Data Warehouse

As per Bill Inmon "A warehouse is a **Historical**, **subject-oriented**, **integrated**, **time-variant** and **non-volatile** collection of data in support of management's decision making process".

By **Historical** we mean, the data is continuously collected from sources and loaded in the warehouse. The previously loaded data is not deleted for long period of time. This results in building historical data in the warehouse.

By **Subject Oriented** we mean data grouped into a particular business area instead of the business as a whole.

By **Integrated** we mean, collecting and merging data from various sources. These sources could be disparate in nature.

By **Time-variant** we mean that all data in the data warehouse is identified with a particular time period.

By **Non-volatile** we mean, data that is loaded in the warehouse is based on business transactions in the past, hence it is not expected to change over time.

An implementation of an informational database used to store sharable data sourced from an operational database of record. It is typically a subject database that allows users to tap into a company’s vast store of operational data to track and respond to business trends and facilitate forecasting and planning efforts. It is a collection of integrated, subject-oriented databases designed to support the DSS function, where each unit of data is relevant to some moment in time. The data warehouse contains atomic data and lightly summarized data.

Data Warehousing (DWH)

Data warehousing is the vast field that includes the **processes** and **methodologies** involved in the creating, populating, querying and maintaining a data warehouse. The objective of DWH is to help users make better business decisions.

Enterprise Data Warehouse (EDW)

Enterprise Data Warehouse (EDW) is a **central normalized repository** containing **multiple subject area** data for an organization. EDW are generally used for feeding data to subject area specific data marts. We must consider building an EDW when

1. We have clarity on multiple business processes requirements.

2. Want a centralized architecture catering to single version of truth.

An EDW is a database that is subject-oriented, integrated, non-volatile (read-only) and time-variant (historical). It is usually used by financial analysts for historical trend analysis reporting, data mining and other activities that need historical data. An EDW keeps growing as you add more historical snapshots, daily, weekly or monthly. Because an EDW has

historical data (and the ODS usually does not), some companies use the EDW as a hub for loading their data marts. CIF stands for Corporate Information Factory in the context of data warehousing. CIF is an overall architecture that includes ODS, EDW, data marts, oper marts, exploration warehouse, Meta data repository, extract, transfer and load (ETL) processes, portals, OLAP and BI applications like dashboards and scorecards, and so on. Since the question was about data mart consolidation, the other option you have is to create a normalized persistent staging area (PSA) database, which is not accessible by end users but serves as a hub for loading your current and future data marts.

Enterprise Data Warehouse is a centralized warehouse which provides service for the entire enterprise. A data warehouse is by essence a large repository of historical and current transaction data of an organization. An Enterprise Data Warehouse is a specialized data warehouse which may have several interpretations.

Several terms used in information technology have been used by a so many different vendors, IT workers and marketing ad campaigns that has left many confused about what really the term Enterprise Data Warehouse means and what makes it different from a general data warehouse.

Enterprise Data Warehouse has emerged from the convergence of opportunity, capability, infrastructure and need for data which has exponentially increased during the last few years as technology has advanced too fast and Business Enterprises tried to do their best to catch up and be on the top of the industry competition.

Enterprise data warehouses replace data marts

Businesses are replacing departmental databases - known as data marts - with enterprise data warehouses designed to pool information across the company.

"Data marts are very expensive," said Gartner analyst Donald Feinberg. "If you have six departments each with its own data mart, you end up with six hardware systems and six database licenses. You also need people who can maintain each data mart."

Businesses often find they end up with the same information replicated in each data mart, he said. They no longer have a single master copy of the data and have to spend more on storage.

"In an enterprise data warehouse, data quality is higher," said Feinberg. "A data warehouse project does not have to take a 'big bang' approach. You can start on a small project and design it around a data warehouse."

Data Mart

When we restrict the scope of a data warehouse to a sub set of business process or subject area we call it a Data Mart. Data Marts can be stand-alone entities, built from warehouses or be used to build warehouse.

1. **Stand-Alone Data Marts** – These are single subject areas data warehouses. Stand alone data marts could cause many stove-pipe/silo data marts in the future.

2. Data Marts fed from Enterprise Data Warehouses (**Top-Down**). This is the Top-Down strategy suggested by Bill Inmon. Here we build an enterprise data warehouse (EDW) hosting data for multiple subject areas. From the EDW we build subject area specific data marts.

