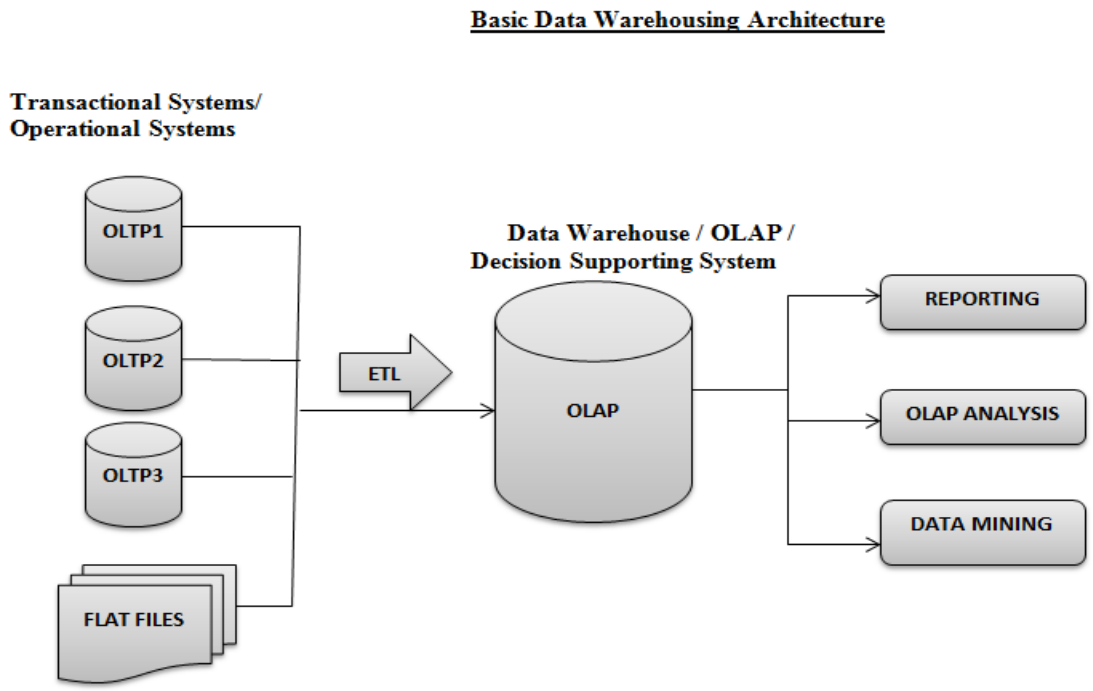
Data Warehousing Definition

Data Warehousing is the process of constructing a Data Warehouse and using it.

Basic Data Warehousing Architecture



Basic Data Warehousing Architecture contains Operational Systems or Transactional Systems which stores the Transactional Data and OLAP / Data Warehouse / Decision Supporting System (DSS) which stores the Analytical Data that is used for Analysis and Reporting. From Operational Systems we will Extract the Data and Transform the Data as per the Organization Needs and load the Data into Data Warehouse using ETL tools (Informatica). From Data Warehouse by using OLAP Tools or Reporting Tools we will generate reports, perform OLAP Analysis and Data Mining.

The Operational Systems may be OLTP Databases or Flat Files etc., which stores Transactional Data.

Transaction & Transactional Data

Transactional data describe about a transaction that takes place in an organization. Examples for Transactions are sales orders, invoices, purchase orders, shipping documents, passport applications, credit card payments, and insurance claims etc.

OLTP / Transactional System / Operational System

- ✓ OLTP stands for Online Transactional Process which stores transactional data.
- ✓ The Operational systems are where the data is put in.
- ✓ The users of an operational system turn the wheels of the organization.
- ✓ Users of an operational system almost always deal with one record at a time.
- ✓ They repeatedly perform the same operational tasks over and over.

OLAP / Data Warehouse / Decision Supporting System (DSS)

- ✓ OLAP stands for Online Analytical Process which stores Analytical data that is used for Analysis and Reporting.
- ✓ Data Warehouse is where we get the data out.
- ✓ The users of the Data Warehouse watch the wheels of organization.
- ✓ Users of the Data Warehouse almost never deal with one row at a time rather, their questions often require hundreds or thousands of rows are searched and compressed into an answer set.
- ✓ Users of a Data Warehouse continuously change the kind of required questions they ask.

Need of Data Warehouse?

Let us assume 'ABC' bank operates in multiple Countries.

And let us say Country1 data is residing in OLTP1, Country2 data in OLTP2 and Country3 data in OLTP3.

If one day ABC Bank requires consolidated Reports,

There are two ways

1. Completely manual, generate different reports from different OLTP's and integrate them to get the consolidated report.

2. Fetch all the data from different OLTP's, made it coherent (consistent manner), load to Data Warehouse and generate the Reports from Data Warehouse.

Obviously second approach is the best.

Issues with OLTP Reporting

1. **Less History**

The OLTP Systems does not maintain Complete History in order to have the better transaction performance. So it is not possible to analyze the data completely for a wide range.

2. **Performance issue**

- ✓ As we are fetching the data from multiple transactional systems to generate consolidated report obviously it takes some time to get the final consolidated report.

- ✓ As the OLTP systems are highly normalized and hence to get the report output we need to join more number of tables.

- ✓ Also it is not recommendable to Insert and Retrieve Data from the same system at the same time.

3. **Data quality and data consistency issues**

As we are generating the Reports from OLTP's on the fly with some data transformations at the Report level to make data consistent and as it is not possible to test the reports output every time so there might be a chance of data quality and data consistency issues and also performance will degrades as we are performing the data transformation.

4. **Confident level**

   Obviously there will be low confident level from the users (BI People)

If we generate reports using Data Warehouse we will overcome the above issues.

Benefits with OLAP Reporting

- ✓ OLAP will maintain complete History so that we can make Better Analysis using complete Data.
- ✓ There will be no performance issues because we have the complete data from all the transactional sources in Data Warehouse.
- ✓ As the data in the Data Warehouse is in de-normalized form, hence to get the report output you no need to join more number of tables.
- ✓ There will be no data quality and data consistency issues as a Data Warehouse gets the preprocessed data from well Designed and Tested ETL code developed using any of the ETL tools (Informatica).
- ✓ Obviously user (BI People) will be confident at the report output.

Benefits of Building a Data Warehouse

- ✓ Data Warehouse provides more accurate and complete data.
- ✓ Data Warehouse provides easy access for end users.

Need of ETL Tools?

- ✓ ETL stands for Extraction, Transformation and Load.
- ✓ ETL tool provides the facility to extract data from different source systems, transform the data and load the data into target systems.
- ✓ To fetch the data from different sources and to make the data coherent or consistent and to load the data into the Data Warehouse we need ETL Tools.

Need of BI Tools?

As the Business Users don't know the SQL to communicate with the Databases so we generate the Reports using any of the BI Tools or Reporting Tools and we will share the Reports to end users for Analysis.

Data Warehouse Definition

A Data Warehouse is a copy of transactional data specifically structured for Reporting and Analysis.

--Ralph Kimball

A Data Warehouse is a Subject Oriented, Integrated, Time Variant and Non Volatile collection of data in support of management's decision making process.

--Bill Inmon

Characteristics of Data Warehouse

1. **Subject Oriented**

Data Warehouse is Subject Oriented rather than Application Oriented. Data in the Data Warehouse belongs to a specific subject.

Example: Sales, Finance, HR etc.

2. **Integrated**

Usually the source data for Data Warehouse is coming from different sources so we will integrate and load data into Data Warehouse.

3. **Time Variant**

Data entered into the Data Warehouse belongs to a specific time period. Data Warehouse will contain entire History of Data and to compare the data we need to maintain the timestamp for every record loaded into the Data Warehouse.

