18CSE487T - DATA WAREHOUSING AND ITS APPLICATIONS Unit-2

Data Warehouse Schema- Introduction

Dimensional Modeling

The Star Schema

The Snowflake Schema

Aggregate Tables

DBMS Schemas for Decision Support

Data Extraction

Data transformation: Basic tasks

Major transformation types

OLAP definition,

Dimensional Analysis

Hypercube

OLAP operations

Drill down

Roll up

Slice

OLAP models

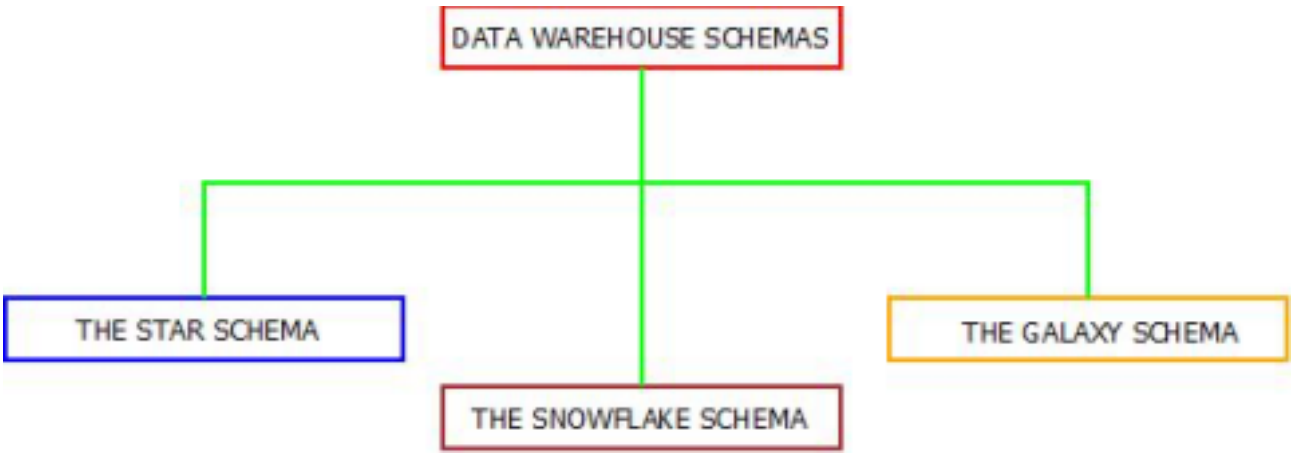
MOLAP

Data Warehouse Schema- Introduction

- Schema in general terms means “Logical Structure”. So, data warehouse schema describes the logical structure of any data warehouse containing records.
- Also, the concept behind schema of data warehouse is same as that in data bases. Relational data models are used by data bases for their logical structure while data warehouses uses schema for the same purpose.
- The schema in data warehouses are used to get the knowledge of complexity of a structure of data warehouse.
- They are basically the representation of the outer model or the way to logically deduce the results

from the figure and these figures are made from combinations of fact tables and dimension tables.

- The conceptual modelling of warehouse comprises of three models. These are:-



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.

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.

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.

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.

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

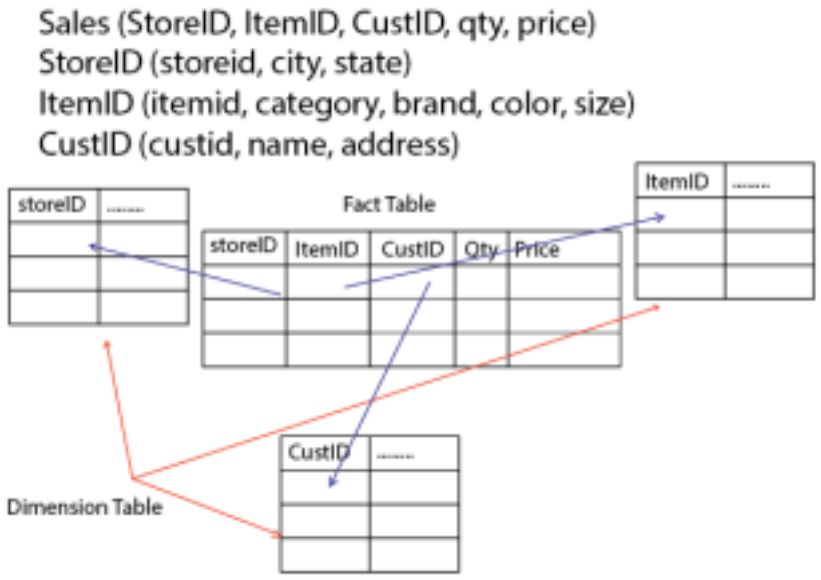
Dimension tables establish the context of the facts. Dimensional tables store fields that describe the facts.

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.

The 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

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.

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.

Characteristics of Star Schema

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

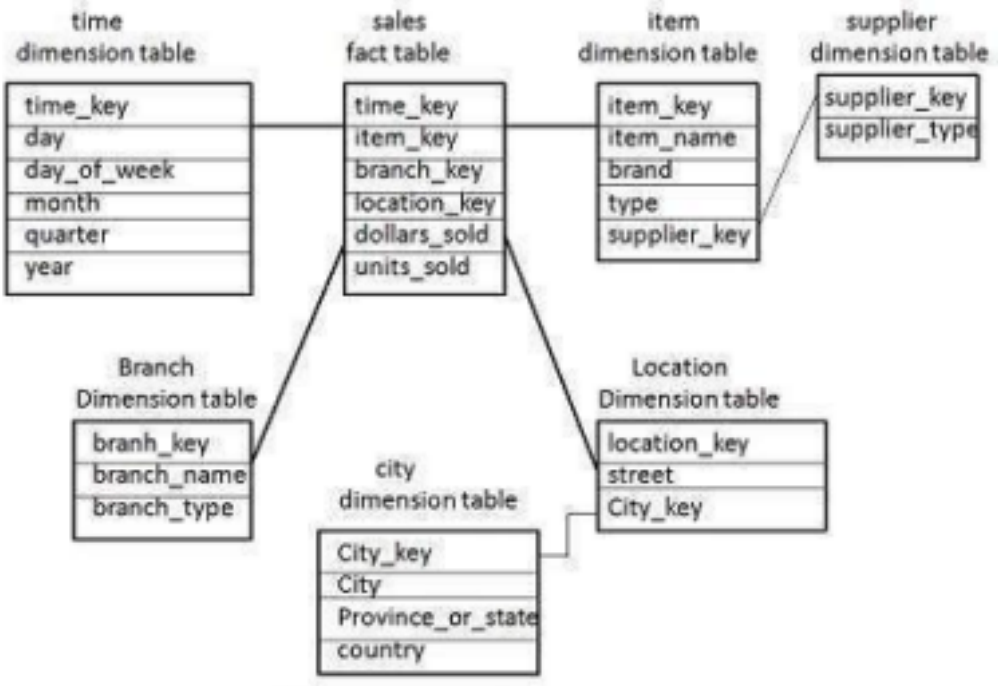
- It creates a DE-normalized database that can quickly provide query responses.
- It provides a flexible design that can be changed easily or added to throughout the development cycle, and as the database grows.
- It provides a parallel in design to how end-users typically think of and use the data.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

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 Snowflake Schema

Some dimension tables in the Snowflake schema are normalized. The normalization splits up the data into additional tables as shown in the following illustration.



