iCEDQ
TORANA INC.

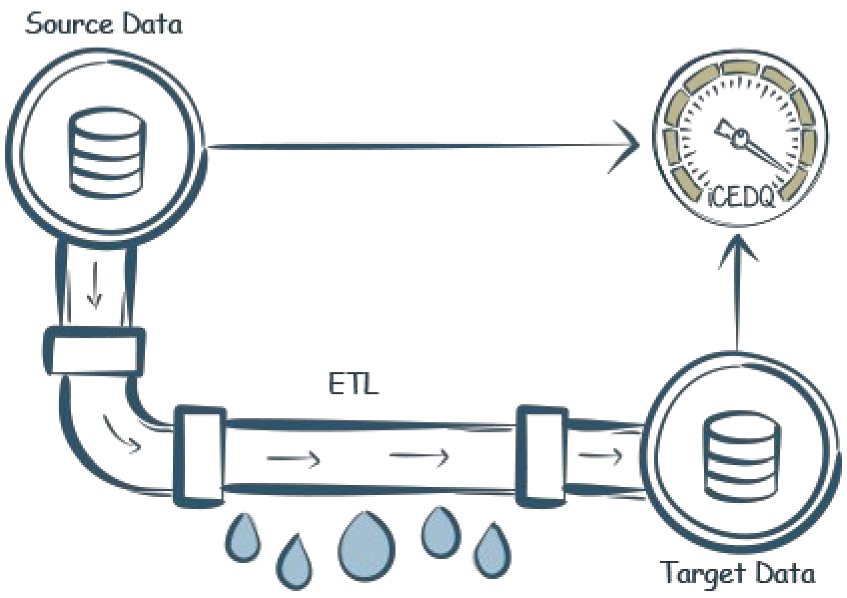
ETL Testing Concepts

This guide provides core concepts of ETL testing. This knowledge is gained while developing iceDQ software and learning from numerous implementations of ETL testing projects. Let’s dive right in.

Torana Inc.
60 Long Ridge Road, Suite 303, Stamford CT 06902.
Phone: (203) 666-4442
contact@icedq.com

What is ETL Testing?

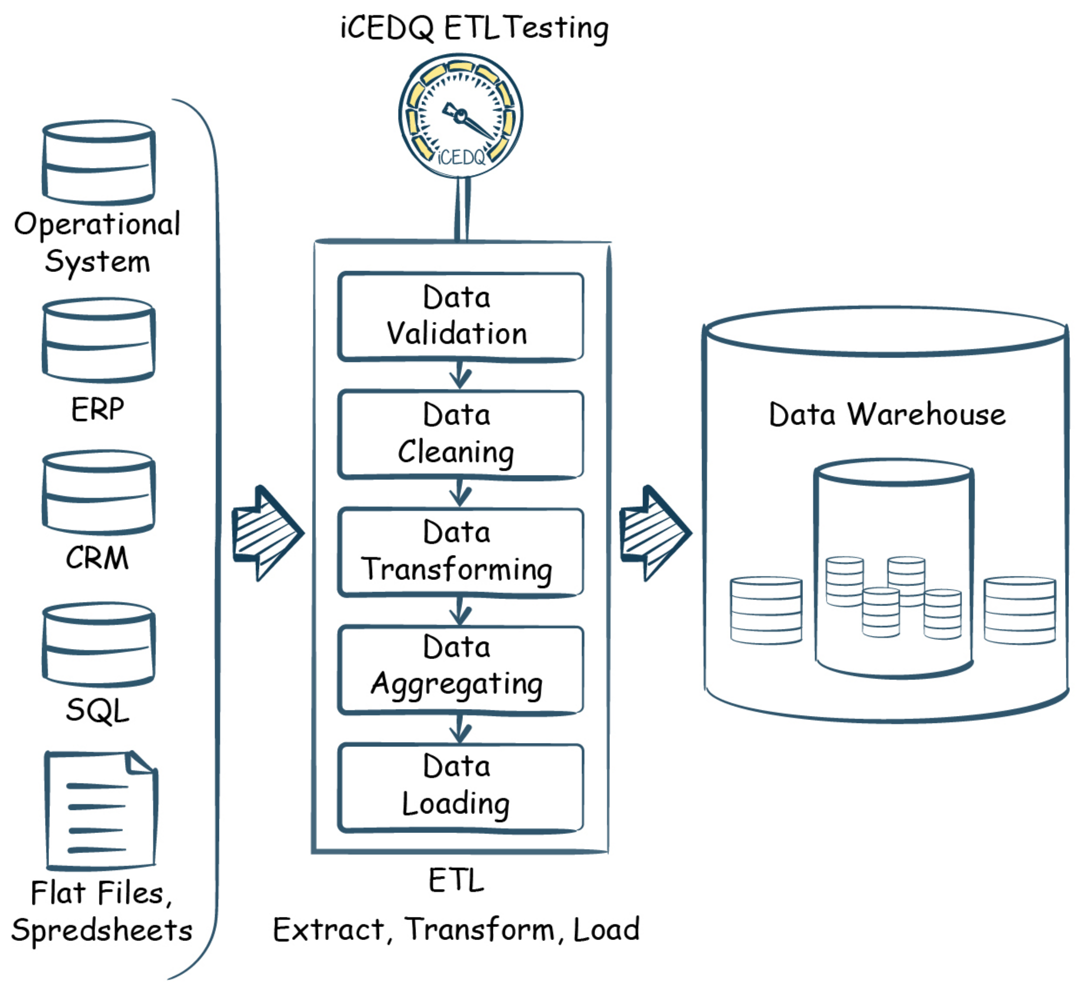
ETL Testing certifies that an ETL process is correctly extracting, transforming, and loading data as per the specifications. ETL testing is done by validating and/or comparing the input and output data transformed by the ETL process.



