on-line data analysis. The most popular data model for a data warehouse is a multidimensional model. Such a model can exist in the form of a star schema, a snowflake schema, or a fact constellation schema.

Star schema: The most common modeling paradigm is the star schema, in which the data warehouse contains (1) a large central table (fact table) containing the bulk of the data, with no redundancy, and (2) a set of smaller attendant tables (dimension tables), one for each dimension. The schema graph resembles a starburst, with the dimension tables displayed in a radial pattern around the central fact table.

Example: Star schema. A star schema for All Electronics sales is shown in Figure . Sales are considered along four dimensions, namely, time, item, branch, and location. The schema contains a central fact table for sales that contains keys to each of the four dimensions, along with two measures: dollars sold and units sold. To minimize the size of the fact table, dimension identifiers (such as time key and item key) are system-generated identifiers.

In the star schema, each dimension is represented by only one table, and each table contains a set of attributes. For example, the location dimension table contains the attribute set location key, street, city, province or state, country. This constraint may introduce some redundancy. For example, “Vancouver” and “Victoria” are both cities in the Canadian province of British Columbia. Entries for such cities in the location dimension table will create redundancy among the attributes province or state and country, that is, (..., Vancouver, British Columbia, Canada) and (..., Victoria, British Columbia, Canada).Moreover, the attributes within a dimension table may form either a hierarchy (total order) or a lattice (partial order).

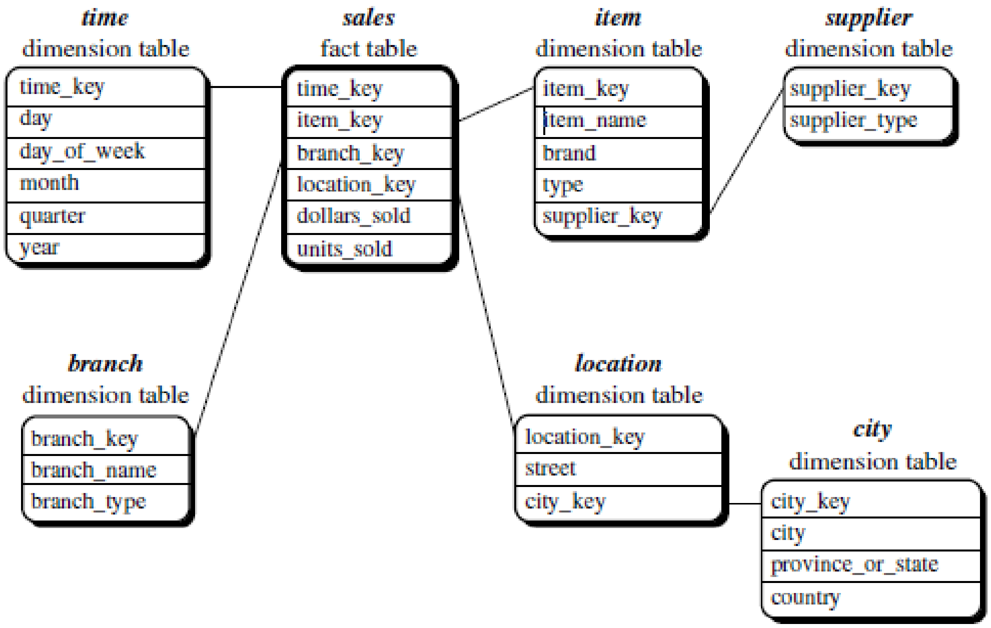


Star schema of a data warehouse for sales.

Snowflake schema: The snowflake schema is a variant of the star schema model, where some dimension tables are normalized, thereby further splitting the data into additional tables. The resulting schema graph forms a shape similar to a snowflake.

The major difference between the snowflake and star schema models is that the dimension tables of the snowflake model may be kept in normalized form to reduce redundancies. Such a table is easy to maintain and saves storage space. However, this saving of space is negligible in comparison to the typical magnitude of the fact table. Furthermore, the snowflake structure can reduce the effectiveness of browsing, since more joins will be needed to execute a query. Consequently, the system performance may be adversely impacted. Hence, although the snowflake schema reduces redundancy, it is not as popular as the star schema in data warehouse design. Snowflake schema.

A snowflake schema for All Electronics sales is given in Figure Here, the sales fact table is identical to that of the star schema .The main difference between the two schemas is in the definition of dimension tables.The single dimension table for item in the star schema is normalized in the

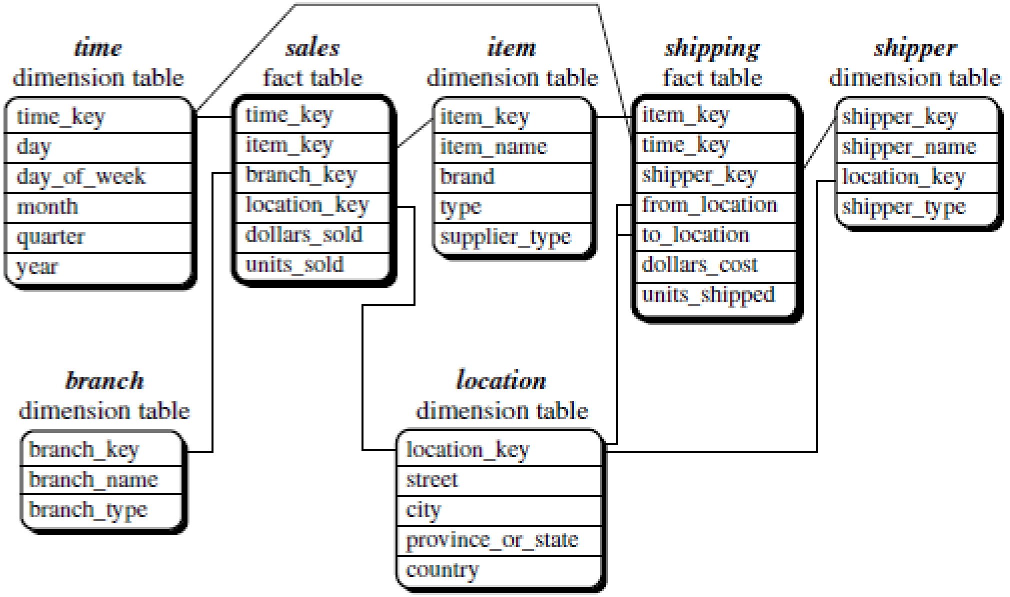


Snowflake schema of a data warehouse for sales.