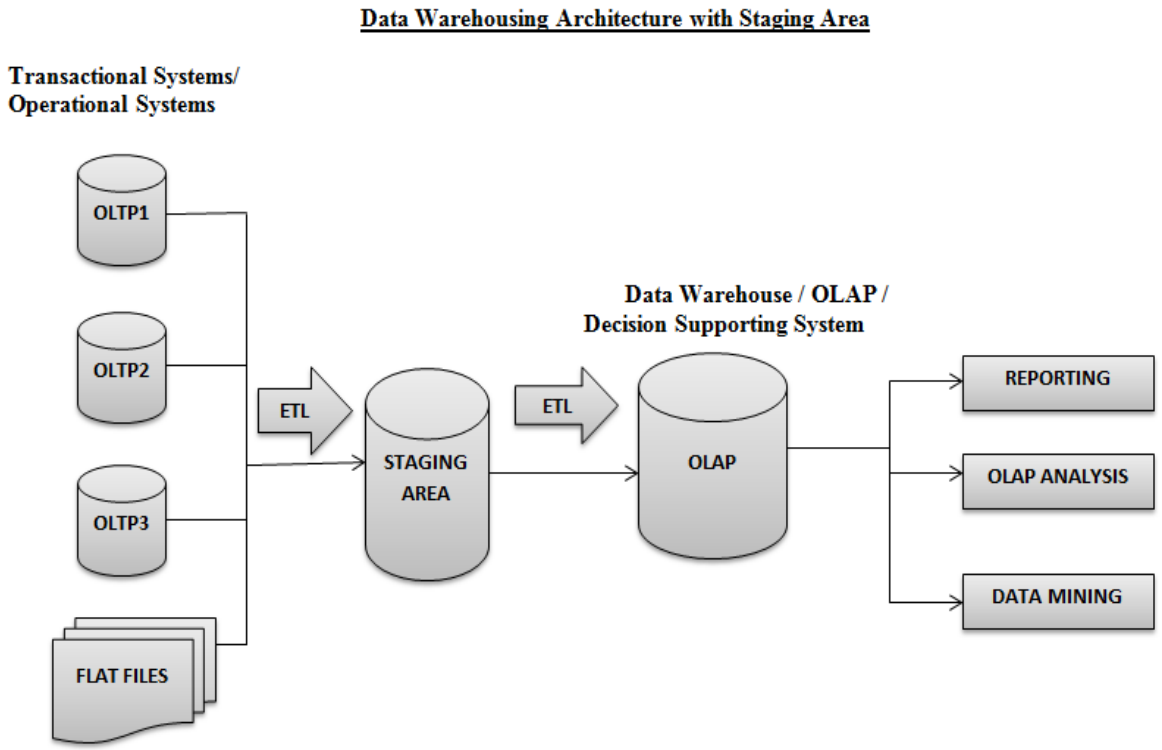
4. **Non Volatile**

Data entered into Data Warehouse never deleted or removed.

Differences between OLTP and OLAP

OLTP	OLAP
OLTP stands for Online Transactional Process which stores Transactional Data	OLAP stands for Online Analytical Process which stores Analytical Data that is used for Analysis
Designed to support business transactional process	Design to support business decision making process
Application Oriented Data	Subject Oriented Data
Less History	Complete History
Designed for Clerical access	Designed for Managerial access
Designed using ER Modeling	Designed using Dimension Modeling
More joins	Less joins
Normalized data (No data redundancy)	De Normalized data (Data Redundancy)

Data Warehousing Architecture with Staging Area



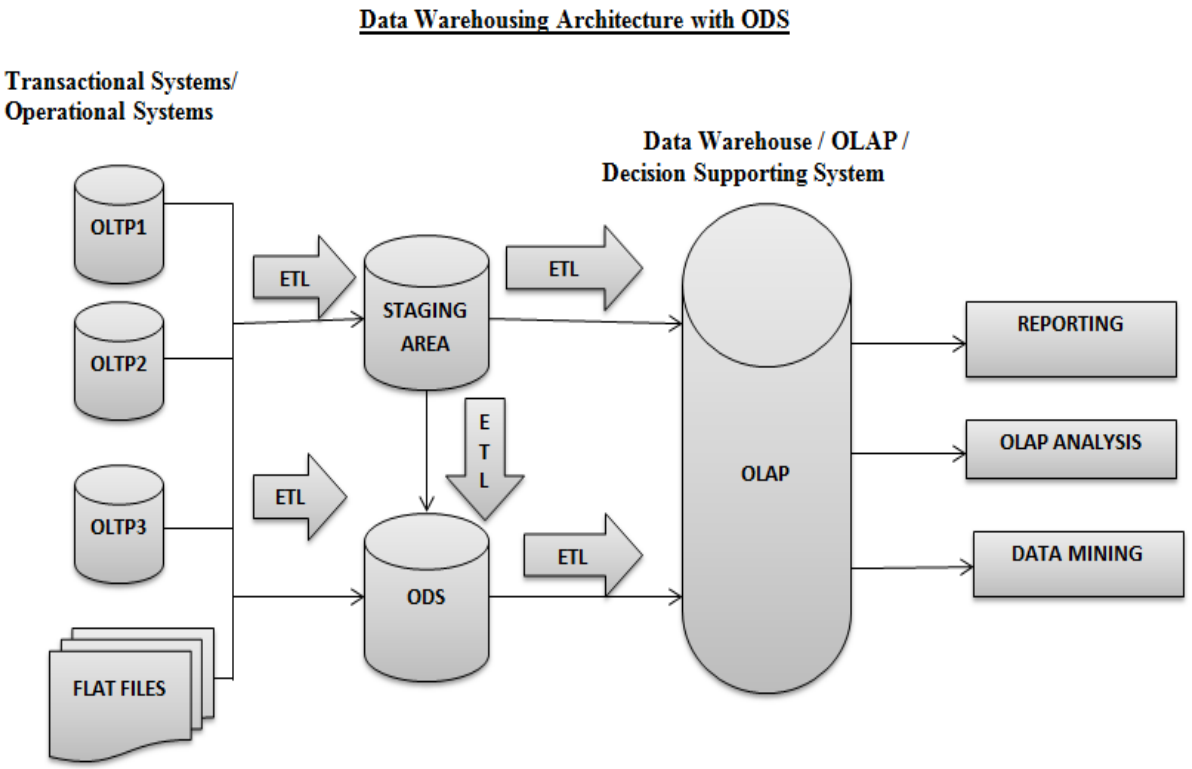
Staging Area

- ✓ Data Staging Area is a database and is an intermediate storage area between the Transactional sources and Data Warehouse / Data Mart.
- ✓ It is an integrated view of multiple Transactional sources.
- ✓ It is usually of temporary in nature, and its contents can be erased after the Data Warehouse / Data Mart has been loaded successfully.

Need of Staging Area

- ✓ If we have different sources for easy communication we use Data Staging Area.
- ✓ As we don't have the access to the data sources throughout the day. So it's better to get the data into staging and then to do the transformations on the Data.
- ✓ Suppose if we need to join 2 Tables data which comes on a particular day. Joining 2 tables in OLTP and filtering the data for a particular day will be time taking process. So we will get particular day data to the staging database and then we will join.

Data Warehousing Architecture with ODS



Operational Data Store (ODS)

ODS is used for many purposes. Let us see few of them.

1. An ODS is a database designed to integrate data from multiple sources for **additional operations on the data.** It may be passed for further operations and to the data warehouse for reporting.

