CHENNAI INSTITUTE OF TECHNOLOGY
(Autonomous)
(Affiliated to Anna University, Approved by AICTE, Accredited by NAAC & NBA)
Sarathy Nagar, Kundrathur, Chennai – 600069, India.

UNIT IV - LECTURE NOTES

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CCS341 DATA WAREHOUSING

III CSE/VI SEMESTER

UNIT IV DIMENSIONAL MODELING AND SCHEMA

Dimensional Modeling- Multi-Dimensional Data Modeling – Data Cube- Star Schema- Snowflake schema- Star Vs Snowflake schema- Fact constellation Schema- Schema Definition - Process Architecture- Types of Data Base Parallelism – Datawarehouse Tools

1.Dimensional Modeling:

Dimensional modeling represents data with a cube operation, making more suitable logical data representation with OLAP data management. The perception of Dimensional Modeling was developed by **Ralph Kimball** and is consist of **"fact"** and **"dimension"** tables.

In dimensional modeling, the transaction record is divided into either **"facts,"** which are frequently numerical transaction data, or **"dimensions,"** which are the reference information that gives context to the facts. For example, a sale transaction can be damage into facts such as the number of products ordered and the price paid for the products, and into dimensions such as order date, user name, product number, order ship-to, and bill-to locations, and salesman responsible for receiving the order.

1.1 Objectives of Dimensional Modeling:

The purposes of dimensional modeling are:

1. To produce database architecture that is easy for end-clients to understand and write queries.
2. To maximize the efficiency of queries. It achieves these goals by minimizing the number of tables and relationships between them.

1.2 Advantages of Dimensional Modeling:

Following are the benefits of dimensional modeling are:

Dimensional modeling is simple: Dimensional modeling methods make it possible for warehouse designers to create database schemas that business customers can easily hold and comprehend. There is no need for vast training on how to read diagrams, and there is no complicated relationship between different data elements.

Dimensional modeling promotes data quality: The star schema enable warehouse administrators to enforce referential integrity checks on the data warehouse. Since the fact information key is a concatenation of the essentials of its associated dimensions, a factual record is actively loaded if the corresponding dimensions records are duly described and also exist in the database.

By enforcing foreign key constraints as a form of referential integrity check, data warehouse DBAs add a line of defense against corrupted warehouses data.

Performance optimization is possible through aggregates: As the size of the data warehouse increases, performance optimization develops into a pressing concern. Customers who have to wait for hours to get a response to a query will quickly become discouraged with the warehouses. Aggregates are one of the easiest methods by which query performance can be optimized.

1.3 Disadvantages of Dimensional Modeling

1. To maintain the integrity of fact and dimensions, loading the data warehouses with a record from various operational systems is complicated.
2. It is severe to modify the data warehouse operation if the organization adopting the dimensional technique changes the method in which it does business.

1.4 Elements of Dimensional Modeling

Fact

It is a collection of associated data items, consisting of measures and context data. It typically represents business items or business transactions.

Dimensions

It is a collection of data which describe one business dimension. Dimensions decide the contextual background for the facts, and they are the framework over which OLAP is performed.

Measure

It is a numeric attribute of a fact, representing the performance or behavior of the business relative to the dimensions.

Considering the relational context, there are two basic models which are used in dimensional modeling:

- Star Model
- Snowflake Model

The star model is the underlying structure for a dimensional model. It has one broad central table (fact table) and a set of smaller tables (dimensions) arranged in a radial design around the primary table. The snowflake model is the conclusion of decomposing one or more of the dimensions.

Fact Table

Fact tables are used to data facts or measures in the business. Facts are the numeric data elements that are of interest to the company.

Characteristics of the Fact table

The fact table includes numerical values of what we measure. For example, a fact value of 20 might means that 20 widgets have been sold.

Each fact table includes the keys to associated dimension tables. These are known as foreign keys in the fact table.

Fact tables typically include a small number of columns.

When it is compared to dimension tables, fact tables have a large number of rows.

Dimension Table

Dimension tables establish the context of the facts. Dimensional tables store fields that describe the facts.

Characteristics of the Dimension table

Dimension tables contain the details about the facts. That, as an example, enables the business analysts to understand the data and their reports better.

The dimension tables include descriptive data about the numerical values in the fact table. That is, they contain the attributes of the facts. For example, the dimension tables for a marketing analysis function might include attributes such as time, marketing region, and product type.

Since the record in a dimension table is denormalized, it usually has a large number of columns. The dimension tables include significantly fewer rows of information than the fact table.

The attributes in a dimension table are used as row and column headings in a document or query results display.

Example: A city and state can view a store summary in a fact table. Item summary can be viewed by brand, color, etc. Customer information can be viewed by name and address

Fact Table

Time ID	Product ID	Customer ID	Unit Sold
4	17	2	1
8	21	3	2
8	4	1	1

In this example, Customer ID column in the facts table is the foreign keys that join with the dimension table. By following the links, we can see that row 2 of the fact table records the fact that customer 3, Gaurav, bought two items on day 8.

Dimension Tables

Customer ID	Name	Gender	Income	Education	Region

1	Rohan	Male	2	3	4
2	Sandeep	Male	3	5	1
3	Gaurav	Male	1	7	3

Hierarchy

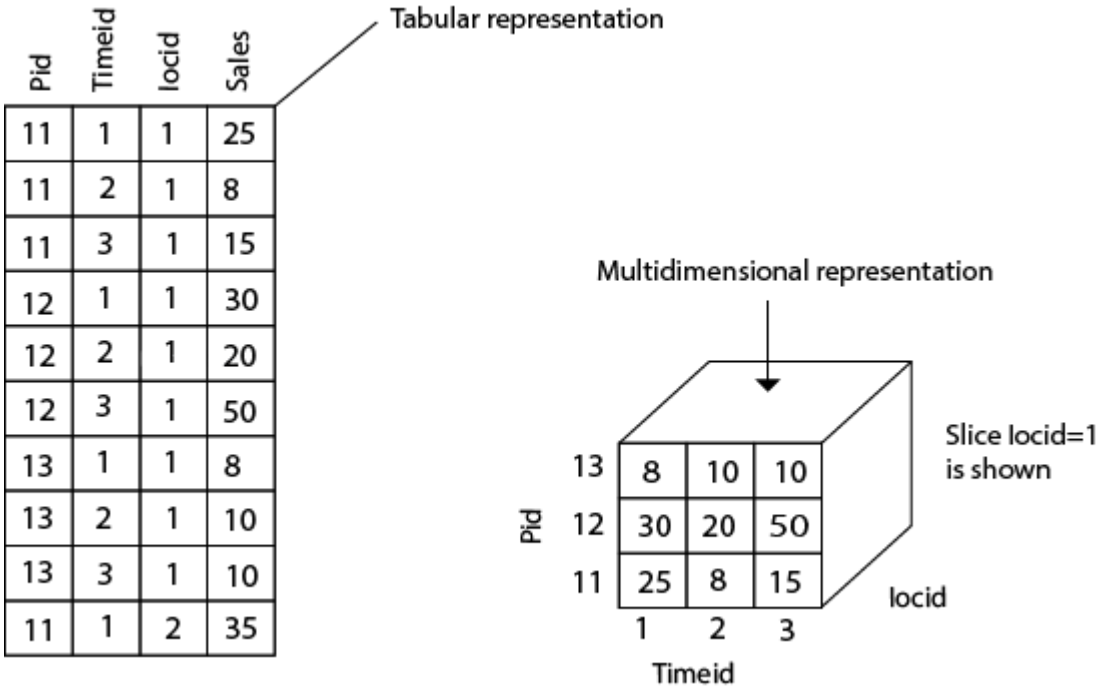
A hierarchy is a directed tree whose nodes are dimensional attributes and whose arcs model many to one association between dimensional attributes team. It contains a dimension, positioned at the tree's root, and all of the dimensional attributes that define it.

2.Multi-Dimensional Data Model:

A multidimensional model views data in the form of a data-cube. A data cube enables data to be modeled and viewed in multiple dimensions. It is defined by dimensions and facts.

The dimensions are the perspectives or entities concerning which an organization keeps records. For example, a shop may create a sales data warehouse to keep records of the store's sales for the dimension time, item, and location. These dimensions allow the save to keep track of things, for example, monthly sales of items and the locations at which the items were sold. Each dimension has a table related to it, called a dimensional table, which describes the dimension further. For example, a dimensional table for an item may contain the attributes item_name, brand, and type.

A multidimensional data model is organized around a central theme, for example, sales. This theme is represented by a fact table. Facts are numerical measures. The fact table contains the names of the facts or measures of the related dimensional tables.



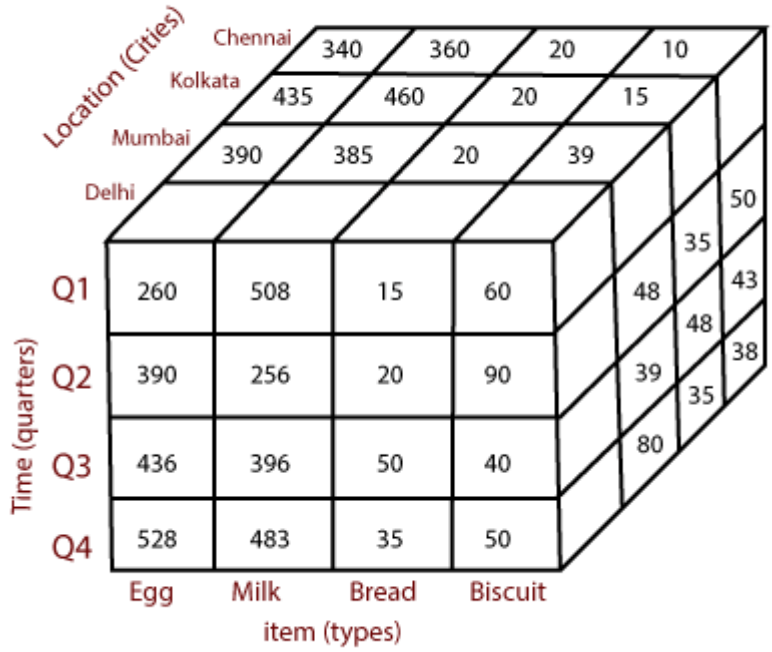
Consider the data of a shop for items sold per quarter in the city of Delhi. The data is shown in the table. In this 2D representation, the sales for Delhi are shown for the time dimension (organized in quarters) and the item dimension (classified according to the types of an item sold). The fact or measure displayed in rupee_sold (in thousands).creen

Location="Delhi"				
Time (quarter)	item (type)			
	Egg	Milk	Bread	Biscuit
Q1	260	508	15	60
Q2	390	256	20	90
Q3	436	396	50	40
Q4	528	483	35	50

Now, if we want to view the sales data with a third dimension, For example, suppose the data according to time and item, as well as the location is considered for the cities Chennai, Kolkata, Mumbai, and Delhi. These 3D data are shown in the table. The 3D data of the table are represented as a series of 2D tables.

