LECTURE 12

Extract Transform Load (ETL)

Learning Goals

- Understand Extract from the operational system
- Understand Transform the data, which includes data cleansing
- Understand Loading the data into the DWH

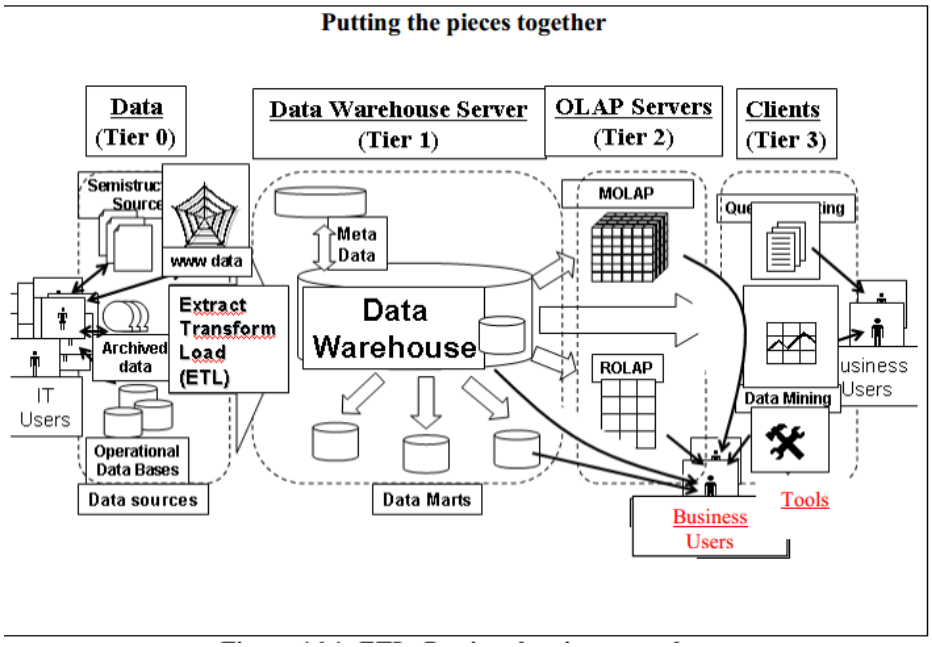
In a DWH project, 50% of the work is related to ETL. ETL stands for Extract Transform Load. Some people call it ETML i.e. Extract Transform Move Load, as you have to move the data into the DWH. There is a significant part after data transformation that involves data cleansing i.e. there is a C also here. However, ETL is a pretty standard definition, and normally when you say ETL people know what you mean.

