**Database**: Database is collection of information organized in a way to easily access, manage and update. This is done in a tabular format which has columns and rows where in columns have attributes and rows are the instances of the table.

**Data Modeling**

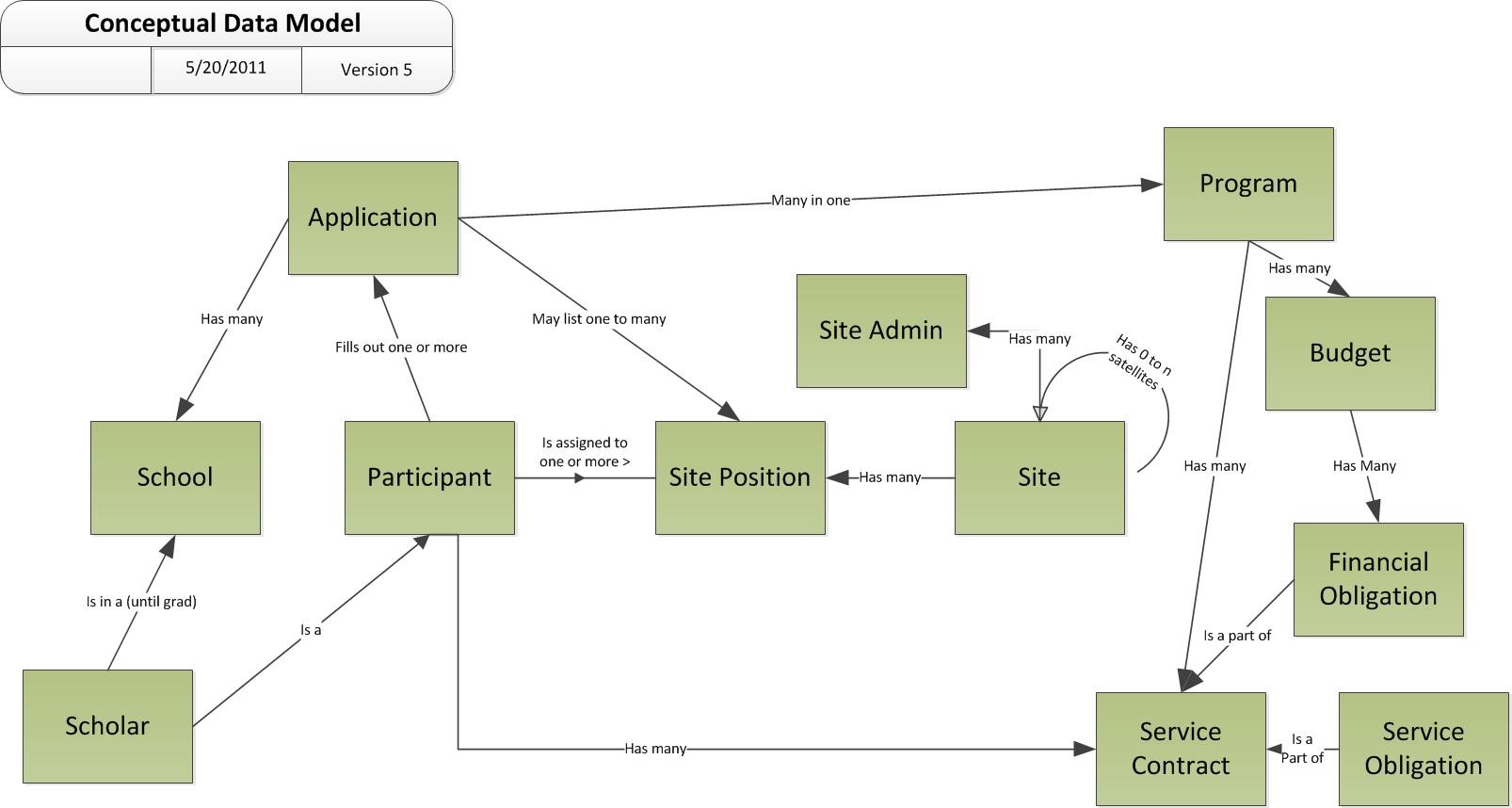
This is a process through which we define and analyze the data requirements needed to support the business processes of the corresponding information system. As a part of this process we define data entities and their relationships in a system to build a strong data base design and architecture for the information system.

We create 3 models as we progress from requirements to the actual database which is

to be used by the information system.

**Conceptual Model:** The data requirements are initially recognized as the conceptual model where we capture the technology independent specifications that is used to discuss the initial requirements with the business owners. Three important elements of conceptual model are

* It includes important entities and their relationships.
* No attributes are defined.
* No primary keys are identified.



**Logical Data Model:**  Implementation of one conceptual model may require multiple logical data models, a logical data model describes the data in as much detail as possible without any details of they are implemented physically on a database.