snowflake schema, resulting in new item and supplier tables. For example, the item dimension table now contains the attributes item key, item name, brand, type, and supplier key, where supplier key is linked to the supplier dimension table, containing supplier key and supplier type information. Similarly, the single dimension table for location in the star schema can be normalized into two new tables: location and city. The city key in the new location table links to the city dimension. Further normalization can be performed on province or state and country in the snowflake schema shown in Figure, when desirable.

Fact constellation: Sophisticated applications may require multiple fact tables to share dimension tables. This kind of schema can be viewed as a collection of stars, and hence is called a galaxy schema or a fact constellation.

Example Fact constellation. A fact constellation schema is shown in Figure . This schema specifies two fact tables, sales and shipping. The sales table definition is identical to that of the star schema . The shipping table has five dimensions, or keys: item key,time key, shipper key, from location, and to location, and two measures: dollars cost and units shipped. A fact constellation schema allows dimension tables to be shared between fact tables. For example, the dimensions tables for time, item, and location are shared between both the sales and shipping fact tables.



Fact constellation schema of a data warehouse for sales and shipping.

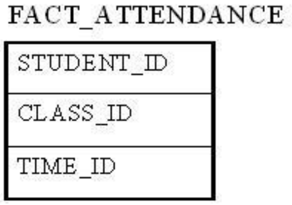
Data warehouses versus Data Marts

In data warehousing, there is a distinction between a data warehouse and a data mart. A data warehouse collects information about subjects that span the entire organization, such as customers, items, sales, assets, and personnel, and thus its scope is enterprise wide. For data warehouses, the fact constellation schema is commonly used, since it can model multiple, interrelated subjects. A data mart, on the other hand, is a department subset of the data warehouse that focuses on selected subjects, and thus its scope is departmentwide. For data marts, the star or snowflake schema are commonly used, since both are geared toward modeling single subjects, although the star schema is more popular and efficient.

Factless Fact Table

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.

Factless fact tables offer the most flexibility in data warehouse design. For example, one can easily answer the following questions with this factless 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 factless fact table, we will need two separate fact tables to answer the above two questions. With the above factless fact table, it becomes the only fact table that's needed.

Updates in Dimension Table

Compared to the fact **table**, the **dimension tables** are more stable and less volatile. However, unlike the fact **table**, which changes through the increase in the number of rows, a **dimension table** does not change just through the increase in the number of rows, but also through changes to the attributes themselves. Consider product dimension table. Every year, rows are added as new models become available. But what about the attributes within the product dimension table? If a particular product is moved to a different product category, then the corresponding values must be changed in the product dimension table. Let us examine the types of changes that affect dimension tables and discuss the ways for dealing with these types.

Slowly Changing Dimensions

In the above example, we have mentioned a change to the product dimension table because the product category for a product was changed. Consider the customer demographics dimension table. What happens when a customer's status changes from rental home to own home? The corresponding row in that dimension table must be changed. Next, look at the payment method dimension table. When finance type changes for one of the payment methods, this change must be reflected in the payment method dimension table. From the consideration of the changes to the dimension tables, we can derive the following principles:

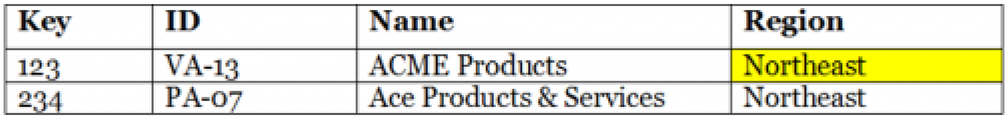
- Most dimensions are generally constant over time
- Many dimensions, though not constant over time, change slowly
- The product key of the source record does not change
- The description and other attributes change slowly over time
- In the source OLTP systems, the new values overwrite the old ones
- Overwriting of dimension table attributes is not always the appropriate option in a data warehouse
- The ways changes are made to the dimension tables depend on the types of changes and what information must be preserved in the data warehouse

  The usual changes to dimension tables may be classified into three distinct types. We will discuss these three types in detail. You will understand why you must use different

Type 1 – For this type of slowly changing dimension you **simply overwrite the existing data** values with new data values. This makes the updating of the dimension easy and limits the growth of the dimension table to only new records. The drawback of this is you lose the historical value of the data because the dimension will always contain the current values for each attribute. For instance, you have a store dimension which has an attribute for a geographic region. If there is a redesign in the regional boundaries some stores may move from one region to another. Because the record is simply updated a store which may have been reporting results in the Northeast district will now report their results to the Mid-Atlantic region. But, as a result of the update, now all of the history for that store before the move is essentially removed from the Northeast and moved to the Mid-Atlantic district.

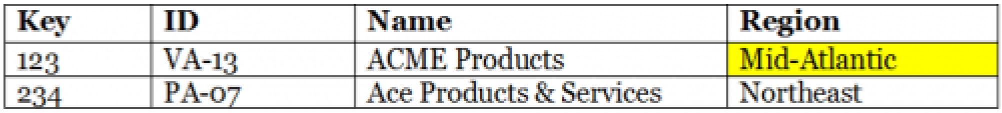
This change will skew the historical reports and the reports run before the update will no longer match the reports run after the update for the same timeframe.