2. An ODS is a database designed to do immediate reporting with current operational data. An ODS must be frequently refreshed so that it contains very current data. An ODS can be updated daily, hourly or even immediately after transactions on operational sources. **It is used for real time and near real time reporting.**

   The Data Warehouse will not update immediately once the transactions happened in operational sources.

- ✓ ODS is directly loaded from Operational sources or Staging Tables.
- ✓ It can optionally serve as a data source for the Data Warehouse.

Data Mart

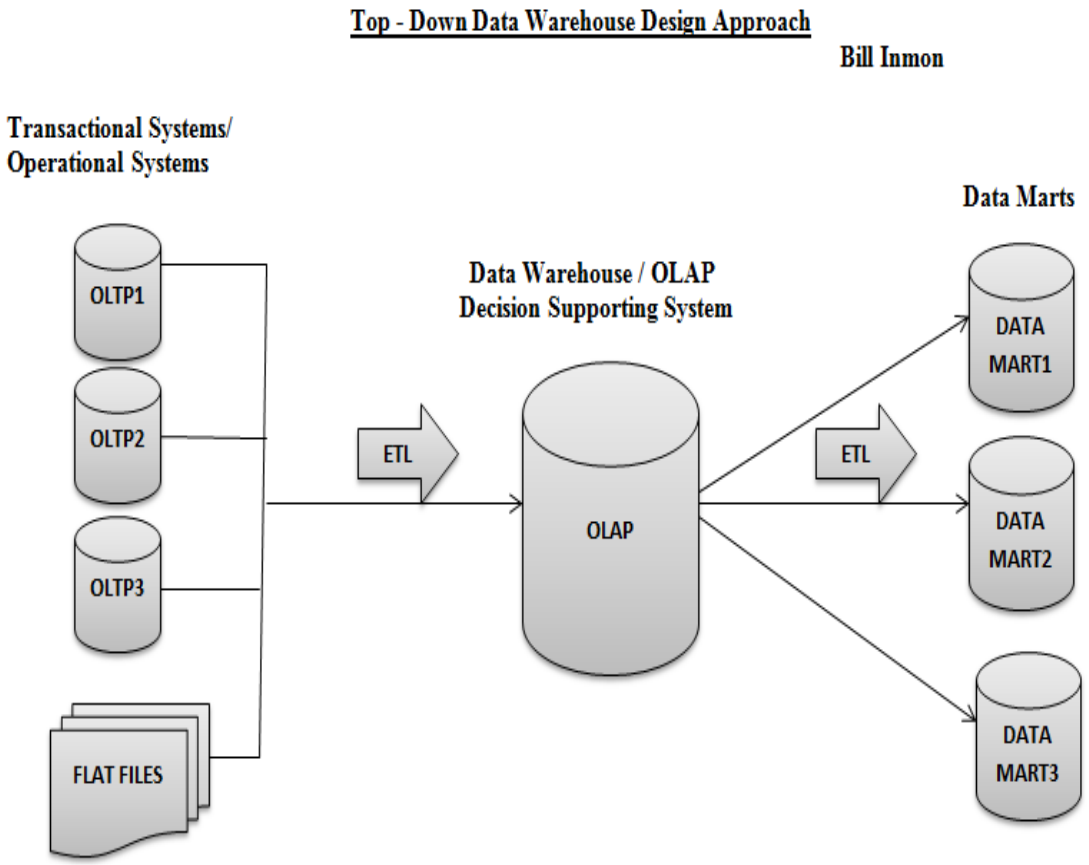
- ✓ A Data Mart is a Subject Oriented database which supports the business needs of specific department business managers.
- ✓ A Data Mart is subset of an Enterprise Data Warehouse.
- ✓ A Data Mart is a single Subject View and integration of multiple subject views is called an Enterprise Data Warehouse.
- ✓ A Data Warehouse is a database provides enterprise business view and Data Mart is also a database provides department specific business view.
- ✓ A Data Mart is known as High Performance Query Structure (HPQS).

Data Warehouse Design Approaches

1. Top - Down Data Warehouse Design Approach
2. Bottom - Up Data Warehouse Design Approach

Top-Down Data Warehouse Design Approach

--Bill Inmon



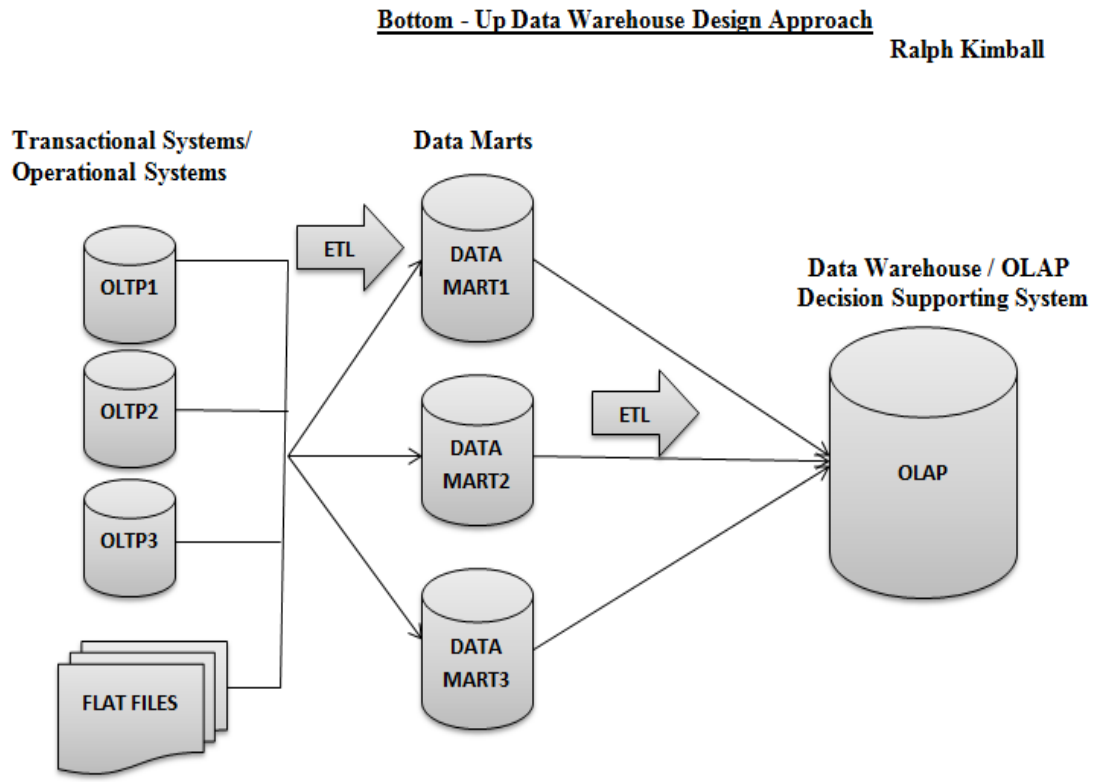
According to **Bill Inmon** first we need design an Enterprise Data Warehouse, from EDWH design the Subject Oriented, department specific databases known Data Marts are Design.

Dependent Data Marts

In Top - Down DWH design approach a Data Mart Development is dependent on EDWH hence such Data Marts are known as dependent Data Marts.