Key features of a logical data model are

* Includes as all entities and relationships
* All attributes for each entity are specified
* The primary key for each entity is specified
* Foreign keys are specified (Resolve many to many relationships)
* Normalization is done

**Physical data model** represents how the model will be built in the database. A physical database model shows all table structures, including column name, column data type, column constraints, primary key, foreign key, and relationships between tables. Features of a physical data model include:

* Specification of all tables and columns.
* Foreign keys are used to identify relationships between tables.
* De-normalization may occur based on user requirements.
* Physical considerations may cause the physical data model to be quite different from the logical data model.
* Physical data model will be different for different RDBMS. For example, data type for a column may be different between MYSQL and SQL Server.

The steps for physical data model design are as follows:

1. Convert entities into table
2. Convert foreign keys into relationships.
3. Convert attributes into columns.
4. Modify the physical
5. Data model based on physical constraints / requirements.

**Data Entity**: This is thing that holds the information (table), Attributes (column) define the facts about the entity. There are multiple types of entities

* **Strong Entity:** An entity which can exist independently in a data base

Department can exist without an employee

* **Weak Entity:** An Entity dependent on a strong entity to exist in a data base

**Employee cannot exist without a department**

* **ID Dependent Weak Entity:** if the relationship between Strong Entity is identifier (Primary Key) dependent then it is called ID dependent. So, Primary Key of strong entity is also primary key for the weak entity.
* **Non-ID dependent weak entity:** if the relationship between Strong Entity is non-id dependent then it is called non-ID dependent. So, Primary Key of strong entity will appear as a foreign key in the weak entity.
* **Exclusive Super type and Subtype Entities (IS A)**:
* **Inclusive Super type and Subtype Entities (IS A)**:
* **Reccursive**

**Attributes:**

**]]Single valued Attributes :** An attribute, that has a single value for a particular entity is known as single valued attributes. For example, age of a employee entity.  
**Multi valued Attributes :**An attributes that may have multiple values for the same entity is known as multi valued attributes. For example colors of a car entity.  
**Compound Attribute/Composite Attribute :** Attribute can be subdivided into two or more other Attribute. For Example, Name can be divided into First name, Middle name and Last name.  
**Simple Attributes/Atomic Attributes :**The attributes which cannot be divided into smaller subparts are called simple or atomic attributes. For example, age of employee entity  
**Stored Attribute :**An attribute, which cannot be derived from other attribute, is known as stored attribute. For example, BirthDate of employee.  
**Derived Attribute :** Attributes derived from other stored attribute. For example age from Date of Birth and Today’s date.  
·         An attribute can be derived from a single attribute. Example age from DOB and current date.  
·         An attribute can be derived from multiple attribute.  
·         An entity can be derived from a separate table. Example,  
RNO  
Name  
DeptNO  
1  
A  
1  
2  
B  
1  
3  
C  
2  
4  
D  
3  
5  
E  
1  
⇒  
DNO  
Dname  
1  
CSE  
2  
IT  
3  
ECE  
**Complex Attributes :** If an attribute fr an entity, is built using composite and multivalued attributes, then these attributes are called complex attributes. For example, a person can have more than one residence and each residence can have multiple phones, an addressphone for a person entity can be specified as –  
**{Addressphone (phone {(Area Code, Phone Number)}, Address(Sector Address (Sector Number,House Number), City, State, Pin))}**  
Here {} are used to enclose multivalued attributes and () are used to enclose composite attributes with comma separating individual attributes./  
**Key Attribute :**

**Simple Attribute:**

**Relationships:**  Entities are related to each other, Database has to capture all the relationships. For example customer and products are strong entities. But the customer buy products and we need to capture that as well.

* **Association (All the table exist independently)**
* **Generalization (bottom up)**
* **Specialization (Top down)**
* **Aggregation (child can exist with out parent object)(Department , Teacher)**
* **Composition(cannot exist without a parent) also called a s death relation.**
* **Recursive (Customer referrals)**

**Cardinality:** It is the numerical mapping between the entities, this describes ‘how many’ of one entity are related to ‘how many’ of other entity. There are three types of cardinalities

* One to one
* One to many
* Many to one

**ER Diagrams:** An Entity-Relationship (ER) diagram provides a graphical model of the things that the organization deals with (entities) and how these things are related to one another (relationships). An ER diagram is a high-level, logical model used by both end users and database designers to document the data requirements of an organization.

**Keys:**

**Primary Key:**

A primary key is a special constraint on a column. A primary key ensures that the column designated has no NULL values, and that every value is unique. Physically, a primary key is implemented by the database system using a unique index, and all the columns in the primary key must have been declared NOT NULL. A table may have only one primary key.

**Natural Key:**

A natural key is a naturally occurring descriptor of the data and one of a table's attributes that has no duplicate values. When you use a natural key as a table's primary key, each of the table's rows is uniquely identified

**Candidate Key:**

A candidate key is a column, or set of columns, in a table that can uniquely identify any database record without referring to any other data. Each table may have one or more candidate keys, but one candidate key is special, and it is called the primary key.

**Super Key:**

A super key is a combination of columns that uniquely identifies any row within a relational database management system (RDBMS) table. A candidate key is a closely related concept where the superkey is reduced to the minimum number of columns required to uniquely identify each row.