ETL testing is used in data-centric projects having a huge amount of data or substantial number of data pipelines. It should not be confused with application testing which usually involves a small amount of transactional data.

Why ETL Testing is Required?

Anytime a piece of software is developed, it must be tested. The ETL process is ultimately a piece of software written by a developer. An ETL process is at the heart of any data-centric system and/or project and mistakes in the ETL process will directly impact the data and the downstream applications.



1. Without ETL testing there is now way of knowing if the process is built to the specifications and as per requirements.
2. Without ETL testing the code cannot be released or deployed in production.
3. ETL testing enables root cause analysis to identify data issues due to the source data or the ETL process.
4. It is very expensive and difficult to fix data issue sin production. ETL testing ensures that the data issues are caught early in the development lifecycle.

ETL Testing Basics

ETL processes read data from a source, transform the data, and then load it in the destination databases. An organization can easily have thousands of such ETL jobs processing their nancial, customer, or operations data.

"Just like any piece of code that is developed, the ETL code must be tested to ensure that the developers have written the code correctly."

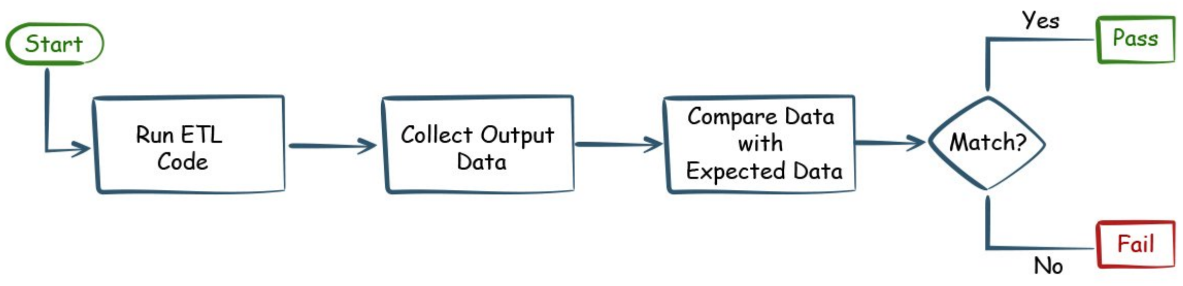
ETL testing is unique since,

- ETL processes are background processes and don't have user screens.
- ETL testing involves a large amount of data.
- ETL processes are like functions where testing requires execution of the ETL process and then the comparison of input and output data.
- The defects in the ETL processes cannot be detected by simply reviewing the ETL code.

How to do ETL Testing?

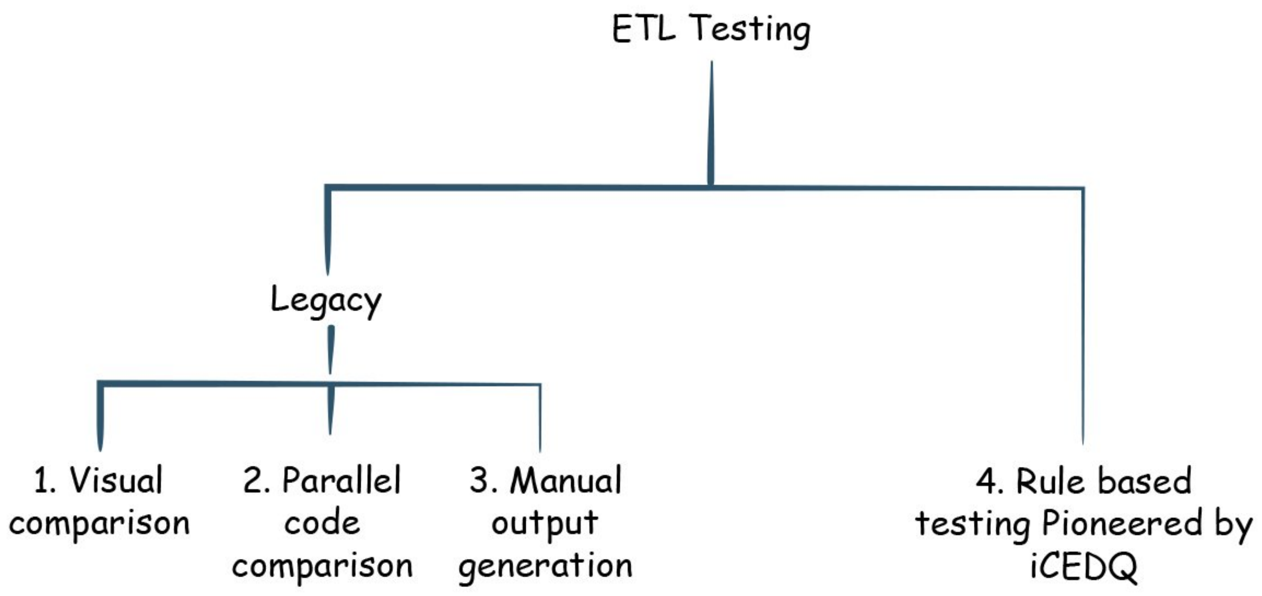
ETL processes are evaluated indirectly through black box testing approach, wherein the ETL process is first executed to create the output data and then by verifying the output data the quality of the ETL process is determined.