For example − The item dimension table in a 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 the normalization in the Snowflake schema, the redundancy is reduced and therefore, it becomes easy to maintain and the save storage space.

Aggregate tables

Aggregate tables, in general, are simply database tables that contain aggregated values. Imagine that you have a fact table like this in which the granularity is date, product and customer:

Customer ID	Item No	Order Date	Unit Sales	Profit
1	143	04/05/2012	1	1.52
1	150	04/05/2012	3	3.9
1	8	10/06/2012	1	2.48
1	77	10/06/2012	1	1.37
1	92	10/06/2012	1	1.33

1	95	10/06/2012	1	2.87
1	18	28/07/2012	1	2.3
1	37	28/07/2012	1	1.03
1	61	28/07/2012	1	2.01
1	83	28/07/2012	1	2.37
1	120	28/07/2012	1	2.24
1	8	13/08/2012	1	2.48
1	58	13/08/2012	1	2.79
1	100	16/12/2012	1	2.18
1	122	16/12/2012	1	2.47
1	83	12/05/2013	1	2.37
1	148	12/05/2013	1	2.17
1	37	29/11/2013	1	1.03
1	116	29/11/2013	1	2.03
2	46	07/04/2012	1	2.24
2	110	07/04/2012	1	2.84
2	49	06/05/2012	1	2.2
2	60	06/05/2012	1	1.91
2	26	22/05/2012	1	1.69

Every value that you see for "unit sales" is the number of units of a particular product that were sold to a particular customer on a particular day. (Here, to help make the point of aggregation slightly clearer, I've shown the day of sale as an actual date rather than as a pointer. And yes, I realize that these European-style dates are still in the future – but we'll treat them as the past for purposes of this demonstration.)

We could run queries against this fact table, and it would return data. For example, we could ask for the total unit sales of Item No. 150 to Customer 1 on May 4, 2012, and the answer we'd get is three.

We could also run a query that returns the total sales of Item No. 150 to the same customer not just

for the 4th of May but for that entire month. In order to do this, the system would scan the fact table looking for 31 separate date entries listing sales of that product to Customer 1 and then aggregate the unit-sale values from the returned rows. That would be a relatively slow process.

An alternative is to create a fact table that already contains one or more levels of aggregation. For example, we could aggregate this fact table by month. That would involve finding all the sales of Item No. 150 to Customer 1 in, say, January 2012, aggregating the data, and putting the results in a single row. We then, of course, would have to do the same for the other months, the other products and the other customers.

The new table would look like this:

Customer ID	Item No	Order date	Unit Sales	Profit
1	150	Jan 2012	11	11.52
1	150	Feb 2012	3	3.9
1	150	March 2012	2	2.48
1	150	April 2012	1	1.37
1	150	May 2012	1	1.33
1	150	June 2012	2	2.87

Now when we had a query that looked for a monthly sales total, we could run it against the aggregate fact table and find the answer much more rapidly. And of course, this isn't the only aggregate fact table that we could generate. For example, if we knew the counties that different customers were located in, we could aggregate their data on that basis.

We could also aggregate by both date and customer up to the level of the month and county, as in this table excerpt (with Herefordshire being a county in England, for the uninitiated):

Customer ID	Item No	Order date	Unit Sales	Profit
Herefordshire	150	Jan 2012	111	121.52
Herefordshire	150	Feb 2010	63	73.9

In other words, we could create a series of aggregate fact tables, and when a query comes in, we could run it against the appropriate aggregated table.

Data Extraction

Data extraction is the process of collecting or retrieving disparate types of data from a variety of sources, many of which may be poorly organized or completely unstructured. Data extraction makes it possible to consolidate, process, and refine data so that it can be stored in a centralized location in order to be transformed. These locations may be on-site, cloud-based, or a hybrid of the two.

Data extraction is the first step in both ETL (extract, transform, load) and ELT (extract, load, transform) processes. ETL/ELT are themselves part of a complete data integration strategy.

Data Extraction and ETL

To put the importance of data extraction in context, it's helpful to briefly consider the ETL process as a whole. In essence, ETL allows companies and organizations to 1) consolidate data from different sources into a centralized location and

2) assimilate different types of data into a common format.

There are three steps in the ETL process:

1. Extraction: Data is taken from one or more sources or systems. The extraction locates and identifies relevant data, then prepares it for processing or transformation. Extraction allows many different kinds of data to be combined and ultimately mined for business intelligence.
2. Transformation: Once the data has been successfully extracted, it is ready to be refined. During the transformation phase, data is sorted, organized, and cleansed. For example, duplicate entries will be deleted, missing values removed or enriched, and audits will be performed to produce data that is reliable, consistent, and usable.
3. Loading: The transformed, high quality data is then delivered to a single, unified target location for storage and analysis.

Data Extraction without ETL

Can data extraction take place outside of ETL? The short answer is yes. However, it's important to keep in mind the limitations of data extraction outside of a more complete data integration process. Raw data which is extracted but not transformed or loaded properly will likely be difficult to organize or analyze, and may be incompatible with newer programs and applications. As a result,

the data may be useful for archival purposes, but little else. If you're planning to move data from a legacy databases into a newer or cloud-native system, you'll be better off extracting your data with a complete data integration tool.

Another consequence of extracting data as a stand alone process will be sacrificing efficiency, especially if you're planning to execute the extraction manually. Hand-coding can be a painstaking process that is prone to errors and difficult to replicate across multiple extractions. In other words, the code itself may have to be rebuilt from scratch each time an extraction takes place.

Benefits of Using an Extraction Tool

Companies and organizations in virtually every industry and sector will need to extract data at some point. For some, the need will arise when it's time to upgrade legacy databases or transition to cloud-native storage. For others, the motive may be the desire to consolidate databases after a merger or acquisition. It's also common for companies to want to streamline internal processes by merging data sources from different divisions or departments.