**Alternate key:**

All the other candidate keys which are not chosen as the primary key become the alternate key of the data.

**Surrogate Key:**

A surrogate key has no business value. The most common type of surrogate key is an incrementing integer

**Data Integrity:**

Data integrity refers to maintaining and assuring the accuracy and consistency of data over its entire life-cycle, and is a critical aspect to the design, implementation and usage of any system which stores, processes, or retrieves data.

There are two ways we can enforce integrity, one is through putting constraints and other is using Triggers. However, using constraints is a preferred way.

**Data Constraints**: A constraint is a limitation that you place on the data that users can enter into a column or group of columns. A constraint is part of the table definition; you can implement constraints when you create the table or later. There can be ***column level constraints*** or ***table level constraints.***

**Types of constraints**

* ***Primary Key Constraint***
* ***Unique Constraint***
* ***Foreign Key Constraints***
* ***Default Constraints***
* ***Check Constraints***

**Data trigger**: A database trigger is special stored procedure that is run when specific actions occur within a database. Most triggers are defined to run when changes are made to a table's data.

Reference: <http://sqlmag.com/database-performance-tuning/certifiably-sql-data-constraints-and-integrity>

**Anomalies:**

In relational database design, we

not only want to create a structure that stores all of the data, but we also want to do it in a way that it minimizes potential errors when we work with the data. The default language for accessing data from a relational database is SQL. In particular, SQL can be used to manipulate data in the following ways: insert new data, delete unwanted data, and update existing data. Similarly, in an un-normalized design, there are 3 problems that can occur when we work with the data:

* **INSERT ANOMALY**: This refers to the situation when it is impossible to insert certain types of data into the database.
* **DELETE ANOMALY:** The deletion of data leads to unintended loss of additional data, data that we had wished to preserve.
* **UPDATE ANOMALY:** This refers to the situation where updating the value of a column leads to database inconsistencies (i.e., different rows on the table have different values).

**Normalization:**  To address the 3 problems above, we go through the process of normalization. When we go through the normalization process, we increase the number of tables in the database, while decreasing the amount of data stored in each table. There are several different levels of database normalization.

There are 5 normal forms but all we need to understand is till third normal form. 4 and 5 are theoretical

**First Normal Form:**

A database is in first normal form if it satisfies the following conditions:

* Contains only atomic values
* There are no repeating groups

An atomic value is a value that cannot be divided. For example, in the table shown below, the values in the [Color] column in the first row can be divided into "red" and "green", hence [TABLE\_PRODUCT] is not in 1NF.

A repeating group means that a table contains two or more columns that are closely related. For example, a table that records data on a book and its author(s) with the following columns: [Book ID], [Author 1], [Author 2], [Author 3] is not in 1NF because [Author 1], [Author 2], and [Author 3] are all repeating the sa

me attribute.

**Second Normal Form:**

A database is in second normal form if it satisfies the following conditions:

* It is in first normal form
* It should be independent of partial dependency
* All non-primary key attributes are fully functionally dependent on the primary key

**Third Normal Form:**

A database is in third normal form if it satisfies the following conditions:

* It is in second normal form
* All non-primary key attributes have non-transitive functional dependency on the primary key
* (it should not have any transitive dependency
* If there is no partial or transitive dependency then we can say a table is in 3rd normal form
* Transitive dependency: when a non-prime attribute depends on another non-prime attributes then it is called transitive dependency

**Data Profiling:**

Data profiling is a process of examining available data in the existing data source and collect statistics and information about the data, improper data profiling can lead to poor quality data and this could adversely affect the overall purpose of the information system.

single version of truth:

There are multiple things one does as a part of Data Profiling at the attribute level.

* Completeness Analysis
  + How often is a given attribute populated, versus blank or null?
* Uniqueness Analysis
  + How many unique (distinct) values are found for a given attribute across all records? Are there duplicates? Should there be?
* Values Distribution Analysis
  + What is the distribution of records across different values for a given attribute?
* Range Analysis
  + What are the minimum, maximum, average and median values found for a given attribute?
* Pattern Analysis
  + What formats were found for a given attribute, and what is the distribution of records across these formats?

The purpose of this analysis and statistics is to

* Find out whether existing data can easily be used for other purposes
* Analyze the customer’s characteristics, value and pattern in their changing behaviors.
* Improve the ability to search the data by tagging it with keywords, descriptions, or assigning it to a category
* Give metrics on data quality including whether the data conforms to particular standards or patterns
* Assess the risk involved in integrating data for new applications, including the challenges of joins
* Discover metadata of the source database, including value patterns and distributions, key candidates, foreign-key candidates, and functional dependencies
* Assess whether known metadata accurately describes the actual values in the source database
* Understanding data challenges early in any data intensive project, so that late project surprises are
* avoided. Finding data problems late in the project can lead to delays and cost overruns.
* Have an enterprise view of all data, for uses such as master data management where key data is needed, or data governance for improving data quality

Data profiling can add value in a wide variety of situations.  The basic litmus test is, “Is the quality of data important for this initiative?”  If the answer is “yes”, then data profiling can help as it can quickly and thoroughly unveil the true content and structure of your data.