Original Record



Key	ID	Name	Region
123	VA-13	ACME Products	Northeast
234	PA-07	Ace Products & Services	Northeast

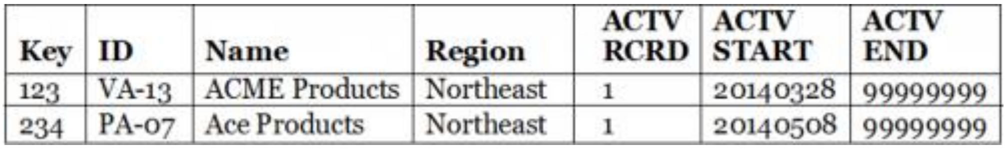
Updated Record



Key	ID	Name	Region
123	VA-13	ACME Products	Mid-Atlantic
234	PA-07	Ace Products & Services	Northeast

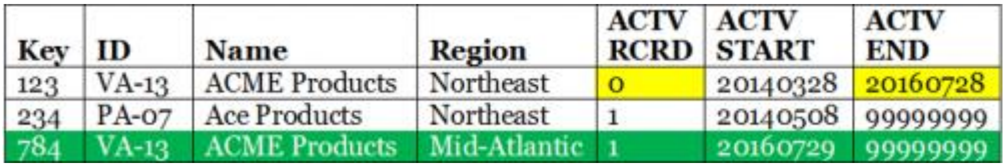
<u>**Type 2**</u> – This is the most commonly used type of slowly changing dimension. For this type of slowly changing dimension, add a new record encompassing the change and mark the old record as inactive. This allows the fact table to continue to use the old version of the data for historical reporting purposes leaving the changed data in the new record to only impact the fact data from that point forward. Several columns should be added to the dimension table (active record start/end dates and a current active record flag) to provide historical change management and ensure optimal use of the active record. Using the same example from the Type 1 dimension above, the change in the district will cause the updating of the current active dimension record's active record end data and active record flag denoting this record is no longer actively in use. This will also spawn the creation of a new active record with a new dimension key. This new dimension key will be used in the generation of the fact table moving forward. This allows the fact table to still use the data stored under the old dimension key for historical reporting. This will ensure that the data remains the same and a historical report for the same timeframe run before the update was made will continue to display the exact same data as before the change was made.

Original Record



				ACTV	ACTV	ACTV
Key	ID	Name	Region	RCRD	START	END
123	VA-13	ACME Products	Northeast	1	20140328	99999999
234	PA-07	Ace Products	Northeast	1	20140508	99999999

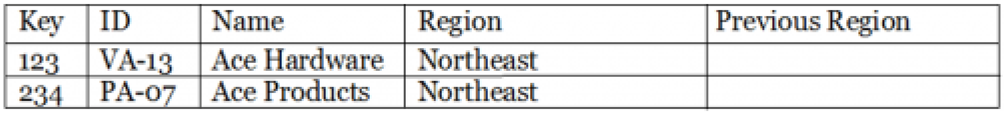
Inserted / Updated Records



Key	ID	Name	Region	ACTV RCRD	ACTV START	ACTV END
123	VA-13	ACME Products	Northeast	0	20140328	20160728
234	PA-07	Ace Products	Northeast	1	20140508	99999999
784	VA-13	ACME Products	Mid-Atlantic	1	20160729	99999999

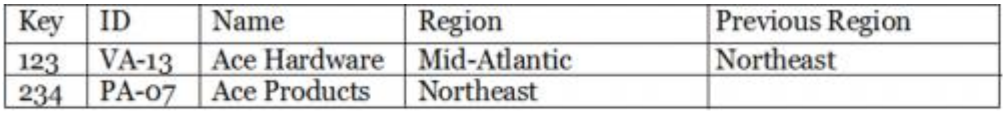
Type 3 – This is a seldom used type of slowly changing dimension. In this type of slowly changing dimension you add a second column to store the most recent past value of the column(s) you wish to be able to report on. When the data is updated the existing value is "moved" to the column defined to store the previous past value and the new value is placed into the reportable column. This allows you the ability to look back at what the value of the data was previously. This can be a challenge when loading/updating the data. The amount of work to design and maintain this solution far exceed the benefit the "fallback snapshot" provides.

Original Record



Key	ID	Name	Region	Previous Region
123	VA-13	Ace Hardware	Northeast	
234	PA-07	Ace Products	Northeast	

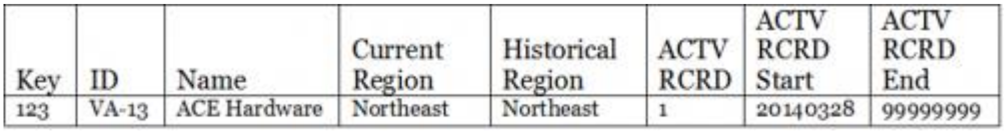
Updated Record



Key	ID	Name	Region	Previous Region
123	VA-13	Ace Hardware	Mid-Atlantic	Northeast
234	PA-07	Ace Products	Northeast	

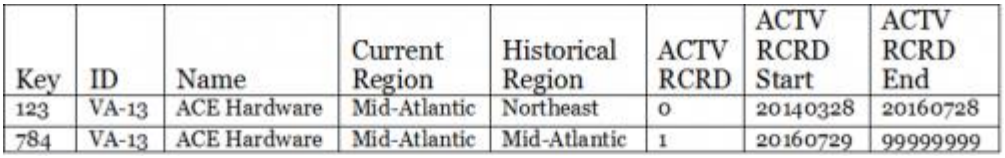
Type 6 – A Type 6 SCD is a very rarely used SCD. In this instance, you combine SCD Type 1, SCD Type 2 and SCD Type 3 (1 + 2 + 3 = 6). To create a Type 6 SCD you would start with a Type 2, add columns for the records you wish to capture the current value as well as the historical value. This allows one to filter or group on the Type 2 value in effect when the measure occurred or the current attribute value.