3. Data Marts building Data Warehouses (**Bottom-Up**). This is the Bottom-Up strategy suggested by Ralph Kimball. Here we build subject area specific data marts first, keeping in mind the shared business attributes (called conformed dimensions). Using the conformed dimensions we build a warehouse.

A subset of a data warehouse that focuses on one or more specific subject areas. The data is usually extracted from the data warehouse and further denormalised and indexed to support intense usage by targeted customers.

Legacy Systems

Legacy system is the term given to a system having **very large storage capabilities**. Mainframes and AS400 are the two examples of legacy systems. Applications running on legacy systems are generally OLTP systems. Due to their large storage capabilities the business transactional data from legacy applications and other OLTP systems is stored on them and maintained for a long period of time. In the world of DWH, legacy systems are often used to extract past historical business data.

Data Profiling

This is a very important activity mostly performed while collecting the requirements and the system study. Data Profiling is a process that **familiarizes you with the data you will be loading into the warehouse/mart**.

In the most basic form, data profiling process would read the source data and report on the

Type of data – Whether data is Numeric, Text, Date etc.

Data statistics – Minimum and maximum values, most occurred value.

Data Anomalies like Junk values, incorrect business codes, invalid characters, values outside a given range.

Using data profiling we can validate business requirements with respect to business codes and attributes present in the sources, define exceptional cases when we get incorrect or inconsistent data.

A lot of Data Profiling tools are available in the market. Some examples are **Trillium Software** and IBM Websphere Data Integrator's **Profile Stage**.

In the simplest form of data profiling, we can even hand code (programming) data profile application specific to projects. Programming languages like PERL, Java, PLSQL and Shell Scripting can be very handy in creating these applications.

Data Modeling

This activity is performed once we have understood the requirements for the data warehouse/mart. Data modeling refers to creating the structure and relationship of the data for the business rules defined in the requirements.

We create the conceptual model followed by logical model followed by physical model. Here we see a **progression in the level of detail** as we proceed toward the physical model.

Conceptual Model

Conceptual Model identifies at a high level the subject areas involved. The objective of creating the conceptual model is to identify the scope of the project in terms of the subject areas involved.

Logical Model

Logical Model identifies the details of the subject areas identified in the conceptual model, and the relationships between the subject areas. Logical model identifies the entities within the subject area and documents in detail the definition of the entity, attributes, candidate keys, data types and the relationship with other entities.

Physical Model

Physical Model identifies the table structure, Columns within the tables, Primary Keys, data types and size of the columns, constraints on the table and the privileges on the tables. Using the physical model we develop the database objects through the DDL's developed in the physical model.

Normalization

This is a technique used in data modeling that emphasizes on avoiding storing the same data element at multiple places. We follow the 3 rules of normalization called the **First Normal Form**, **Second Normal Form**, and **Third Normal Form** to achieve a normalized data model.

A normalized data model may result in many tables/entities having multiple levels of relationships, example table1 related to table2, table2 further related to table3, table3 related to table 4 and so on.

First Normal Form – The attributes of the entity must be atomic and must depend on the Key.

Second Normal Form – This rule demands that every aspect of each and every attribute depends on Key.

Third Normal Form (3NF) – This rule demands that every aspect of each and every attributes depends on nothing but the key.

Theoretically We have further rules called the Boyce-Codd Normal Form, Fourth Normal Form and the Fifth Normal form. In practice we don't use the rules beyond 3NF.

Dimensional Modeling

This is the form of data modeling used in data warehousing where we store business related attributes in tables called **Dimension Tables** and the business related metrics/measures in tables called **Fact Tables**. The business metrics are associated with the business attributes by relating the dimension tables with the fact table. The dimension tables are not related to one another.

A fact table could be related to many dimension table and hence form the center of the structure when graphically represented. This structure is also called the **star schema**.

Please note that here we do not emphasize on normalizing the data instead have lesser levels of relationships.

Extract Transform Load (ETL)

The back-end processes involved in collecting data from various sources, preparing the data with respect to the requirements and loading it in the warehouse/mart.

Extraction – This is the process where we select the data from various source systems and transport it in an ETL server. Source systems could range from one to many in number, and similar or completely disparate in nature.