Some example scenarios include:

* Data Warehousing / Business Intelligence (DW/BI) projects
  + These projects involve gathering data from disparate systems for the purpose of reporting and analysis. Data profiling can help ensure project success by:
    - Identifying data quality issues that must be corrected in the source system
    - Identifying issues that can be corrected in ETL processing
    - Discovering unanticipated business rules
    - Even potentially providing a “no-go” decision on the project as a whole
* Data conversion / Migration projects
  + These involve moving data from a legacy system to a new system.  Data profiling can help reduce project risk by:
    - Identifying data quality issues that must be handled in the code that moves data from the legacy system to the new system
    - Identifying data issues that may require a change to the target (new) system
* Source System Data Quality Initiative
  + These projects endeavor to assess and improve the data quality of a given source system, seeking to fix existing issues as well as avoid those issues in the future. Data profiling can help maximize project ROI by:
    - Highlighting the areas within the system suffering from the most serious and/or numerous data quality issues
    - Identifying issues that may be the result of bad user input or errant system interfaces

***Tools used are***

* ***Informatica Data quality***
* ***Datiris Profiler***

**Data Mapping:**

Data mapping specification is a special type of data dictionary that shows how data in one information system maps with the data in another information system.

**Data Mapping will be valuable in**

***Data Migration –*** When source data is migrated to a new target data repository.Data migration is necessary when an organization decides to use a new computing system or DBMS that is incompatible with the current system. (Ex: Legacy to ERP). In this type, the data from the old system is completed deleted after transferring to the new system.

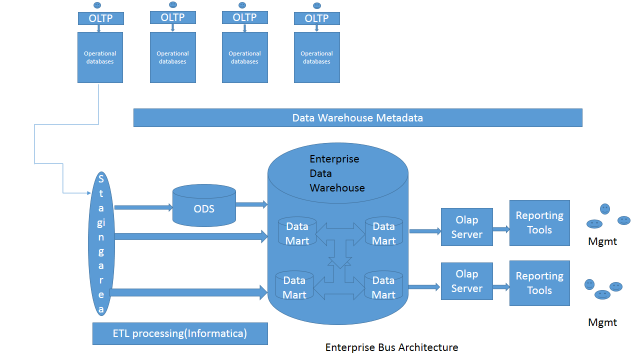
***Data Integration –*** When source data is sent to a target data repository on a regular basis and the two data sources do not share a common data model. The integration can be hourly, daily, weekly, monthly, or even in real-time as is typically required for system integration

Data Mapping Specifications constitute **Source attributes, types, description and corresponding list of attributes, types and description\**

***Tools used are***

* ***Altoa Mapforce***
* ***MetaDapper***
* ***Mapping Manager***

Data warehouse Architecture:



**Data warehousing:** *A data warehouse is a subject-oriented, integrated, time-variant and non-volatile collection of data in support of management's decision making process.*

A Data warehouse is a place where you store the data from multiple data sources to be used for historical and trend analysis reporting.

**Architecture Types:**

There are Five types of Architecture:

* + Centralized Architecture.(Normalized relational data)
  + Independent Data Mart Architecture.(atomic summarized data)
  + Data Mart Bus architecture(Dimensional Modeling)
  + Hub and Spoke(Normalized data)
  + Federated Architecture(Connects the existing data marts/ integrates the common items with keys)

**Implementation Types:**

* + Top-down Approach (High initial cost, Time Consuming, Easy Maintenance)
  + Bottom Up Approach (Low initial cost, Less time taking, Difficult Maintenance)

**Data modeling:** The Data in the Data warehouse can be modeled in two ways.

* ER modeling
* Dimension modeling

**Dimensional Modeling**: Dimensional modeling uses three basic concepts: measures, facts, and dimensions. Dimensional modeling is powerful in representing the requirements of the business user in the context of database tables.

**Facts:** A Fact is a collection of related data items, consisting of measure and context of the data. Each fact typically represents a business item, business transaction or an event that could be analyzed for the business process.

**Dimensions**: A Dimension is a collection of members or units of same type of views. Every data point in the fact table is associated with one and only one member from each of the multiple dimensions. Dimensions are the parameters over which we want to perform Online Analytical Processing (OLAP).

**Dimension Members:** A dimension contains many dimension members. A dimension member is a distinct name or identifier used to determine a data items position.

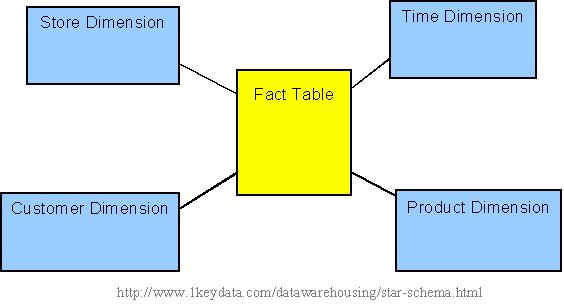
**Dimension Hierarchy:** We can arrange the members of a dimension into one or more hierarchies. Each hierarchy can also have multiple hierarchy levels. Every member of a dimension does not locate on one hierarchy structure.