Bottom - Up Data Warehouse Design Approach

--Ralph Kimball

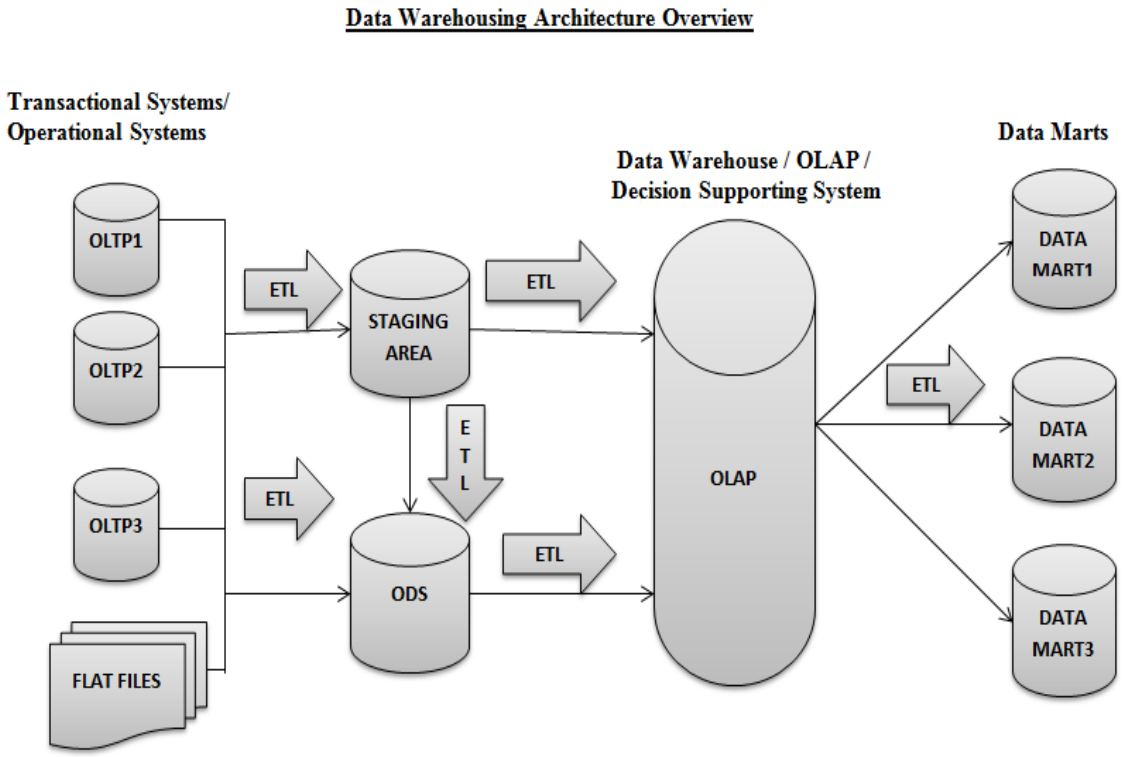


According to **Ralph Kimball** first we need to design subject specific databases know as Data Marts and the integrate Data Marts into an EDWH.

Independent Data Marts

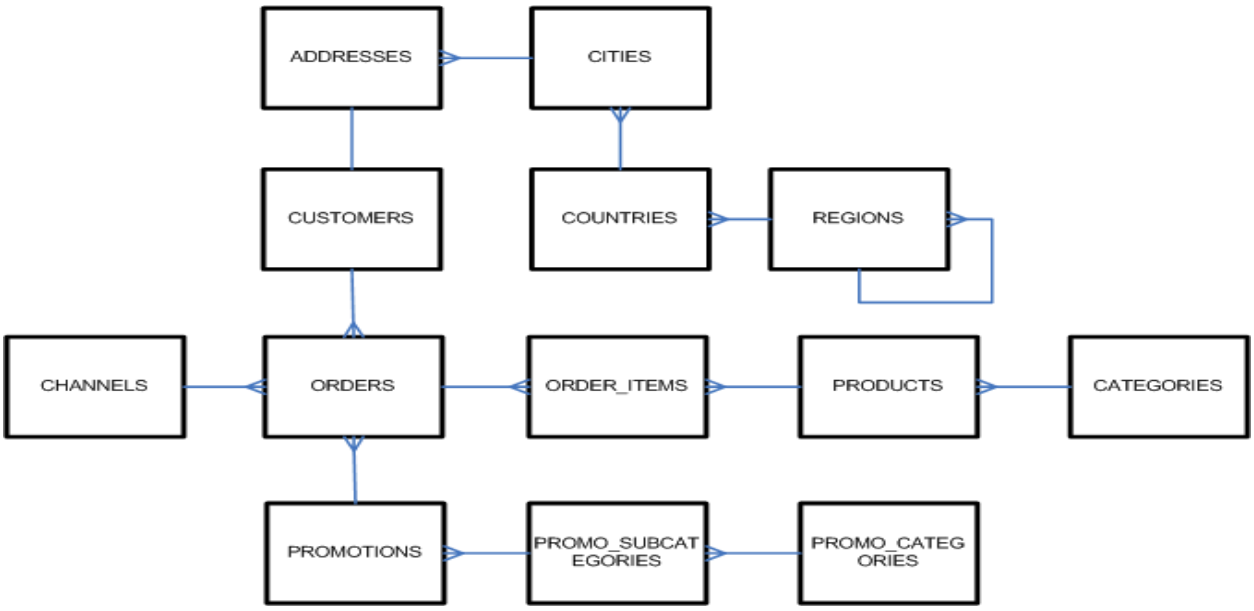
In a Bottom – Up Data Warehouse design approach a Data Mart development is independent on EDWH hence such Data Marts are known as independent Data Marts.

Data Warehousing Architecture Overview



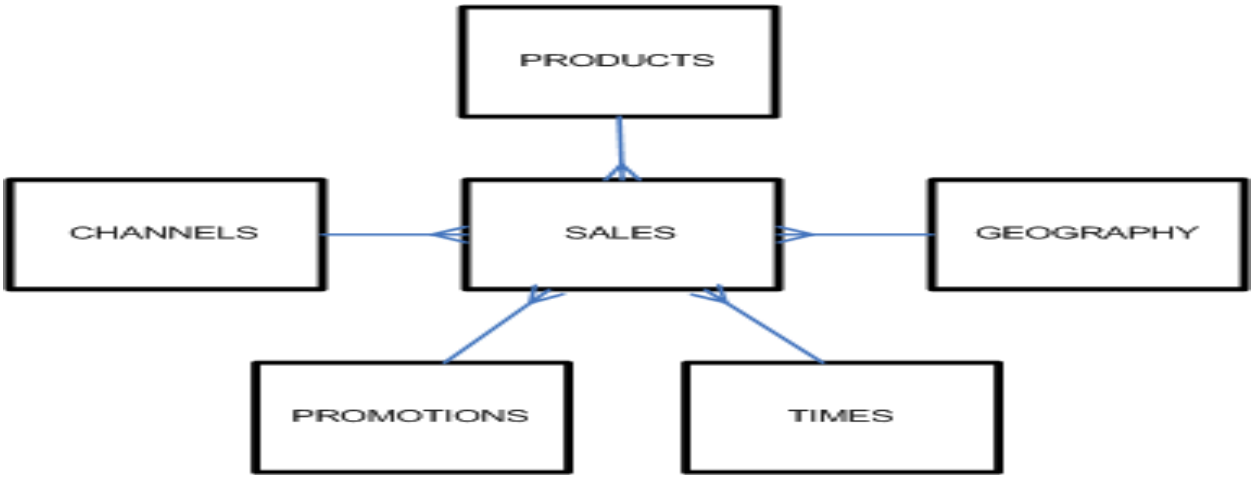
ER Modeling (Entity Relationship Modeling)

- ✓ Entity Relationship Modeling is used to design OLTP databases.
- ✓ Entity – Relationship modeling is a particular design methodology of data modeling wherein the goal of modeling is to **normalize** the data by reducing data redundancy.

