ELTMaestro and the ELT Paradigm Shift

Abstract

ELT – extract, load, and transform – is replacing ETL (extract, transform, load) as the usual method of populating data warehouses. Modern data warehouse appliances and appliance-like systems, such as IBM Pure Data / Netezza, are ideally suited for an ELT environment. ELTMaestro is a framework for ELT development, deployment, and production. ELTMaestro is not an ETL tool that has been repurposed for ELT; rather, it is designed from the ground up with ELT in mind, and fully supports the ELT paradigm. At the same time, ELTMaestro is familiar and easy to learn for developers with experience in traditional ETL tools.

In the beginning…

Data warehousing emerged from the realization that databases that worked well for operational purposes usually did not work well for analytical purposes. In a bank, for example, one operational database might keep track of customers' accounts. Another operational database, from a different vendor, and maintained in a different department, contains demographic information about the bank's customers. A marketing analyst wants to know the average account for customers in a certain demographic category. This is difficult to determine from the operational databases because (1) information from two different vendors' databases must be combined, and (2) large operations (such as looking at every account balance) put the bank's operations at risk, since they take computing resources away from the basic business of serving customers.

To address these kinds of problems, banks and other businesses started using *data warehouses.* Data warehouses collected all of the information needed for analytical tasks into an environment where analysts could do their work without affecting a business' operations. Data warehouses were periodically refreshed with new information from the operational systems, preferably during times when demand on the operational systems was low, like the middle of the night.

It was soon obvious that it wasn't enough simply to move the data from the operational systems to the warehouse unchanged.[1] The data from the operational systems needed to be rearranged, summarized, combined with data from other systems, and in general *transformed* in order to become useful to data warehouse users. This need gave rise to the *Extract, Transform, and Load,* or ETL paradigm: data was *extracted* from operational systems, suitably *transformed*, and then *loaded* to the data warehouse. The three-step ETL process was conceived as a process involving three servers: the original operational server, an intermediate server which extracted and transformed the data, and then the data warehouse itself, which was typically implemented in a traditional database on a third server. An ecosystem of tools and developers evolved to fill the role of the intermediate step of the ETL process. Notable players in this field included Ascential DataStage, Informatica, and Ab Initio. Traditional database vendors such

[1] This was sometimes done as an intermediate step, however, resulting in what was called an *operational data warehouse* or ODW.

Copyright 2013, 2014 by David Rosenthal, IQ Associates, Inc. All rights reserved.

as Oracle also offered ETL tools together with editions of their databases adapted for use as data warehouses. Some Business Intelligence (BI) vendors such as Business Objects, also had ETL offerings.

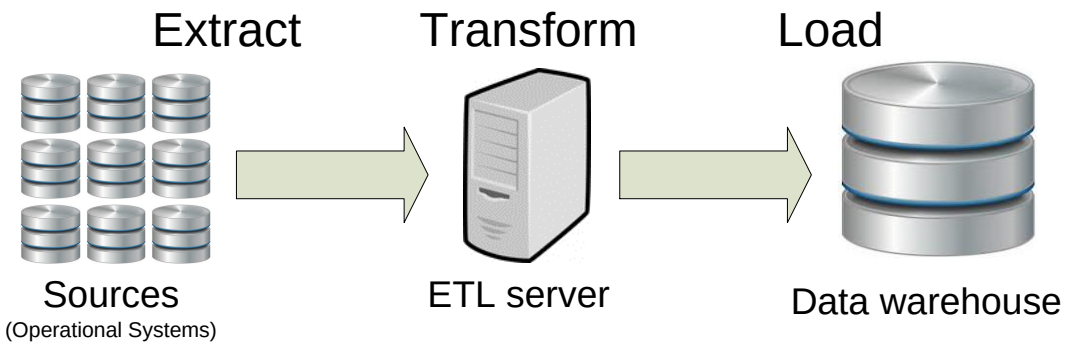

Figure 1. Traditional ETL Architecture

As ETL implementations became more widespread, the following considerations became dominant:

1. Data volumes rapidly increased. Both the supply of data available from operational systems, and users' demand that that data be available in the warehouse tended to grow exponentially.
2. ETL bottlenecks became important. ETL systems usually collected data from operational systems at night, and were expected to have incorporated the new data by the next morning. As data volumes increased, ETL systems found it difficult to meet their service level agreements.
3. ETL bottlenecks often occurred in data movement from one stage to the next.
4. Users submitted increasingly difficult queries to data warehouses, and wanted the results of those queries returned quickly.
5. It was observed that databases that performed well on OLTP tasks – basically, many small queries involving few table rows – did not necessarily perform well as data warehouses, use of which involved generally fewer queries on a large number of table rows.