	Location="Chennai"				Location="Kolkata"				Location="Mumbai"				Location="Delhi"			
	item				item				item				item			
Time	Egg	Milk	Bread	Biscuit	Egg	Milk	Bread	Biscuit	Egg	Milk	Bread	Biscuit	Egg	Milk	Bread	Biscuit
Q1	340	360	20	10	435	460	20	15	390	385	20	39	260	508	15	60
Q2	490	490	16	50	389	385	45	35	463	366	25	48	390	256	20	90
Q3	680	583	46	43	684	490	39	48	568	594	36	39	436	396	50	40
Q4	535	694	39	38	335	365	83	35	338	484	48	80	528	483	35	50

Conceptually, it may also be represented by the same data in the form of a 3D data cube, as shown in fig:

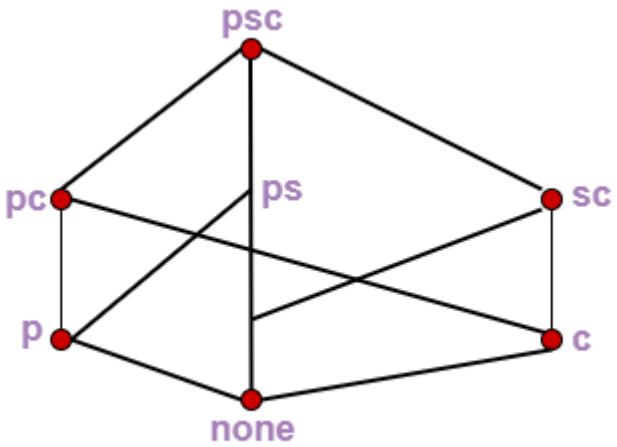


2.1 Data Cube:

When data is grouped or combined in multidimensional matrices called Data Cubes. The data cube method has a few alternative names or a few variants, such as "Multidimensional databases," "materialized views," and "OLAP (On-Line Analytical Processing)."

The general idea of this approach is to materialize certain expensive computations that are frequently inquired.

For example, a relation with the schema sales (part, supplier, customer, and sale-price) can be materialized into a set of eight views as shown in fig, where **psc** indicates a view consisting of aggregate function value (such as total-sales) computed by grouping three attributes part, supplier, and customer, **p** indicates a view composed of the corresponding aggregate function values calculated by grouping part alone, etc.



Eight views of data cubes for sales information.

A data cube is created from a subset of attributes in the database. Specific attributes are chosen to be measure attributes, i.e., the attributes whose values are of interest. Another attributes are selected as dimensions or functional attributes. The measure attributes are aggregated according to the dimensions.

For example, XYZ may create a sales data warehouse to keep records of the store's sales for the dimensions time, item, branch, and location. These dimensions enable 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 identify with it, known as a dimensional table, which describes the dimensions. For example, a dimension table for items may contain the attributes item_name, brand, and type.

Data cube method is an interesting technique with many applications. Data cubes could be sparse in many cases because not every cell in each dimension may have corresponding data in the database.

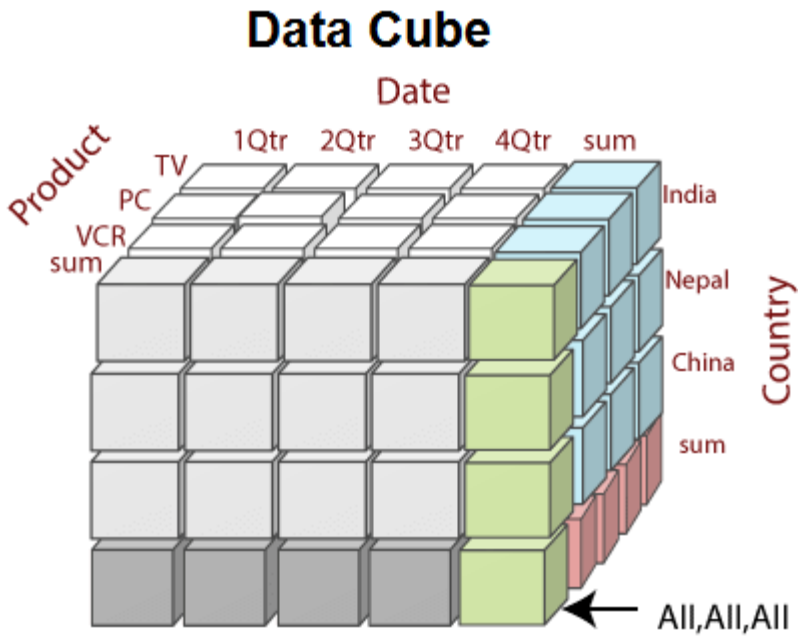
Techniques should be developed to handle sparse cubes efficiently.

If a query contains constants at even lower levels than those provided in a data cube, it is not clear how to make the best use of the precomputed results stored in the data cube.

The model view data in the form of a data cube. OLAP tools are based on the multidimensional data model. Data cubes usually model n-dimensional data.

A data cube enables data to be modeled and viewed in multiple dimensions. A multidimensional data model is organized around a central theme, like sales and transactions. A fact table represents this theme. Facts are numerical measures. Thus, the fact table contains measure (such as Rs_sold) and keys to each of the related dimensional tables.

Dimensions are a fact that defines a data cube. Facts are generally quantities, which are used for analyzing the relationship between dimensions.



Example: In the **2-D representation**, we will look at the All Electronics sales data for **items sold per quarter** in the city of Vancouver. The measured display in dollars sold (in thousands).

2-D view of Sales Data

location ="Vancouver"				
	item (type)			
time (quarter)	home entertainment	computer	phone	security
Q1	605	825	14	400
Q2	680	952	31	512
Q3	812	1023	30	501
Q3	927	1038	38	580

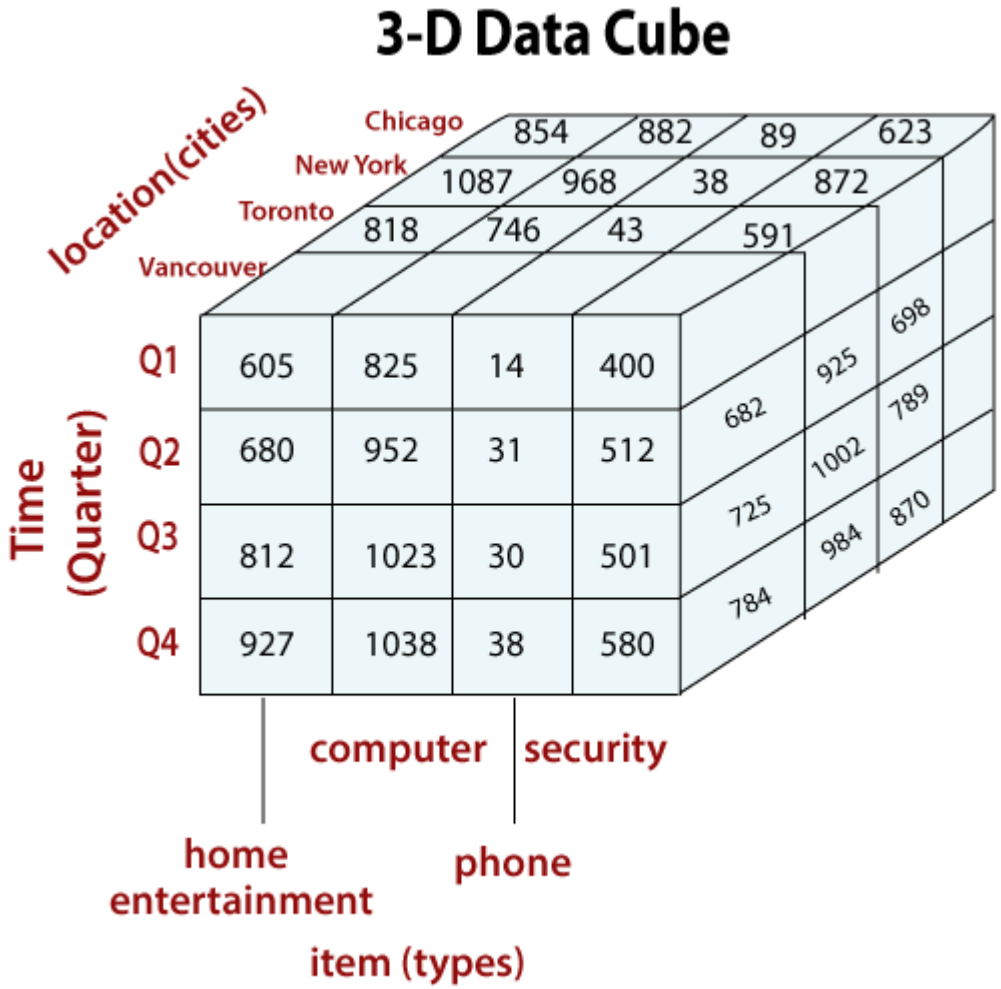
3-Dimensional Cuboids

Let suppose we would like to view the sales data with a third dimension. For example, suppose we would like to view the data according to time, item as well as the location for the cities Chicago, New York, Toronto, and Vancouver. The measured display in dollars sold (in thousands). These 3-D data are shown in the table. The 3-D data of the table are represented as a series of 2-D tables.

3-D view of Sales Data

location ="Chicago"					location ="New York"				location ="Toronto"			
item					item				item			
time	home ent.	comp.	phone	sec.	time	home comp.	phone	sec.	home ent.	comp.	phone	sec.
Q1	854	882	89	623	1087	968	38	872	818	746	43	591
Q2	943	890	64	698	1130	1024	41	925	894	769	52	682
Q3	1032	924	59	789	1034	1048	45	1002	940	795	58	728
Q4	1129	992	63	870	1142	1091	54	984	978	864	59	784

Conceptually, we may represent the same data in the form of 3-D data cubes, as shown in fig:



Let us suppose that we would like to view our sales data with an additional fourth dimension, such as a supplier.

In data warehousing, the data cubes are n-dimensional. The cuboid which holds the lowest level of summarization is called a **base cuboid**.

For example, the **4-D cuboid** in the figure is the base cuboid for the given time, item, location, and supplier dimensions.