If the prospect of extracting data sounds like a daunting task, it doesn't have to be. In fact, most companies and organizations now take advantage of data extraction tools to manage the extraction process from end-to-end. Using an ETL tool automates and simplifies the extraction process so that resources can be deployed toward other priorities. The benefits of using a data extraction tool include:

- **More control.** Data extraction allows companies to migrate data from outside sources into their own databases. As a result, you can avoid having your data siloed by outdated applications or software licenses. It's your data, and extraction let's you do what you want with it.
- **Increased agility.** As companies grow, they often find themselves working with different types of data in separate systems. Data extraction allows you to consolidate that information into a centralized system in order to unify multiple data sets.
- **Simplified sharing.** For organizations who want to share some, but not all, of their data with external partners, data extraction can be an easy way to provide helpful but limited data access. Extraction also allows you to share data in a common, usable format.
- **Accuracy and precision.** Manual processes and hand-coding increase opportunities for errors, and the requirements of entering, editing, and re-enter large volumes of data take their toll on data integrity. Data extraction automates processes to reduce errors and avoid time spent on resolving

them.

Types of Data Extraction

Data extraction is a powerful and adaptable process that can help you gather many types of information relevant to your business. The first step in putting data extraction to work for you is to identify the kinds of data you'll need. Types of data that are commonly extracted include: • **Customer Data**: This is the kind of data that helps businesses and organizations understand their customers and donors. It can include names, phone numbers, email addresses, unique identifying numbers, purchase histories, social media activity, and web searches, to name a few. • **Financial Data**: These types of metrics include sales numbers, purchasing costs, operating margins, and even your competitor's prices. This type of data helps companies track performance, improve efficiencies, and plan strategically.

- **Use, Task, or Process Performance Data**: This broad category of data includes information related to specific tasks or operations. For example, a retail company may seek information on its shipping logistics, or a hospital may want to monitor post-surgical outcomes or patient feedback.

Once you've decided on the type of information you want to access and analyze, the next steps are 1) **figuring out where you can get it** and 2) **deciding where you want to store it**. In most cases, that means moving data from one application, program, or server into another. A typical migration might involve data from services such as SAP, Workday, Amazon Web Services, MySQL, SQL Server, JSON, SalesForce, Azure, or Google Cloud. These are some examples of widely used applications, but data from virtually any program, application, or server can be migrated.

Data transformation: Basic tasks

Data Transformation

Data transformation is the process in which data gets converted from one format to another. The most common data transformation process involves collecting raw data and converting it into clean, usable data.

Data transformation increases the efficiency of business and analytic processes, and it enables businesses to make better data-driven decisions. During the data transformation process, an analyst will determine the structure of the data. This could mean that data transformation may be:

- **Constructive**: The data transformation process adds, copies, or replicates data.

• **Destructive**: The system deletes fields or records.

• **Aesthetic**: The transformation standardizes the data to meet requirements or parameters. • **Structural**: The database is reorganized by renaming, moving, or combining columns. They'll also perform data mapping and extract the data from its original source before executing the transformation. Finally, they'll store the transformed data within the appropriate database technology.

Basic Tasks

The major tasks performed during this phase vary depending on the application; however the basic tasks are discussed here.

Selection: This takes place at the beginning of the whole process of data transformation. You select either whole records or parts of several records from the source systems. The task of selection usually forms part of the extraction function itself. However, in some cases, the composition of the source structure may not be supporting selection of the necessary parts during data extraction. In these cases, it is advised to extract the whole record and then do the selection as part of the transformation function.

Splitting/joining: This task includes the types of data manipulation you need to perform on the selected parts of source records. Sometimes (uncommonly), you will be splitting the selected parts even further during data transformation. Joining of parts selected from many source systems is more widespread in the data warehouse environment.

Conversion: This is an all-inclusive task. It includes a large variety of rudimentary conversions of single fields for two primary reasons (i) to standardize among the data extractions from disparate source systems, and (ii) to make the fields usable and understandable to the users. **Summarization:** Sometimes you may find that it is not feasible to keep data at the lowest level of detail in your data warehouse. It may be that none of your users ever need data at the lowest granularity for analysis or querying. For example, for a grocery chain, sales data at the lowest level of detail for every transaction at the checkout may not be needed. Storing sales by product by store by day in the data warehouse may be quite adequate. So, in this case, the data transformation function includes summarization of daily sales by product and by store.

Enrichment: This task is the rearrangement and simplification of individual fields to make them more useful for the data warehouse environment. You may use one or more fields from the same input record to create a better view of the data for the data warehouse. This principle is extended

when one or more fields originate from multiple records, resulting in a single field for the data warehouse.

To better understand lets discuss conversion and enrichment with examples.

Data Transformation Basic Tasks: Conversion (example)

Data representation change

EBCIDIC to ASCII

Operating System Change

Mainframe (MVS) to UNIX

UNIX to NT or XP

Data type change

Program (Excel to Access), database format (FoxPro to Access).

Character, numeric and date type.

Fixed and variable length

Major Transformation Types

1. Format Revision

Format revisions fix problems that stem from fields having different data types. Some fields might be numeric, and others might be text. One data system could treat text versus numeric information differently, so you might have to standardize the formats to integrate source data with the target data schema. This could involve the conversion of male/female, date/time, measurements, and other information into a consistent format.

Field lengths can also be an issue—especially if the target schema has smaller character limits. In these cases, it may be necessary to standardize the length of fields by breaking up long serial numbers into smaller parts and putting them into separate columns.

Additionally, format revision could involve splitting up a comma-separated list of words or numbers into multiple columns.

2. Data Derivation

Data derivation involves the creation of special rules to “derive” the specific information you want from the data source. For example, you might have a database that includes total revenue data from sales, but you’re only interested in loading the profit figures after subtracting costs and tax liabilities. Data derivation allows you to create a set of transformation rules that subtract costs and taxes from the total revenue information.

3. Data Splitting

Data splitting refers to dividing a single column into multiple columns. This is critical for analyzing the available data; splitting the single column into multiple columns can be useful to develop "training" and "testing" sets, for example. The "training" gets used for experimental analysis and making models, while the "testing" set is the untouched, "control" element. Data splitting can be helpful with a large amount of data gathered over a significant amount of time.

4. Data Joining

Joining data is one of the most important functions of data transformation. A “join” is an operation in the SQL database language that allows you to connect two or more database tables by their matching columns. This allows you to establish a relationship between multiple tables, which merges table data together so you can query correlating data on the tables.

5. Data Summarization

It refers to the creation of different business metrics through the calculation of value totals. You could sum up the total revenue of all the sales made by the individual salespeople on your staff, then create sales metrics that reveal total sales for individual time periods.

6. Key Restructuring

When the tables in a data warehouse have keys with built-in meanings, serious problems can develop. For example, if a client phone number serves as a primary key, changing the phone number in the original data source means that the number would have to change everywhere it appears in the data system. That would cause a cascade of updates that over-burden or slow down the system.

Through key restructuring, you can transform any keys with built-in meanings to generic keys—i.e., random numbers that reference back to the source database with the actual information. By drawing key connections from one table to another, key restructuring optimizes the data warehouse for speed and efficiency.