Cleansing – Once the data has been extracted, we need to check whether the data is **valid** and **ensure consistency** when coming from disparate sources. We check for valid values of data by performing ETL data validations rules. The rules could be defined to check for correct data formats (Numeric, Date, Text), data within the range, correct business codes, check for junk values. We can ensure consistency of data when it is collected from disparate sources by making sure that the same business attribute will have the same representation. Example would be the gender of customer may be represented as "m" and "f" in one source system and "male" and "female" in another source system. In this case we will make sure we convert "male" to "m" and "female" to "f" so that the gender coming from the second source system is consistent with that from the first source system.

Transformation – This is the process where we apply the business rules to the source data before loading it to the warehouse/mart. Transformation as the term suggests converts the data based on the business rules. Transformation also adds further business context to the data. The types of transformations we generally use in ETL are sort data, merge data, compare data, generate running numbers, aggregate data, change data type etc.

Loading – Once the source data is transformed by applying business rules, we load it into the warehouse/mart. There are two types of loads we perform in data warehousing.

History Load / Bulk Load – This is the process of loading the historical data over a period of time into the warehouse. This is usually a one time activity where we take the historical business data and bulk load into the warehouse/mart.

Incremental Load – This follows the history load. Based on the availability of the current source data we load the data into the warehouse. This is an ongoing process and is performed periodically based on the availability of source data (eg daily, weekly, monthly etc).

Operational Data Store (ODS)

Operational Data Store (ODS) is layer of repository that resides **between a data warehouse and the source system**. The current, detailed data from disparate source systems is integrated, grouped subject areas wise in the ODS. The ODS is updated more frequently than a data warehouse. The updates happen shortly after a source system data changes. There are two objectives of an ODC

1. Source DWH with detailed, integrated data.

2. Provide organization with operational Integration.

An ODS is an integrated database of operational data. Its sources include legacy systems and it contains current or near-term data. An ODS may contain 30 to 60 days of information, while a data warehouse typically contains years of data. An operational data store is basically a database that is used for being an interim area for a data warehouse. As such, its primary purpose is for handling data which are progressively in use such as transactions, inventory and collecting data from Point of Sales. It works with a data warehouse but unlike a data warehouse, an operational data store does not contain static data. Instead, an operational data store contains data which are constantly updated through the course of the business operations.

An ODS is a database that is subject-oriented, integrated, volatile and current. It is usually used by business managers, analysts or customer service representatives to monitor manage and improve daily business processes and customer service. An ODS is often loaded daily or multiple times a day with data that represents the current state of operational systems.

An operational data store (ODS) is a type of database that's often used as an interim logical area for a data warehouse.

While in the ODS, data can be scrubbed, resolved for redundancy and checked for compliance with the corresponding business rules. An ODS can be used for integrating disparate data from multiple sources so that business operations, analysis and reporting can be carried out while business operations are occurring. This is the place where most of the

data used in current operation is housed before it's transferred to the data warehouse for longer term storage or archiving.

An ODS is designed for relatively simple queries on small amounts of data (such as finding the status of a customer order), rather than the complex queries on large amounts of data typical of the data warehouse. An ODS is similar to your short term memory in that it stores only very recent information; in comparison, the data warehouse is more like long term memory in that it stores relatively permanent information.

ODS is specially designed such that it can quickly perform relatively simply queries on smaller volumes of data such as finding orders of a customer or looking for available items in the retails store. This is in contrast to the structure of a data warehouse wherein one needs to perform complex queries on high volumes of data. As a simple analogy, a data store may be a company's short term memory storing only the most recent information while the data warehouse is the long term memory which also serves as a company's historical data repository whose data stored are relatively permanent.

In recent years, data warehouses have started becoming operational in nature with a flavor of warehouse data that is both current or almost current and static as well. Banks are trying to maintain data in two ways. One is more static in nature for daily reporting requirements, whiles the other, also called an operational data store (ODS), is more dynamic, offering a current view of data for immediate reporting and analysis requirements.

ODS is part of the data warehouse architecture. It is the first stop for the data on its way to the warehouse. It is here where I can collect and integrate the data. I can ensure its accuracy and completeness. In many implementations, all transformations can't be completed until a full set of data is available. If the data rate is high, I can capture it without constantly changing the data in the warehouse. I can even supply some analysis of near current data to those impatient business analysts. However, not every DW needs an ODS. Typically the ODS is a normalized structure that integrates data based on a subject area, not a specific application or applications. For example, one client site I worked at had 60+ premium applications. A "premium" subject area ODS was created which collected data using a feed from each application to provide near real- time premium data across the enterprise. This ODS was refreshed to stay current, and the history was sent to the data warehouse.