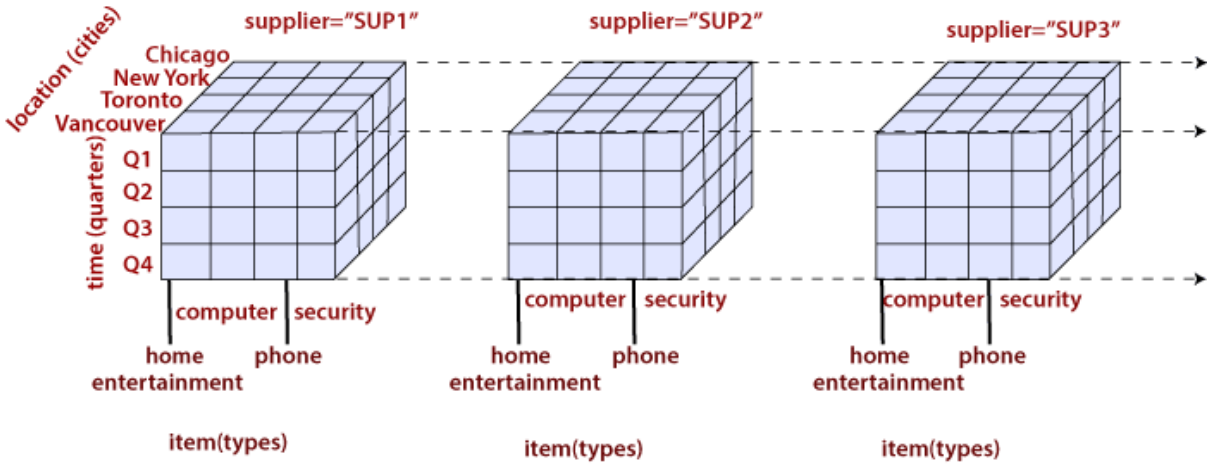
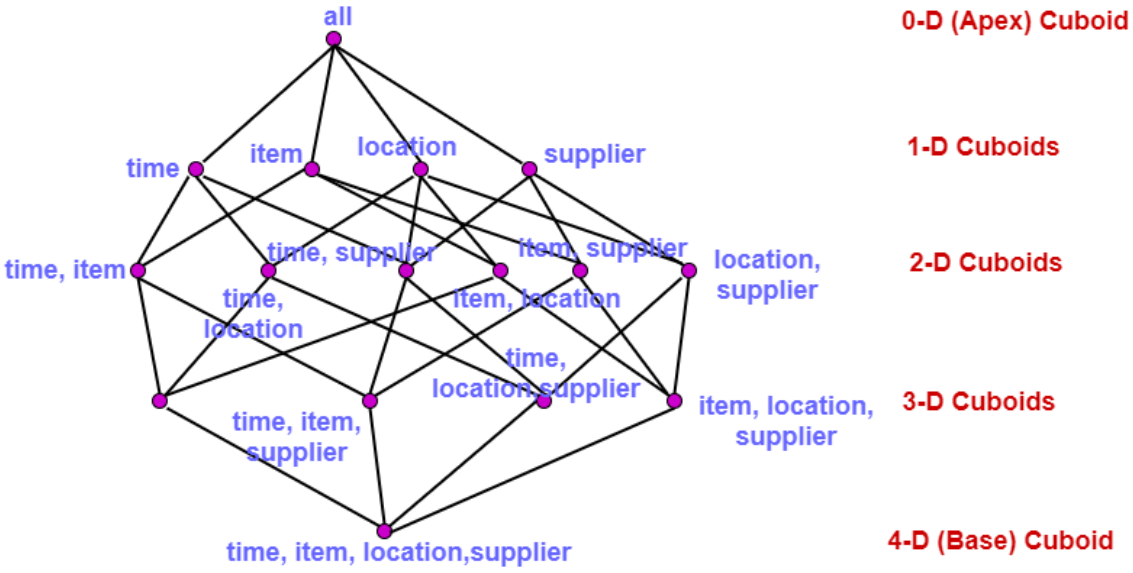


Figure is shown a **4-D data cube** representation of sales data, according to the dimensions time, item, location, and supplier. The measure displayed is dollars sold (in thousands).

The topmost **0-D cuboid**, which holds the highest level of summarization, is known as the apex cuboid. In this example, this is the total sales, or dollars sold, summarized over all four dimensions.

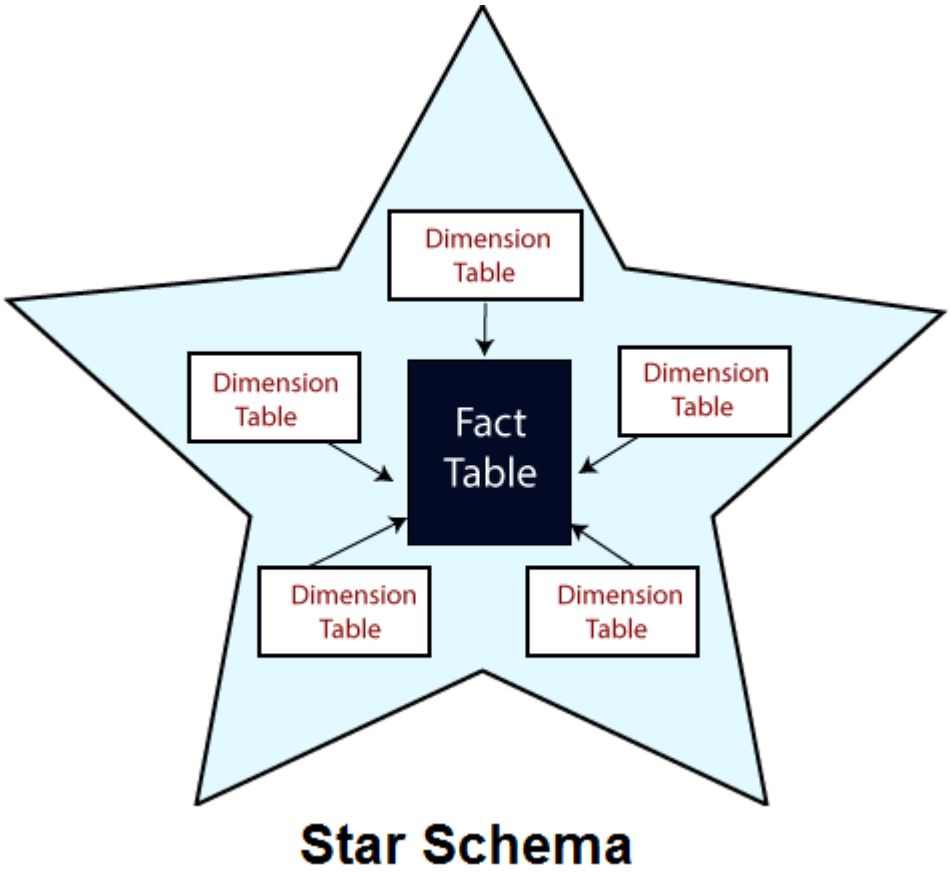
The lattice of cuboid forms a data cube. The figure shows the lattice of cuboids creating 4-D data cubes for the dimension time, item, location, and supplier. Each cuboid represents a different degree of summarization.



3.Star Schema:

A star schema is the elementary form of a dimensional model, in which data are organized into **facts** and **dimensions**. A fact is an event that is counted or measured, such as a sale or log in. A dimension includes reference data about the fact, such as date, item, or customer.

A star schema is a relational schema where a relational schema whose design represents a multidimensional data model. The star schema is the explicit data warehouse schema. It is known as **star schema** because the entity-relationship diagram of this schemas simulates a star, with points, diverge from a central table. The center of the schema consists of a large fact table, and the points of the star are the dimension tables.



Star Schema

3.1 Fact Tables

A table in a star schema which contains facts and connected to dimensions. A fact table has two types of columns: those that include fact and those that are foreign keys to the dimension table. The primary key of the fact tables is generally a composite key that is made up of all of its foreign keys.

A fact table might involve either detail level fact or fact that have been aggregated (fact tables that include aggregated fact are often instead called summary tables). A fact table generally contains facts with the same level of aggregation.

3.2 Dimension Tables

A dimension is an architecture usually composed of one or more hierarchies that categorize data. If a dimension has not got hierarchies and levels, it is called a **flat dimension** or **list**. The primary keys of each of the dimensions table are part of the composite primary keys of the fact table. Dimensional attributes help to define the dimensional value. They are generally descriptive, textual values. Dimensional tables are usually small in size than fact table.

Fact tables store data about sales while dimension tables data about the geographic region (markets, cities), clients, products, times, channels.

3.3 Characteristics of Star Schema

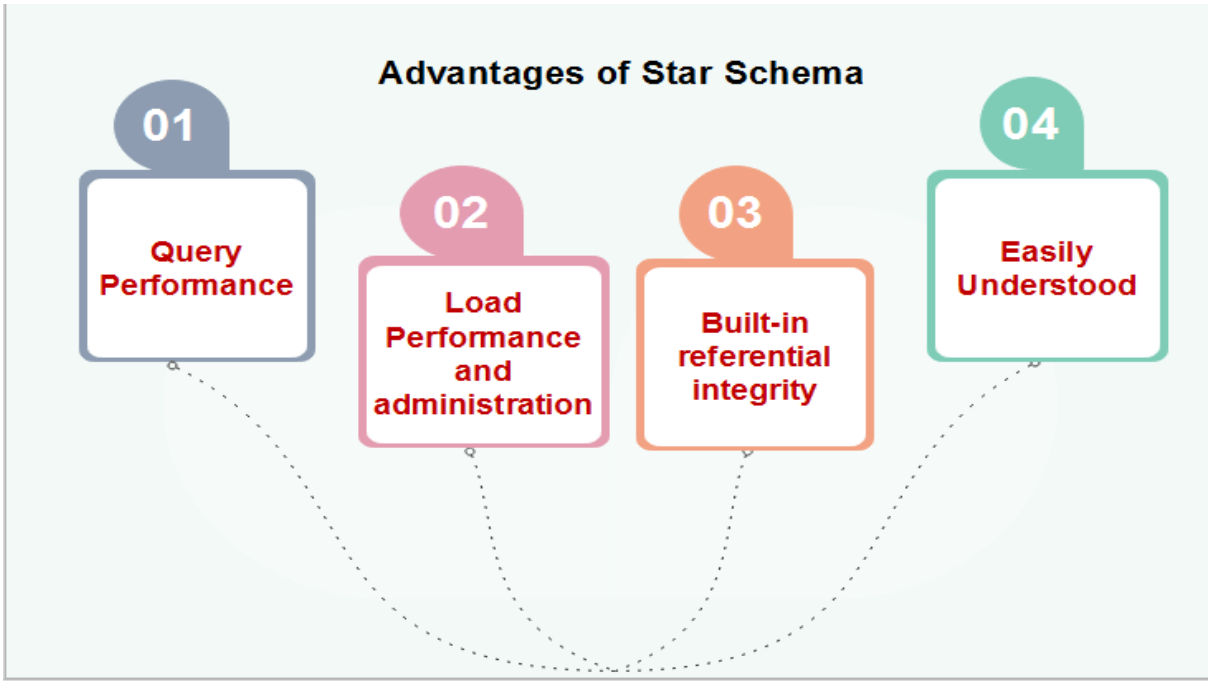
The star schema is intensely suitable for data warehouse database design because of the following features:

- It creates a DE-normalized database that can quickly provide query responses.
- It provides a flexible design that can be changed easily or added to throughout the development cycle, and as the database grows.
- It provides a parallel in design to how end-users typically think of and use the data.
- It reduces the complexity of metadata for both developers and end-users.

3.4 Advantages of Star Schema

Star Schemas are easy for end-users and application to understand and navigate. With a well-designed schema, the customer can instantly analyze large, multidimensional data sets.

The main advantage of star schemas in a decision-support environment are:



3.5 Query Performance

A star schema database has a limited number of table and clear join paths, the query run faster than they do against OLTP systems. Small single-table queries, frequently of a dimension table, are almost instantaneous. Large join queries that contain multiple tables takes only seconds or minutes to run.

In a star schema database design, the dimension is connected only through the central fact table. When the two-dimension table is used in a query, only one join path, intersecting the fact tables, exist between those two tables. This design feature enforces authentic and consistent query results.