7. Data Deduplication

Data deduplication is a data compression process where you identify and remove duplicate or

repeated copies of information. Also referred to as single-instance storage, intelligent compression, commonality factoring, or data reduction, deduplication allows you to store one unique copy of data in your data warehouse or database.

The deduplication process analyzes incoming data and compares it to data that's already stored in the system. If the data is already there, deduplication algorithms delete the duplicate information while creating a reference to it. If you upload a changed version of a previous file, the system will back up a said file while adding the changes to the data segment. Deduplication algorithms also keep track of outgoing data to delete duplicates, which speeds up the information transfer process.

OLAP

OLAP stands for **On-Line Analytical Processing**. OLAP is a classification of software technology which authorizes analysts, managers, and executives to gain insight into information through fast, consistent, interactive access in a wide variety of possible views of data that has been transformed from raw information to reflect the real dimensionality of the enterprise as understood by the clients.

OLAP implement the multidimensional analysis of business information and support the capability for complex estimations, trend analysis, and sophisticated data modeling. It is rapidly enhancing the essential foundation for Intelligent Solutions containing Business Performance Management, Planning, Budgeting, Forecasting, Financial Documenting, Analysis, Simulation Models, Knowledge Discovery, and Data Warehouses Reporting. OLAP enables end-clients to perform ad hoc analysis of record in multiple dimensions, providing the insight and understanding they require for better decision making.

Who uses OLAP and Why?

OLAP applications are used by a variety of the functions of an organization.

Finance and accounting:

- Budgeting
- Activity-based costing
- Financial performance analysis
- And financial modeling

Sales and Marketing

- Sales analysis and forecasting

- Market research analysis
- Promotion analysis
- Customer analysis
- Market and customer segmentation

Production

- Production planning
- Defect analysis

OLAP cubes have two main purposes. The first is to provide business users with a data model more intuitive to them than a tabular model. This model is called a Dimensional Model. The second purpose is to enable fast query response that is usually difficult to achieve using tabular models.

How OLAP Works?

Fundamentally, OLAP has a very simple concept. It pre-calculates most of the queries that are typically very hard to execute over tabular databases, namely aggregation, joining, and grouping. These queries are calculated during a process that is usually called 'building' or 'processing' of the OLAP cube. This process happens overnight, and by the time end users get to work - data will have been updated.

DIMENSIONAL ANALYSIS

One approach to data warehouse design is to develop and implement a dimensional model. This has given rise to dimensional analysis (sometimes generalized as multi-dimensional analysis). It was noticed quite early on when data warehouses started to be developed that, whenever decision makers were asked to describe the kinds of questions they would like to get answers to regarding their organizations, they almost always wanted the following:

- Summarized information with the ability to break the summaries into more detail
- Analysis of the summarized information across their own organizational components such as departments or regions
- Ability to slice and dice the information in any way they chose
- Display of the information in both graphical and tabular form
- Capability to view their information over time

So as an example, they might wish to see a report showing Wine Sales by Product, or a report showing Sales by Customer, or even Sales by Product by Customer

Table shows a typical report of sales by product.

Table The Wine Club ” Analysis of Sales by Product for July.

Product Name	Quantity Sold (Cases)	Cost Price	Selling Price	Total Revenue	Total Cost	Gross Profit
Chianti	321	26.63	42.95	13,787	8,548	5,239
Bardolino	1,775	15.10	31.35	55,646	26,802	28,844
Barolo	275	46.72	70.95	19,511	12,848	6,663
Lambrusco	1,105	23.25	41.45	45,802	25,691	20,111
Valpolicella	2,475	12.88	32.45	80,313	31,878	48,435

This dimensional approach led Ted Codd to make the following observation: *There are typically a number of different dimensions from which a given pool of data can be analyzed . This plural perspective, or multidimensional conceptual view, appears to be the way most business persons naturally view their enterprise.*

”E. F. Codd, 1993

So the concept of dimensional analysis became a method for defining data warehouses. The approach is to determine, by interviewing the appropriate decision makers in an organization, which is the *subject area* that they are most interested in, and which are the most important *dimensions of analysis.*

Recall that one of the characteristics of a data warehouse is that it is subject oriented. The subject area reflects the subject-oriented nature of the warehouse.

In the example above, the subject area would be Sales. The dimensions of analysis would be Customers and Products. The requirement is to analyze sales by customer and sales by product. This requirement is depicted in the following three-dimensional cube. Figure shows Sales (the shaded area) having axes of:

1. Customer
2. Product

3. Time

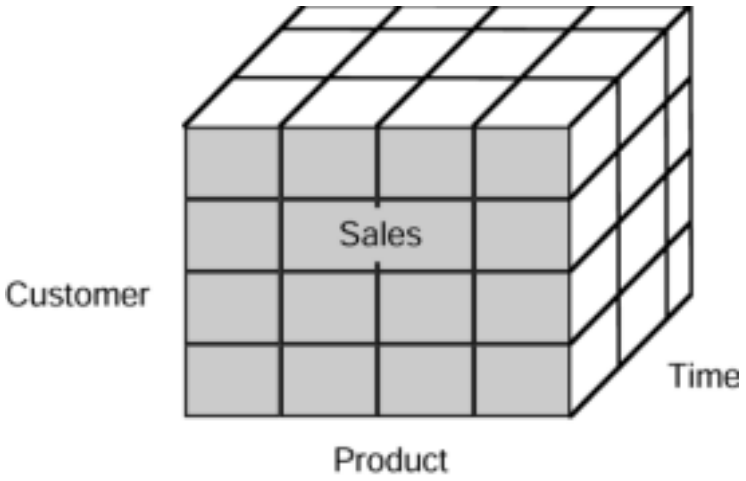


Figure Wine sales dimensional model for the Wine Club.

Notice that **time** has not been examined so far. Time is regarded as a necessary dimension of analysis (recall that time variance is another characteristic of data warehouses) and so is always included as one of the dimensions of analysis.

This means that Sales can be analyzed by Customer by Product over Time. So each element of the cube (each minicube) contains a value for sales to a particular customer, of a particular product, at a particular point in time.

The multidimensional cube in Figure shows sales as the subject with three dimensions of analysis. There is no real limit to the number of dimensions that can used in a dimensional model, although there is, of course, a limit to the number of dimensions we can draw! Now let's return to the Wine Club.

The directors of The Wine Club need answers to questions about sales. Looking back at the five example questions, they all concerned sales: Sales by Product, Sales by Customer, Sales by Area. As in the example above, the subject area for their data warehouse is clearly Sales. So what are the dimensions of analysis? Well, we've just mentioned three:

1. Product
2. Customer
3. Area

So is that it? Not quite; we must not forget the Time dimension. So now we have it, a subject area and four dimensions of analysis.

As we cannot draw four-dimensional models, we can represent the conceptual dimensional model as shown in Figure.