Original Record

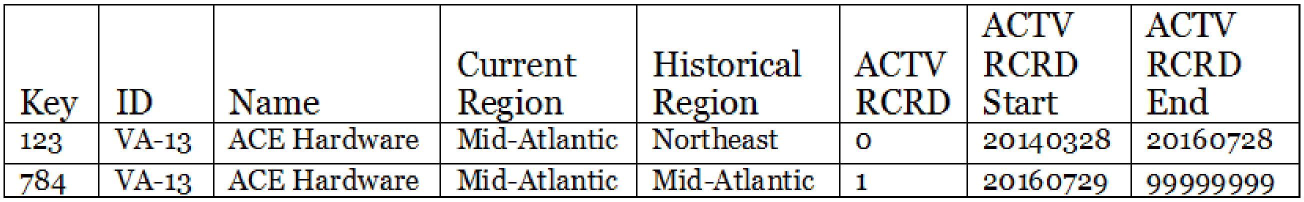


Key	ID	Name	Current Region	Historical Region	ACTV RCRD	ACTV RCRD Start	ACTV RCRD End
123	VA-13	ACE Hardware	Northeast	Northeast	1	20140328	99999999

1st Update

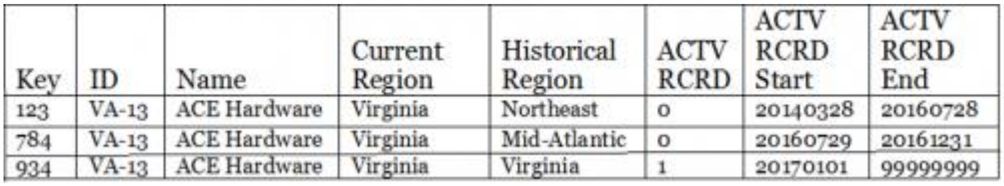


Key	ID	Name	Current Region	Historical Region	ACTV RCRD	ACTV RCRD Start	ACTV RCRD End
123	VA-13	ACE Hardware	Mid-Atlantic	Northeast	0	20140328	20160728
784	VA-13	ACE Hardware	Mid-Atlantic	Mid-Atlantic	1	20160729	99999999

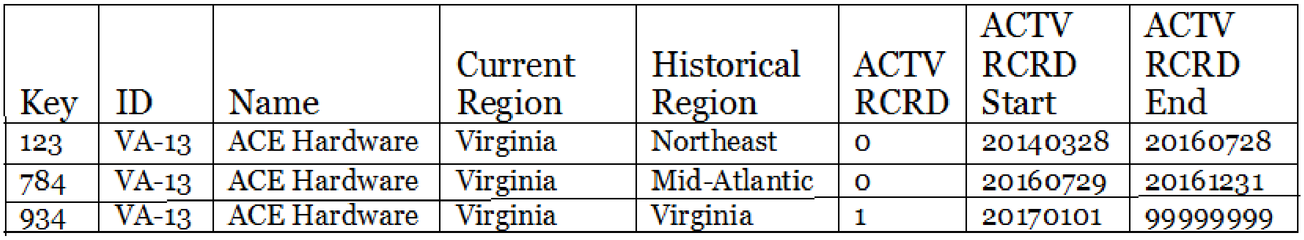


Key	ID	Name	Current Region	Historical Region	ACTV RCRD	ACTV RCRD Start	ACTV RCRD End
123	VA-13	ACE Hardware	Mid-Atlantic	Northeast	0	20140328	20160728
784	VA-13	ACE Hardware	Mid-Atlantic	Mid-Atlantic	1	20160729	99999999

2nd Update



Key	ID	Name	Current Region	Historical Region	ACTV RCRD	ACTV RCRD Start	ACTV RCRD End
123	VA-13	ACE Hardware	Virginia	Northeast	0	20140328	20160728
784	VA-13	ACE Hardware	Virginia	Mid-Atlantic	0	20160729	20161231
934	VA-13	ACE Hardware	Virginia	Virginia	1	20170101	99999999



Key	ID	Name	Current Region	Historical Region	ACTV RCRD	ACTV RCRD Start	ACTV RCRD End
123	VA-13	ACE Hardware	Virginia	Northeast	0	20140328	20160728
784	VA-13	ACE Hardware	Virginia	Mid-Atlantic	0	20160729	20161231
934	VA-13	ACE Hardware	Virginia	Virginia	1	20170101	99999999

Aggregates Fact Tables

Aggregate table contains the summary of existing data warhouse , it contains data which is group accordingly (month , quartely etc) depending upon the business needs.For example you may contain transaction in fact table for all the customers but sometime you want the sum of amount for a particular customer over a month in that case you need to run the query in fact table which contains millions of row and time consuming. so avoid this problem we do the aggregation of amount in aggregate over a month for each customers, so this will improve the performance and you can retrieve the result very fastly.

University Solutions on Star and Snowflake schema

Q1

i. Design star & snowflake schema for "Hotel Occupancy" considering

dimensions like Time, Hotel, Room, etc.

ii. Calculate the maximum number of base fact table records for the values given

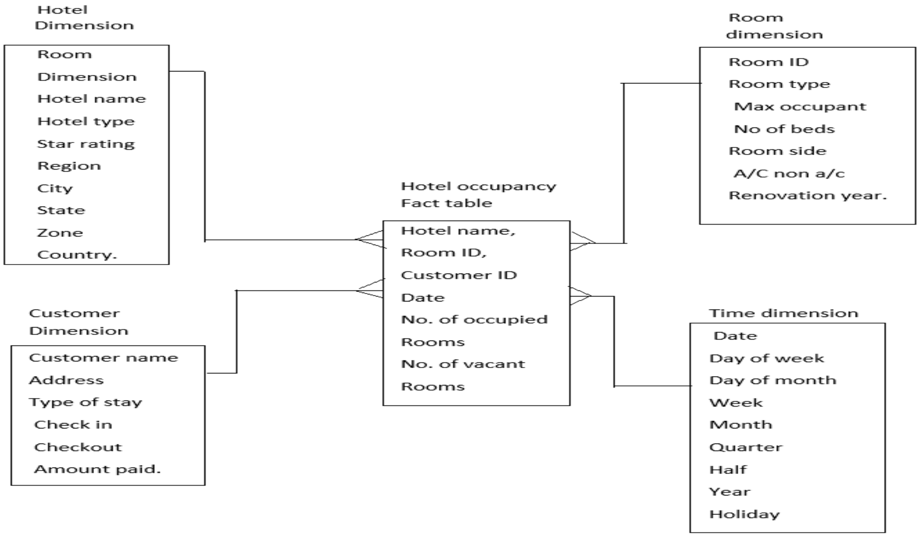
below:

Time period: 5 years

Hotels: 150

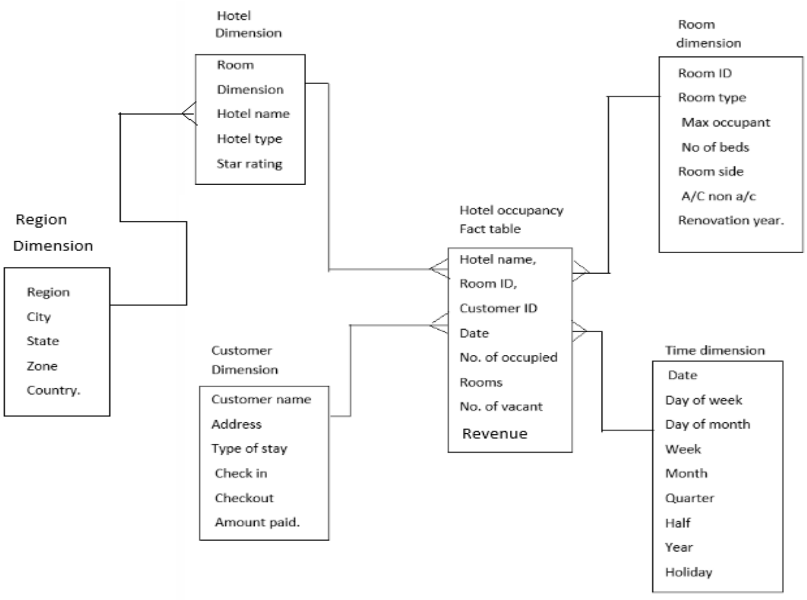
Rooms: 750 rooms in each Hotel (about 400 occupied in each hotel daily)

Star scheme = For hotel occupancy.
Facts : No. of occupied rooms
No. of vacant rooms revenue
Dimensions : Time, Hotel, Room, Customer,
Dimension hierarchy :
Time : Data, day of week, day of month, week, month, quarter, half, year, holiday.
Hotel : Hotel name, hotel type, start rating, regime, city, state, zone, country.
Room : Room ID, room type, max occupant, No of beds, Room side, A/C non a/c, renovation year.
Customer : Customer ID, customer name, address, type of stay, check in, checkout, amount paid.
Star scheme contains fact table and dimension tables. All the facts are recorded in the fact table. All the hierarchies are grouped in dimension tables.



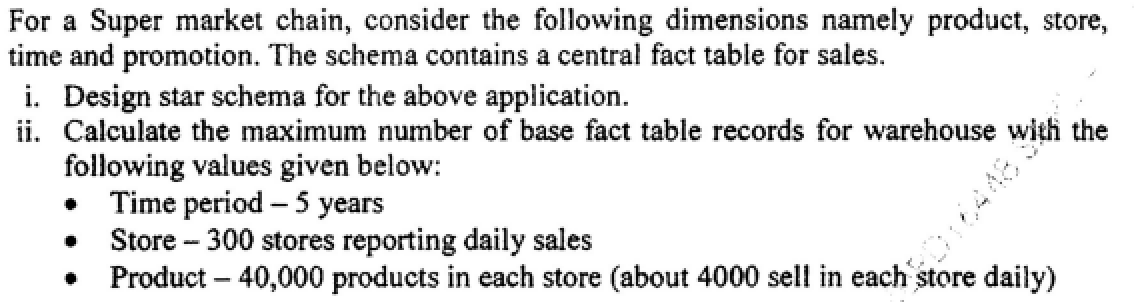
Star Schema

Snowflake schema has one or more normalized dimensions.
The hotel dimension in the above star schema can be normalized.



Snow flake schema

Q2)