ETL testing process is summarized in the following three steps:



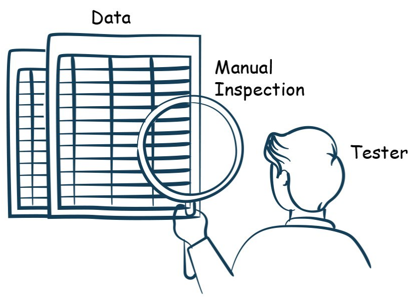
A. First, the ETL code is executed to generate the output data.
B. Then the output data is compared with the predetermined expected data.
C. Based on the comparison results, the quality of the ETL process is determined.

For ETL testing you can follow the legacy approach, which is outdated, or the newer rules based ETL testing pioneered by iceDQ.


ETL Testing
Legacy
1. Visual comparison
2. Parallel code comparison
3. Manual output generation
4. Rule based testing Pioneered by iCEDQ

1. Difference between Manual Testing and ETL Testing
2. Legacy- Pseudo code-based ETL Testing
3. Legacy- Golden Copy based ETL Testing
4. Rules Based ETL Testing

1. Difference between manual testing and ETL Testing



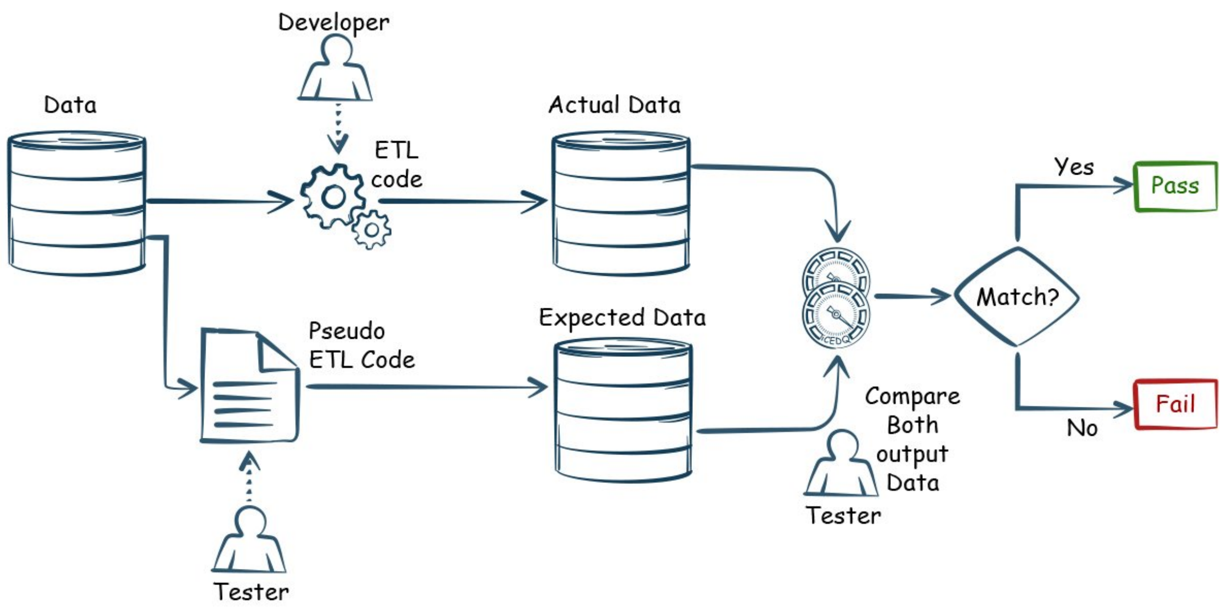
In this approach, the data created by the ETL process is sampled and inspected visually by a tester. If the data output is as excepted the ETL process is certified.

This manual testing approach is not scalable as humans are not capable of dealing with more than a few hundred records.

ETL testing vs manual testing:

- It is resource-intensive, hence very costly.
- Testing is based on few sampled records.
- It is ad hoc hence not repeatable.
- The tester must do all the testing on his desktop.
- Incomplete test coverage.
- Regression testing is not possible.