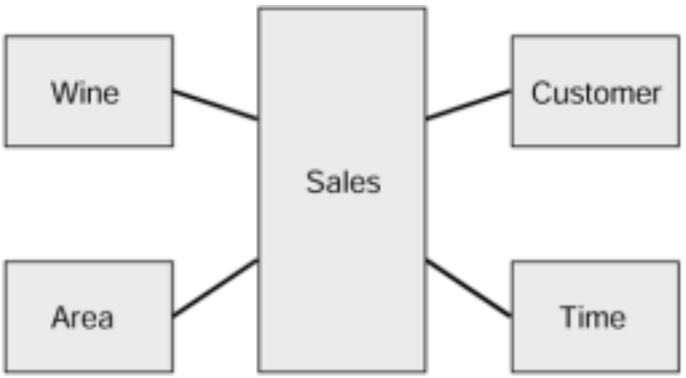
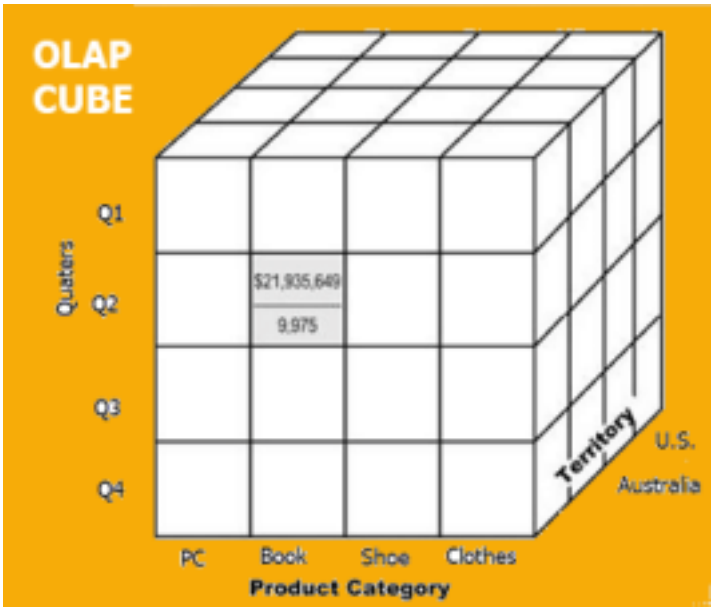


Figure Wine sales dimensional model for the Wine Club.

The diagram in Figure is often referred to as a *Star Schema* because the diagram loosely resembles a star shape. The subject area is the center of the star and the dimensions of analysis form the points of the star. The subject area is often drawn long and thin because the table itself is usually long and thin in that it contains a small number of columns but a very large number of rows. The Star Schema is the most commonly used diagram for dimensional models.

Hypercube

At the core of the OLAP concept, is an OLAP Cube. The OLAP cube is a data structure optimized for very quick data analysis.



The OLAP Cube consists of numeric facts called measures which are categorized by dimensions. OLAP Cube is also called the **hypercube**.

Usually, data operations and analysis are performed using the simple spreadsheet, where data values are arranged in row and column format. This is ideal for two-dimensional data. However,

OLAP contains multidimensional data, with data usually obtained from a different and unrelated source. Using a spreadsheet is not an optimal option. The cube can store and analyze multidimensional data in a logical and orderly manner.

How does it work?

A Data warehouse would extract information from multiple data sources and formats like text files, excel sheet, multimedia files, etc.

The extracted data is cleaned and transformed. Data is loaded into an OLAP server (or OLAP cube) where information is pre-calculated in advance for further analysis.

OLAP Operations

OLAP helps in analyzing different extracts and in viewing business data from different points of view. It is often required to group aggregate and join data. The structure is basically called the OLAP cube. The OLAP cube is a data structure that is optimized for proper data analysis. It mainly consists of numeric facts which can be called dimensions, at the same time where the OLAP cubes are termed as 'Hyper cubes', which will allow the user to perform Multidimensional Analytical querying for the required data using the basic OLAP operations such as Drill-down, Roll-up, Slicing, Dicing and Pivot.

Types of Operations on OLAP

There are four types of OLAP operations that can be performed. These areas below:

1. Roll up
2. Drill down
3. Slice and dice
4. Pivot

Roll-Up

The roll-up operation **(also known as drill-up or aggregation operation)** performs aggregation on a data cube, by climbing down concept hierarchies, i.e., dimension reduction. Roll up is like **zooming-out** on the data cubes. Figure shows the result of roll-up operations performed on the dimension location. The hierarchy for the location is defined as the Order Street, city, province, or

state, country. The roll-up operation aggregates the data by ascending the location hierarchy from the level of the city to the level of the country.

When a roll-up is performed by dimensions reduction, one or more dimensions are removed from the cube. For example, consider a sales data cube having two dimensions, location and time. Roll-up may be performed by removing, the time dimensions, appearing in an aggregation of the total sales by location, relatively than by location and by time.

Example

Consider the following cubes illustrating temperature of certain days recorded weekly:

Temperature	64	65	68	69	70	71	72	75	80	81	83	85
Week1	1	0	1	0	1	0	0	0	0	0	1	0
Week2	0	0	0	1	0	0	1	2	0	1	0	0

Consider that we want to set up levels (hot (80-85), mild (70-75), cool (64-69)) in temperature from the above cubes. To do this, we have to group column and add up the value according to the concept hierarchies. This operation is known as a roll-up.

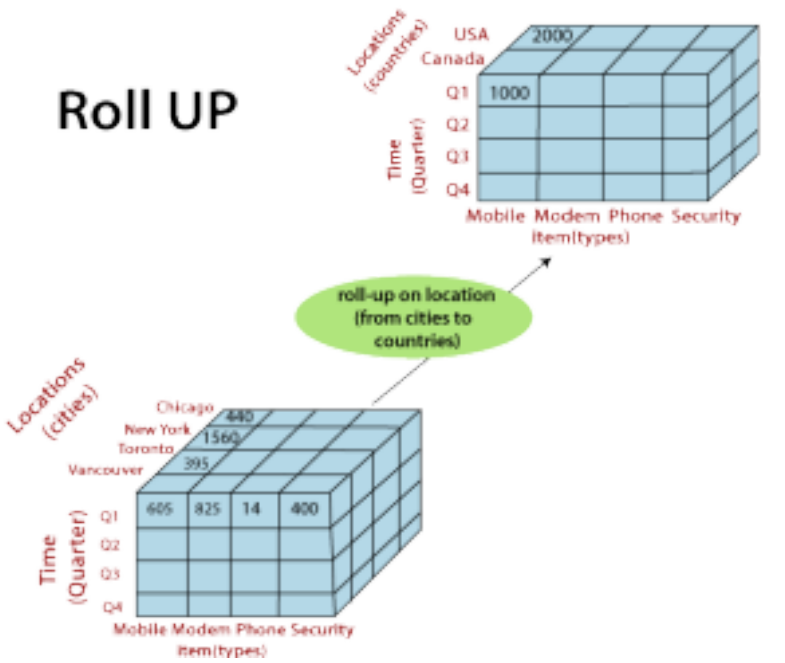
By doing this, we contain the following cube:

Temperature	**cool**	**mild**	**hot**
Week1	2	1	1
Week2	2	1	1

The roll-up operation groups the information by levels of temperature.