Vendors such as Netezza began to develop systems to address these issues (particularly, at first, issues 4 and 5). Netezza produced a machine that specialized in quickly answering queries that were too large and complex for traditional databases. Netezza called their machines "data warehousing appliances."

Data warehouse appliances and the ELT paradigm

Initially, the success of specialized systems such as the Netezza data warehouse appliance lay in their ability to address items 4 and 5 in the list above. It has turned out, however, that the same characteristics that make the appliances excellent at answering difficult queries also make them ideal platforms for performing transformations. This means that the appliances can support a new paradigm for populating data warehouses.

Copyright 2013, 2014 by David Rosenthal, IQ Associates, Inc. All rights reserved.

In the ELT (Extract, Load, Transform) paradigm, operational data is loaded directly to the server that hosts the data warehouse, into tables that have the same structure as the source tables. The data is then transformed "in place," and loaded into the data warehouse tables.

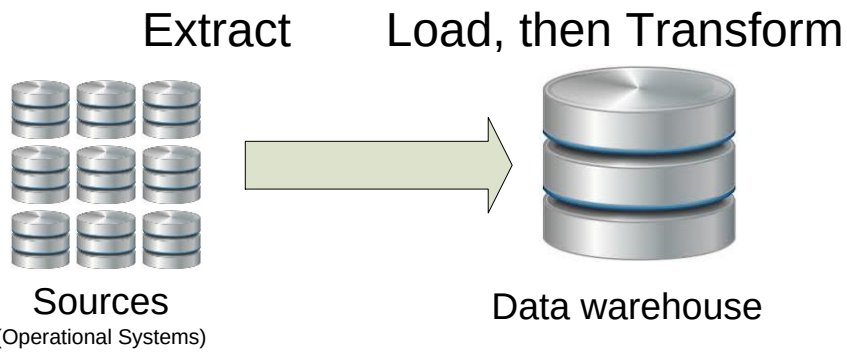

Figure 2. ELT architecture

The first thing to notice is that compared with ETL, there is one less "hop" between machines; in fact, there is one less platform altogether. This is important because, as we mentioned, data movement is a major source of bottlenecks. Having one less platform to buy and maintain, and one less environment for developers to learn are also obvious advantages. These advantages might make the case for ELT even if the target machine performed less well than traditional ETL servers on the transformation task. But Netezza machines often perform much better.

The Netezza architecture makes queries run faster by distributing them over tens or hundreds of parallel processing nodes. The architecture maintains detailed maps of the slices of data allotted to each node, and employs sophisticated algorithms to minimize the amount of data that needs to be moved to produce the desired result. Netezza will often run queries faster than traditional databases "out of the box," that is, with no changes to the queries to accommodate the Netezza architecture. The queries will run even faster if the database designer uses domain knowledge to help Netezza decide how to store table data.

These characteristics certainly improve query performance, by orders of magnitude in many cases. But these same characteristics – knowing the contents of each slice of data, minimizing data movement, and organizing data in optimal configurations – are often just as useful for the tasks involved in *transforming* data. Simply put, Netezza isn't just good at select statements – it's good at SQL and data manipulation in general. In fact, Netezza often outperforms traditional dedicated ETL systems at transformation tasks.

The price/performance advantages of the ELT/Netezza approach over traditional ETL are especially striking in a side-by-side comparison of processing power. Enterprise ETL tools are most commonly licensed *per processor.* ETL software licenses for 48 or 96 processors (typical Netezza configurations) will cost multiple millions of dollars – a price that only includes only software and is many times the price of Netezza hardware *and* software. This is a potent consideration at a time when an increasing number of companies with moderate IT budgets are dealing with terabyte-range data volumes, and require processing power in this range.

Copyright 2013, 2014 by David Rosenthal, IQ Associates, Inc. All rights reserved.

Problems with the ELT approach

We see that there are several reasons why ELT would be a tempting choice for a data warehouse architect (and in fact there are other reasons that we haven't mentioned)[2], especially an architect with a Netezza machine at their disposal. Even so, there are reasons why an architect might hesitate to pursue the ELT path. Designers of ELT systems must deal with the following issues:

1. *Lack of tools and resources.* The ecosystem of tools and developers that has evolved to support ETL does not exist for ELT. It is difficult to find developers that can implement complex transformations on a Netezza machine.