***\*\* The most popular way of visualizing a dimensional model is to draw a cube. We can represent a three-dimensional model using a cube.***

**There are two models that can be used in Dimensional Data modeling:**

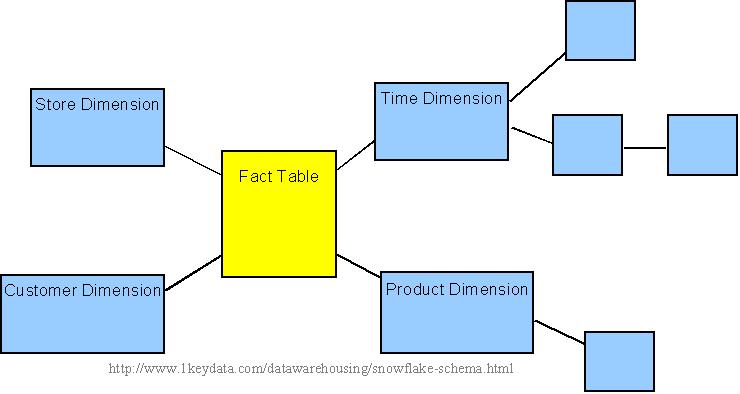
**Star Schema:** *The star model is the basic structure for a dimensional model. It typically has one large central table (called the fact table) and a set of smaller tables (called the dimension tables) arranged in a radial pattern around the fact table.*

In the star schema design, a single object (the fact table) sits in the middle and is radically connected to other surrounding objects (dimension lookup tables) like a star. Each dimension is represented as a single table. The primary key in each dimension table is related to a foreign key in the fact table.



**Snow Flake Schema:** *Dimensional modeling typically begins by identifying facts and dimensions, after the business requirements have been gathered. The initial dimensional model is usually star-like in appearance, with one fact in the center and one level of several dimensions around it. The snowflake model is the result of decomposing one or more of the dimensions, which sometimes have hierarchies themselves. We can define the many-to-one relationships among members within a dimension table as a separate dimension table, forming a hierarchy.*

The snowflake schema is an extension of the star schema, where each point of the star explodes into more points. In a star schema, each dimension is represented by a single dimensional table, whereas in a snowflake schema, that dimensional table is normalized into multiple lookup tables, each representing a level in the dimensional hierarchy.



**Types of Data models:**

## Types of Data Model

They fall into three broad categories –

* Object Based Data Models
* Physical Data Models
* Record Based Data Models

The object based and record based data models are used to describe data at the conceptual and external levels, the physical data model is used to· describe data at the internal level.

### Object Based Data Models

Object based data models use concepts such as entities, attributes, and relationships. An entity is a distinct object (a person, place, concept, and event) in the organization that is to be represented in the database. An attribute is a property that describes some aspect of the object that we wish to record, and a relationship is an association between entities.

Some of the more common types of object based data model are –

* Entity-Relationship
* Object Oriented
* Semantic
* Functional

The Entity-Relationship model has emerged as one of the main techniques for modeling database design and forms the basis for the database design methodology. The object oriented data model extends the definition of an entity to include, not only the attributes that describe the state of the object but also the actions that are associated with the object, that is, its behavior. The object is said to encapsulate both state and behavior. Entities in semantic systems represent the equivalent of a record in a relational system or an object in an OO system but they do not include behaviour (methods). They are abstractions 'used to represent real world (e.g. customer) or conceptual (e.g. bank account) objects. The functional data model is now almost twenty years old. The original idea was to' view the database as a collection of extensionally defined functions and to use a functional language for querying the database.

### Physical Data Models

Physical data models describe how data is stored in the computer, representing information such as record structures, record ordering, and access paths. There are not as many physical data models as logical data models, the most common one being the Unifying Model.

### Record Based Logical Models

Record based logical models are used in describing data at the logical and view levels. In contrast to object based data models, they are used to specify the overall logical structure of the database and to provide a higher-level description of the implementation. Record based models are so named because the database is structured in fixed format records of several types. Each record type defines a fixed number of fields, or attributes, and each field is usually of a fixed length.

The three most widely accepted record based data models are –

* Hierarchical Model
* Network Model
* Relational Model

**Implementation Process:**

**Extraction:**

Extraction is process of extracting the data from a source system which are usually the transaction process applications and storing them in a staging area for further processing into Data warehouse environment. The staging area is temporary database which is transient in nature. The data in this space would be deleted before performing every extraction process

Designing and creating an Extraction process is one of the most time consuming tasks in the entire ETL process. The source system could be complex and poorly documents, making the decisions on how to pull the data, how much data to pull, what components of the data must we extract etc…  The data has to be extracted normally not only once, but several times in a periodic manner to supply all changed data to the warehouse and keep it up-to-date. Moreover, the source system typically cannot be modified, nor can its performance or availability be adjusted, to accommodate the needs of the data warehouse extraction process.