Difference between ODS and DWH

One of the key challenges in a data warehouse is to extract data from multiple non-integrated systems to a defined single space, all the while ensuring linkages between data elements such as customer, transaction, account and other dimensional data. By providing a foundation to integrated operational results, the ODS easily helps resolve this challenge and is often regarded as a common link between banks' operational systems and the data warehouse. However, it is always advisable to have a clear distinction between the ODS and the DWH decision systems. While the DWH should store only historical data, the ODS needs to be more current in nature. It should be designed to rapidly update data so as to provide snapshots in an online, real-time environment.

There is also a marked trend towards keeping both the ODS and warehouse in the same place as the interchangeability is very thin

Often, to avoid investing in technology to develop an ODS, banks end up using the old warehouse for operational reporting requirements as well. However, it needs to be remembered that at the technology front, both the ODS and DWH need different architectures. While the ODS needs 'update and record' oriented functionality, the warehouse requires a 'load and access' oriented functionality.

An important recent trend is of keeping some functionalities of a DWH in the ODS, such as summary or aggregated data for easy and fast query. Although it is very difficult to have both current and summary data stored in an ODS, this helps to meet quickly and easily, different requirements from the same place. However, as the ODS data gets regularly updated, it is important to realize that the accuracy of the summary data in an

ODS is only for the specific point when it is accessed. So compared to a DWH, the reconstruction of same balances in an ODS will be difficult.

Essentially, the ODS derives some of its essence from that of a data warehouse with certain key differences. An ODS can be defined as an application that enables any institution to meet its operational data analysis and processing needs. The key features of an ODS are as below:

- **Subject oriented**
- **Integrated**
- **Volatile - updated data**
- **Near time or current time data**

Notably, it is the last two points that differentiate an ODS from a warehouse.

When we say an ODS needs to be subject oriented, it means that the basic design should cater to specific business areas or domains for easy access and analysis of data. It will be easier to store data in sub segments like customer, transaction and balance with an inter linkage between all the segments. That brings the requirement of using a common key for all data files to establish the linkage. Along with using the key, aggregation of data for certain time blocks as discussed above, it will also be helpful for easy and fast recovery of query results.

The following key functions are generally required for a successful ODS

- **Converting data**
- **Selection of best data from multiple sources**
- **Decoding/encoding data**
- **Summarizing data**
- **Recalculating data**
- **Reformatting data and many more.**

Apart from changes in the structure of an ODS in terms of handling large volume volatile transactional data from multiple sources with a scope of aggregating them as well, there are also infrastructure related changes happening today. While the basic technology of loading and processing high volume data remains the same, when the system of records changes a

little, the impact on the ODS can be very significant and has to be managed very carefully. Therefore, even if the system of records may comprise a very small portion of the system, it should be ensured that the underlying infrastructure is able to handle high volumes at all times.

How to start building an ODS

It is advisable to start building a subject-oriented small ODS in the first place with customer centric activities and transactions. Later, with integration in mind, other subject areas can be added.

Conclusion

Although there are many similarities between a data warehouse and an ODS and it is often difficult to differentiate between them, there is a specific need for an ODS in banks today where large volume of data is required to be processed in a fast and accurate manner to support the operational data reporting and analytical needs of the organization. As ODS contains specific data required for a set of business users, it helps in reducing the time to source information and with the added functionality of summary or aggregated data, it will definitely be an important element in any large bank's IT systems infrastructure.

Differences between OLTP and OLAP

Online Transaction Processing Systems (OLTP)

Online Transaction Processing Systems (OLTP) are t**ransactional systems** that perform the **day to day business operations**. In other words, OLTP are the automated day to day business processes. In the world of DWH, they are usually one of the types of source system from where the day to day business transactional data is extracted. OLTP systems contain the current valued business data that is updated based on business transactions.

OLTP online transactional processing (supports 3rd Normal Form)

OLTP is nothing but Online Transaction Processing, which contains a normalized tables and online data, which have frequent insert/updates/delete.