2. *Preference for dataflow diagrams.* ETL architects, developers and the business users they serve tend to think in terms of dataflow diagrams like the one below, rather than abstract set operations that comprise SQL. Traditional ETL tools usually offer a GUI based on the concept of data flows.

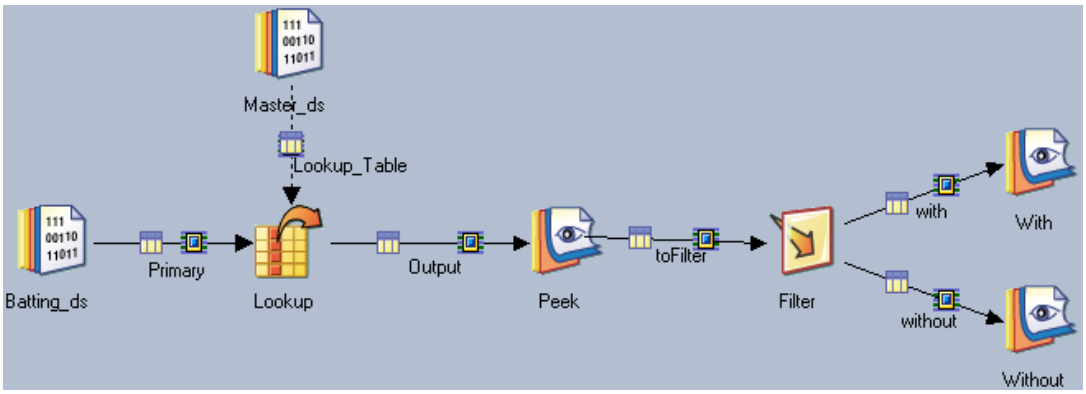

Figure 3. Dataflow diagram.

3. *Deployment.* It is not enough to write a working program that transforms and loads data to the warehouse; the program must then be *deployed* to testing and development environments. To support deployment, all program features that depend on a particular environment's hardware, drivers, connections, auxiliary programs, etc. must be parameterized. Production environments are substantially different from development environments; typically they are more isolated and are implemented on more powerful machines. It is usually difficult or impossible to test directly on production machines, for fear of contaminating them or interfering with production programs. On the other hand, when a new program or version is taken to production, it is expected to work flawlessly! Deployment is obviously not a problem unique to ELT, ETL, or data warehousing, but data warehouse practitioners have more experience and comfort with deploying ETL systems.

4. *Restartability.* Loading data warehouses involves processes and transactions that may take many hours to complete. One must assume that these loads will occasionally fail. It is more

[2] http://www.dataacademy.com/files/ETL-vs-ELT-White-Paper.pdf

Copyright 2013, 2014 by David Rosenthal, IQ Associates, Inc. All rights reserved.

than likely that failures will occur at inconvenient times, since data warehouses are most often loaded during non-business hours. This is not a good time to deal with complex manual procedures to unwind and clean up the partially-accomplished work so that the load can be resumed. On the other hand, production probably won't be able to afford the time to restart from scratch if a significant amount of work has been done. What is needed is a process that automatically determines the "last known good point" from which work can be resumed, and automatically clears (or preserves) any temporary files as necessary to produce the correct result. Restartability that meets these criteria should not be added as an afterthought, but rather designed into the process from the beginning.

5. *Audit, balance, and control.* It is a data warehousing best practice to maintain a database of information about the loading process. Examples of this kind of information include:

    - The date and time that a certain target table was loaded from a certain source table.
    - How many rows were loaded.
    - Whether the load transaction succeeded or failed.
    - If appropriate, whether or not the load "balanced;" whether some total or checksum computed from the target matched a checksum in the source.

    Maintaining records concerning the movement and handling of important information is almost always recommended; in some cases it is legally mandated. Audit, balance, and control (ABC) records can support restartability, mentioned above, and can be an important source of debugging information.

Again, restartability and ABC are important issues independently of whether an ETL or and ELT approach is employed, but they will be implemented differently, and most architects will have more experience with them in the context of ETL. Architects who have dealt with these issues in an ETL context may be hesitant to re-solve them in an ELT context.

Role of ELTMaestro

ELTMaestro is an environment for ELT development on the Netezza/Pure Data platform. ELTMaestro is partly a dataflow-style GUI tool which compiles to SQL commands to be run on Netezza, and partly a framework for supporting the data warehouse development lifecycle. ELTMaestro enables and encourages the ELT approach by addressing the issues raised in the last section.