Designing this extraction process means making decisions about the following two main aspects:

* Which extraction method do I choose?
* This influences the source system, the transportation process, and the time needed for refreshing the warehouse.
* How do I provide the extracted data for further processing?
* This influences the transportation method, and the need for cleaning and transforming the data.

**Extraction Methods:**

**Logical Extraction Method**

Full extraction (Big bang): The data is extracted completely from the source system. Since this extraction reflects all the data currently available on the source system, there’s no need to keep track of changes to the data source since the last successful extraction.

Incremental Extraction: At a specific point in time, only the data that has changed since a well-defined event back in history will be extracted.

**Physical Extraction methods**

Offline extraction: The data is not extracted directly from the source system but is staged explicitly outside the original source system. The data already has an existing structure (for example, redo logs, archive logs or transportable table spaces) or was created by an extraction routine.

Online extraction: The data is extracted directly from the source system itself. The extraction process can connect directly to the source system to access the source tables themselves or to an intermediate system that stores the data in a preconfigured manner (for example, snapshot logs or change tables). Note that the intermediate system is not necessarily physically different from the source system.

Change Data Capture: If a data warehouse extracts data from an operational system on a nightly basis, then the data warehouse requires only the data that has changed since the last extraction (that is, the data that has been modified in the past 24 hours).

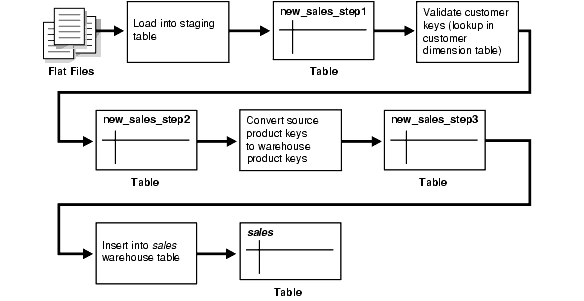
**\*\*\*Slowly Changing Dimensions**

One can use Time Stamps, Triggers

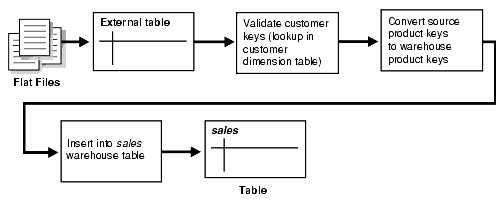
**Transformation:** Data transformations are often the most complex and, in terms of processing time, the most costly part of the extraction, transformation, and loading (ETL) process. They can range from simple data conversions to extremely complex data scrubbing techniques.

From an architectural perspective you can transform data in two ways

**Multistage Transformation:** The data transformation logic for most data warehouses consists of multiple steps. For example, in transforming new records to be inserted into a sales table, there may be separate logical transformation steps to validate each dimension key.



Pipelined Transformation:



Transformation tasks:

* applying business rules (so-called derivations, e.g., calculating new measures and dimensions),
* Cleaning (e.g., mapping NULL to 0 or “Male” to “M” and “Female” to “F” etc.),
* filtering (e.g., selecting only certain columns to load),
* splitting a column into multiple columns and vice versa,
* joining together data from multiple sources (e.g., lookup, merge),
* transposing rows and columns,
* applying any kind of simple or complex data validation (e.g., if the first 3 columns in a row are empty then reject the row from processing)

**Loading Mechanisms:**

* [Loading a Data Warehouse with SQL\*Loader](https://docs.oracle.com/cd/B19306_01/server.102/b14223/transform.htm#BABJDJIC)
* [Loading a Data Warehouse with External Tables](https://docs.oracle.com/cd/B19306_01/server.102/b14223/transform.htm#i1006296)
* [Loading a Data Warehouse with OCI and Direct-Path APIs](https://docs.oracle.com/cd/B19306_01/server.102/b14223/transform.htm#i1006366)
* [Loading a Data Warehouse with Export/Import](https://docs.oracle.com/cd/B19306_01/server.102/b14223/transform.htm#i1006369)

**Data Analysis:**

Data warehouse is built to provide an ease to access source of high quality data and perform analysis to help devise management decisions and strategies. There are three types of Data analysis.

* 1. Query and Reporting
  2. Multi-dimensional analysis
  3. Data mining.

**Query and Reporting:** This could be done on the by storing the data in the data warehouse in the simple 3rd normal form, run required sql queries, manipulate and perform simple calculations on the report it using the front end reporting tools.

**Multi-dimensional Analysis**: Multidimensional analysis has become a popular way to extend the capabilities of query and reporting. That is, rather than submitting multiple queries, data is structured to enable fast and easy access to answers to the questions that are typically asked.

The Data is modelled in different dimensions thereby pre-building results to various queries in the form of a cube. There by reducing the effort of querying repetitively. This enables both the drill down operations and Roll up operations.

**Data Mining**: Data mining is a relatively new data analysis technique. It is very different from query and reporting and multidimensional analysis in that is uses what is called a discovery technique. That is, you do not ask a particular question of the data but rather use specific algorithms that analyze the data and report what they have discovered.