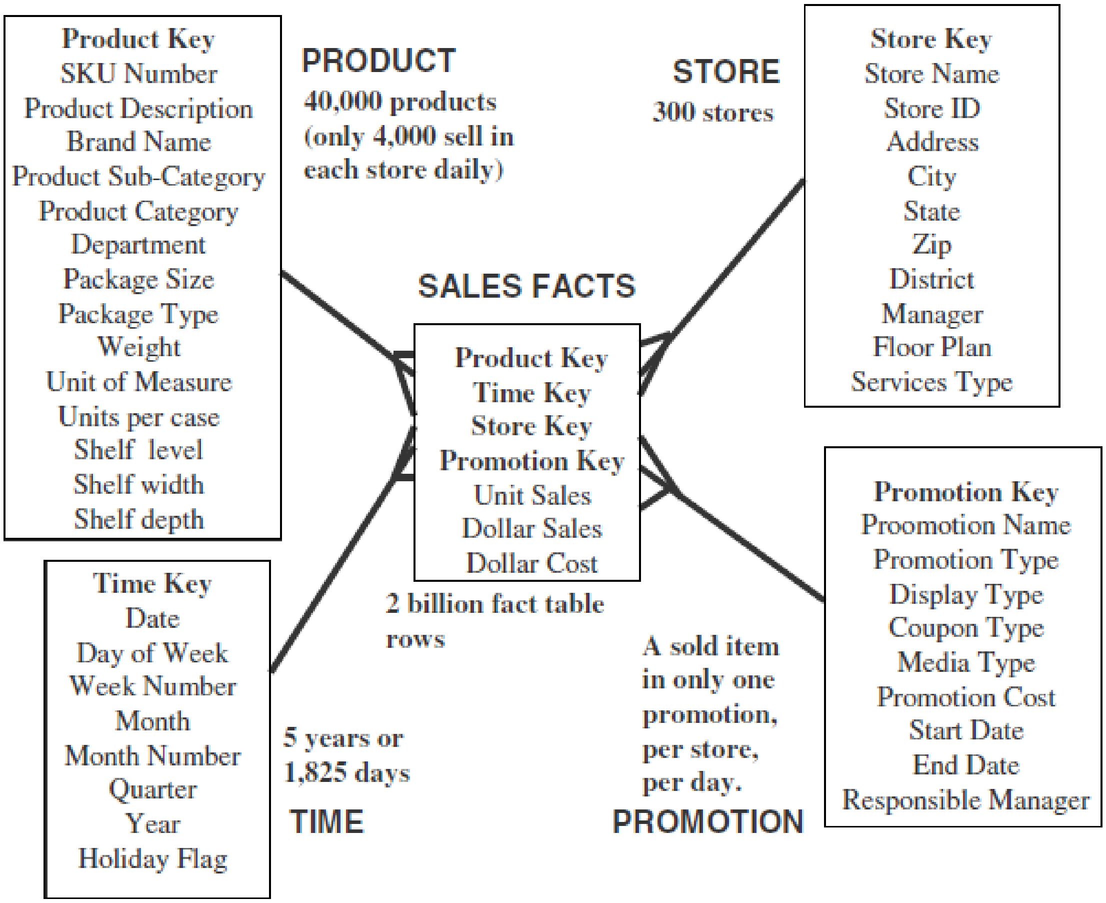


For a Super market chain, consider the following dimensions namely product, store, time and promotion. The schema contains a central fact table for sales.

i. Design star schema for the above application.
ii. Calculate the maximum number of base fact table records for warehouse with the following values given below:
   - Time period – 5 years
   - Store – 300 stores reporting daily sales
   - Product – 40,000 products in each store (about 4000 sell in each store daily)

Answer:

ii) Time dimension: 5 years × 365 days = 1825
Store dimension: 300 stores reporting daily sales
Product dimension: 40,000 products in each store (about 4000 sell in each store daily)
Promotion dimension: a sold item may be in only one promotion in a store on a given day
Maximum number of base fact table records: 1825 × 300 × 4000 × 1 = 2 billion



STAR schema: grocery chain.

Here are a few more estimates of the fact table sizes in other typical cases:

Telephone Call Monitoring

Time dimension: 5 years = 1825 days

Number of calls tracked each day: 150 million

Maximum number of base fact table records: 274 billion

Credit Card Transaction Tracking

Time dimension: 5 years = 60 months

Number of credit card accounts: 150 million

Average number of monthly transactions per account: 20

Maximum number of base fact table records: 180 billion

Q) The data warehouse has to allow analyzing the company's situation at least with respect to the Furniture , Customer and Time. More ever, the company needs to analyses: The furniture with respect to its type category and material. The customer with respect to their spatial location, by considering at least cities, regions and states. The company is interested in learning the quantity, income and discount of its sales

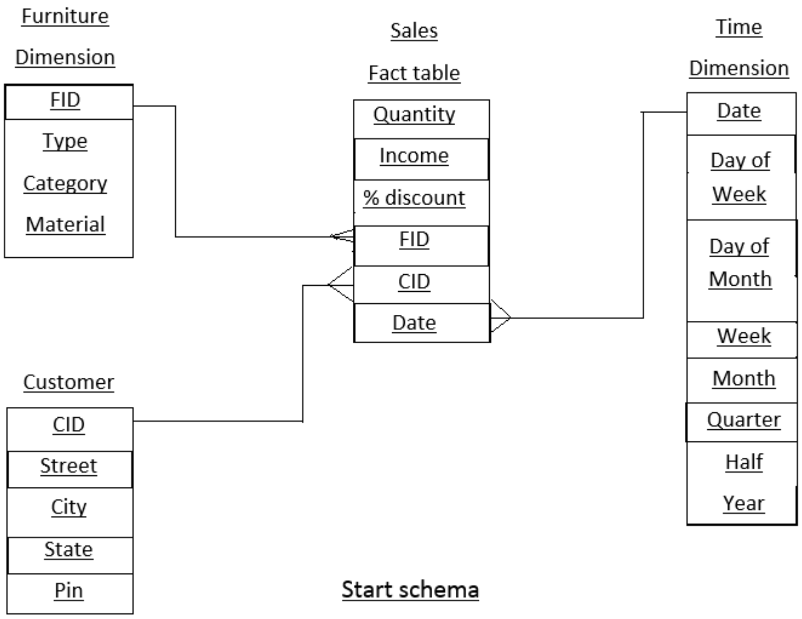
Dimensional model is used to analyze data/business facts with respect to business dimensions example customers, products, etc.

DW for furniture company.

Dimensions:

1. Furniture : type, category, material.
2. Customer : Name, street, city, state, min.
3. Time **:** date, day of week, day of month, week, month, quarter, half year.

Facts : Quantity income discount



Start schema

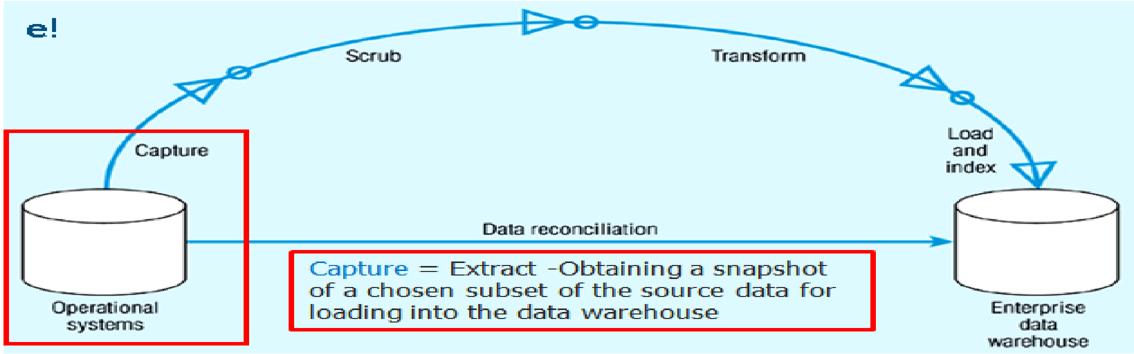
Chapter 2 : ETL Process

What are the steps involved in ETL process? There are mainly 4 steps in the Informatica ETL process, let us now understand them in depth:

1. Extract or Capture
2. Scrub or Clean
3. Transform
4. Load and Index

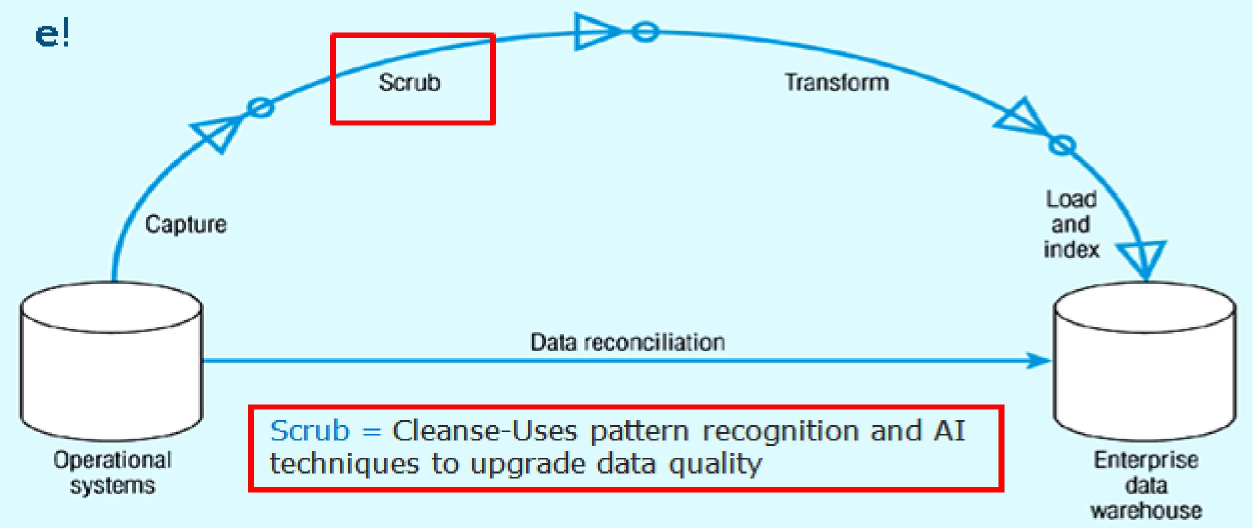
1. Extract or Capture: As seen in the image below, the Capture or Extract is the first step of Informatica ETL process. It is the process of obtaining a snapshot of the chosen subset of data from the source, which has to be loaded into the data warehouse. A snapshot is a read-only static view of the data in the database. The Extract process can be of two types:

- Full extract: The data is extracted completely from the source system and there's no need to keep track of changes to the data source since the last successful extraction.
- Incremental extract: This will only capture changes that have occurred since the last full extract.


e!
Scrub
Transform
Capture
Load and index
Data reconciliation
Capture = Extract -Obtaining a snapshot of a chosen subset of the source data for loading into the data warehouse
Operational systems
Enterprise data warehouse

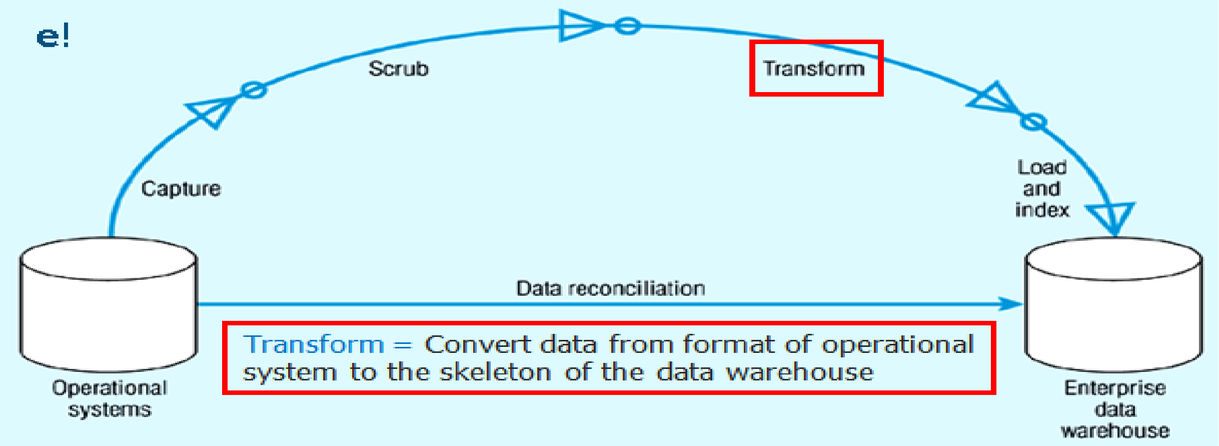
Phase 1: Extract or Capture

2.Scrub or Clean: This Scrub or Clean: This is the process of cleaning the data coming from the source by using various pattern recognition and AI techniques to upgrade the quality of data taken forward. Usually, the errors like misspellings, erroneous dates, incorrect field usage, mismatched addresses, missing data, duplicate data, inconsistencies are highlighted and then corrected or removed in this step. Also, operations like decoding, reformatting, time stamping, conversion, key generation, merging, error detection/logging, locating missing data are done in this step. As seen in the image below, this is the second step of Informatica ETL process.


e!
Scrub
Transform
Capture
Load and index
Data reconciliation
Scrub = Cleanse-Uses pattern recognition and AI techniques to upgrade data quality
Operational systems
Enterprise data warehouse

Phase 2: Scrubbing or Cleaning of data

3. Transform: As seen in the image below, this is the third and most essential step of Informatica ETL process. Transformations is the operation of converting data from the format of the source system to the skeleton of Data Warehouse. A Transformation is basically used to represent a set of rules, which define the data flow and how the data is loaded into the targets.