3.6 Load performance and administration

Structural simplicity also decreases the time required to load large batches of record into a star schema database. By describing facts and dimensions and separating them into the various table, the impact of a load structure is reduced. Dimension table can be populated once and occasionally refreshed. We can add new facts regularly and selectively by appending records to a fact table.

3.7 Built-in referential integrity

A star schema has referential integrity built-in when information is loaded. Referential integrity is enforced because each data in dimensional tables has a unique primary key, and all keys in the fact table are legitimate foreign keys drawn from the dimension table. A record in the fact table which is not related correctly to a dimension cannot be given the correct key value to be retrieved.

3.8 Easily Understood

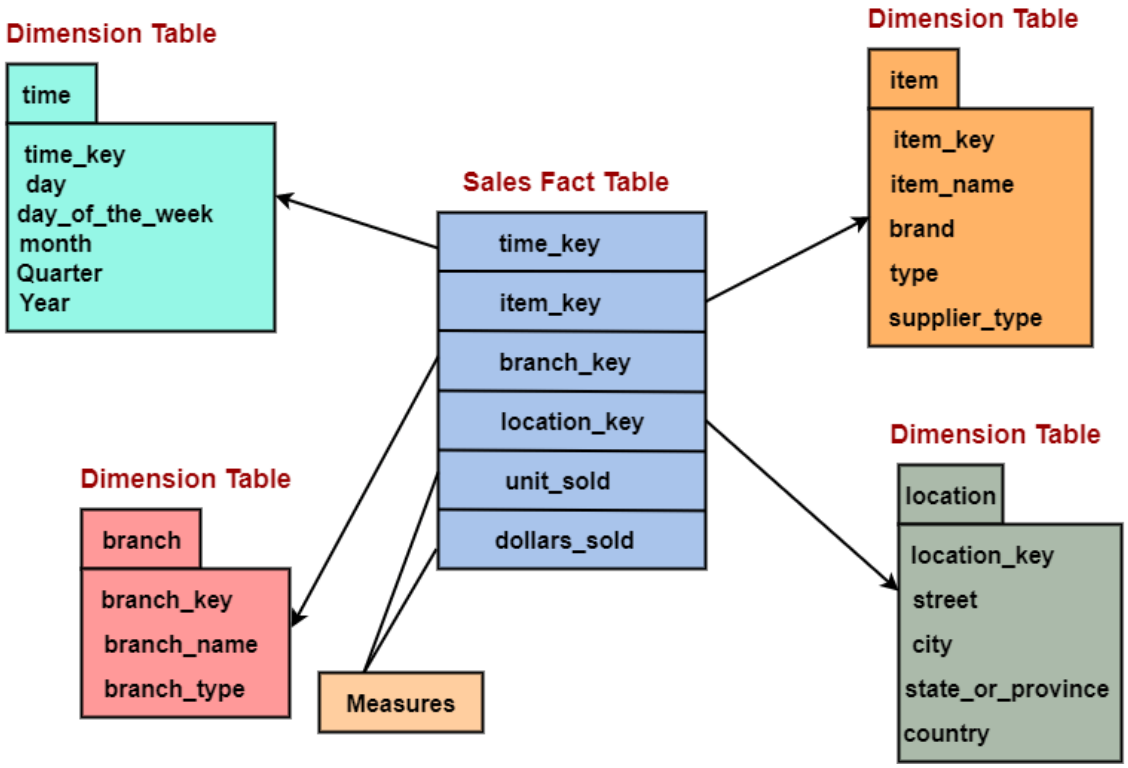
A star schema is simple to understand and navigate, with dimensions joined only through the fact table. These joins are more significant to the end-user because they represent the fundamental relationship between parts of the underlying business. Customer can also browse dimension table attributes before constructing a query.

3.9 Disadvantage of Star Schema

There is some condition which cannot be meet by star schemas like the relationship between the user, and bank account cannot describe as star schema as the relationship between them is many to many.

Example: Suppose a star schema is composed of a fact table, SALES, and several dimension tables connected to it for time, branch, item, and geographic locations.

The TIME table has a column for each day, month, quarter, and year. The ITEM table has columns for each item_Key, item_name, brand, type, supplier_type. The BRANCH table has columns for each branch_key, branch_name, branch_type. The LOCATION table has columns of geographic data, including street, city, state, and country.



In this scenario, the SALES table contains only four columns with IDs from the dimension tables, TIME, ITEM, BRANCH, and LOCATION, instead of four columns for time data, four columns for ITEM data, three columns for BRANCH data, and four columns for LOCATION data. Thus, the size of the fact table is significantly reduced. When we need to change an item, we need only make a single change in the dimension table, instead of making many changes in the fact table.
We can create even more complex star schemas by normalizing a dimension table into several tables. The normalized dimension table is called a **Snowflake**.

4.Snowflake Schema:

A snowflake schema is equivalent to the star schema. "A schema is known as a snowflake if one or more dimension tables do not connect directly to the fact table but must join through other dimension tables."

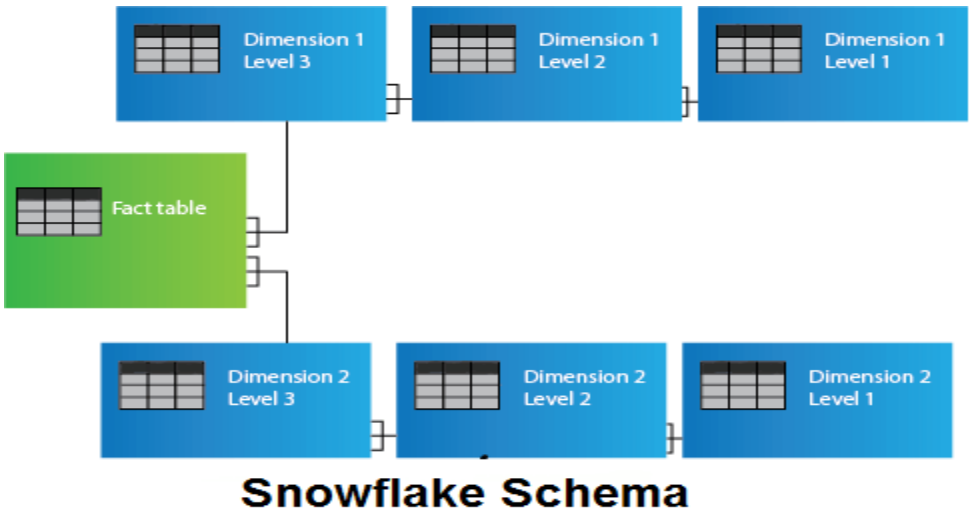
The snowflake schema is an expansion of the star schema where each point of the star explodes into more points. It is called snowflake schema because the diagram of snowflake schema resembles a snowflake. **Snowflaking** is a method of normalizing the dimension tables in a STAR schemas. When we normalize all the dimension tables entirely, the resultant structure resembles a snowflake with the fact table in the middle.

Snowflaking is used to develop the performance of specific queries. The schema is diagramed with each fact surrounded by its associated dimensions, and those dimensions are related to other dimensions, branching out into a snowflake pattern.

The snowflake schema consists of one fact table which is linked to many dimension tables, which can be linked to other dimension tables through a many-to-one relationship. Tables in a

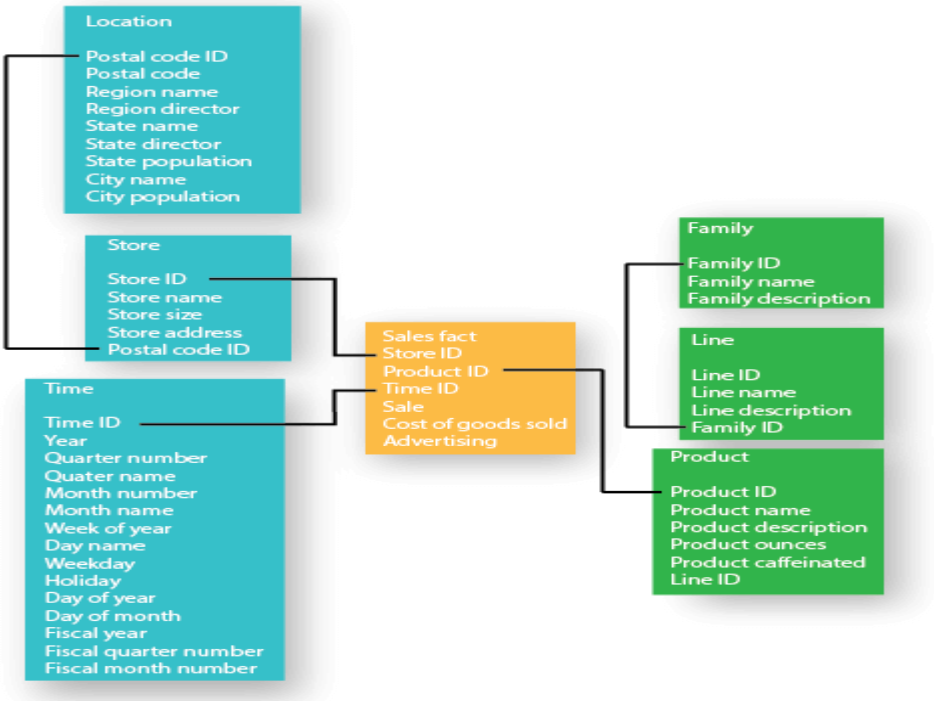
snowflake schema are generally normalized to the third normal form. Each dimension table performs exactly one level in a hierarchy.

The following diagram shows a snowflake schema with two dimensions, each having three levels. A snowflake schemas can have any number of dimension, and each dimension can have any number of levels.



Snowflake Schema

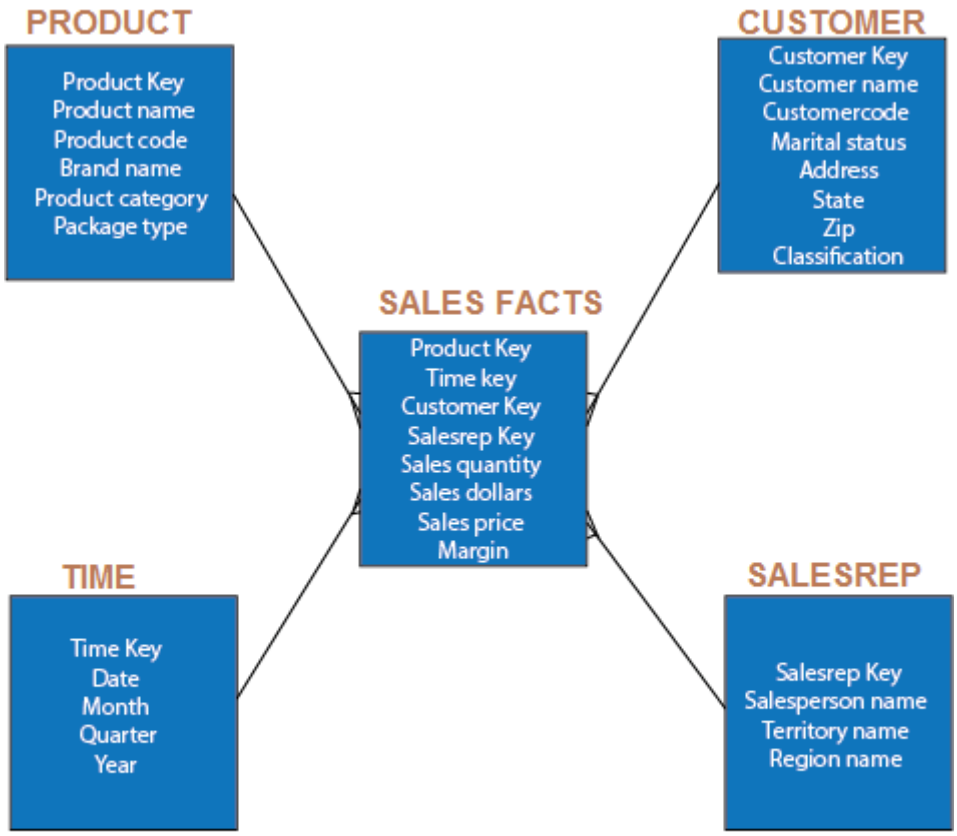
Example: Figure shows a snowflake schema with a Sales fact table, with Store, Location, Time, Product, Line, and Family dimension tables. The Market dimension has two dimension tables with Store as the primary dimension table, and Location as the outrigger dimension table. The product dimension has three dimension tables with Product as the primary dimension table, and the Line and Family table are the outrigger dimension tables.



