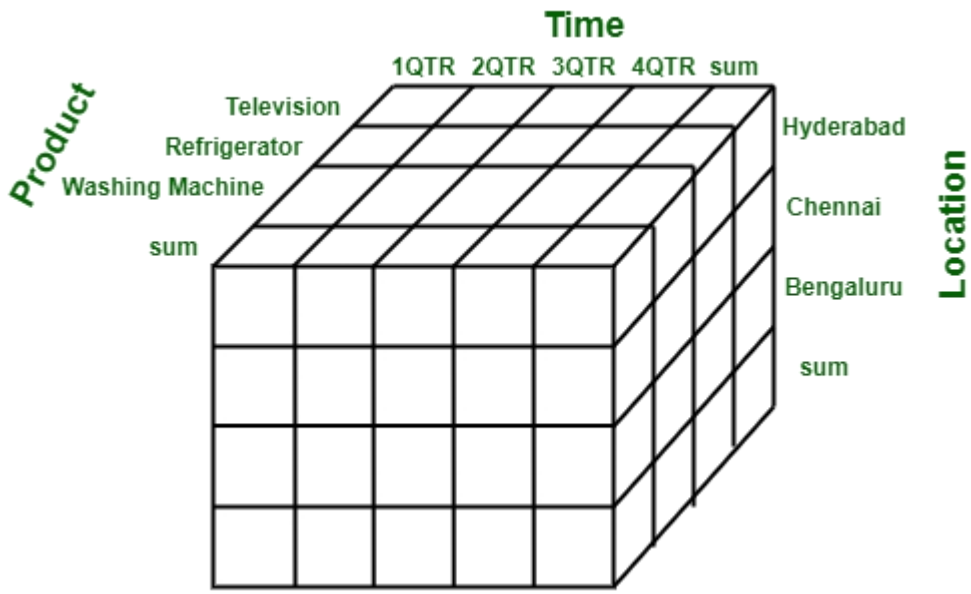
MultiDimensional Data Model

The multi-Dimensional Data Model is a method which is used for ordering data in the database along with good arrangement and assembling of the contents in the database.

The Multi Dimensional Data Model allows customers to interrogate analytical questions associated with market or business trends, unlike relational databases which allow customers to access data in the form of queries. They allow users to rapidly receive answers to the requests which they made by creating and examining the data comparatively fast.

OLAP (online analytical processing) and data warehousing uses multi dimensional databases. It is used to show multiple dimensions of the data to users.

It represents data in the form of data cubes. Data cubes allow to model and view the data from many dimensions and perspectives. It is defined by dimensions and facts and is represented by a fact table. Facts are numerical measures and fact tables contain measures of the related dimensional tables or names of the facts.



Working on a Multidimensional Data Model

On the basis of the pre-decided steps, the Multidimensional Data Model works.

The following stages should be followed by every project for building a Multi Dimensional Data Model :

Stage 1 : Assembling data from the client : In first stage, a Multi Dimensional Data Model collects correct data from the client. Mostly, software professionals

provide simplicity to the client about the range of data which can be gained with the selected technology and collect the complete data in detail.

Stage 2 : Grouping different segments of the system : In the second stage, the Multi Dimensional Data Model recognizes and classifies all the data to the respective section they belong to and also builds it problem-free to apply step by step.

Stage 3 : Noticing the different proportions : In the third stage, it is the basis on which the design of the system is based. In this stage, the main factors are recognized according to the user's point of view. These factors are also known as "Dimensions".

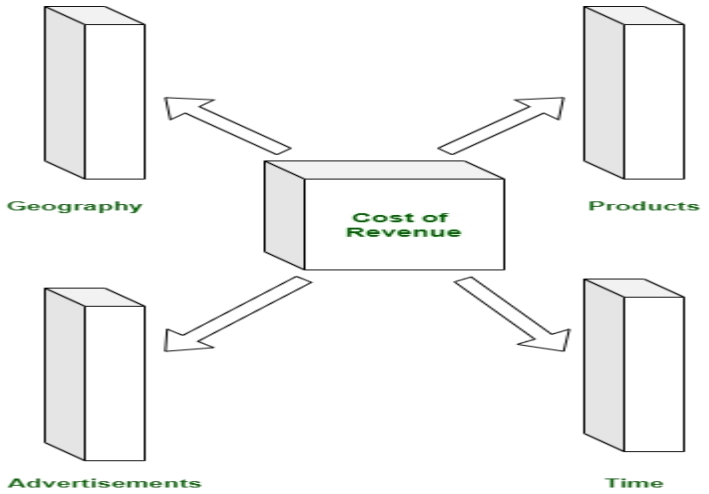
Stage 4 : Preparing the actual-time factors and their respective qualities : In the fourth stage, the factors which are recognized in the previous step are used further for identifying the related qualities. These qualities are also known as **"attributes"** in the database.