The following diagram illustrates how roll-up works.

The figure shows data cubes for sales of a shop. The cube contains the dimensions, location, and time and item, where the **location** is aggregated with regard to city values, **time** is aggregated with respect to quarters, and an **item** is aggregated with respect to item types.



Drill-Down

The drill-down operation **(also called roll-down)** is the reverse operation of **roll-up**. Drill down is like **zooming-in** on the data cube. It navigates from less detailed record to more detailed data. Drill-down can be performed by either **stepping down** a concept hierarchy for a dimension or adding additional dimensions.

Figure shows a drill-down operation performed on the dimension time by stepping down a concept hierarchy which is defined as day, month, quarter, and year. Drill-down appears by descending the time hierarchy from the level of the quarter to a more detailed level of the month. Because a drill-down adds more details to the given data, it can also be performed by adding a new dimension to a cube. For example, a drill-down on the central cubes of the figure can occur by introducing an additional dimension, such as a customer group.

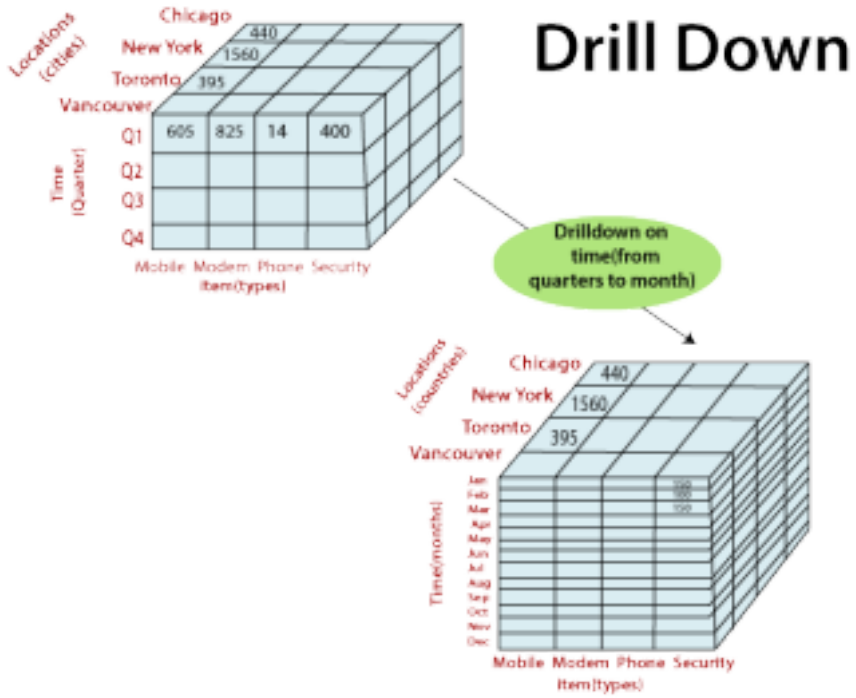
Example

Drill-down adds more details to the given data

Temperature	**cool**	**mild**	**hot**
Day 1	0	0	0
Day 2	0	0	0

Day 3	0	0	1
Day 4	0	1	0
Day 5	1	0	0
Day 6	0	0	0
Day 7	1	0	0
Day 8	0	0	0
Day 9	1	0	0
Day 10	0	1	0
Day 11	0	1	0
Day 12	0	1	0
Day 13	0	0	1
Day 14	0	0	0

The following diagram illustrates how Drill-down works.

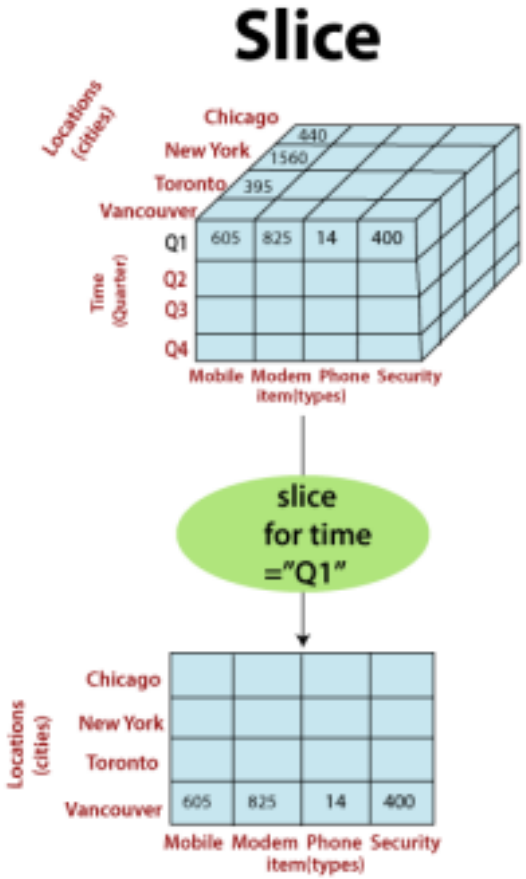


Slice

A **slice** is a subset of the cubes corresponding to a single value for one or more members of the dimension. For example, a slice operation is executed when the customer wants a selection on one dimension of a three-dimensional cube resulting in a two-dimensional site. So, the Slice operations perform a selection on one dimension of the given cube, thus resulting in a subcube.

For example, if we make the selection, temperature=cool we will obtain the following cube:

Temperature	**cool**
Day 1	0
Day 2	0
Day 3	0
Day 4	0
Day 5	1
Day 6	1
Day 7	1
Day 8	1
Day 9	1
Day 11	0
Day 12	0
Day 13	0
Day 14	0



Here Slice is functioning for the dimensions "time" using the criterion time = "Q1".