A star schema store all attributes for a dimension into one denormalized table. This needed more disk space than a more normalized snowflake schema. Snowflaking normalizes the dimension by moving attributes with low cardinality into separate dimension tables that relate to the core dimension table by using foreign keys. Snowflaking for the sole purpose of minimizing disk space is not recommended, because it can adversely impact query performance.

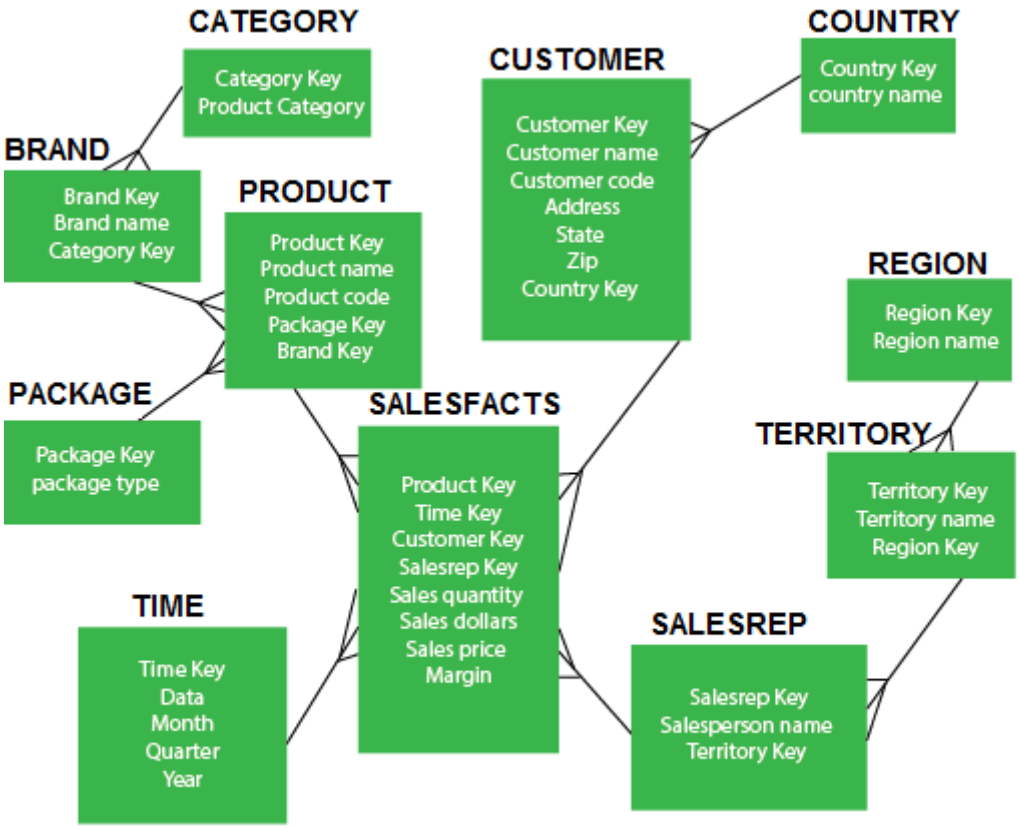
In snowflake, schema tables are normalized to delete redundancy. In snowflake dimension tables are damaged into multiple dimension tables.

Figure shows a simple STAR schema for sales in a manufacturing company. The sales fact table include quantity, price, and other relevant metrics. SALESREP, CUSTOMER, PRODUCT, and TIME are the dimension tables.



STAR Schema

The STAR schema for sales, as shown above, contains only five tables, whereas the normalized version now extends to eleven tables. We will notice that in the snowflake schema, the attributes with low cardinality in each original dimension tables are removed to form separate tables. These new tables are connected back to the original dimension table through artificial keys.



Snowflake Schema

A snowflake schema is designed for flexible querying across more complex dimensions and relationship. It is suitable for many to many and one to many relationships between dimension levels.

4.1 Advantage of Snowflake Schema

1. The primary advantage of the snowflake schema is the development in query performance due to minimized disk storage requirements and joining smaller lookup tables.
2. It provides greater scalability in the interrelationship between dimension levels and components.
3. No redundancy, so it is easier to maintain.

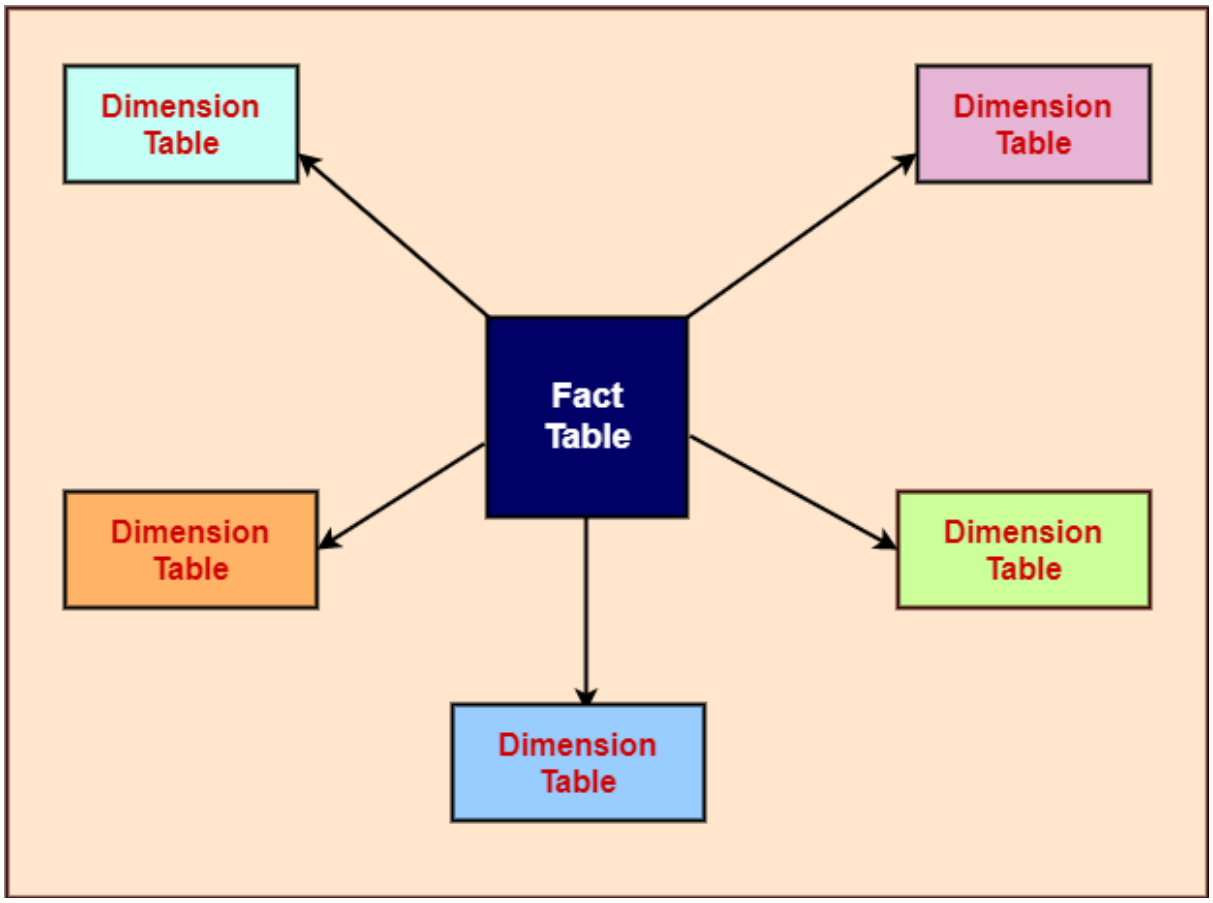
4.2 Disadvantage of Snowflake Schema

1. The primary disadvantage of the snowflake schema is the additional maintenance efforts required due to the increasing number of lookup tables. It is also known as a multi fact star schema.
2. There are more complex queries and hence, difficult to understand.
3. More tables more join so more query execution time.