1. ELTMaestro gives developers a tool as powerful as the best traditional ETL tools, but which is designed to implement the ELT/Netezza paradigm described above, currently the highest performing and most cost-effective solution available. ELTMaestro is lightweight, easy to administer, and inexpensive; and in all of these respects, by orders of magnitude compared with the best-known ETL tools. This is in no small measure because the "engine" of ELTMaestro, normally the largest component of an ETL tool, is in this case already provided by Netezza.

Copyright 2013, 2014 by David Rosenthal, IQ Associates, Inc. All rights reserved.

2. ELTMaestro uses a dataflow-based GUI. Those familiar with traditional ETL tools understand ELTMaestro jobs when they see them, and can construct them with little to no training.

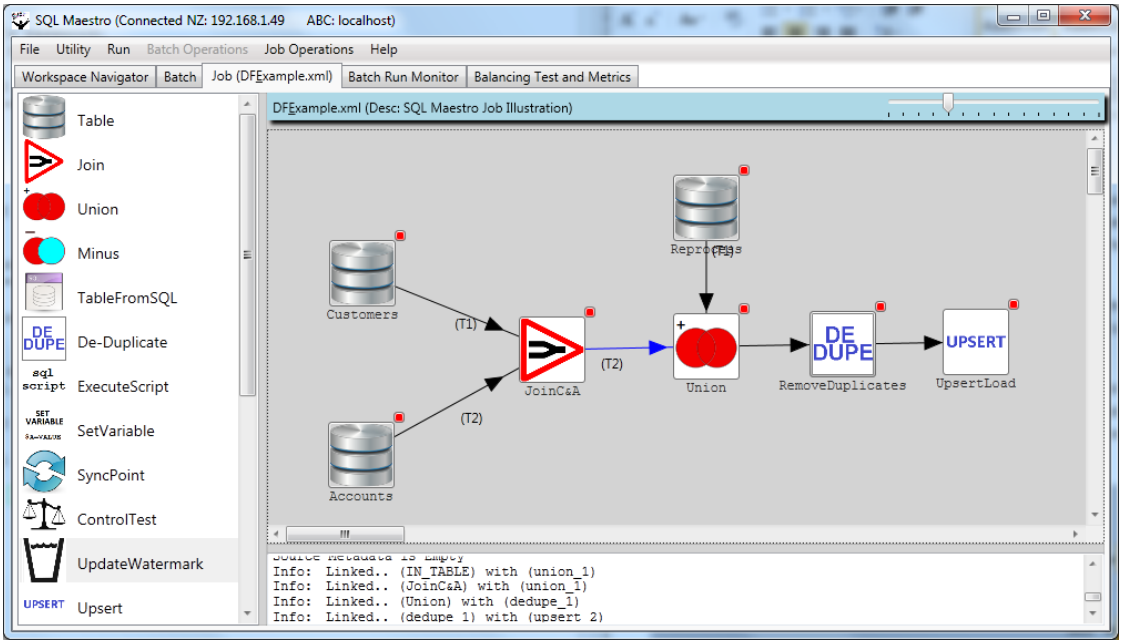

Figure 4. ELTMaestro GUI.

3. ELTMaestro includes a framework for testing and deployment. Developers, testers, and production engineers who have deployed ETL systems will find the concepts couched in terms and practices which are familiar from their previous experience, but appropriately adapted to ELT and Netezza.

4. ELTMaestro includes an ABC database. The ABC database may reside in a DBMS of the installer's choosing, or in a PostgreSQL instance which is part of the installation. The ABC database supports restartability and other aspects of audit, balance, and control.

Conclusion

Many data warehouse architects have concluded that with currently available hardware, ELT has marked performance and cost advantages over ETL as a way of loading data warehouses, especially if high-performing data warehouse appliances such as Netezza are available. Objections to ELT are mainly at the level of pragmatics: there are more ETL tools available, developers and other stakeholders have more experience, etc. – in short, ELT is newer, and therefore involves more change.

ELTMaestro unlocks the potential of ELT by obviating these objections. A data warehouse hosted on Netezza but loaded by an ETL system is allowing two expensive systems to do work that could be accomplished, faster, with one of them, by using ELT. ELTMaestro provides the path to remove this inefficiency, and realize the potential performance and cost savings.

Copyright 2013, 2014 by David Rosenthal, IQ Associates, Inc. All rights reserved.