Stage 5 : Finding the actuality of factors which are listed previously and their qualities : In the fifth stage, A Multi Dimensional Data Model separates and differentiates the actuality from the factors which are collected by it. These actually play a significant role in the arrangement of a Multi Dimensional Data Model.

Stage 6 : Building the Schema to place the data, with respect to the information collected from the steps above : In the sixth stage, on the basis of the data which was collected previously, a Schema is built.

For Example :

1. Let us take the example of a firm. The revenue cost of a firm can be recognized on the basis of different factors such as geographical location of firm's workplace, products of the firm, advertisements done, time utilized to flourish a product, etc.



Example 1

2. Let us take the example of the data of a factory which sells products per quarter in Bangalore. The data is represented in the table given below :

Location = "Bangalore"				
Time (quarter)	Type of item			
	Jam	Bread	Sugar	Milk
Q1	350	389	35	50
Q2	260	528	50	90
Q3	483	256	20	60
Q4	436	396	15	40

2D factory data

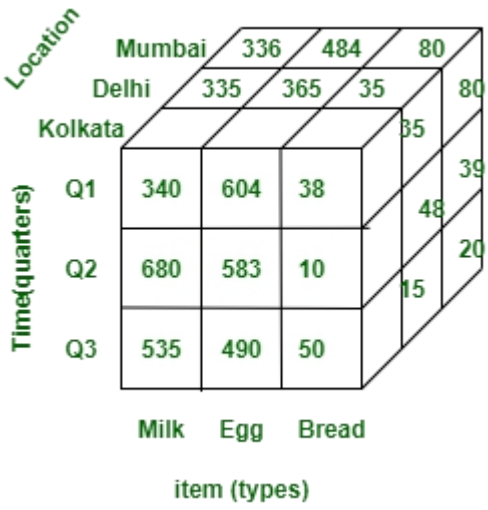
In the above given presentation, the factory's sales for Bangalore are, for the time dimension, which is organized into quarters and the dimension of items, which is sorted according to the kind of item which is sold. The facts here are represented in rupees (in thousands).

Now, if we desire to view the data of the sales in a three-dimensional **table**, then it is represented in the diagram given below. Here the data of the sales is represented as a two **dimensional table**. Let us consider the data according to item, time and location (like Kolkata, Delhi, Mumbai). Here is the table :

	Location="Kolkata"			Location="Delhi"			Location="Mumbai"		
	item			item			item		
Time	Milk	Egg	Bread	Milk	Egg	Bread	Milk	Egg	Bread
Q1	340	604	38	335	365	35	336	484	80
Q2	680	583	10	684	490	48	595	594	39
Q3	535	490	50	389	385	15	366	385	20

3D data representation as 2D

This data can be represented in the form of three dimensions conceptually, which is shown in the image below :



3D data representation

Advantages of Multi Dimensional Data Model

The following are the advantages of a multi-dimensional data model :

- A multi-dimensional data model is easy to handle.
- It is easy to maintain.
- Its performance is better than that of normal databases (e.g. relational databases).
- The representation of data is better than traditional databases. That is because the multi-dimensional databases are multi-viewed and carry different types of factors.
- It is workable on complex systems and applications, contrary to the simple one-dimensional database systems.
- The compatibility in this type of database is an upliftment for projects having lower bandwidth for maintenance staff.