4.3 Difference between Star and Snowflake Schemas:

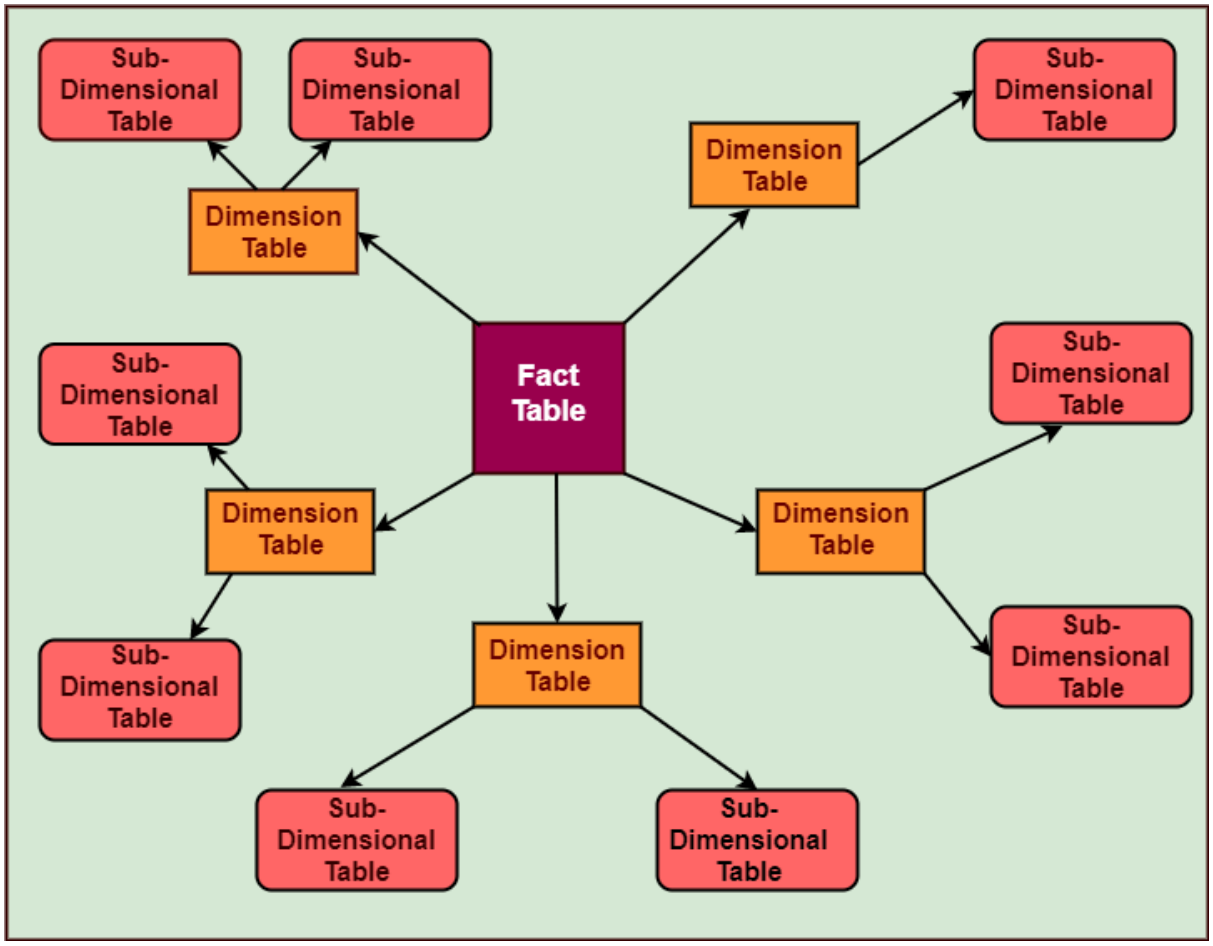
Star Schema

- In a star schema, the fact table will be at the center and is connected to the dimension tables.
- The tables are completely in a denormalized structure.
- SQL queries performance is good as there is less number of joins involved.
- Data redundancy is high and occupies more disk space.

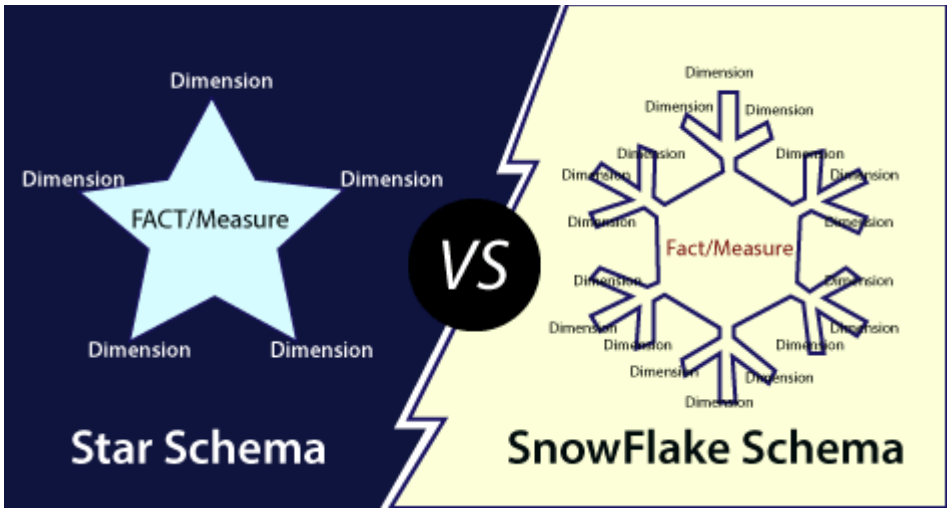


Snowflake Schema

- A snowflake schema is an extension of star schema where the dimension tables are connected to one or more dimensions.
- The tables are partially denormalized in structure.
- The performance of SQL queries is a bit less when compared to star schema as more number of joins are involved.
- Data redundancy is low and occupies less disk space when compared to star schema.



Let's see the differentiate between Star and Snowflake Schema.

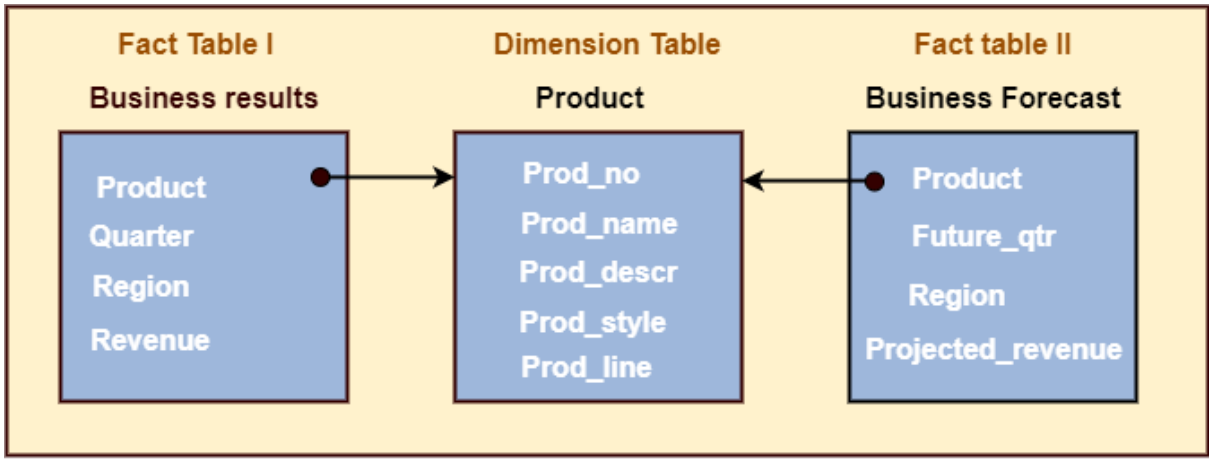


Basis for Comparison	Star Schema	Snowflake Schema
Ease of Maintenance/change	It has redundant data and hence less easy to maintain/change	No redundancy and therefore more easy to maintain and change
Ease of Use	Less complex queries and simple to understand	More complex queries and therefore less easy to understand
Parent table	In a star schema, a dimension table will not have any parent table	In a snowflake schema, a dimension table will have one or more parent tables
Query Performance	Less number of foreign keys and hence lesser query execution time	More foreign keys and thus more query execution time
Normalization	It has De-normalized tables	It has normalized tables
Type of Data Warehouse	Good for data marts with simple relationships (one to one or one to many)	Good to use for data warehouse core to simplify complex relationships (many to many)
Joins	Fewer joins Higher	number of joins
Dimension Table	It contains only a single dimension table for each dimension	It may have more than one dimension table for each dimension
Hierarchies	Hierarchies for the dimension are stored in the dimensional table itself in a star schema	Hierarchies are broken into separate tables in a snowflake schema. These hierarchies help to drill down the information from topmost hierarchies to the lowermost hierarchies.
When to use	When the dimensional table contains less number of rows, we can go for Star schema.	When dimensional table store a huge number of rows with redundancy information and space is such an issue, we can choose snowflake schema to store space.
Data Warehouse system	Work best in any data warehouse/ data mart	Better for small data warehouse/data mart.

5.Fact Constellation Schema:

A Fact constellation means two or more fact tables sharing one or more dimensions. It is also called **Galaxy schema**.

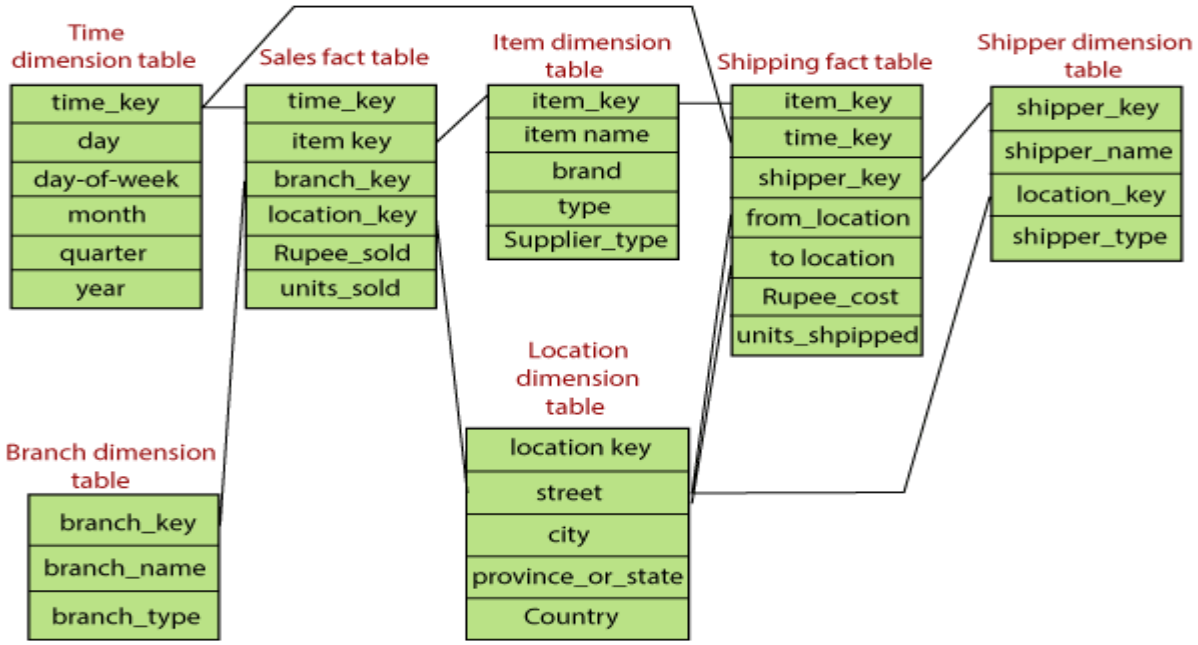
Fact Constellation Schema describes a logical structure of data warehouse or data mart. Fact Constellation Schema can design with a collection of de-normalized FACT, Shared, and Conformed Dimension tables.



FACT Constellation Schema

Fact Constellation Schema is a sophisticated database design that is difficult to summarize information. Fact Constellation Schema can implement between aggregate Fact tables or decompose a complex Fact table into independent simplex Fact tables.

Example: A fact constellation schema is shown in the figure below.



This schema defines two fact tables, sales, and shipping. Sales are treated along four dimensions, namely, time, item, branch, and location. The schema contains a fact table for sales that includes keys to each of the four dimensions, along with two measures: Rupee_sold and units_sold. The shipping table has five dimensions, or keys: item_key, time_key, shipper_key, from_location, and to_location, and two measures: Rupee_cost and units_shipped.

The primary disadvantage of the fact constellation schema is that it is a more challenging design because many variants for specific kinds of aggregation must be considered and selected.