OLTP databases are fully normalized and are designed to consistently store operational data, one transaction at a time. It performs day-to -day operations and not support historical data

OLTP is customer-oriented, used for data analysis and querying by clerks, clients and IT professionals.

OLTP manages current data, very detail-oriented.

OLTP adopts an entity relationship(ER) model and an application-oriented database design

Current data
Detailed data
normalized data
Short database transactions
Online update/insert/delete
Normalization is promoted
High volume transactions
Transaction recovery is necessary
In OLTP Few Indexes and Many Joins
No. of concurrent users are very high.
Data is volatile.
Size of oltp system is low.
Subject oriented
gives operational view of data

Online transactional processing (OLTP) is designed to efficiently process high volumes of transactions, instantly recording business events (such as a sales invoice payment) and reflecting changes as they occur.

OLAP - online analytical processing (Supports Star/snowflake Schemas)

OLAP (Online Analytical Programming) contains the history of OLTP data, which is, non-volatile, acts as a Decisions Support System and is used for creating forecasting reports.

OLAP is a set of specifications which allows the client applications (reports) in retrieving the data from data ware house

OLAP is market-oriented, used for data analysis by knowledge workers (managers, executives, analysis).

OLAP manages large amounts of historical data, provides facilities for summarization and aggregation, stores information at different levels of granularity to support decision making process.

OLAP adopts star, snowflake or fact constellation model and a subject-oriented database design.

Online analytical processing (OLAP) is designed for analysis and decision support, allowing exploration of often hidden relationships in large amounts of data by providing unlimited views of multiple relationships at any cross-section of defined business dimensions.

OLTP (On-line Transaction Processing) is characterized by a large number of short on-line transactions (INSERT, UPDATE, and DELETE). The main emphasis for OLTP systems is put on very fast query processing, maintaining data integrity in multi-access environments and an effectiveness measured by number of transactions per second. In OLTP database there is detailed and current data, and schema used to store transactional databases is the entity model (usually 3NF).
Current and historical data
Summarized data
Denormalized data
Long database transactions
Batch update/insert/delete

Low volume transactions
Transaction recovery is not necessary
In OLAP Many Indexes and Few Joins
no. of concurrent users are low.
Data is nonvolatile.
Integrated and subject oriented database.
Give analytical view.
W/H maintains subject oriented, time variant, and historical data. Where as OLTP applications maintain current data, and it gets updates regularly. ex: ATM machine, Railway reservations etc..

OLAP (On-line Analytical Processing) is characterized by relatively low volume of transactions. Queries are often very complex and involve aggregations. For OLAP systems a response time is an effectiveness measure. OLAP applications are widely used by Data Mining techniques. In OLAP database there is aggregated, historical data, stored in multi-dimensional schemas (usually star schema).

The following table summarizes the major differences between OLTP and OLAP system design.

	OLTP System Online Transaction Processing (Operational System)	OLAP System Online Analytical Processing (Data Warehouse)
Source of data	Operational data; OLTPs are the original source of the data.	Consolidation data; OLAP data comes from the various OLTP Databases
Purpose of data	To control and run fundamental business tasks	To help with planning, problem solving, and decision support
What the data	Reveals a snapshot of ongoing business processes	Multi-dimensional views of various kinds of business activities
Inserts and Updates	Short and fast inserts and updates initiated by end users	Periodic long-running batch jobs refresh the data
Queries	Relatively standardized and simple queries Returning relatively few records	Often complex queries involving aggregations
Processing Speed	Typically very fast	Depends on the amount of data involved; batch data refreshes and complex queries may take many hours; query speed can be improved by creating indexes
Space Requirements	Can be relatively small if historical data is archived	Larger due to the existence of aggregation structures and history data; requires more indexes than OLTP

Database Design	Highly normalized with many tables	Typically de-normalized with fewer tables; use of star and/or snowflake schemas
Backup and Recovery	Backup religiously; operational data is critical to run the business, data loss is likely to entail significant monetary loss and legal liability	Instead of regular backups, some environments may consider simply reloading the OLTP data as a recovery method

Online Analytical Processing (OLAP)

Online Analytical Processing (OLAP) is the approach involved in providing **analysis to multidimensional data**. There are various forms of OLAP namely

Multidimensional OLAP (MOLAP) – This uses a cube structure to store the business attributes and measures. Conceptually it can be considered as multi-dimensional graph having various axes. Each axis represents a category of business called a dimension (eg location, product, time, employee etc). The data points are the business transactional values called measures. The coordinate of each data point related it to the business dimension. The cubes are generally stored in proprietary format.