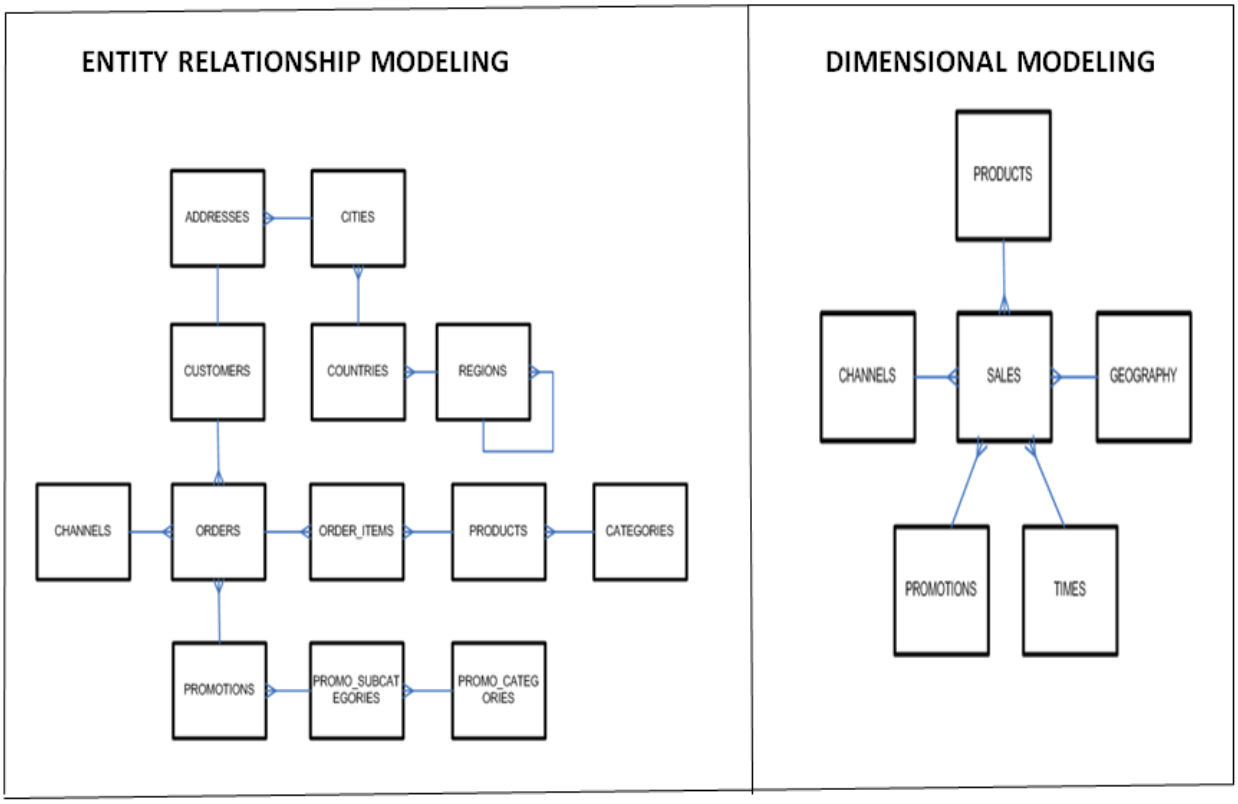


Dimensional Modeling

- ✓ Dimensional Modeling is used to design OLAP databases.
- ✓ Dimensional Modeling is a particular design methodology of data modeling wherein the goal of modeling is to improve **query performance**.



ER Modeling Vs Dimensional Modeling



- ✓ **REGIONS**
- ✓ **COUNTRIES**
- ✓ **CITIES**
- ✓ **GEOGRAPHY**

Entity Relationship Modeling Table Structure

REGION	
REGION_CODE (P.K)	**REGION_NAME**
STH	SOUTH
NTH	NORTH
WST	WEST
EST	EAST

COUNTRY		
COUNTRY_CODE(P.K)	**COUNTRY_NAME**	**REGION_CODE(F.K)**
IND	INDIA	STH
SL	SRI LANKA	STH
PAK	PAKISTAN	STH
SA	SOUTH AFRICA	STH

CITY		
CITY_CODE(P.K)	CITY_NAME	COUNTRY_CODE(F.K)
AP	ANDHRA PRADESH	IND
TG	TELANGANA	IND
TN	TAMIL NADU	IND
KA	KARNATAKA	IND

Dimensional Modeling Table Structure

GEOGRAPHY						
GEOGRAPHY_KEY(P.K)	REGION_CODE	REGION_NAME	COUNTRY_CODE	COUNTRY_NAME	CITY_CODE	CITY_NAME
1001	STH	SOUTH	IND	INDIA	AP	ANDHRA PRADESH
1002	STH	SOUTH	IND	INDIA	TG	TELANGANA
1003	STH	SOUTH	IND	INDIA	TN	TAMIL NADU
1004	STH	SOUTH	IND	INDIA	KA	KARNATAKA

In the process of Data Modeling to design a Data Warehouse using Dimensional Modeling we will be getting any one of the below schemas as a result.

Types of Dimensional Schemas

1. Star Schema
2. Snow Flake Schema
3. Galaxy Schema or Fact Constellation Schema

- ✓ Data Warehouse Schemas are designed using Dimensional Modeling. A Schema consists of dimension tables and fact tables.
- ✓ A dimension table contains dimensions and primary keys.
- ✓ A fact table contains facts and foreign keys to the dimension tables.

Dimension

- ✓ A dimension is a descriptive data which describes the key performance indicators know as facts.

E.g. Product, Customer Name, Date etc.

Fact

- ✓ A fact is something that is measurable or quantifiable.
- ✓ Fact is the metric that business users would use for making business decisions.
- ✓ Measurable information of the business which can be analyzed and has the impact on the business is known as fact.

E.g. Profit, Revenue, Price etc.

Without dimensions we cannot measure the facts.

Profit is a fact, and if I say 1000$ as profit it doesn't have any meaning and we need to add some dimensions to a fact to give a meaningful sentence.

Product, Month and Region are the dimensions and, if I say we had a profit of 1000$ by selling Samsung Mobiles (Product) for the Month of Nov 2015 under the region South, then it give us the meaningful sentence.

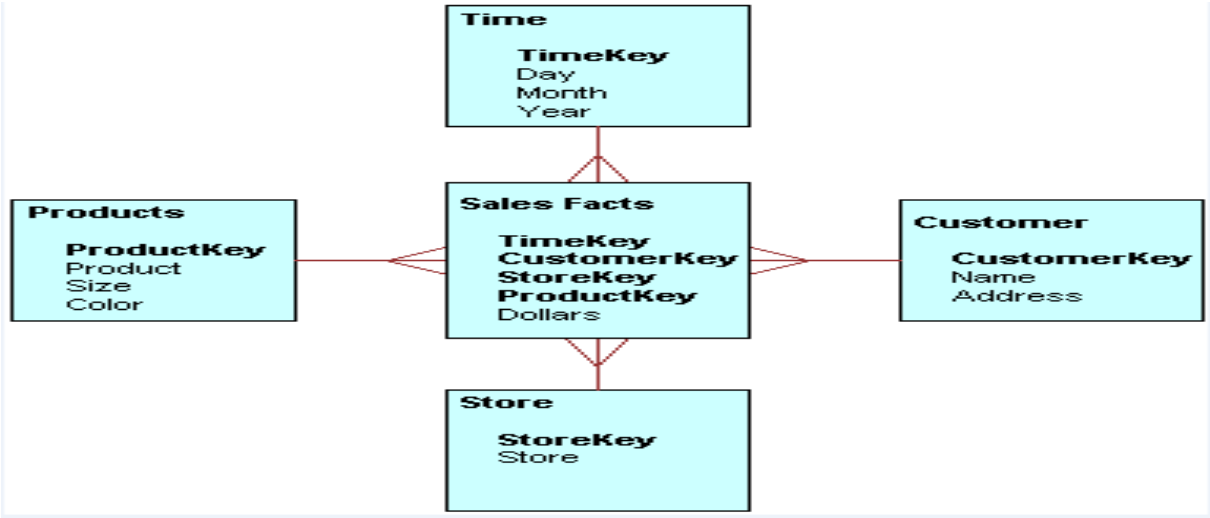
A fact table works with dimension table.