6.Schema Definition

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

6.1 Syntax for Cube Definition

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

6.2 Syntax for Dimension Definition

define dimension < dimension_name > as (< attribute_or_dimension_list >)

6.3 Star Schema Definition

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

define cube sales star [time, item, branch, location]:
dollars sold = sum(sales in dollars), units sold = count(*)
define dimension time as (time key, day, day of week, month, quarter, year)
define dimension item as (item key, item name, brand, type, supplier type)
define dimension branch as (branch key, branch name, branch type)
define dimension location as (location key, street, city, province or state, country)

6.4 Snowflake Schema Definition

Snowflake schema can be defined using DMQL as follows −

define cube sales snowflake [time, item, branch, location]:
dollars sold = sum(sales in dollars), units sold = count(*)
define dimension time as (time key, day, day of week, month, quarter, year)
define dimension item as (item key, item name, brand, type, supplier (supplier key, supplier type))
define dimension branch as (branch key, branch name, branch type)
define dimension location as (location key, street, city (city key, city, province or state

7. Data Warehouse Applications

The application areas of the data warehouse are:

7.1 Information Processing

It deals with querying, statistical analysis, and reporting via tables, charts, or graphs. Nowadays, information processing of data warehouse is to construct a low cost, web-based accessing tools typically integrated with web browsers.

7.2 Analytical Processing

It supports various online analytical processing such as drill-down, roll-up, and pivoting. The historical data is being processed in both summarized and detailed format.

OLAP is implemented on data warehouses or data marts. The primary objective of OLAP is to support ad-hoc querying needed for support DSS. The multidimensional view of data is fundamental to the OLAP application. OLAP is an operational view, not a data structure or schema. The complex nature of OLAP applications requires a multidimensional view of the data.

7.3 Data Mining

It helps in the analysis of hidden design and association, constructing scientific models, operating classification and prediction, and performing the mining results using visualization tools.

Data mining is the technique of designing essential new correlations, patterns, and trends by changing through high amounts of a record save in repositories, using pattern recognition technologies as well as statistical and mathematical techniques.

It is the phase of selection, exploration, and modeling of huge quantities of information to determine regularities or relations that are at first unknown to access precise and useful results for the owner of the database.

It is the process of inspection and analysis, by automatic or semi-automatic means, of large quantities of records to discover meaningful patterns and rules.

8. Data Warehouse Process Architecture

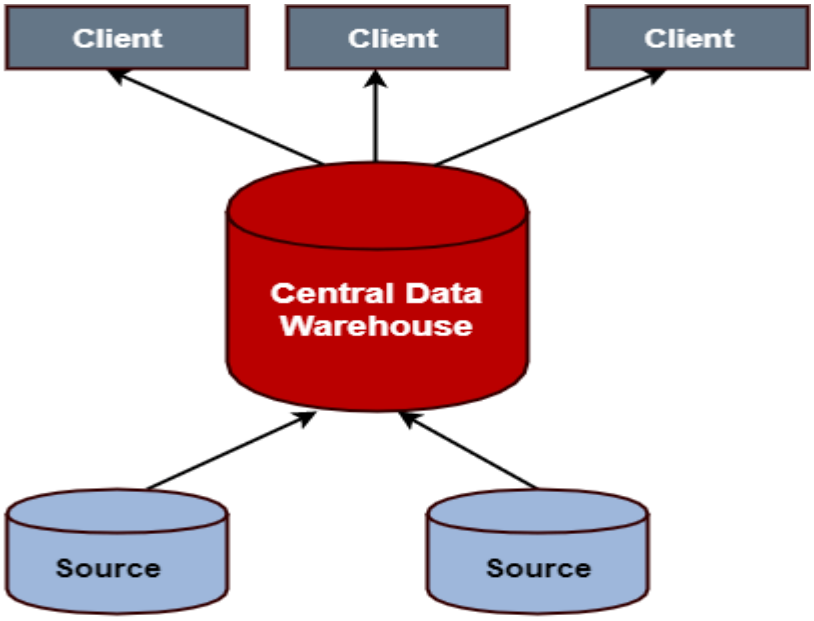
The process architecture defines an architecture in which the data from the data warehouse is processed for a particular computation.

Following are the two fundamental process architectures:

8.1 Centralized Process Architecture:

In this architecture, the data is collected into single centralized storage and processed upon completion by a single machine with a huge structure in terms of memory, processor, and storage.

Centralized process architecture evolved with transaction processing and is well suited for small organizations with one location of service.It requires minimal resources both from people and system perspectives. It is very successful when the collection and consumption of data occur at the same location.



Centralized Process Architecture

8.2 Distributed Process Architecture:

In this architecture, information and its processing are allocated across data centers, and its processing is distributed across data centers, and processing of data is localized with the group of the results into centralized storage.Distributed architectures are used to overcome the limitations of the centralized process architectures where all the information needs to be collected to one central location, and results are available in one central location.

There are several architectures of the distributed process:

8.2.1 Client-Server

In this architecture, the user does all the information collecting and presentation, while the server does the processing and management of data.

8.2.2 Three-tier Architecture

With client-server architecture, the client machines need to be connected to a server machine, thus mandating finite states and introducing latencies and overhead in terms of record to be carried between clients and servers.

8.2.3 N-tier Architecture

The n-tier or multi-tier architecture is where clients, middleware, applications, and servers are isolated into tiers.

8.2.4 Cluster Architecture

In this architecture, machines that are connected in network architecture (software or hardware) to approximately work together to process information or compute requirements in parallel.

Each device in a cluster is associated with a function that is processed locally, and the result sets are collected to a master server that returns it to the user.

8.2.5 Peer-to-Peer Architecture

This is a type of architecture where there are no dedicated servers and clients. Instead, all the processing responsibilities are allocated among all machines, called peers. Each machine can perform the function of a client or server or just process data.

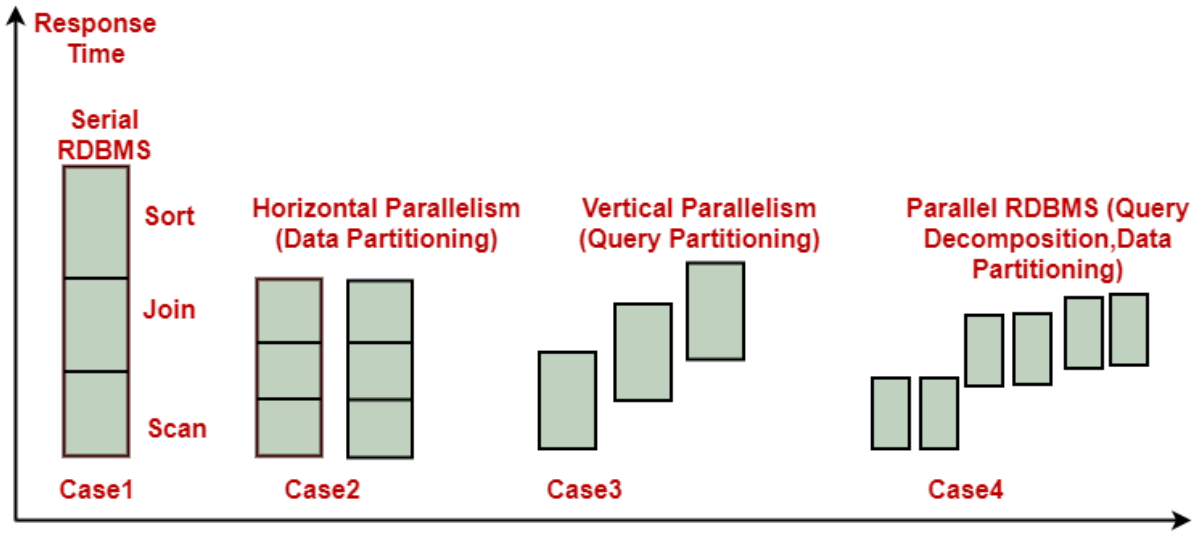
9.Types of Database Parallelism

Parallelism is used to support speedup, where queries are executed faster because more resources, such as processors and disks, are provided. Parallelism is also used to provide scale-up, where increasing workloads are managed without increase response-time, via an increase in the degree of parallelism.

Different architectures for parallel database systems are shared-memory, shared-disk, shared-nothing, and hierarchical structures.

(a)Horizontal Parallelism: It means that the database is partitioned across multiple disks, and parallel processing occurs within a specific task (i.e., table scan) that is performed concurrently on different processors against different sets of data.

(b)Vertical Parallelism: It occurs among various tasks. All component query operations (i.e., scan, join, and sort) are executed in parallel in a pipelined fashion. In other words, an output from one function (e.g., join) as soon as records become available.



c) Intraquery Parallelism

Intraquery parallelism defines the execution of a single query in parallel on multiple processors and disks. Using intraquery parallelism is essential for speeding up long-running queries.

Interquery parallelism does not help in this function since each query is run sequentially.

To improve the situation, many DBMS vendors developed versions of their products that utilized intraquery parallelism.

This application of parallelism decomposes the serial SQL, query into lower-level operations such as scan, join, sort, and aggregation.

These lower-level operations are executed concurrently, in parallel.

d) Interquery Parallelism

In interquery parallelism, different queries or transaction execute in parallel with one another.

This form of parallelism can increase transactions throughput. The response times of individual transactions are not faster than they would be if the transactions were run in isolation.

Thus, the primary use of interquery parallelism is to scale up a transaction processing system to support a more significant number of transactions per second.

Database vendors started to take advantage of parallel hardware architectures by implementing multiserver and multithreaded systems designed to handle a large number of client requests efficiently.

This approach naturally resulted in interquery parallelism, in which different server threads (or processes) handle multiple requests at the same time.

Interquery parallelism has been successfully implemented on SMP systems, where it increased the throughput and allowed the support of more concurrent users.

10. Shared Disk Architecture

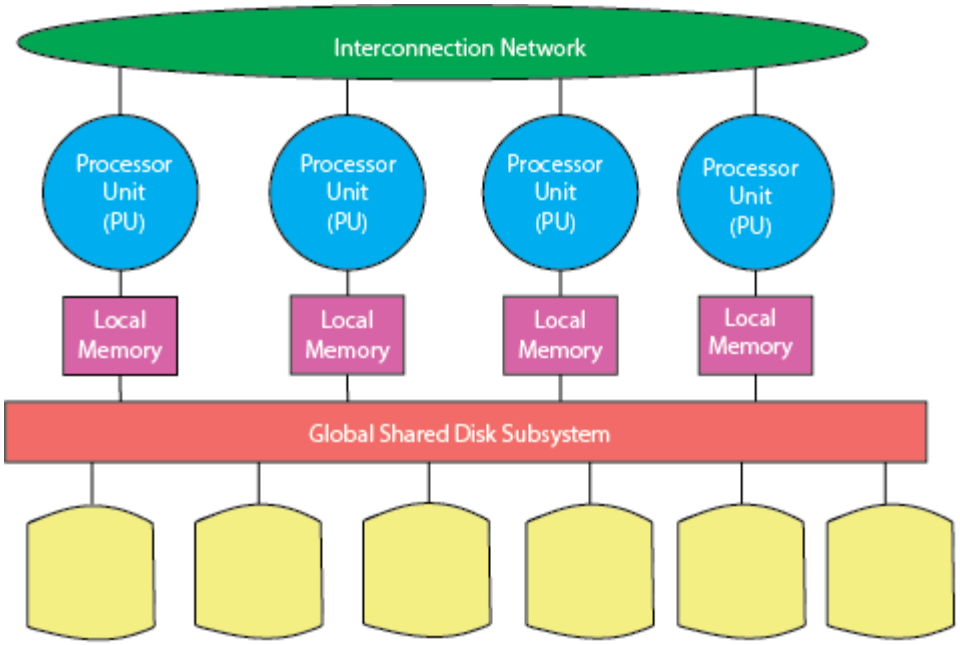
Shared-disk architecture implements a concept of shared ownership of the entire database between RDBMS servers, each of which is running on a node of a distributed memory system.