E. Extract from the operational system

T. Transform the data, which includes data cleansing

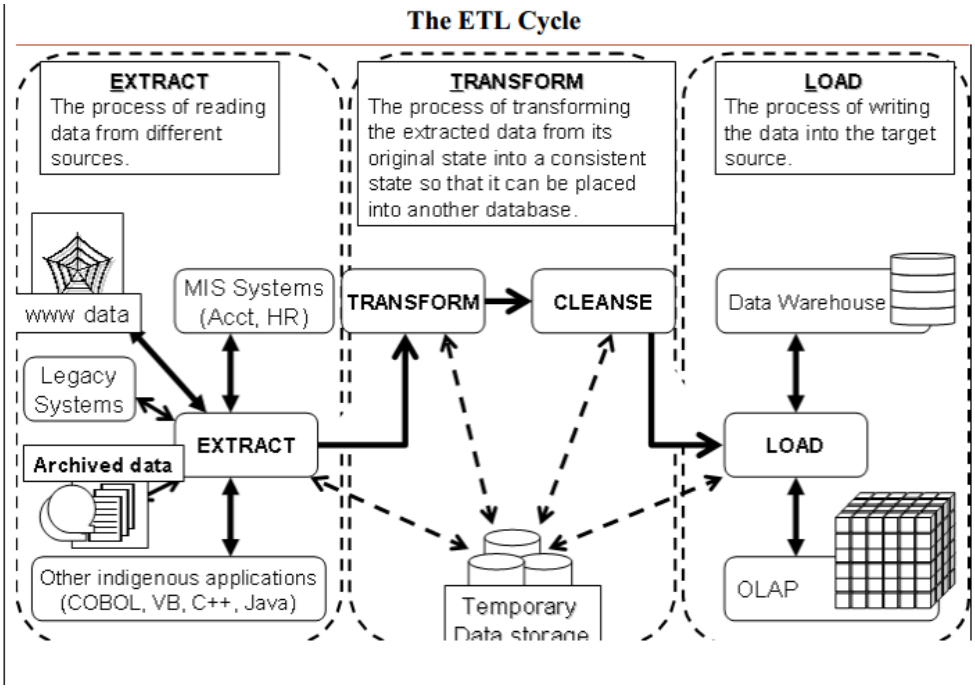
L. Loading the data into the DWH

Putting the pieces together



ETL: Putting the pieces together

Figure-16.1 shows a multi-tiered data warehousing architecture. We can see that at the lowest level (tier 0) lies the data sources like operational, archived and/or web etc. Data Warehouse Server lays in the next level (tier 1). We can see in the Figure 16.1 that ETL box lies at the boundaries of both the tiers 0 and 1. That means it serves as a bridge between data source systems and the actual data warehouse. Thus the source data is processed using some set of activities before bringing it into the data warehouse environment. These set of activities comprise the ETL process.



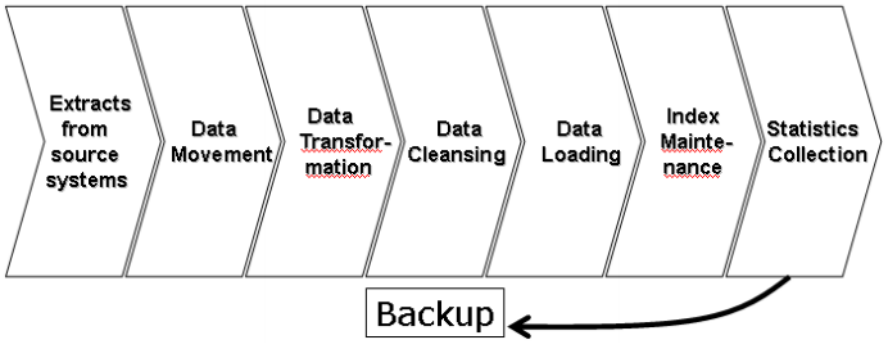
Now what are those activities in the ETL process shown as a box in the Figure 16.1? Let's have a look inside the box to find the sequence of activities that constitute the ETL process. As shown in Figure 16.2 ETL process consists of three major activities separated by dotted lines. The first step is the Extract which is the activity of reading data from multiple heterogeneous data sources. Next comes the Transform step which is the activity of transforming the input data from multiple heterogeneous environments into a single consistent state. Another important activity at this level is the data cleansing which is the activity of noise removal from input data before bringing it in the DWH environment. The final step is the Load which is an activity of loading cleansed data in to the target system.

ETL Processing

ETL is independent yet interrelated steps.

It is important to look at the big picture.

Data acquisition time may include…



Back-up is a major task, its a DWH not a cube

ETL Processing

When we look at ETL processing, it is important to look at the big picture as shown in Figure 16.3. It is not just the transformation; there is a whole lot of work you have to consider in the design and architecture of the ETL model. You have to design to extract from source system, how to get the data out of the operational databases. You have to move the data to the dedicated server where the transformations are being done or on the very least if the transformations will be done on the same systems, start moving the data into the warehouse environment. You have to look at the data movement and all the network implications of data movement. The transformations we will be talking about and the data loading strategies all have to be considered. It is not just good enough to talk about data loading, we have to do index maintenance, statistics collection, summary, data maintenance, data mart construction, data backups, a whole process has to be followed each time you refresh or reload the data warehouse. The architects really need to consider all of these. We will be focusing on all the major operations of ETL, so as to understand the big picture. It is certainly true that the volume of the data warehouse will make a difference on how you do the backups and the technology. Consider a cube which is not a data warehouse, it is a data mart. Typically a data mart is much smaller and probably pretty easier to backup just part of a file system backup on wherever your cube server is. The data warehouse is usually much bigger. So for a data warehouse you usually do a full tape backup, and a tape-

backup would ideally use a robotic tape library and all the associated setup. Typically you don't need robotic libraries for the cubes and data marts, but you do need them for data warehouse. But again you have to come back to economics, robotic tape libraries are very expensive, and so you may actually use manual tape mounting, because it is not worth paying for robots when you can have humans do it on a very different price range.

Overview of Data Extraction

First step of ETL, followed by many.

Source systems for extraction are typically OLTP systems.