Relational OLAP (ROLAP)– This uses database to store the business categories as dimension tales (containing textual business attributes and codes) and the fact tables to store the business transactional values (numeric measures). The fact tables are connected to the dimension tables through the Foreign-Key relationships.

Hybrid OLAP (HOLAP) – This uses a cube structure to store some business attributes and database tables to hold the other attributes.

Business Intelligence (BI)

Business Intelligence (BI) is a generic term used for the **processes, methodologies and technologies** used for **collecting, analyzing and presenting business information**.

The objective of BI is to **support users make better business decisions**. The objective of DWH and BI go hand-in-hand, that of enabling users make better business decisions.

Data marts/warehouses provide easily accessible historical data to the BI tools to provide end users with forecasting/predictive analytical capabilities and historical business patters and trends.

BI Tools are generally accompanied with reporting features like S**lice-and-dice, Pivot-table-analysis, Visualization, and statistical functions** that make the decision making process even more sophisticated and advanced.

Decision Support Systems (DSS)

A Decision Support System (DSS) is used for **management based decision making reporting**. It is basically an automated system to gather business data and help make management business decisions.

Data Mining

Data mining in the broad sense **deals with extracting relevant information from very large amount of data** and make intelligent decision based on the information extracted.

Building blocks of Data Warehousing

If you had to build a Data Warehouse, you would need the following components

Source System(s) – A source system is generally a collection of **OLTP System and/or legacy systems** that contains day to day business data.

ETL Server – These are generally **Symmetric Multi-Processing (SMP)** machines or **Massively Parallel Processing Machines (MPP)**.

ETL Tool – These are sophisticated GUI based applications that enable the operation of Extracting data from source systems, transforming and cleaning data and loading the data into databases. ETL tool resides on the ETL server.

Database Server – Usually we have the database reside on the ETL server itself. In case of very large amount of data we may want to have a separate Symmetric Multi-Processing machines or Massively Parallel Processing Machines clustered to exploit parallel processing of the RDBMS.

Landing Zone – Usually a file-system and sometimes a database used to land the source data from various source systems. Generally the data in the landing zone is in the raw format.

Staging Area -Usually a file-system and a database used to house the transformed and cleansed source data before loading it to the warehouse or mart. Generally we avoid any further data cleansing and transformation after we have loaded data in the staging area. The format of the data in staging area is similar to that of in the warehouse or mart.

Data Warehouse – A Database housing the normalized (top-Down) or conformed star schemas (bottom-up) Reporting Server - These are generally Symmetric Multi-Processing machines or Massively Parallel Processing Machines.

Reporting Tool – A sophisticated GUI based application that enables users to view, create and schedule reports on data warehouse.

Metadata Management Tool – Metadata is defined as description of the data stored in your system. In data warehousing the Reporting tool, database and ETL tool have their own metadata. But if we look at the overall picture, we need to have Metadata defined starting

from source systems, landing zone, staging area, ETL, warehouse, mart and the reports. An integrated metadata definition system containing metadata definition from source, landing, staging, ETL, warehouse, mart and reporting is advisable.

Star Schema:

A single fact table associated to N dimension tables.

Definition: The star schema is the simplest data warehouse schema. It is called a star schema because the diagram resembles a star, with points radiating from a center.

A single Fact table (center of the star) surrounded by multiple dimensional tables (the points of the star).

This schema is denormalised and hence has good performance as this involves a simple join rather than a complex query.

Star Schema is a relational database schema for representing multimensional data. It is the simplest form of data warehouse schema that contains one or more dimensions and fact tables. It is called a star schema because the entity-relationship diagram between dimensions and fact tables resembles a star where one fact table is connected to multiple dimensions. The center of the star schema consists of a large fact table and it points towards the dimension tables. The advantage of star schema is slicing down, performance increase and easy understanding of data.

Star schema is simplest data warehouse schema .It is called star schema because ER diagram of this schema looks like star with points originating from center. Center of star schema consists of large fact table and points of star are dimensional table.