Disadvantages of Multi Dimensional Data Model

The following are the disadvantages of a Multi Dimensional Data Model :

- The multi-dimensional Data Model is slightly complicated in nature and it requires professionals to recognize and examine the data in the database.
- During the work of a Multi-Dimensional Data Model, when the system caches, there is a great effect on the working of the system.
- It is complicated in nature due to which the databases are generally dynamic in design.
- The path to achieving the end product is complicated most of the time.

As the Multi Dimensional Data Model has complicated systems, databases have a large number of databases due to which the system is very insecure when there is a security break.

Data Warehousing - Schemas

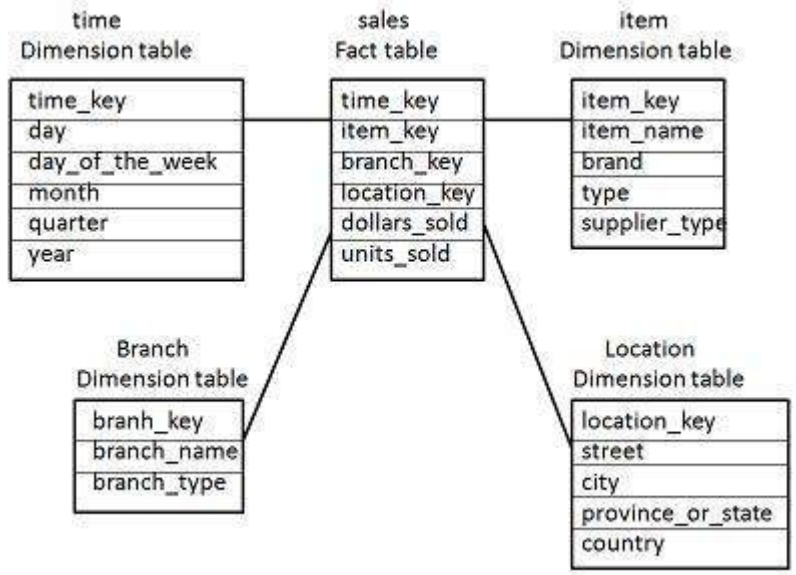
Schema is a logical description of the entire database. It includes the name and description of records of all record types including all associated data-items and aggregates. Much like a database, a data warehouse also requires to maintain a schema. A database uses relational model, while a data warehouse uses Star, Snowflake, and Fact Constellation schema. In this chapter, we will discuss the schemas used in a data warehouse.

Star Schema

Each dimension in a star schema is represented with only one-dimension table.

This dimension table contains the set of attributes.

The following diagram shows the sales data of a company with respect to the four dimensions, namely time, item, branch, and location.



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

The fact table also contains the attributes, namely dollars sold and units sold.

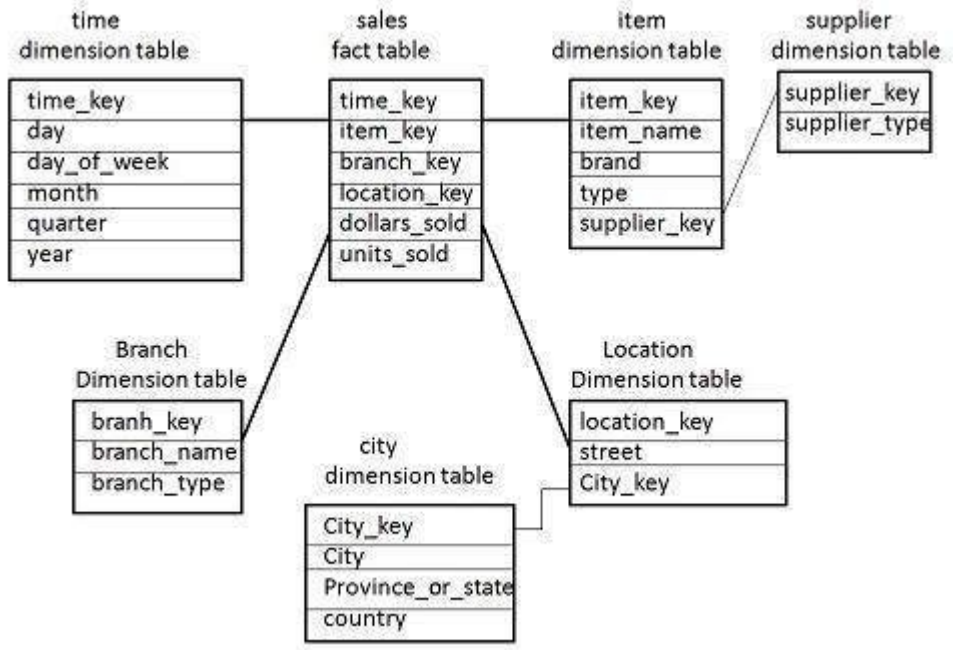
Note – Each dimension has only one dimension table and each table holds a set of attributes. For example, the location dimension table contains the attribute set {location_key, street, city, province_or_state,country}. This constraint may cause data redundancy. For example, 'Vancouver' and 'Victoria' both the cities are in the Canadian province of British Columbia. The entries for such cities may cause data redundancy along the attributes province_or_state and country.