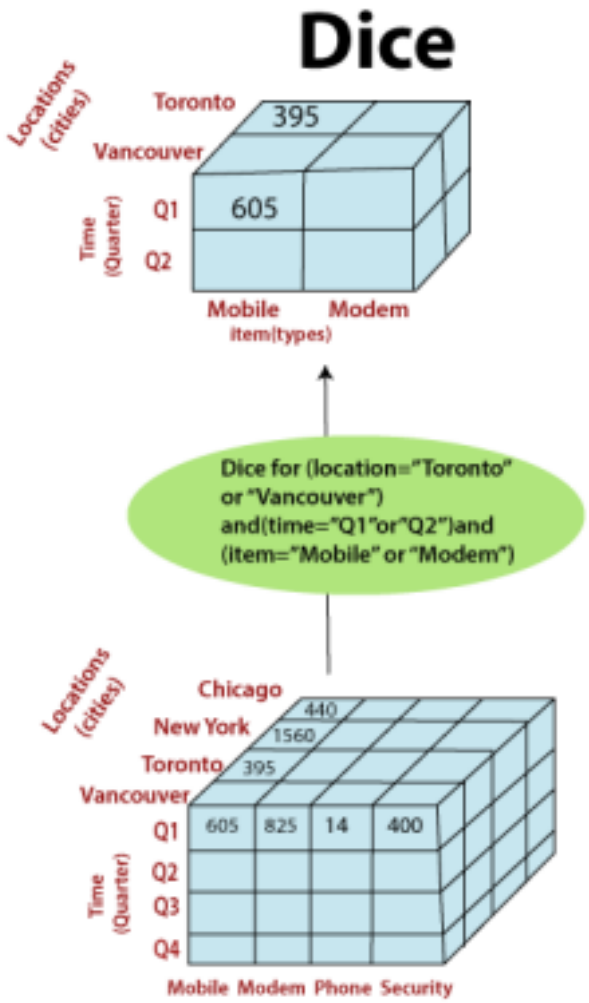
It will form a new sub-cubes by selecting one or more dimensions.

Dice

The dice operation describes a subcube by operating a selection on two or more dimension. **For example**, Implement the selection (time = day 3 OR time = day 4) AND (temperature = cool OR temperature = hot) to the original cubes we get the following subcube (still two-dimensional)

Temperature	**cool**	**hot**
Day 3	0	1
Day 4	0	0

Consider the following diagram, which shows the dice operations.

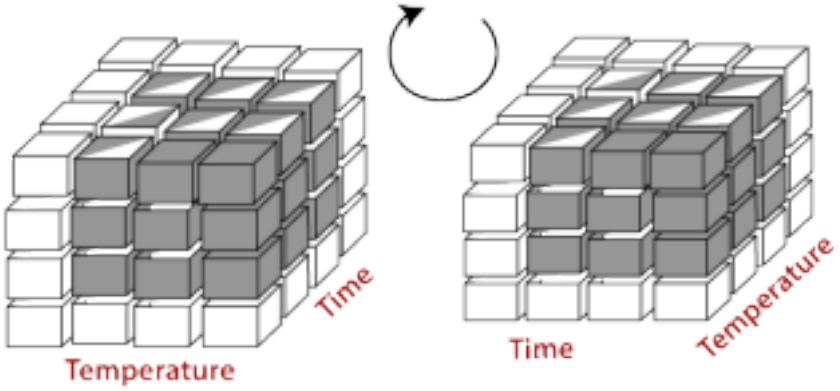


The dice operation on the cubes based on the following selection criteria involves three dimensions.

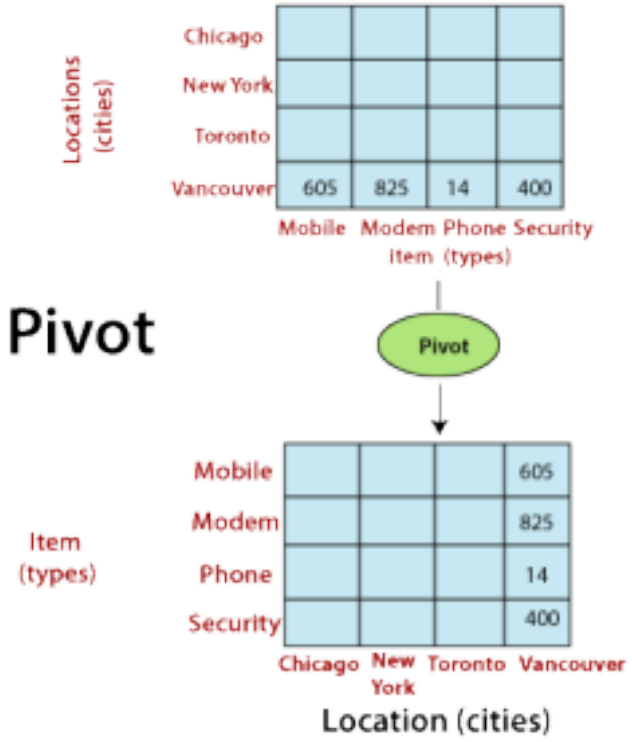
- (location = "Toronto" or "Vancouver")
- (time = "Q1" or "Q2")
- (item =" Mobile" or "Modem")

Pivot

The pivot operation is also called a rotation. Pivot is a visualization operations which rotates the data axes in view to provide an alternative presentation of the data. It may contain swapping the rows and columns or moving one of the row-dimensions into the column dimensions.



Consider the following diagram, which shows the pivot operation.



Other OLAP Operations

Other OLAP operations may contain ranking the top-N or bottom-N elements in lists, as well as calculate moving average, growth rates, and interests, internal rates of returns, depreciation, currency conversions, and statistical tasks.

OLAP offers analytical modeling capabilities, containing a calculation engine for determining ratios, variance, etc. and for computing measures across various dimensions. It can generate summarization, aggregation, and hierarchies at each granularity level and at every dimensions intersection.

OLAP Models

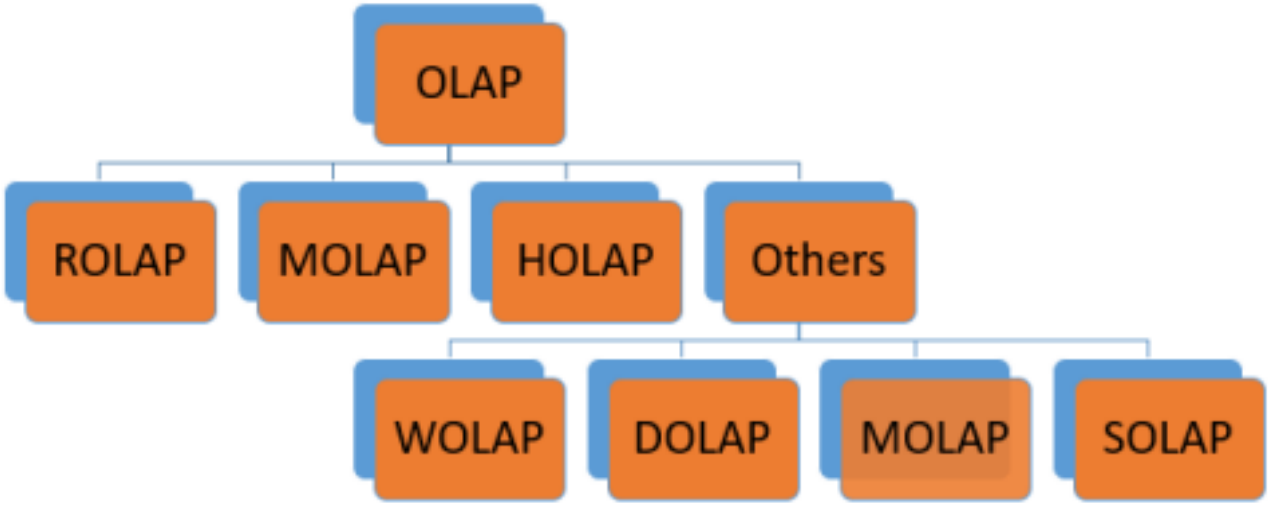


Figure: OLAP Hierarchical Structure

There are 3 main types of OLAP servers are as following:

- ROLAP
- MOLAP
- HOLAP

ROLAP stands for Relational OLAP, an application based on relational DBMSs. **MOLAP** stands for Multidimensional OLAP, an application based on multidimensional DBMSs. **HOLAP** stands for Hybrid OLAP, an application using both relational and multidimensional techniques.