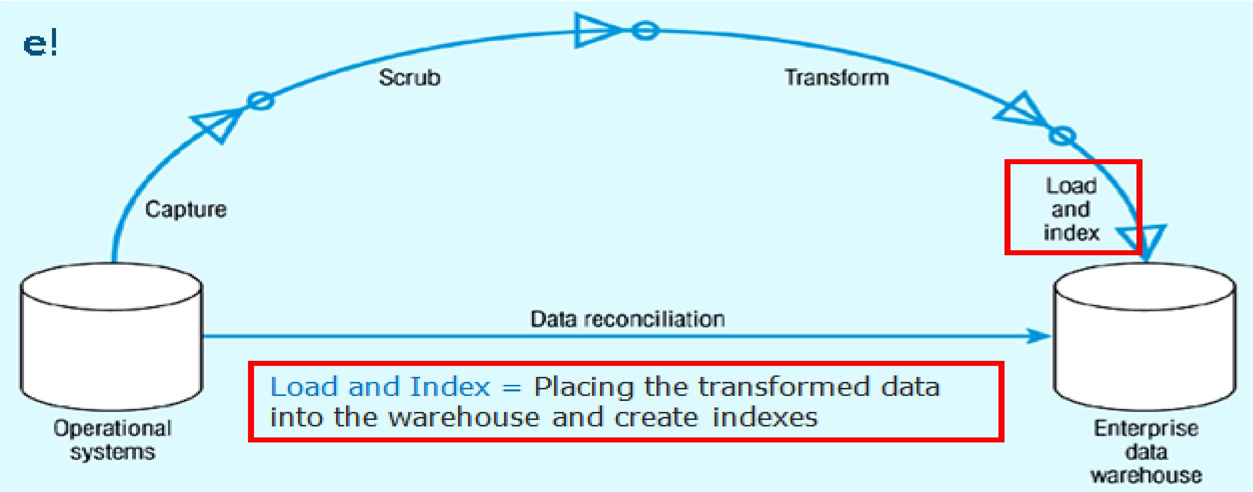

e!
Scrub
Transform
Capture
Load and index
Data reconciliation
Transform = Convert data from format of operational system to the skeleton of the data warehouse
Operational systems
Enterprise data warehouse

Phase 3: Transformation

4. Load and Index: This is the final step of Informatica ETL process as seen in the image below. In this stage, we place the transformed data into the warehouse and

create indexes for the data. There are two major types of data load available based on the load process.:

- Full Load or Bulk Load: The data loading process when we do it at very first time. The job extracts entire volume of data from a source table and loads into the target data warehouse after applying the required transformations. It will be a one time job run after then changes alone will be captured as part of an incremental extract.
- Incremental load or Refresh load: The modified data alone will be updated in target followed by full load. The changes will be captured by comparing created or modified date against the last run date of the job. The modified data alone extracted from the source and will be updated in the target without impacting the existing data.


e!
Scrub
Transform
Capture
Load and index
Data reconciliation
Load and Index = Placing the transformed data into the warehouse and create indexes
Operational systems
Enterprise data warehouse

Phase 4: Load and Index

OLAP(Online analytical Processing):

- OLAP is an approach to answering multi-dimensional analytical (MDA) queries swiftly.
- OLAP is part of the broader category of business intelligence, which also encompasses relational database, report writing and data mining.

- OLAP tools enable users to analyze multidimensional data interactively from multiple perspectives.

OLAP consists of three basic analytical operations:

- ➢ Consolidation (Roll-Up)
- ➢ Drill-Down
- ➢ Slicing And Dicing

- Consolidation involves the aggregation of data that can be accumulated and computed in one or more dimensions. For example, all sales offices are rolled up to the sales department or sales division to anticipate sales trends.

- The drill-down is a technique that allows users to navigate through the details. For instance, users can view the sales by individual products that make up a region's sales.

- Slicing and dicing is a feature whereby users can take out (slicing) a specific set of data of the OLAP cube and view (dicing) the slices from different viewpoints.

1.10.1 Types of OLAP:

1. Relational OLAP (ROLAP):

- ROLAP works directly with relational databases. The base data and the dimension tables are stored as relational tables and new tables are created to hold the aggregated information. It depends on a specialized schema design.
- This methodology relies on manipulating the data stored in the relational database to give the appearance of traditional OLAP's slicing and dicing functionality. In essence, each action of slicing and dicing is equivalent to adding a "WHERE" clause in the SQL statement.
- ROLAP tools do not use pre-calculated data cubes but instead pose the query to the

standard relational database and its tables in order to bring back the data required to answer the question.

- ROLAP tools feature the ability to ask any question because the methodology does not limit to the contents of a cube. ROLAP also has the ability to drill down to the lowest level of detail in the database.

2. Multidimensional OLAP (MOLAP):

- MOLAP is the 'classic' form of OLAP and is sometimes referred to as just OLAP.

- MOLAP stores this data in an optimized multi-dimensional array storage, rather than in a relational database. Therefore it requires the pre-computation and storage of information in the cube - the operation known as processing.

- MOLAP tools generally utilize a pre-calculated data set referred to as a data cube. The data cube contains all the possible answers to a given range of questions.

- MOLAP tools have a very fast response time and the ability to quickly write back data into the data set.

3. Hybrid OLAP (HOLAP):

- There is no clear agreement across the industry as to what constitutes Hybrid OLAP, except that a database will divide data between relational and specialized storage.
- For example, for some vendors, a HOLAP database will use relational tables to hold the larger quantities of detailed data, and use specialized storage for at least some aspects of the smaller quantities of more-aggregate or less-detailed data.
- HOLAP addresses the shortcomings of MOLAP and ROLAP by combining the capabilities of both approaches.
- HOLAP tools can utilize both pre-calculated cubes and relational data sources.