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

Advantage of Star Schema:

1.Provide a direct mapping between the business entities and the schema design.
2. Provide highly optimized performance for star queries.
3.It is widely supported by a lot of business intelligence tools.
Simplest DW schema

Star Schema has data redundancy, so the query performance is good.
Easy to understand

Easy to Navigate between the tables due to less number of joins.

Most suitable for Query processing

can have less number of joins

Disadvantage of Star Schema:
There are some requirement which can not be meet by star schema like relationship between customer and bank account can not represented purely as star schema as relationship between them is many to many.

- **Occupies more space**
- **Highly Denormalized**

Snowflake schema:

Definition: A Snowflake schema is a Data warehouse Schema which consists of a single Fact table and multiple dimensional tables. These Dimensional tables are normalized. A variant of the star schema where each dimension can have its own dimensions.

SNOWFLAKING is a method of normalizing the dimension tables in a star schema.

Snow flake: can have more number of joins.

Snow flake: is normalized, so does not has data redundancy. Can have performance issues.

Advantages:

- **These tables are easier to maintain**
  saves the storage space.

Disadvantages:

- **Due to large number of joins, it is complex to navigate**

Star flake schema - Hybrid structure that contains a mixture of (Denormalized) STAR and (normalized) SNOWFLAKE schemas.

Star schema having one fact table surrounded by dimension table but in case of snowflack schema more than one fact table surrounded by dimensiontable.

Snowflake is bit more complex than star schema. It is called snow flake schema because diagram of snowflake schema resembles snowflake.

In snowflake schema tables are normalized to remove redundancy. In snowflake dimension tables are broken into multiple dimension tables, for example product table is broken into tables product and sub product.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.
For example, the Time Dimension that consists of 2 different hierarchies:

1. Year → Month → Day
2. Week → Day

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

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

Advantage of Snowflake Schema:

1.It provides greater flexibility in interrelationship between dimension levels and components.
2.No redundancy so it is easier to maintain.

Disadvantage of Snowflake Schema :

1.There are More complex queries and hence difficult to understand
2.More tables more joins so more query execution time.

A snowflake schema is a term that describes a star schema structure normalized through the use of outrigger tables. i.e dimension table hierarchies are broken into simpler tables.

In a star schema every dimension will have a primary key.

☐ In a star schema, a dimension table will not have any parent table.
☐ Whereas in a snow flake schema, a dimension table will have one or more parent tables.
☐ Hierarchies for the dimensions are stored in the dimensional table itself in star schema.
☐ Whereas hierarchies are broken into separate tables in snow flake schema. These hierarchies help to drill down the data from topmost hierarchies to the lowermost hierarchies.

If u r taking the snow flake it requires more dimensions, more foreign keys, and it will reduce the query performance, but it normalizes the records.,

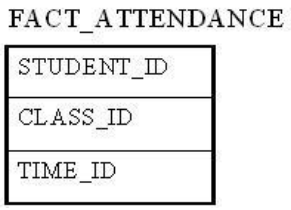
depends on the requirement we can choose the schema

Both these schemas are generally used in Data warehousing.

FACT LESS FACT TABLE

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

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



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

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

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

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

What are fact tables and dimension tables?

The data in a warehouse comes from the transactions. Fact table in a data warehouse consists of facts and/or measures. The nature of data in a fact table is usually numerical.

On the other hand, dimension table in a data warehouse contains fields used to describe the data in fact tables. A dimension table can provide additional and descriptive information (dimension) of the field of a fact table.

Dimension Table features

1. It provides the context /descriptive information for fact table measurements.
2. Provides entry points to data.
3. Structure of Dimension - Surrogate key , one or more other fields that compose the natural key (nk) and set of Attributes.
4. Size of Dimension Table is smaller than Fact Table.
5. In a schema more number of dimensions are presented than Fact Table.
6. Surrogate Key is used to prevent the primary key (pk) violation(store historical data).
7. Values of fields are in numeric and text representation.

Fact Table features

1. It provides measurement of an enterprise.
2. Measurement is the amount determined by observation.
3. Structure of Fact Table - foreign key (fk), Degenerated Dimension and Measurements.
4. Size of Fact Table is larger than Dimension Table.
5. In a schema less number of Fact Tables observed compared to Dimension Tables.
6. Compose of Degenerate Dimension fields act as Primary Key.
7. Values of the fields always in numeric or integer form.