**OLAP SERVERS --**

In computing, online analytical processing, or OLAP is an approach to answer multi-dimensional analytical (MDA) queries swiftly. Typical applications of OLAP include business reporting for sales, marketing, management reporting, business process management, budgeting and forecasting, financial reporting and similar areas.

OLAP stands for OnLine Analytical Processing. The first attempt to provide a definition to OLAP was by Dr. Codd, who proposed 12 rules for OLAP. Later, it was discovered that this particular white paper was sponsored by one of the OLAP tool vendors, thus causing it to lose objectivity. The OLAP Report has proposed the FASMI test, Fast Analysis of Shared Multidimensional Information.

For people on the business side, the key feature out of the above list is Multidimensional. In other words, the ability to analyze metrics in different dimensions such as time, geography, gender, product, etc. For example, sales for the company are up. What region is most responsible for this increase? Which store in this region is most responsible for the increase? What particular product category or categories contributed the most to the increase? Answering these types of questions in order means that you are performing an OLAP analysis.

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 |

Depending on the underlying technology used, OLAP can be broadly divided into two different camps: MOLAP and ROLAP.

**MOLAP:** This is the more traditional way of OLAP analysis. In MOLAP, data is stored in a multidimensional cube. The storage is not in the relational database, but in proprietary formats.

Advantages:

* Excellent performance: MOLAP cubes are built for fast data retrieval, and are optimal for slicing and dicing operations.
* Can perform complex calculations: All calculations have been pre-generated when the cube is created. Hence, complex calculations are not only doable, but they return quickly.

**Disadvantages:**

* Limited in the amount of data it can handle: Because all calculations are performed when the cube is built, it is not possible to include a large amount of data in the cube itself. This is not to say that the data in the cube cannot be derived from a large amount of data. Indeed, this is possible. But in this case, only summary-level information will be included in the cube itself.
* Requires additional investment: Cube technology are often proprietary and do not already exist in the organization. Therefore, to adopt MOLAP technology, chances are additional investments in human and capital resources are needed.

**ROLAP:** This methodology relies on manipulating the data stored in the relational database to give the appearance of traditional OLAP's slicing and dicing functionality. In essence, each action of slicing and dicing is equivalent to adding a "WHERE" clause in the SQL statement.

Advantages:

* Can handle large amounts of data: The data size limitation of ROLAP technology is the limitation on data size of the underlying relational database. In other words, ROLAP itself places no limitation on data amount.
* Can leverage functionalities inherent in the relational database: Often, relational database already comes with a host of functionalities. ROLAP technologies, since they sit on top of the relational database, can therefore leverage these functionalities.

Disadvantages:

* Performance can be slow: Because each ROLAP report is essentially a SQL query (or multiple SQL queries) in the relational database, the query time can be long if the underlying data size is large.
* Limited by SQL functionalities: Because ROLAP technology mainly relies on generating SQL statements to query the relational database, and SQL statements do not fit all needs (for example, it is difficult to perform complex calculations using SQL), ROLAP technologies are therefore traditionally limited by what SQL can do. ROLAP vendors have mitigated this risk by building into the tool out-of-the-box complex fun
* ctions as well as the ability to allow users to define their own functions.

**HOLAP:** HOLAP technologies attempt to combine the advantages of MOLAP and ROLAP. For summary-type information, HOLAP leverages cube technology for faster performance. When detail information is needed, HOLAP can "drill through" from the cube into the underlying relational data.

**What is slicing and dicing in business objects?**

Slicing and dicing of business objects is used for a detailed analysis of the data. It allows changing the position of data by interchanging rows and columns. It is used to rotate the cube to view it from different perspectives. Slicing and Dicing are used for analyzing data in different views and perspectives.

Slicing - The cube is sliced based on its dimension. Slice is the act of picking a rectangular subset of a cube by choosing a single value for one of its dimensions, creating a new cube with one fewer dimension

Dicing - The cube is rotated without the need of the dimension (independent of dimension). The dice operation produces a sub-cube by allowing the analyst to pick specific values of multiple dimensions.

**Role of BA in Data warehousing:**

1. Identifying the type of reports the Business Stakeholder needs
2. Identifying the number of reports by the subject area.
3. Creating the Data requirements document
4. Help ETL people in Data mapping
5. Identify report requirements
6. Analyze each report, data content, drill path, subject areas.
7. Fact Data(atomic level(most detailed granularity),fact data field, summary data)
8. Participate in the Data modeling activity
9. Work with ETL(Data mapping)
10. Data Analysis of source
11. Data Profiling testing
12. Data Transformation.

**Data mapping documents**

Source-to target attribute mapping document.

Dependencies mapping document.

**Testing data warehouse**:

The success of any Data Warehouse (DWH) solution lies in its ability to not only analyze huge amounts of data over time but also to provide stakeholders and end-users meaningful options that are based on real-time data.

While details are elaborated below, it is essential that a good DWH test strategy should cover validation of loading of all required rows, correct execution of all transformations and successful completion of the cleansing operation. The team also needs to thoroughly test SQL queries, stored procedures or queries that produce aggregate or summary tables. Keeping in tune with emerging trends, it is also important for test teams to design and execute a set of test cases that are focused on customer experience.