2. Legacy- Pseudo code-based ETL Testing:

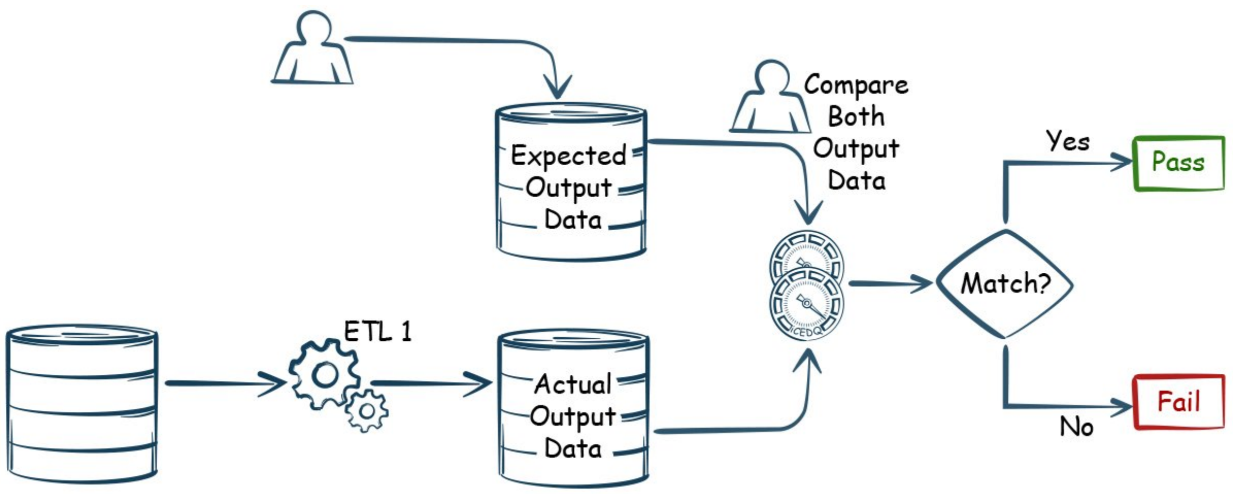


QA recreates pseudo ETL code in parallel to the developers' actual ETL process. This pseudo ETL code processes a subset of data and generates an output. The actual ETL process also uses the same input data and generates data. Then the ETL tester compares the data generated by both the processes and documents the differences. If the data is an exact match, then the ETL process passes the quality test.

Parallel development of pseudo ETL code by QA team is ridiculous because:

- Crazy amount of time, money and resources are wasted in reproducing the pseudo ETL code by the QA team.
- Since the pseudo ETL code is also development, there is no guarantee that the pseudo ETL code developed by the QA team is also error free and hence, the data output generated by it.

3. Legacy- Golden Copy based ETL Testing:



In this method, QA manually creates expected data output based on their understanding of the data processing logic. First the manual data, also called as golden copy is created and is stored in a database. Next, the ETL process is executed, and the data generated by the ETL process is compared to the golden copy of data that was created by the QA team. The ETL process is certified based on the comparison results.

This approach is also severely limited because:

- Only works with sampled data.
- Does not ensure test coverage.
- Does not work if the input data changes.

4. Rules Based ETL Testing:

An ETL process is a set of actions that transforms an input data into a desired data output. Rules based ETL testing understands these data transformation requirements and derive data audit rules which are later used to test the ETL processes.

The prior three legacy methods have inherent limitations. Beyond the obvious economic reasons there few fundamental flaws because of which the conventional approach towards ETL testing will never work.

- **Large Data Volume:** The recent exponential growth in the data volumes as well as growth in the number of ETL processes have made the above three approaches pretty much useless. It is no longer feasible to manually test the ETL processes.
- **Data Sampling:** Data sampling does not work because there are many corner cases that will be only discovered if almost all the data is processed, and the output generated by the ETL process is inspected.
- **Dynamic Input Data:** The output generated by an ETL process is totally dependent on the input data; and the input is dynamic. Hence any conventional ETL testing that uses a predetermined output, will not work.
- **Cross database comparisons:** Since ETL processes data from one system and loads into another, it is almost impossible to bring the data in one place and then do the comparison.

The rule based ETL testing is designed to avoid all the above pitfalls. Its concept is derived from auditing of financial transactions. Example, if $100 is transferred from account ‘A’ to account ‘B’. Then for the financial audit, the balance in account ‘A’ must reduce while the balance in account ‘B’ must stimulatingly increase by the same amount.

Account ‘A’ (Original Balance – New Balance) = Account “B” (New Balance – Original Balance)

This same concept has been expanded in for ETL testing. Wherein the logicof ETL transformation is understood and the audit rules are created to certify the ETL process.

Advantages of Rules based ETL Testing:

- The data can change but the rules don't.
- No need to recreate the ETL processes.
- There is no limit on data volume.
- No manual interventions.
- No data sampling is required.
- The rules are stored in central knowledge repository.
- The rules are executed on the server via a schedule or on-demand.
- Cross database testing is possible.
- The rules can be stateless to supports dynamic change in the input data.