Relational OLAP

ROLAP servers are placed between relational back-end server and client front-end tools. To store and manage warehouse data, ROLAP uses relational or extended-relational DBMS. ROLAP includes the following −

- Implementation of aggregation navigation logic.
- Optimization for each DBMS back end.
- Additional tools and services.
-

Multidimensional OLAP

MOLAP uses array-based multidimensional storage engines for multidimensional views of data. With multidimensional data stores, the storage utilization may be low if the data set is sparse. Therefore, many MOLAP server use two levels of data storage representation to handle dense and sparse data sets.

Hybrid OLAP

Hybrid OLAP is a combination of both ROLAP and MOLAP. It offers higher scalability of ROLAP and faster computation of MOLAP. HOLAP servers allows to store the large data volumes of detailed information. The aggregations are stored separately in MOLAP store. Other OLAP models are

Desktop OLAP (DOLAP)	In Desktop OLAP, a user downloads a part of the data from the database locally, or on their desktop and analyze it. DOLAP is relatively cheaper to deploy as it offers very few functionalities compares to other OLAP systems.
Web OLAP (WOLAP)	Web OLAP which is OLAP system accessible via the web browser. WOLAP is a three-tiered architecture. It consists of three components: client, middleware, and a database server.
Mobile OLAP	Mobile OLAP helps users to access and analyze OLAP data using their mobile devices
Spatial OLAP	SOLAP is created to facilitate management of both spatial and non-spatial data in a Geographic Information system (GIS)

Multidimensional OLAP (MOLAP)

A MOLAP system is based on a native logical model that directly supports multidimensional data and operations. Data are stored physically into multidimensional arrays, and positional techniques are used to access them.

One of the significant distinctions of **MOLAP** against a **ROLAP** is that data are summarized and are stored in an optimized format in a multidimensional cube, instead of in a relational database. In MOLAP model, data are structured into proprietary formats by client's reporting requirements with the calculations pre-generated on the cubes.

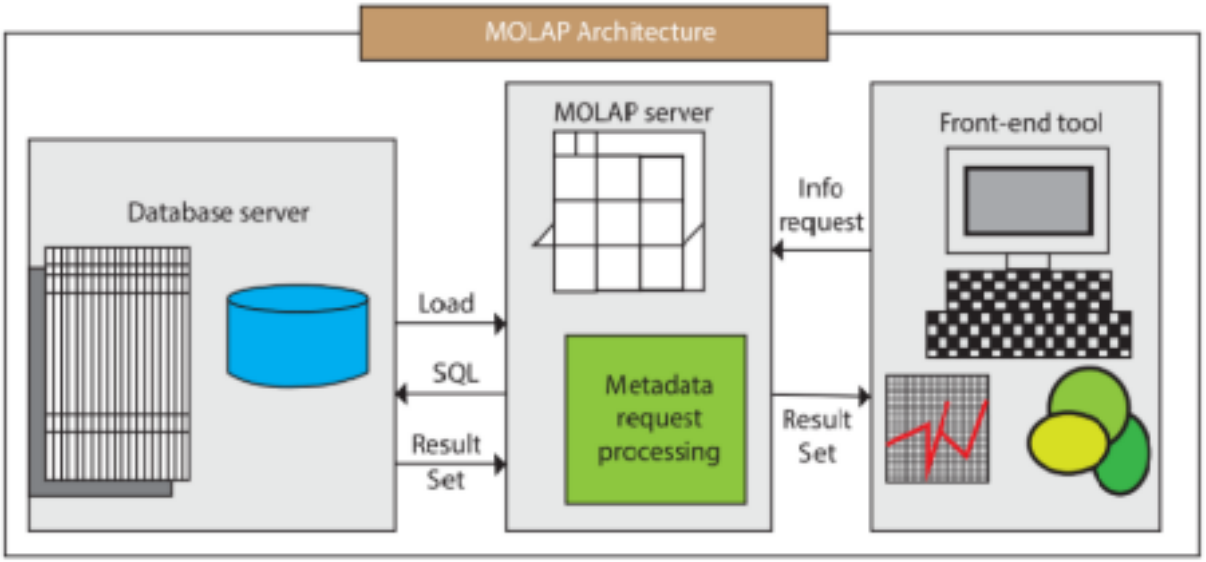
MOLAP Architecture

MOLAP Architecture includes the following components

• Database server.

• MOLAP server.

• Front-end tool.



MOLAP structure primarily reads the precompiled data. MOLAP structure has limited capabilities to dynamically create aggregations or to evaluate results which have not been pre-calculated and stored.

Applications requiring iterative and comprehensive time-series analysis of trends are well suited for MOLAP technology (e.g., financial analysis and budgeting).

Examples include Arbor Software's Essbase. Oracle's Express Server, Pilot Software's Lightship Server, Sniper's TM/1. Planning Science's Gentium and Kenan Technology's Multiway. Some of the problems faced by clients are related to maintaining support to multiple subject areas in an RDBMS. Some vendors can solve these problems by continuing access from MOLAP tools to detailed data in and RDBMS.

This can be very useful for organizations with performance-sensitive multidimensional analysis requirements and that have built or are in the process of building a data warehouse architecture that contains multiple subject areas.

An example would be the creation of sales data measured by several dimensions (e.g., product and sales region) to be stored and maintained in a persistent structure. This structure would be provided to reduce the application overhead of performing calculations and building aggregation during initialization. These structures can be automatically refreshed at predetermined intervals established by an administrator.

Advantages

Excellent Performance: A MOLAP cube is built for fast information retrieval, and is optimal for slicing and dicing operations.

Can perform complex calculations: All evaluation have been pre-generated when the cube is created. Hence, complex calculations are not only possible, but they return quickly.

Disadvantages

Limited in the amount of information it can handle: Because all calculations are performed when the cube is built, it is not possible to contain a large amount of data in the cube itself.

Requires additional investment: Cube technology is generally proprietary and does not already exist in the organization. Therefore, to adopt MOLAP technology, chances are other investments in human and capital resources are needed.