A very complex task due to number of reasons:

- Very complex and poorly documented source system.
- Data has to be extracted not once, but number of times.

The process design is dependent on:

- Which extraction method to choose?
- How to make available extracted data for further processing?

Extraction is the operation of extracting data from a source system for further use in a data warehouse environment. This is the first step of the ETL process. After the extraction, this data can be transformed, cleansed and loaded into the data warehouse.

The source systems for a data warehouse are typically transaction processing applications. For example, one of the source systems for a sales analysis data warehouse might be an order entry system that records all of the current order activities.

Designing and creating the extraction process is often one of the most time-consuming tasks in the ETL process and, indeed, in the entire data warehousing process. The source systems might be very complex and poorly documented, and thus determining which data needs to be extracted can be difficult. 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. These are important considerations for extraction and ETL in general.

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

? Which extraction method to choose?

This influences the source system and the time needed for refreshing the warehouse.

? How to make available extracted data for further processing?

This influences the transportation method, and the need for cleaning and transforming the data.

Types of Data Extraction

- **Logical Extraction**
- Full Extraction
- Incremental Extraction
- **Physical Extraction**
- Online Extraction
- Offline Extraction
- Legacy vs. OLTP

There are different types of data extraction which can be broadly categorized into two data extraction techniques, logical and physical. The extraction method you should choose is highly dependent on the source system and also on the business needs in the target data warehouse environment. Very often, there's no possibility to add additional logic to the source systems to enhance an incremental extraction of data due to the performance or the increased workload of these systems. Sometimes even the customer is not allowed to add anything to an out-of-the-box (shrink wrapped CD installable) application system as the performance of the source system has been maximized and carefully calibrated and monitored. The reason being creation of the DWH does not directly help the operational people in any way, for them asking for data every now and then is an annoyance.

The estimated amount of the data to be extracted and the stage in the ETL process (initial load or maintenance of data i.e. data already loaded) may also effect the decision of how to extract, from a logical and a physical perspective. Basically, you have to decide how to extract data logically

and physically. Let's discuss logical techniques first in detail.

Logical Data Extraction

Full Extraction

- The data extracted completely from the source system.
- No need to keep track of changes.
- Source data made available as-is w/o any additional information.

Incremental Extraction

- Data extracted after a well defined point/event in time.
- Mechanism used to reflect/record the temporal changes in data (column or table).
- Sometimes entire tables off-loaded from source system into the DWH.
- Can have significant performance impacts on the data warehouse server.

The two logical data extraction types are full and incremental extraction techniques. Let's look at the two in detail.

Full Extraction

The data is extracted completely from the source system. Since this extraction reflects all the data currently available in the source system, there's no need to keep track of changes to the data source since the last successful extraction. The source data will be provided as- is and no additional logical information (for example, timestamps etc) is necessary on the source site. An example for a full extraction may be an export file of a distinct table or a remote SQL statement scanning the complete source table.

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. This event may be the last time of extraction or a more complex business event like the last day of balancing of accounts. To identify this incremental change there must be a mechanism to identify all the changed information since this specific time event.

This information can be either provided by the source data itself like an application column, reflecting the last-changed timestamp or a change table where an appropriate additional mechanism keeps track of the changes besides the originating transactions. In most cases, using the latter method means adding extraction logic to the source system.

Many data warehouses do not use any change-capture techniques as part of the extraction process. Instead, entire tables from the source systems are extracted to the data warehouse or staging area, and these tables are compared with a previous extract from the source system to identify the changed data. This approach may not have significant impact on the source systems, but it can clearly place considerable burden on the data warehouse processes, particularly if the data volumes are large.

Physical Data Extraction…

Online Extraction

- Data extracted directly from the source system.
- May access source tables through an intermediate system.
- Intermediate system usually similar to the source system.

Offline Extraction

- Data NOT extracted directly from the source system, instead staged explicitly outside the original source system.
- Data is either already structured or was created by an extraction routine.
- Some of the prevalent structures are:
- Flat files
- Dump files
- Redo and archive logs
- Transportable table-spaces

Physical Extraction Methods

Depending on the chosen logical extraction method and the capabilities and restrictions on the source side, the extracted data can be physically extracted by two mechanisms. The data can either be extracted online from the source system or from an offline structure. Such an offline

structure might already exist or it might be generated 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. With online extractions, you need to consider whether the distributed transactions are using original source objects or prepared source objects.