Each RDBMS server can read, write, update, and delete information from the same shared database, which would need the system to implement a form of a distributed lock manager (DLM).

DLM components can be found in hardware, the operating system, and separate software layer, all depending on the system vendor.

On the positive side, shared-disk architectures can reduce performance bottlenecks resulting from data skew (uneven distribution of data), and can significantly increase system availability.

The shared-disk distributed memory design eliminates the memory access bottleneck typically of large SMP systems and helps reduce DBMS dependency on data partitioning.



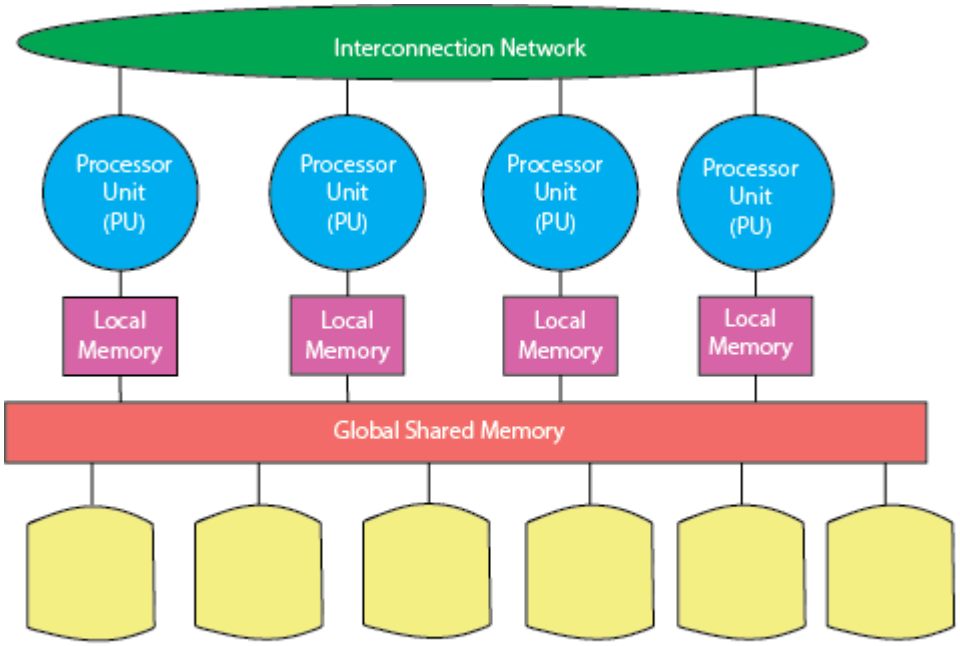
Distributed memory shared-disk architecture

11.Shared-Memory Architecture

Shared-memory or shared-everything style is the traditional approach of implementing an RDBMS on SMP hardware.

It is relatively simple to implement and has been very successful up to the point where it runs into the scalability limitations of the shared-everything architecture.

The key point of this technique is that a single RDBMS server can probably apply all processors, access all memory, and access the entire database, thus providing the client with a consistent single system image.



Shared-Memory Architecture

In shared-memory SMP systems, the DBMS considers that the multiple database components executing SQL statements communicate with each other by exchanging messages and information via the shared memory.

All processors have access to all data, which is partitioned across local disks.

12. Shared-Nothing Architecture

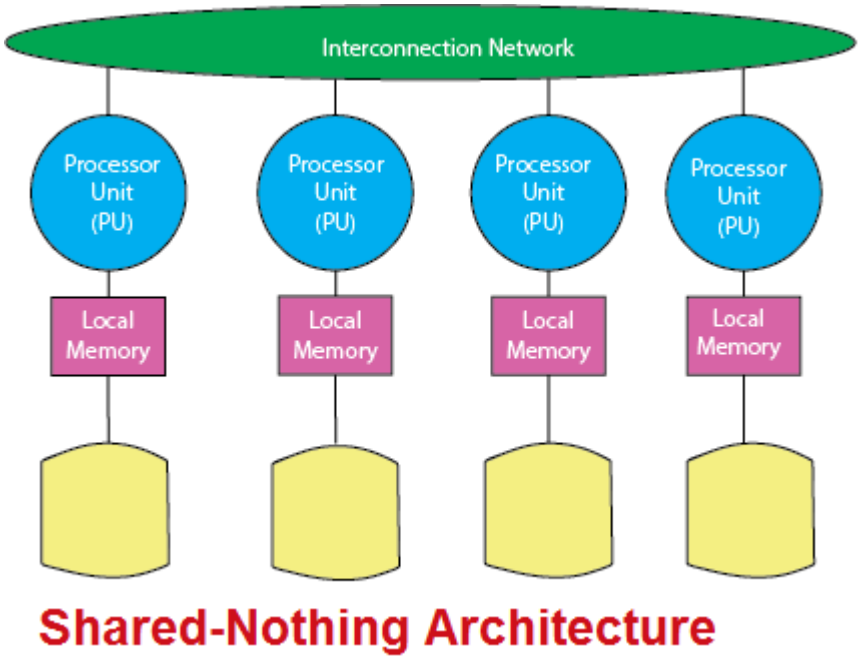
In a shared-nothing distributed memory environment, the data is partitioned across all disks, and the DBMS is "partitioned" across multiple co-servers, each of which resides on individual nodes of the parallel system and has an ownership of its disk and thus its database partition.

A shared-nothing RDBMS parallelizes the execution of a SQL query across multiple processing nodes.

Each processor has its memory and disk and communicates with other processors by exchanging messages and data over the interconnection network.

This architecture is optimized specifically for the MPP and cluster systems.

The shared-nothing architectures offer near-linear scalability. The number of processor nodes is limited only by the hardware platform limitations (and budgetary constraints), and each node itself can be a powerful SMP system.



13. Data Warehouse Tools

The tools that allow sourcing of data contents and formats accurately and external data stores into the data warehouse have to perform several essential tasks that contain: Data consolidation and integration.

- Data transformation from one form to another form.
- Data transformation and calculation based on the function of business rules that force transformation.
- Metadata synchronization and management, which includes storing or updating metadata about source files, transformation actions, loading formats, and events.

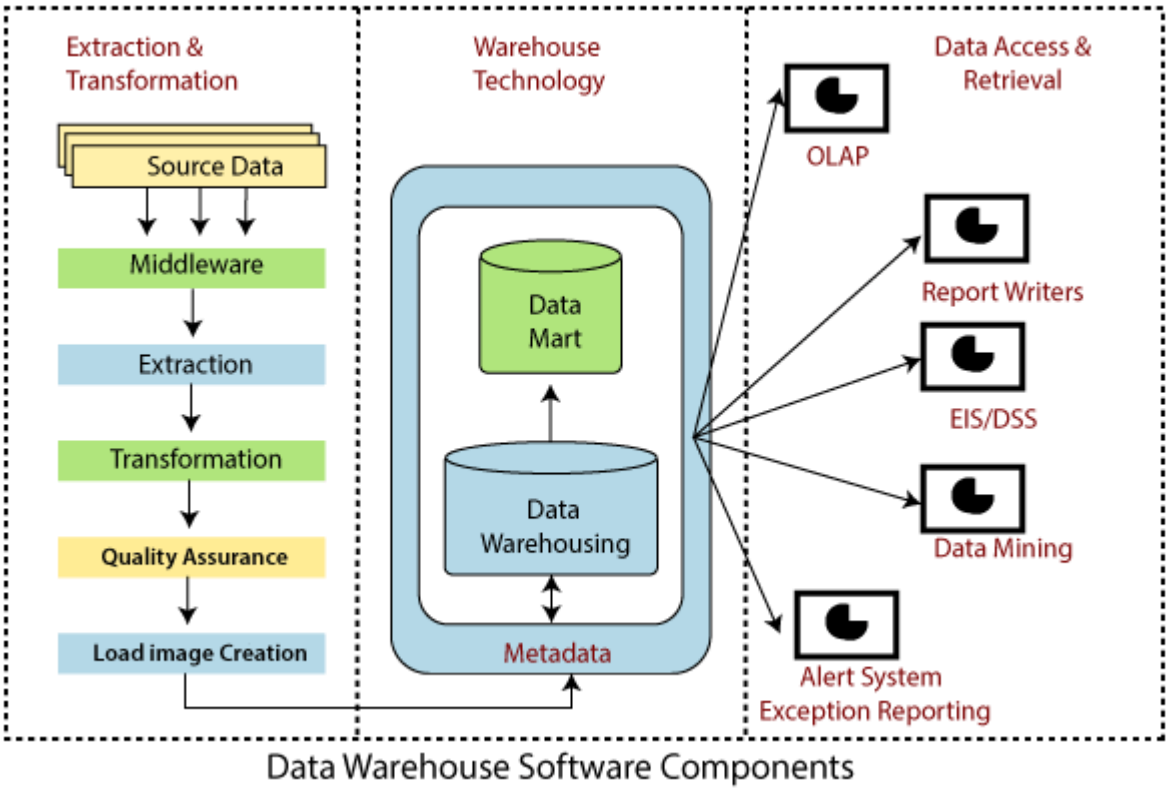
There are several selection criteria which should be considered while implementing a data warehouse:

1. The ability to identify the data in the data source environment that can be read by the tool is necessary.
2. Support for flat files, indexed files, and legacy DBMSs is critical.
3. The capability to merge records from multiple data stores is required in many installations.
4. The specification interface to indicate the information to be extracted and conversation are essential.
5. The ability to read information from repository products or data dictionaries is desired.
6. The code develops by the tool should be completely maintainable.
7. Selective data extraction of both data items and records enables users to extract only the required data.
8. A field-level data examination for the transformation of data into information is needed.
9. The ability to perform data type and the character-set translation is a requirement when moving data between incompatible systems.

10. The ability to create aggregation, summarization and derivation fields and records are necessary.
11. Vendor stability and support for the products are components that must be evaluated carefully.

13.1 Data Warehouse Software Components

A warehousing team will require different types of tools during a warehouse project. These software products usually fall into one or more of the categories illustrated, as shown in the figure.



Data Warehouse Software Components

Extraction and Transformation

The warehouse team needs tools that can extract, transform, integrate, clean, and load information from a source system into one or more data warehouse databases. Middleware and gateway products may be needed for warehouses that extract a record from a host-based source system.

Warehouse Storage

Software products are also needed to store warehouse data and their accompanying metadata. Relational database management systems are well suited to large and growing warehouses.

Data access and retrieval

Different types of software are needed to access, retrieve, distribute, and present warehouse data to its end-clients.