 Fig 1: Key components of an effective data warehouse test strategy

The focus of Data Warehouse test strategy is primarily on four key aspects:

* Data quality validation
* End user & BI / report testing
* Load and performance testing
* End-to-End (E2E) regression and integration testing

**Data quality validation:**

This is core to any data warehouse tests and includes tests for data completeness, data transformation and data quality.

* **Data completeness tests:** Designed to verify if all the expected data loads into the data warehouse, ensuring that all records are completely loaded without errors in content quality or quantity.
* **Data transformation tests:** Designed to verify the accuracy of the transformation logic or transformation business rules.
* **Data quality tests:** Designed to validate system behavior when data is rejected (example: data inaccuracy or missing data) during data correction and substitution. Scenario-based tests and validation tests for the solutions’ reporting feature are part of this test.

Data quality validation should ensure:

* Extraction of data to the required fields
* Proper functioning of the extraction logic for each source system (historical and incremental loads)
* Security access to source systems for extraction scripts
* Updates to extract audit log and time stamping
* Completeness and accuracy of “source to extraction destination” transaction scripts, which are transforming the data as per the expected logic
* Historical & incremental load transformation for historical snap-shots
* Creation of relevant detailed and aggregated data sets
* Transaction audit log and time stamping
* No pilferage of data during transformation process and also during historical and incremental load
* Real-time or near-real time data loading without impacting performance adversely
* Temporary tables updating through multi-pass SQL statements with real-time or near-real time reporting and analytics

**End user and BI / report testing**Extreme care should be taken while testing and the reports should be as clear and self-explanatory as possible. Usability, performance, data accuracy and preview and/or export to different formats are areas where most of the failures occur.

* When designing tests for end user and BI / report testing, key points to address include:
* Data display on the business views and dashboard are as expected
* Users being able to see reports according to their user profile (authentication and authorization)
* Verification of report format and content by appropriate end users
* Verification of the accuracy and completeness of the scheduled reports
* OLAP, drill down report, cross tab report, parent / child report etc. are all working as expected
* ‘Analysis functions’ and ‘data analysis’ are working
* No pilferage of data between the source systems and the views
* Testing of replicated reports from old system to new system
* Previewing and/or exporting of reports to different formats such as spreadsheet, PDF, HTML, e-mail displays accurate and consistent data
* Print facility
* Where graphs and data in tabular format exist, both should reflect consistent data

**Load and performance testing:**

With an increase in the volume of data, stability and scalability become critical test parameters. Under stress from large transactional data volumes, data warehouses will typically not scale, and will eventually fail, unless they are tested and issues are fixed. To avoid such problems, the test team must design and execute series of tests under different loads. As part of this activity, the following tests can be executed:

* Shutdown the server during batch process and validate the result
* Perform ETL with load that is twice or thrice the maximum possible imagined data (for which the capacity is planned)
* Run huge volumes of ad-hoc queries mimicked from multiple users simultaneously
* Run large numbers of scheduled reports
* Monitor the timing of the reject processes and check system behavior when handling large volumes of rejected data

**E2E integration and regression testing**

Integration tests show how the application fits into the overall flow of all upstream and downstream applications. When designing Integration Tests, the focus of the tester should be on:

* How the overall process can break and focus on the integrations between different systems and their subsystems
* Validating system behavior when different types of data (different user profiles, different data types, different data volumes etc.) get processed and communication to the subsequent system
* Running custom-designed regression tests that simulate end user behavior (ensures success of user-acceptance tests).
* Usage of techniques like scenarios based testing, risk based testing and model based testing will enhance the effectiveness of testing.

To sum it up, it is important for test teams to understand that testing a Data Warehouse implementation is a different ball game. Since a Data Warehouse primarily deals with data, a major portion of the test effort is spent on planning, designing and executing tests that are data oriented. These tests include running SQL queries, validating that ETL executes as expected, exceptions are handled effectively, application performance meets the SLAs and finally, ensuring that the integration points are working as expected. It will also be helpful if the team members have experience in debugging performance bottlenecks.

Another dimension of Data Warehouse testing is the dependency of the tests on the test environment. Since it is a known fact that in general, a test environment will not be robust (high end servers, clustering, load balancing, data volumes and data accuracy), this will have an impact on some of the tests. For example, simulated or masked data that might not be reflecting all the characters of production data may restrict the accuracy of performance tests. In other cases, some of the jobs may not fail under simulated test environment. Test teams should be wary of such limitations.

Detecting all possible defects may sometimes be complex, but planning will best help in identifying the most obvious and costly defects early in the life cycle. Finally, a group of Data Warehouse Architects, Business Analysts and Test teams working together during the initial planning and design phase is one of the time tested approaches that can help in identifying and eliminating potential failures.

**Data completeness:** Data completeness refers to an indication of whether or not all the data necessary to meet the current and future business information demand are available in the data resource. It deals with determining the data needed to meet the business information demand and ensuring those data are captured and maintained in the data resource so they are available when needed.