ETL Testing Concepts with Examples

Rules based ETL testing operates under the two assumptions:

– Input data provided will change for each ETL run.
– The data transformation rules and conditions remain same unless there are changes in business requirements.
For ETL testing you must understand the concepts of Static ETL testing and the Dynamic ETL testing.

1. Static ETL Testing

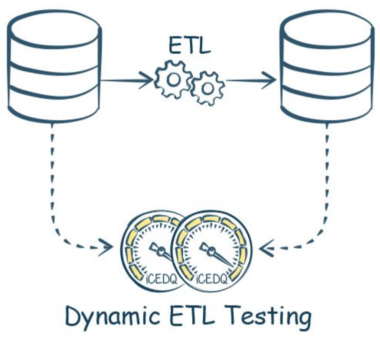
iCEDQ
Static ETL Testing

For static ETL testing only the output data is used for certifying the ETL process. The output value generated by ETL process is compared to a fixed value(s) or a condition that is previously known or defined by the tester. The input data used by the ETL process is not taken into considerations.

Here are few examples:

a. Static Value Example: The Account Category column can only have one of the three values, 'Checking, Savings, Trading'
b. Static Condition Example: Net amount must equal Gross Amount minus sum of Tax, Commission and Fees.

2. Dynamic ETL Testing



For dynamic ETL testing both input and output data is taken into considerations while testing the ETL process. In many cases the output data of an ETL process is totally dependent on the input data provided at runtime to the ETL process. Even though the transformation logic is fixed, the final value cannot be determined without knowing the input values used by the ETL process at runtime.

Thus, the ETL testing must support the dynamic nature of input data that is provided during the execution. This can be represented by the following simplistic equation.

Input Data + Transformation = Output Data

The ETL testing example below will explain the dynamic ETL testing concept:

An ETL is processing customer list. The list contains two types of customers: corporate and individuals. The ETL developer is asked to only load individual customers and not corporate customers. To test this ETL process the total of individual customers in the source must exactly match the customers in the target.

How many customers should be in the target table? That can only be known by counting individual customers in the source that were provided to the ETL process at runtime.

Types of ETL Testing

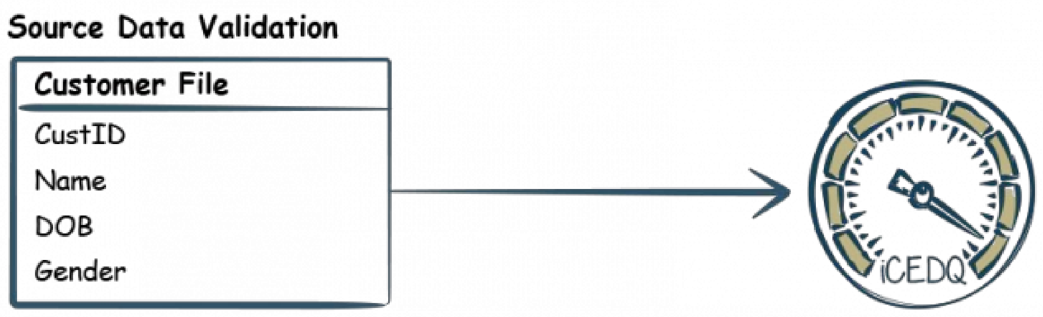
Completeness data qualitrep eht sa den ed si noisnemid ycentage of data populated vs. the possibility otnemll luf %001 f.

The types of ETL testing are listed below

1. ETL Source Data Validation Testing
2. ETL Source to Target Data Reconciliation Testing
3. ETL Data Transformation testing
4. ETL Target Data Validation Testing
5. ETL Referential Integrity Testing
6. ETL Integration Testing
7. ETL Performance Testing

1. ETL Source Data Validation Testing

This ETL testing checks input data for validity. Because if the input data itself is not valid you cannot expect the ETL process to transform the data correctly or for the process to even execute at all.

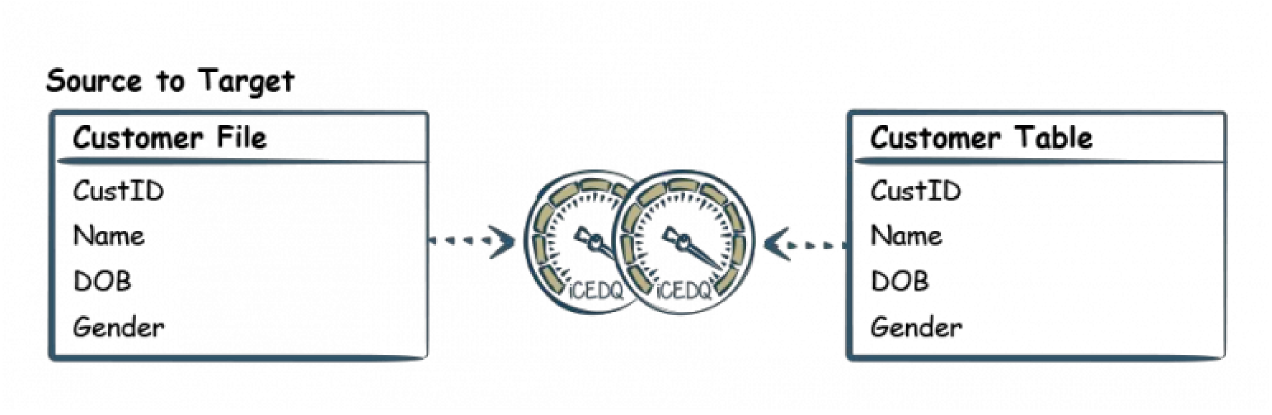


The test involves checking for nulls, formats, reference values, duplicates, etc. For example,

