ETL basically stands for Extract Transform Load - which simply implies the process where you extract data from Source Tables, transform them in to the desired format based on certain rules and finally load them onto Target tables. There are numerous tools that help you with ETL process - Informatica, Control-M being a few notable ones.

So ETL Testing implies - Testing this entire process using a tool or at table level with the help of test cases and Rules Mapping document.

In ETL Testing, the following are prima facie validated –

1) Data File loads from Source system on to Source Tables.
2) The ETL Job that is designed to extract data from Source tables and then move them to staging tables. (Transform process)
3) Data validation within the Staging tables to check all Mapping Rules / Transformation Rules are followed.
4) Data Validation within Target tables to ensure data is present in required format and there is no data loss from Source to Target tab

A data warehouse is the main repository of any organization's historical data. It contains the material and information for management's decision support system. Most of the organization runs their businesses on the basis of collection of data for strategic decision- making. To take a competitive edge the organization should have the ability to review historical trends and monitor real-time functional data.

Most importantly, we never appreciate if the bug is detected at the later stage of testing cycles because it could easily lead to very high financial losses to the project. So data warehouse testing following the best practice is unavoidable to remain at the top of the business.

Data warehousing testing phases

While implementing the best practices at our testing we follow the various phases in our data warehouse testing. They are:

1) Business understanding
    a. High Level Test Approach
    b. Test Estimation
    c. Review Business Specification
    d. Attend Business Specification and Technical Specification walkthroughs
2) Test plan creation, review and walkthrough
3) Test case creation, review and walkthrough
4) Test Bed & Environment setup
5) Receiving test data file from the developers
6) Test predictions creation, review (Setting up the expected results)
7) Test case execution and (regression testing if required).
    a. Comparing the predictions with the actual results by testing thebusiness rules in the test environment.
    b. Displaying the compare result in the separate worksheet.
8) Deployment
a. Validating the business rule in the production environment.

When we test, we take sample data from the designed architecture and the test data files are usually provided to the testers by the developers. Of course, the test data should be able to cover all the possible scenarios with respect to the requirements while we do the predictions to define the expected test case results, however if the data provided to us is not supportive enough to cover all the business rules then we go for the data mocking.

What is ETL?

ETL stands for extract, transform, and load. It can consolidate the scattered data for any organization while working with different departments. It can very well handle the data coming from different departments.

For example, a health insurance organization might have information on a customer in several departments and each department might have that customer's information listed in a different way. The membership department might list the customer by name, whereas the claims department might list the customer by number. ETL can bundle all this data and consolidate it into a uniform presentation, such as for storing in a database or data warehouse.

ETL can transform not only data from different departments but also data from different sources altogether. For example, any organization is running its business on different environments like SAP and Oracle Apps for their businesses. If the higher management wants to take discussion on their business, they want to make the data integrated and used it for their reporting purposes. ETL can take these two source system data and make it integrated in to single format and load it into the tables.

How is data warehouse testing different from normal testing?

Generally the normal testing steps are:

- Requirements Analysis
- Testing Methodologies
- Test Plans and approach
- Test Cases
- Test Execution
- Verification and Validation
- Reviews and Walkthroughs

The main difference in testing a data warehouse (DW) is that we basically involve the SQL queries in our test case documents. It is vital to test both the initial loads of the Data Warehouse from the source i.e. when it gets extracted and then updating it on the target table i.e. the loading step. In specific cases, where trouble shooting is required, we verify intermediate steps as well.

A defect or bug detection can be appreciated if and only if it is detected early and is fixed at the right time without leading to a high cost. So to achieve it, it is very important to set some basic testing rules. They are:

- No Data losses
- Correct transformation rules
- Data validation
- Regression Testing
- Oneshot/ retrospective testing
- Prospective testing

• View testing
• Sampling
• Post implementation

Testing Goals

There is an exponentially increasing cost associated with finding software defects later in the development lifecycle. In data warehousing, this is compounded because of the additional business costs of using incorrect data to make critical business decisions. Given the importance of early detection of software defects, let's first review some general goals of testing an ETL application:

- **Data completeness.** Ensures that all expected data is loaded.
- **Data transformation.** Ensures that all data is transformed correctly according to business rules and/or design specifications.
- **Data quality.** Ensures that the ETL application correctly rejects, substitutes default values, corrects or ignores and reports invalid data.
- **Performance and scalability.** Ensures that data loads and queries perform within expected time frames and that the technical architecture is scalable.
- **Integration testing.** Ensures that the ETL process functions well with other upstream and downstream processes.
- **User-acceptance testing.** Ensures the solution meets users' current expectations and anticipates their future expectations.
- **Regression testing.** Ensures existing functionality remains intact each time a new release of code is completed.