A fact table holds the data to be analyzed, and a dimension table stores data about the ways in which the data in the fact table can be analyzed.

Example

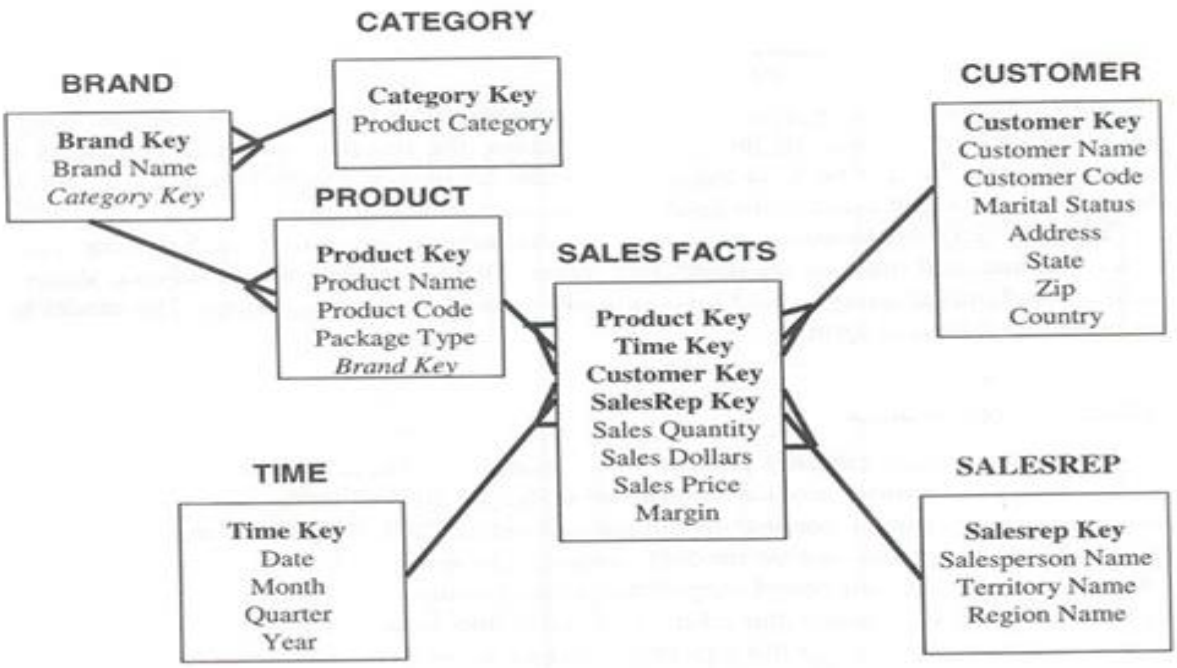
- ✓ South Region Profit
- ✓ Mobile sales in the year 2015 under North Region

Star Schema



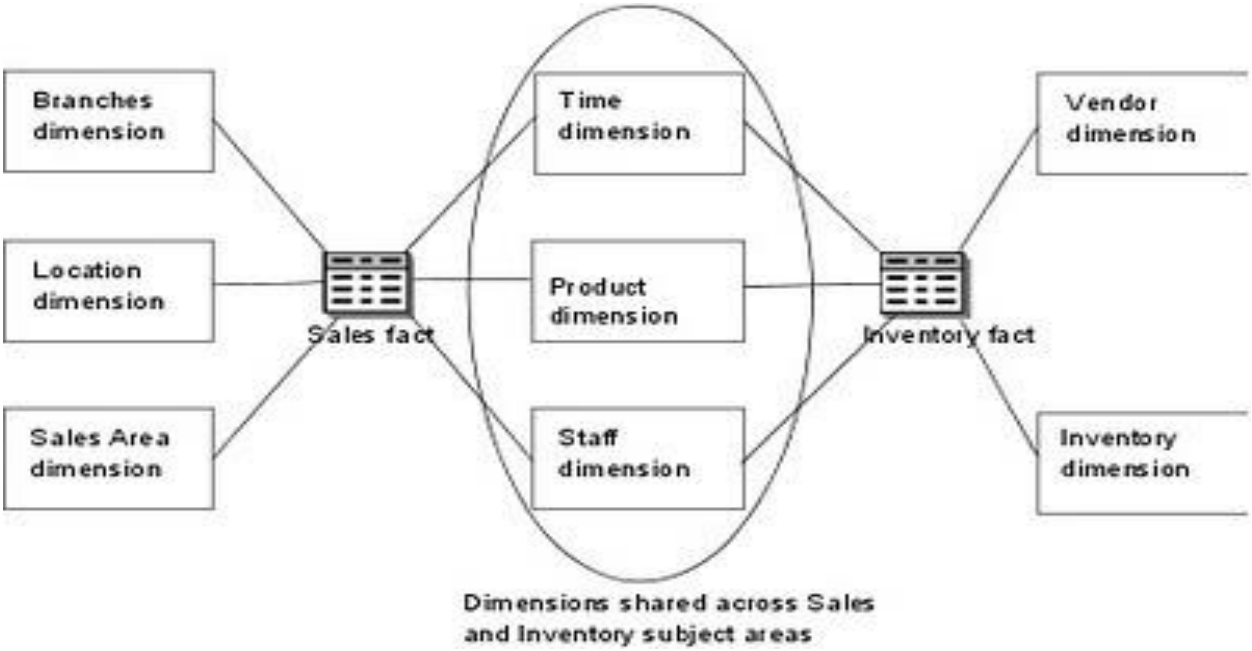
- ✓ A Star Schema is a Data Warehouse database design which contains a centrally located fact table which is surrounded by multiple dimension tables.
- ✓ In star schema all the dimensional tables directly connect to the fact table.

Snow Flake Schema



- ✓ A Snow Flake Schema consists of a fact table surrounded by multiple dimension tables which can be connected to other dimension tables.
- ✓ In snow flake schema some of the dimensions will not directly connect to fact table.
- ✓ When dimension tables stores large number of rows with redundancy of data and space is such an issue, we can use snow flake schema to save space.
- ✓ The tables are partially de normalized in structure.
- ✓ Performance of SQL queries are a bit less when compared to star schema as more number of joins are involved.
- ✓ Data Redundancy is low and occupies less disc space when compared to star schema.
- ✓ In the above figure PRODUCT Dimension is partially normalized.