– Verify that there are no null values in attribute “Name” attribute
– The format of the date in the DOB column should be “YYYY-MM-DD”

2. ETL Source to Target Data Reconciliation Testing

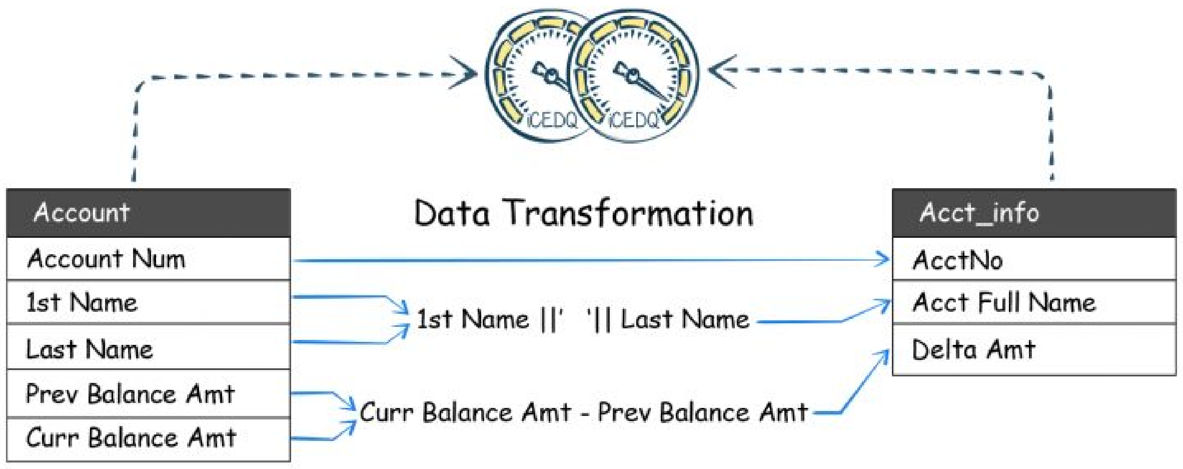
This test is mostly done to prove that there is no leakage while transporting or staging the data. Comparing the source (input) data and the target (output) data ensures that data completeness and consistency is not lost because of any issues in the ETL process. For example,



– Make sure the row count between the source and the target table is matching.
– Compare all the customer data in the source and the table to ensure that ETL loaded the data in the destination table as per the mapping rules.

3. ETL Data Transformation Testing

Data Transformation Tests ensures that every row has transformed successfully based on the mapping document. Testing Data transformations involve reconciling the data between source and destination to verify that the ETL is transforming the data as expected. For example,



Test the transformation of first name and last name source column into full name target column.

– Make sure the ETL is calculating the values correctly.

4. ETL Target Data Validation Testing

Data Validation Tests is used to validate a single data source, be it a database table, data extracts, dimension table, or a fact table.

– Check if there are any nulls in the name column.

– Format of the email should be valid.

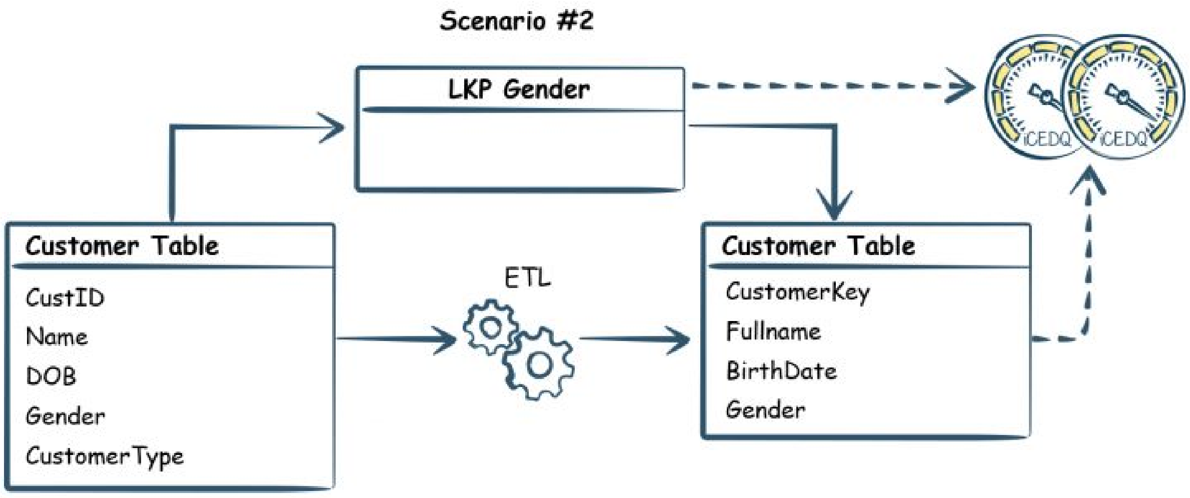
– There should be only one active record in a dimension table.