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. You should consider the following structures:

? Flat files

? Data in a defined, generic format. Dump files

? DBMS-specific format. Redo and archive logs

? Transportable table-spaces

Physical Data Extraction

Legacy vs. OLTP

- Data moved from the source system
- Copy made of the source system data
- Staging area used for performance reasons

During extraction, data may be removed from the source system or a copy made and the original data retained in the source system. It is common to move historical data that accumulates in an operational OLTP system to a data warehouse to maintain OLTP performance and efficiency. Legacy systems may require too much effort to implement such offload processes, so legacy data is often copied into the data warehouse, leaving the original data in place. Extracted data is loaded into the data warehouse staging area (a relational database usually separate from the data

warehouse database), for manipulation by the remaining ETL processes. Data extraction processes can be implemented using SQL stored procedures, Vendor tools, or custom applications developed in programming or scripting languages.

Data Transformation

Basic tasks

- Selection
- Splitting/Joining
- Conversion
- Summarization
- Enrichment

The next operation in the ETL process is the Data Transformation. The major tasks performed during this phase vary depending on the application; however the basic tasks are discussed here.

Selection: This takes place at the beginning of the whole process of data transformation.
You select either whole records or parts of several records from the source systems. The task of selection usually forms part of the extraction function itself. However, in some cases, the composition of the source structure may not be supporting selection of the necessary parts during data extraction. In these cases, it is advised to extract the whole record and then do the selection as part of the transformation function.
Splitting/joining: This task includes the types of data manipulation you need to perform on the selected parts of source records. Sometimes (uncommonly), you will be splitting the selected parts even further during data transformation. Joining of parts selected from many source systems is more widespread in the data warehouse environment.

Conversion: This is an all-inclusive task. It includes a large variety of rudimentary conversions of single fields for two primary reasons (i) to standardize among the data extractions from disparate source systems, and (ii) to make the fields usable and understandable to the users.

Summarization: Sometimes you may find that it is not feasible to keep data at the lowest level of detail in your data warehouse. It may be that none of your users ever need data at the lowest granularity for analysis or querying. For example, for a grocery chain, sales data at the lowest

level of detail for every transaction at the checkout may not be needed. Storing sales by product by store by day in the data warehouse may be quite adequate. So, in this case, the data transformation function includes summarization of daily sales by product and by store.

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

To better understand lets discuss conversion and enrichment with examples

Data Transformation Basic Tasks: Conversion

- **Convert common data elements into a consistent form i.e. name and address.**

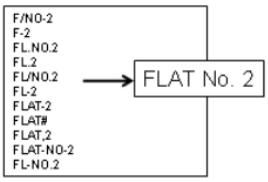
Table-16.1 (a)

Field format		Field data
First-Family-title	→	Muhammad Ibrahim Contractor
Family-title-comma-first	→	Ibrahim Contractor, Muhammad
Family-comma-first-title	→	Ibrahim, Muhammad Contractor

- **Translation of dissimilar codes into a standard code.**

Table-16.1(b)

Natl. ID	→ NID
National ID	→ NID



As discussed earlier, conversion is performed to standardize data by converting data from multiple heterogeneous sources into a single consistent form making it usable and understandable. For example, a data field containing name and job title can be represented in a number of ways in different source systems as shown in Table 16.1(a). Here three different formats have been shown for the same field. Similarly, Table 16.1(b) shows another example of code standardization of dissimilar code representations. Here, two different codes have been shown for representing *National Identity* which can be converted to a consistent form like NID. Same is the case for address representation like house number and flat number etc.

Data Transformation Basic Tasks: Conversion

- Data representation change

1. EBCIDIC to ASCII

- Operating System Change

1. Mainframe (MVS) to UNIX
2. UNIX to NT or XP

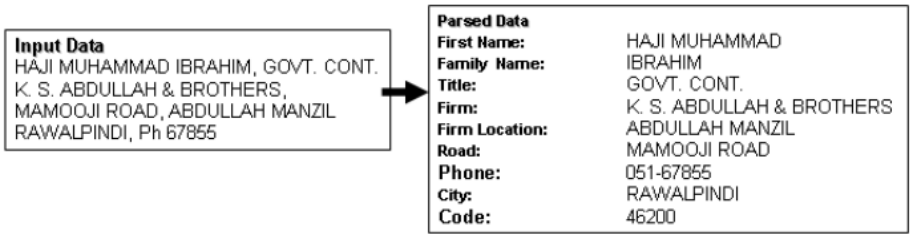
- Data type change

1. Program (Excel to Access), database format (FoxPro to Access).
2. Character, numeric and date type.
3. Fixed and variable length.