Fact Constellation Schema or Galaxy Schema



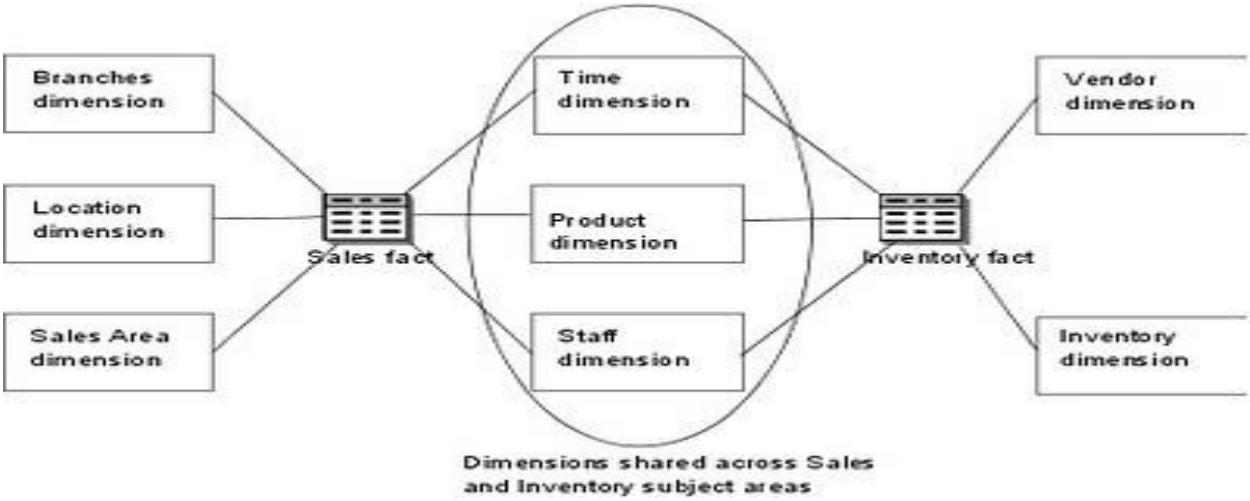
- ✓ This schema is viewed as collection of stars hence called Galaxy Schema or Fact Constellation Schema.
- ✓ In a Galaxy schema a single dimension table is shared with multiple fact tables.

Benefits of Dimensional Model

- ✓ Faster data Retrieval
- ✓ Better Understandability
- ✓ Easy Extensibility

Types of Dimensions and Dimension Tables

Conformed Dimension Table



A Dimension Table that is shared across multiple subject areas or relates to multiple fact tables within the same data warehouse is known as conformed dimension.

Multiple fact tables are used in Data Warehouse that address multiple business functions such as Sales, Inventory and Finance etc.

Each business function will typically have its own Schema (Subject Area) and each schema contains a fact table, some conforming dimension tables and some dimension tables unique to the specific business function.

Date or Time Dimension is a common conformed dimension because its attributes (day, week, month, quarter, year etc.) have the same meaning when joined to any fact table.

Junk Dimension Table

A Junk Dimension is a collection of random transactional codes or flags or text attributes that are unrelated to any particular dimension.

The Junk dimension is simply a structure that provides a convenient place to store the Junk attributes.

Assume that we have a gender dimension and Marital Status dimension as below

Dim_Gender	
Gender_Key	Gender_Value
1001	M
1002	F

Dim_Marital_Status	
Marital_Key	Marital_Value
2001	Y
2002	N

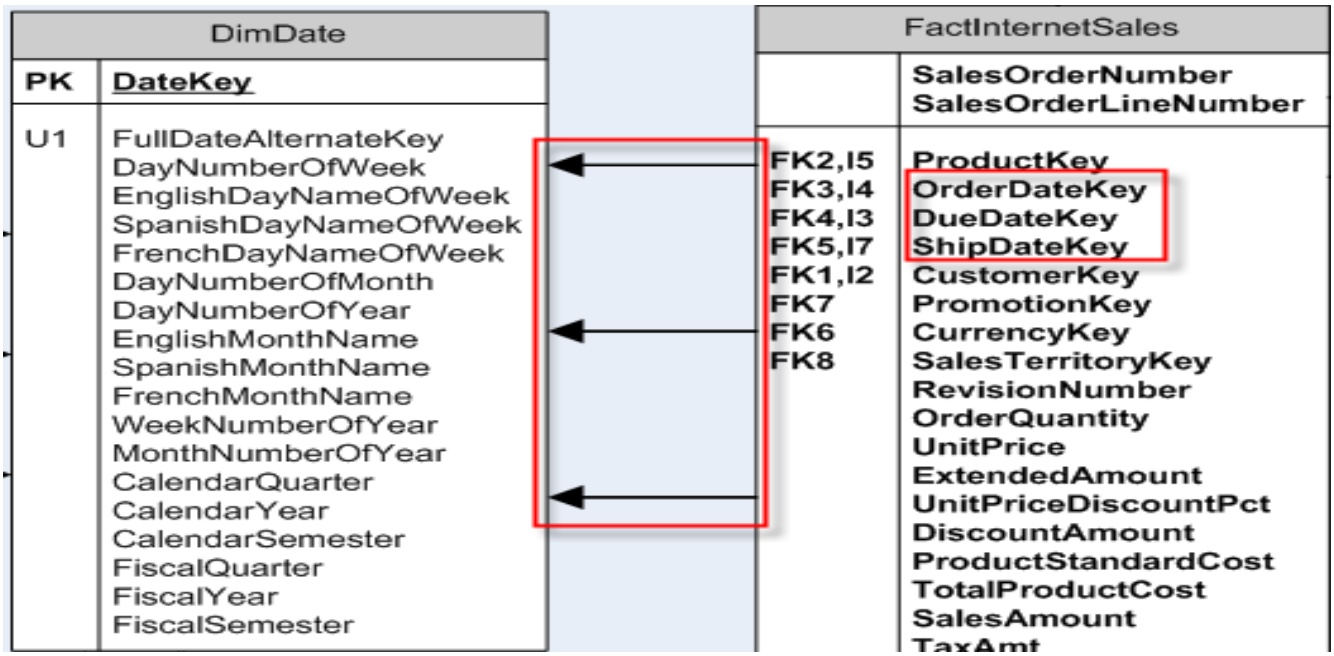
If we have two dimension tables then in the fact table we need to maintain two keys referring to these dimensions.

Instead create a Junk dimension **Dim_Junk** which has all the combinations of Gender and Martial Status (cross join Gender and Martial Status) as below. Now we can maintain only one key in the fact table.

Dim_Junk		
Junk_Key	Gender_value	Marital_status_value
1001	M	Y
1002	M	N
1003	F	Y
1004	F	N

With this Junk dimensional table we will avoid small dimension tables and maintaining them and also we will save space in the fact table by avoiding multiple foreign keys.

Role Playing Dimension Table



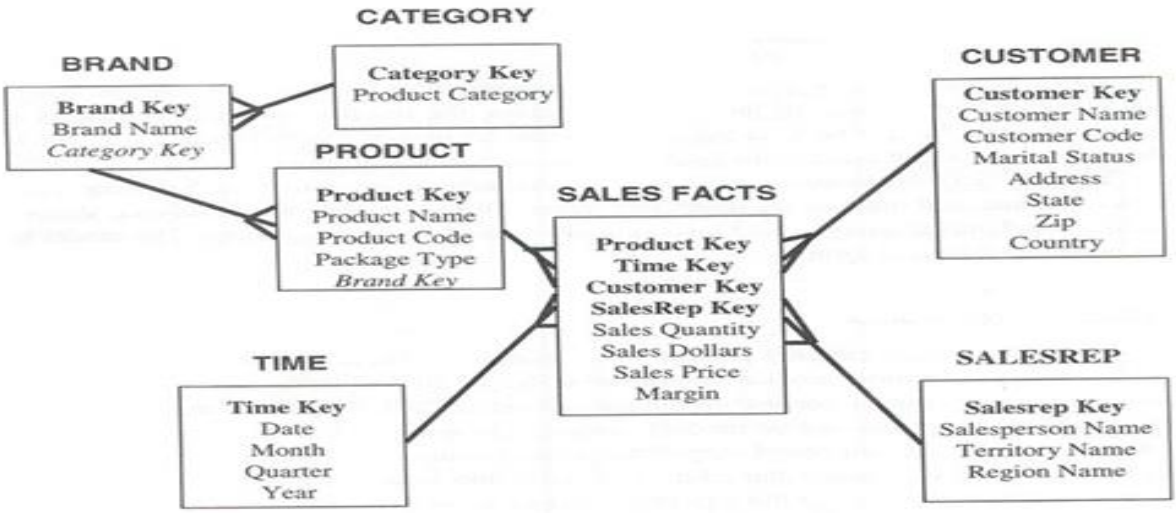
Role playing dimension refers to a dimension that can play different roles in a fact table depending on the content.