– Date in birth date column should be a valid date.

– Check if the Net amount cannot be less than zero.

5. ETL Referential Integrity Testing

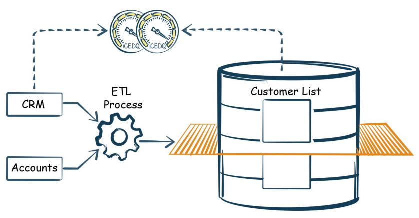
The referential integrity testing ensures that the child table only have foreign key values that exists in the parent table.



In the case above the gender table has M, F and Others. The ETL testing involves reconciling so that the Gender attributes in the customer table will only have one of those three values.

6. ETL Integration Testing

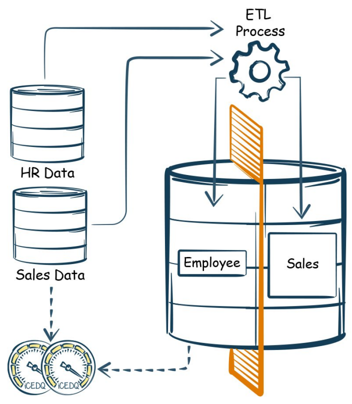
ETL integration testing is done to verify that the ETL process has integrated the data correctly. One of the key purposes of an ETL process is to integrate data from multiple data sources or multiple subject areas.



Vertical Integration Testing: In this case data is brought in from multiple data sources and integrated into a table. Example in this type of integration customer list from CRM system and accounting system is integrated in a single unified list. The integration must ensure that:

Attributes from multiple sources are mapped correctly to the destination

No duplicate records exist.



Horizontal Integration Testing:

In this scenario data from multiple subject areas and sources are linked together to form meaningful relationship. A typical example is to link the salesperson data with sales data to calculate the commission.

Mostly referential integrity /foreign keys are created, and different tables are linked to together.

ETL integration testing involves creation of multiple ETL testing rules to verify if the data integration is done correctly. This is true because even though there might be one ETL process that integrates the data, it nevertheless contains multiple business rules for data transformation. ETL testing must ensure that each of those integration rules are implemented correctly. This testing includes all the above types of testing.

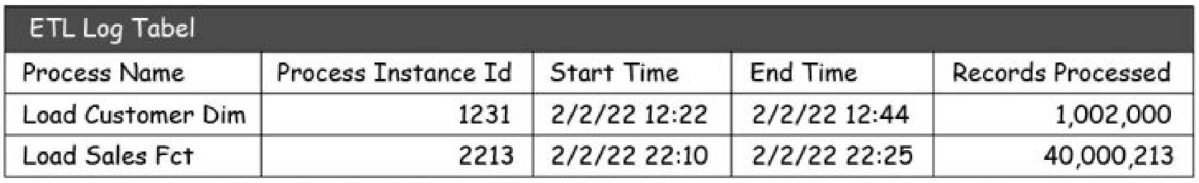
– Ensure the data is going to the respective attributes
– No duplicate entities exists and at the same time no unrelated entities are unified.

– Ensure the entities are linked correctly.

7. ETL Performance Testing

Even if the ETL process is coded correctly it is possible that, when executed it takes unreasonably more time to finish the job. ETL performance testing measure and the time taken to finish processing a certain number of records vs. user expectations. The ETL performance metrics are usually measured in the number of rows processed per seconds.

To measure performance three metric are needed, ETL processes start time, ETL process end time and number of records processed. The sources for the above metrics are:



ETL Log Tabel

Process Name	Process Instance Id	Start Time	End Time	Records Processed
Load Customer Dim	1231	2/2/22 12:22	2/2/22 12:44	1,002,000
Load Sales Fct	2213	2/2/22 22:10	2/2/22 22:25	40,000,213

– Special ETL log table which captures all the ETL process execution stats.
– Some of the metrics are derived from the target table with row level logging attributes such as record insert datetime, record update date time.

There is no universal standard for performance testing numbers, so it all depends on the expectations. However, some parameters must be taken into consideration that directly affects the ETL process performance numbers.

– Number of records inserted.
- Number of records updated or deleted.
– Logging is enabled or not in the target database.
– Row level locking setting in destination tables.
– Presence of indexes.
– The size of the processing machine.

ETL Testing Scenarios

Following ETL testing scenarios should be considered for any data projects.

1. Record level ETL tests
2. Attribute Data level ETL tests
3. Aggregate Data level ETL tests
4. Execution Level ETL tests

The ETL testing scenarios repeat in multiple situations regardless of the type of data being processed.