Snowflake Schema

Some dimension tables in the Snowflake schema are normalized.

The normalization splits up the data into additional tables.

Unlike Star schema, the dimensions table in a snowflake schema are normalized. For example, the item dimension table in star schema is normalized and split into two dimension tables, namely item and supplier table.



Now the item dimension table contains the attributes item_key, item_name, type, brand, and supplier-key.

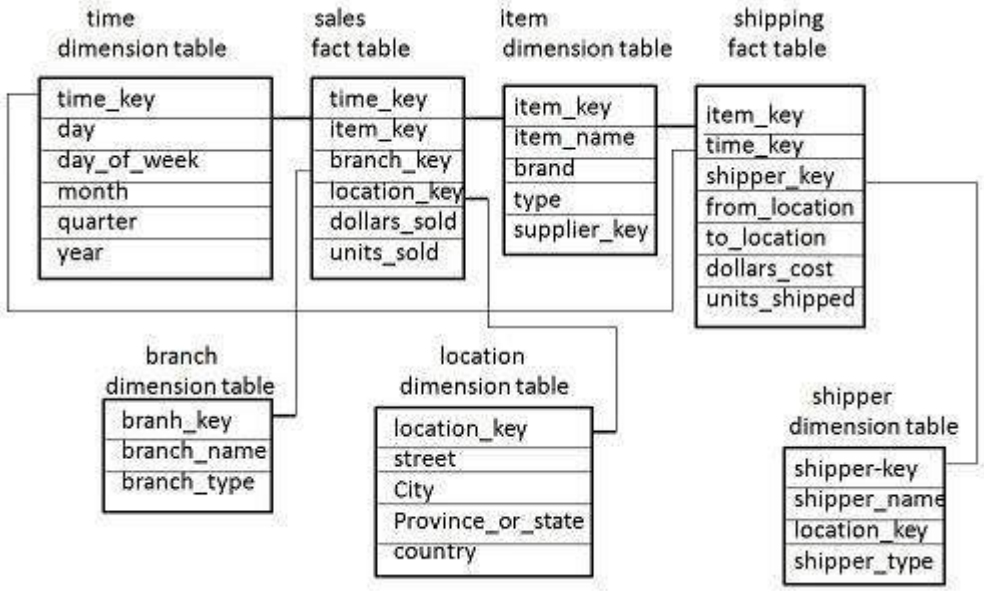
The supplier key is linked to the supplier dimension table. The supplier dimension table contains the attributes supplier_key and supplier_type.

Note – Due to normalization in the Snowflake schema, the redundancy is reduced and therefore, it becomes easy to maintain and the save storage space.

Fact Constellation Schema

A fact constellation has multiple fact tables. It is also known as galaxy schema.

The following diagram shows two fact tables, namely sales and shipping.



The sales fact table is same as that in the star schema.

The shipping fact table has the five dimensions, namely item_key, time_key, shipper_key, from_location, to_location.

The shipping fact table also contains two measures, namely dollars sold and units sold.

It is also possible to share dimension tables between fact tables. For example, time, item, and location dimension tables are shared between the sales and shipping fact table.

Schema Definition

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

Syntax for Cube Definition

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

Syntax for Dimension Definition

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

Star Schema Definition

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

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

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

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

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,
country))
```

Fact Constellation Schema Definition

Fact constellation schema can be defined using DMQL as follows −

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

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

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

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

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