If one dimension key is attached to multiple foreign keys in the fact table then such a kind of dimension is known as role playing dimension.

For example, the Date Dimension can be used for the order date, scheduled shipping date, and shipped date in an order fact table.

In the data warehouse you will have a single dimension table for Dates and you will have multiple warehouse foreign keys from the fact table to the same dimension.

Shrunken Dimension

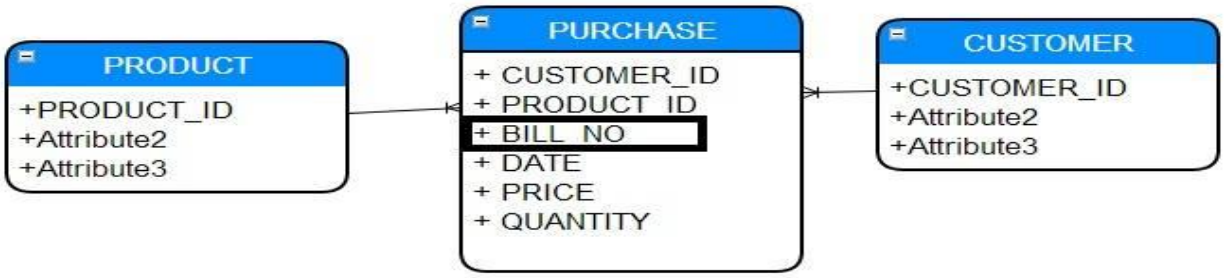


A shrunken dimension is a subset of another dimension.

E.g. In the above figure Product is the shrunken dimension of Brand and Brand is the shrunken dimension of Category.

Similarly Quarter is the Shrunken Dimension of Year and Month is the Shrunken Dimension of Quarter.

Degenerated Dimension



A degenerated dimension is a dimension that is stored in the fact table rather than the dimension table.

A degenerated dimension is a dimension that is derived from fact table.

A dimension such as transaction number, receipt number, Invoice number, Bill No etc. does not have any more associated attributes and hence cannot be designed as a dimension table.

E.g. If you have a dimension that only has Bill number, you would have a 1:1 relation with the fact table.
Do you want to have two tables with a billion rows or one table with a billion rows?
Therefore, this would be a degenerate dimension and Bill number would be stored in the fact table.

Inferred Dimensions

Early arriving facts, late arriving Dimensions, Inferred Dimensions

While loading fact records, a dimension record may not yet be ready. One solution is to generate a surrogate key with Null for all the other attributes. This should technically be called an inferred member, but is often called an inferred dimension.

Based on how frequently the data inside a dimension table changes, we can further classify dimension as

Unchanged Dimension / Static Dimension Table

Static dimensions are not extracted from the original data source, but are created within the context of the data warehouse. A static dimension can be loaded manually for example with Status codes or it can be generated by a procedure for example Date or Time dimension.

Slowly Changing Dimension Table

A dimension is considered to be a slowly changing dimension if its attributes changes over a period of time.

E.g. Employee Dimension Table

Attributes like Age, Location changes over a period of time in Employee Dimension table.

- ✓ SCD Type1 – Contains Current data
- ✓ SCD Type2 – Contains Entire History

Rapidly Changing Dimension

A dimension is considered to be a rapidly changing dimension if one or more of its attributes changes frequently in many rows.

E.g. Currency Dimension

Attribute Currency conversion Rate will be changing frequently in Currency Dimension.

Types of Fact

Additive Fact

Additive facts are facts that can be summed up through all of the dimensions in the fact table.

A sales fact table is a good example for additive facts.

Fact_Sales
Date_Key
Store_Key
Region_Key
Sales
Profit

Date, Store and Region are Dimensions in the fact table, Sales and Profit are facts.

We can measure day wise sales and profit, Store wise sales and Profit, Region wise sales and profit and all will provide us the correct results.

Hence Sales and Profit are additive measures or additive facts.

Semi Additive Fact

Semi Additive facts are facts that can be summed up for some of the dimensions in the fact table, but not the others.

Daily Balances fact table is example for Semi Additive Facts.

Fact_Daily_Balances
Day_Key
Customer_Key
Balance

Day and Customer are the Dimensions in the fact table, Balance is a fact.

We can measure the current balance of the customer and will give the meaningful output.

We can also measure the current balance of Day but it doesn't have any meaning.

Non Additive Fact

Non Additive facts are the facts that cannot be summed up for any of the dimensions present in the fact table.

E.g. Percentages, Ratios are examples for Non Additive facts.

Types of Fact Tables

Fact less Fact Table

A fact table without any facts and has only foreign keys to the dimension tables are known as fact less fact table.

E.g. A fact table which has only product key and date key is a fact less fact table. There are no measures in this table. But still you can get the number of products sold over a period of time.

Detailed Fact Table

A fact tables which stores the details of the transactions is known as detailed fact table.

Additive Fact Table or Cumulative Fact Table

A fact table which stores summary of transactions is known as Additive fact table or Cumulative fact table.

E.g. This fact table may describe the total sales by Product or by Store or by Day.

Granularity

Granularity refers to the level of detail of the data stored in any tables of a data warehouse.

High granularity refers to data that is at or near the transaction level. Data that is at the transaction level is usually referred to as atomic level data.
Low granularity refers to data that is summarized or aggregated, usually from the atomic level data.

Surrogate Key

Surrogate Key is sequentially generated meaningless unique number attached with each and every record in a table in any Data Warehouse.

This Surrogate Key is generated using database sequences or Sequence Generator Transformation in Informatica.

Primary Key Vs. Surrogate Key

Both are primary keys. A natural primary key has some meaning and a surrogate primary key has no meaning.

A Primary key would be anything that has some meaning and will be used to identify the row uniquely.

E.g. EmpNo – A7164500
A7164501

A Surrogate key is something which is having the sequence generated numbers with no meaning, and just used to identify the row uniquely.

E.g. Sequences 10001
10002
10003

What is ETL?

ETL stands for Extraction, Transformation and Loading.

We can perform this ETL process in two ways.

1. Code Based ETL
2. GUI Based ETL

Code Based ETL

An ETL application can be developed using programming languages such as SQL and PL/SQL is said to be a code based ETL.

GUI Based ETL

An ETL application can be designed using simple Graphical User Interface, Point and Click techniques is said to be GUI based ETL.

Examples for GUI based ETL tools

- ✓ Informatica
- ✓ Datastage
- ✓ ODI (Oracle Data Integrator)
- ✓ Ab Initio

Data Acquisition

Data Acquisition is the process of Extracting, Transforming and Loading the data.

Data Acquisition will be having below processes

- ✓ Data Extraction (E)
- ✓ Data Transformation (T)
- ✓ Data Loading (L)

Data Extraction

It is the process of reading or extracting the data from various types of source systems.

Data Transformation

It is the process of transforming the data into the required business format.

The following are the some of the data processing activities takes place.

- ✓ Data Cleansing – Removing unwanted data
- ✓ Data Merging – Joins, Union
- ✓ Data Scrubbing – Deriving New Attributes
- ✓ Data Aggregation – Summaries of the detailed data using aggregate functions.

Data Loading

It is the process of inserting the data into the destination systems.

There are two types of data loading

- ✓ Initial Load or Full Load
- ✓ Incremental Load Or Delta Load

Initial Load

It's the process of inserting the data records into an empty target table.
At first time load all the required data inserts into destination table.

Incremental Load

It's the process of inserting only new records after initial load or any Load.