ETL Test Scenarios	**Test Description**
Record Level Scenarios	These are record level ETL tests
Record Count Testing	This is a primary test, to check if all the available records are populated – Nothing more, nothing less. This test ensures that the ETL process has loaded all the records. But it does not know if the data in the records is correct.
Duplicate Records Testing	Duplicate records happens if primary key or unique key constraints are not implemented in the database. In such cases specific ETL Tests are needed to ensure duplicate records are not generated by the ETL process.
Record Aggregation Test	In many scenarios transaction level records are aggregated by time, or other dimensions. Test are needed to ensure that the dimension chosen for the aggregation of records are correct.
Row Filter Testing	Often ETL developers miss or adding filters or sometimes, forget to remove filters that were added during testing. Create ETL tests to ensure proper data filters are implemented as per requirements.
Type II dimension Testing	The type ii dimensions ETL logic retires old records and inserts new records. This Test to ensure that only one valid record is present, and the expiry dates don't overlap.
Attribute Level Scenarios	**These are attribute level tests.**
Data mapping Testing	During the development of the ETL process the developer might do mistake in mapping the source and target attributes. This ETL test ensure that the data is getting populated in the correct target attributes.

Calculations Numeric and date	There are many mathematical calculations used to populate calculated fields. This ETL test ensures that the calculations are done correctly by the ETL process.
Expressions String	Various string manipulation and operations such as CONACT, SUBSTRING, TRIM, are done on strings. This test ensures string transformations are done correctly by the ETL process.
Data Truncation	Many time the data processed by the ETL process truncate the data and/or if the target column has shorter size the data can be get truncated. This ETL test ensures string data is not truncated by the ETL process or during the load time.
Data Rounding Numbers and dates	This can happen if the datatype is not chosen correctly in either the ETL process variables or the target table datatypes. Numbers can get rounded; dates can lose time or second components. Ensure decimal data is not rounded incorrectly.
Formatting Issues -Date and Strings	This mostly happens with string datatypes as it accepts data in almost any format. Many cases dates are p The date Ensure the date, or string data is formatted correctly.
Reference Data or Dimension Lookup	Ensure that the child or transaction attributes have reference data that are present in the master.
Aggregate Scenarios	**This involves testing of summarized (balances, snapshot, aggregates) data.**
Aggregate calculation	Ensure the data aggregations of data is done correctly.
Simple Row counts	Ensure the number of records populated is not more and/or less than the expected number of records. The row count in the destination matches to the source system.
Simple Sums	Match the sums of numeric values between source and target to ensure the numbers are correct.

Grouped Row Count	Reconcile counts for different groups between source and target.
Group Sums	Reconcile aggregate sums for different groups between source and target.
Execution Scenarios	**This testing involves testing of ETL processes related to their executions.**
Incremental Load	Often data is loaded in increments based on delta logic. This ETL Test ensures the incremental loads are reconciling correctly with source and no gaps or overlapping are generated.
Multi Execution Tests	Normally you won't expect same data to be processed again. But in many situations the data is reprocessed or accidently executed. This test ensures multiple reruns of the ETL process with the same data do not generate extra records.
ETL Performance Test	The data processing must finish within the required timeframe. ETL performance test ensures that the ETL processing time is acceptable by checking the run logs.

Scope of ETL Testing

The scope of ETL testing is restricted to ensuring the ETL process is correctly developed, and it is processing data as per the business requirements.

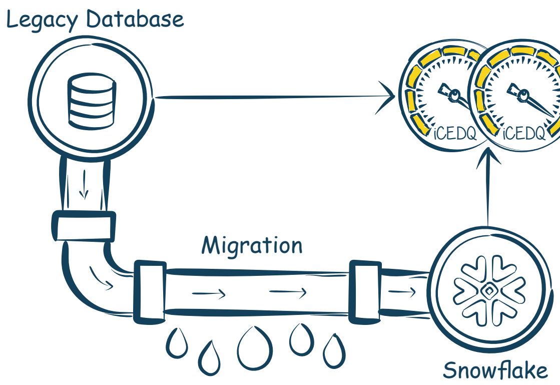
– Data Transformation and data loading is correct.
– ETL process does not create duplicate data.
– ETL process execution is done in proper order.
– The ETL process has correct incremental logic for processing data.
– The ETL emits proper exit codes on errors.
– The ETL Processes do not crash due to data exceptions.
– The ETL process logs metadata about its process.

Conclusion

There are many ways to do ETL testing. Some do it manually while others use legacy approaches. Hope this becomes clear that going forward the rules based ETL testing is the only viable solution to do ETL testing at scale. Let us know what you think in the comments below.

About iCEDQ

iCEDQ is an industry leading DataOps platform for Data Test Automation and Production Data Monitoring. iCEDQ is a certified Snowflake technology partner. Our clients, such as Albertsons, Fidelity, and BMC are extensively using iCEDQ and RuleX software for their Snowflake data migration testing. Also, consider iCEDQ DataOps Platform for:



- ETL / Data Warehouse
- Cloud Data Migration
- BI Report Testing Big
- Data Testing
- Production Data

Contact Us: 60 Long Ridge Road, Suite 303, Stamford CT 06902 | contact@iCEDQ.com | +1 (203) 666-4442