Why differences in data from different sources? Three common reasons are because data at different locations may have different data representation codes, different operating systems and different data types.

Data Transformation Basic Tasks: Enrichment

- **Data elements are mapped from source tables and files to destination fact and dimension tables.**



(a) **(b)**

- **Default values are used in the absence of source data.**
- **Fields are added for unique keys and time elements.**

Data Transformation Basic Tasks: Enrichment

Enrichment is one of the basic tasks of data transformation as shown in Figure 16.2. Figure-16(a) shows input data that does not have any associated information that links semantics with the given data i.e. what means what? Hover after enrichment the contents of the input data are assigned to the corresponding attributes or fields. Recognize that this could be a very difficult

task if the order of the contents of the input varies across the records. Assuming that is not the case, the assignment can be very straightforward. There are certain values that are not given in the input but are implied, such as the postal code, or the phone code etc. In such a case default values are used based on the input data using standard default values. Other information added could be unique keys for identification of data, such that the keys are independent of the business rules, also the data could be time stamped to ascertain its "freshness".

Significance of Data Loading Strategies

- Need to look at:

1. Data freshness
2. System performance
3. Data volatility

- Data Freshness

1. Very fresh low update efficiency
2. Historical data, high update efficiency
3. Always trade-offs in the light of goals

- System performance

1. Availability of staging table space
2. Impact on query workload

- Data Volatility

1. Ratio of new to historical data
2. High percentages of data change (batch update)

Once data has been transformed, the *loading* operation is performed. There are different loading strategies and choosing the one depends on data freshness and performance. We also need to look into data volatility, in other words, how much data is changed within the window of the refresh. So in general if real time or near real-time availability of data is required, meaning low update efficiency, because if want the data to be very fresh, then I would not allow accumulation of lot of data, because it means waiting a long time before data is inserted. Whereas, if I want the maximum update efficiency i.e. highest performance, then batch processing would be required. Thus as always there is a tradeoff between performance and data freshness and you need to look

at the goals and objectives of your data warehouse, how you follow them and how important these characteristics are. So again there is no one right or wrong answer, depends on what are the goals of the data warehouse, and then you design appropriate to those goals.

The data loading strategy will also depend on the data storage requirements. Depending upon which strategy you use, some require more data storage than others, staging tables etc. You want to look at the impact on query workloads, so when loading the data, at the same time these queries are running, what does that mean? Meaning what is allowable and what is not allowable.

It is also important to look at the ratio of existing to new data? And that will determine or help determine what the strategy should be for implementation. So we need to look at what are the characteristics of workloads that I am interested in. So there are clearly some tradeoffs here. If we look at a loading strategy with a high percentage of data changes per data block, this normally means I am doing a batch update. In other words, I have got a lot of data that I am inserting into the table, and therefore each data block has a lot of rows that are changing per data block.

Three Loading Strategies

Once we have transformed data, there are three primary loading strategies:

Full data refresh with BLOCK INSERT or 'block slamming' into empty table.

Incremental data refresh with BLOCK INSERT or 'block slamming' into existing (populated) tables.

Trickle/continuous feed with constant data collection and loading using row level insert and update operations.

There can be a couple of lectures on loading strategies, but in this course will limit ourselves to the basic concepts only. Once the data has been transformed, it is ready to be loaded into the DWH, for this purpose three loading strategies are prevalent. When the tables are populated for the first time it is a full data refresh, depending on the tool or the environment being used, this may be called as BLOCK INSERT or "block slamming" i.e. large blocks of data are loaded, usually in the form of batch updates. As the DWH evolves over time, there may be no empty tables, but already filled table have to be updated. For this purpose DWH has to be refreshed incrementally. Depending on the line of business and other reporting requirements the refresh

period will vary and also the amount of data to be loaded. In extreme cases this may boil down to BLOCK INSERT or “block slamming” into the main DWH tables. This will have certain performance implications with reference to availability of data and the outcome of the queries while the data is being refreshed and requirement of additional memory space in the form of “shadow tables”, but the discussion is beyond the scope of this course. The third and final loading strategy is trickle feed. This strategy is more in the context of active data warehousing and has a high update overhead. Usually the criterion when to load depends on the time interval between uploads or the amount of data recorded or their combination.