COMPARISON

vs.

Talend, Pentaho

by Petr Uher

TABLE OF CONTENTS

BENCHMARKS SET UP

In this review we compare several open and closed source ETL tools:

- CloverETL Designer 2.8.1
- Talend's Open Studio 3.1.3
- Pentaho Data Integration (PDI) 3.2.0.

It is neither easy nor straightforward to compare ETL performance – we are not aware of any standardized industry benchmarks for ETL tools. On the other hand, there is a well-defined TPC-H[1] set of open benchmarks designed by industry leaders to judge the relative strengths and weaknesses of relational database systems. The TPC-H is focused on decision support processing.

To bring a bit of standardization into our comparison, we have converted the TPC-H benchmarks to ETL domain and used them to test the ETL tools mentioned.

TPC benchmarks consists of a suite of business oriented ad-hoc queries and concurrent data modifications. The queries and data populating the database have been chosen to have a broad industry-wide relevance. These benchmarks illustrate decision support systems that examine large volumes of data, execute queries with a high degree of complexity, and give answers to critical business questions.

For our ETL comparisons we will use TPC-H Q1 and Q3 queries converted to ETL mappings and run on 1GB and 10GB data sets generated by *dbgen* utility available on tpc.org web site. Detailed data model of all entities and their relationships used in these tests can be found in Appendix C - TPC-H Tests Data Model (ERA).

Used Hardware:

The tests were executed on two different systems:

System A) laptop PC with Windows Vista Home Premium 32bit, 2GHz CPU (Intel Core 2 Duo) and 2 GB of memory

System B) server class machine with Windows Server 2003 Enterprise Edition SP1 32bit, 2 x 2.33GHz Intel Xeon quad core 12MB L2 cache – 8 CPUs, 8GB of memory

The database used in this test was installed on separate server class machine with SUSE LINUX 11.1 (i586) 32bit, 2xIntel Xeon CPU 3.06GHz 512kB L2 cache, 4GB of memory, and disks in RAID 0.

[1] TPC-Transaction Processing Performance Council (www.tpc.org)

Tests description

TPC-H Q1 : PRICING SUMMARY REPORT QUERY

This query reports the amount of business that was billed, shipped, and returned.

Business Question

The Pricing Summary Report Query provides a summary pricing report for all line items shipped as of a given date. The query lists totals for extended price, discounted extended price and discounted extended price plus tax, average quantity, average extended price, and average discount. These aggregates are grouped by RETURNFLAG and LINESTATUS, and listed in ascending order of RETURNFLAG and LINESTATUS. A count of the number of lineitems in each group is included. The query should select approximately 95% of all records.

Functional Query Definition

```
select
l_returnflag,
l_linestatus,
sum(l_quantity) as sum_qty,
sum(l_extendedprice) as sum_base_price,
sum(l_extendedprice*(1-l_discount)) as sum_disc_price,
sum(l_extendedprice*(1-l_discount)*(1+l_tax)) as sum_charge,
avg(l_quantity) as avg_qty,
avg(l_extendedprice) as avg_price,
avg(l_discount) as avg_disc,
count(*) as count_order
from lineitem
where l_shipdate <= date '1998-09-02'
group by l_returnflag, l_linestatus
order by l_returnflag, l_linestatus;
```

TPC-H Q3 : SHIPPING PRIORITY QUERY

This query retrieves unshipped orders with the highest value.

Business Question

The Shipping Priority Query retrieves the shipping priority and potential revenue, defined as the sum of l_extendedprice * (1-l_discount), of the orders having the largest revenue among those that had not been shipped as of a given date. Orders are listed in decreasing order of revenue.

Functional Query Definition

```
select
l_orderkey,
sum(l_extendedprice*(1-l_discount)) as revenue,
o_orderdate,
o_shippriority
from customer, orders, lineitem
where c_mktsegment = 'BUILDING'
and c_custkey = o_custkey
and l_orderkey = o_orderkey
and o_orderdate < date '1995-03-15'
and l_shipdate > date '1995-03-15'
group by l_orderkey, o_orderdate, o_shippriority
order by revenue desc, o_orderdate;
```

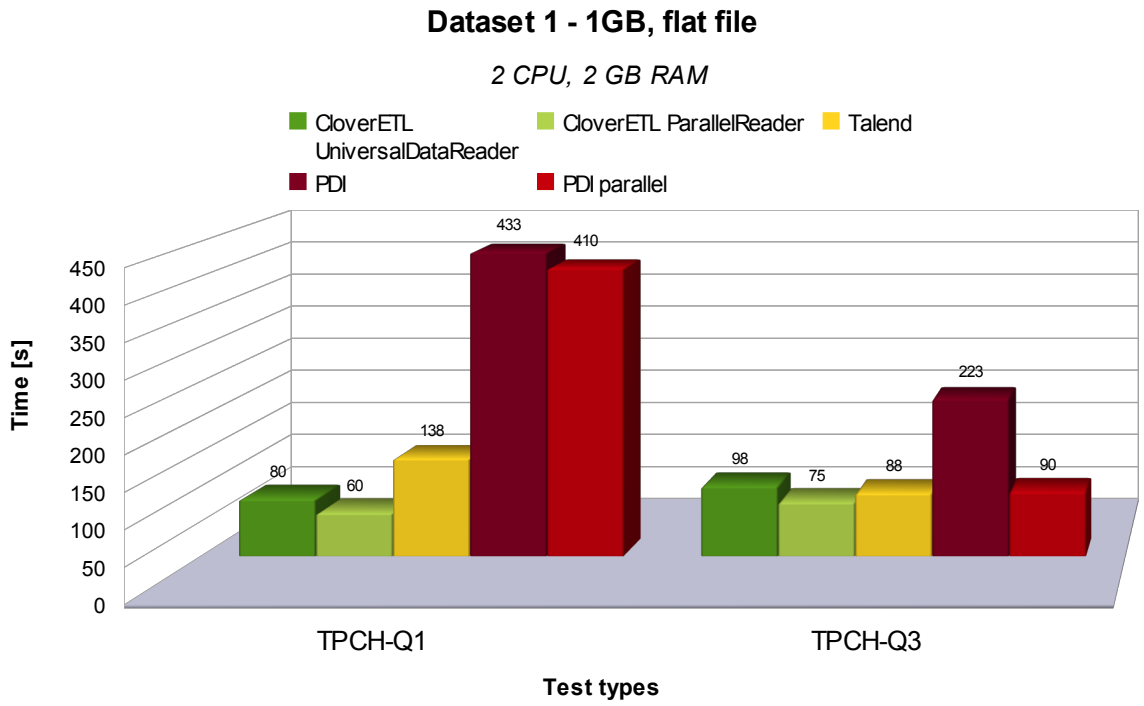
PERFORMANCE RESULTS

We have run both tests Q1 and Q2 with two datasets. The smaller one (Dataset 1) on System A, the size of data is 1GB (~ 7.5 million records) and the larger one (Dataset 2) on System B, 10GB of data (~ 75 million records).

For the first run we decided to read and write data directly from/to text files on local filesystem in order to minimize external factors. The tests on System B were subsequently run with using following database system:

- MySQL Enterprise Edition 5.1.31

Dataset 1 results on System A, data stored in flat file



Dataset 1 on System A	**TPCH-Q1** [s]	**TPCH-Q3** [s]
CloverETL UniversalDataReader	**80**	**98**
CloverETL ParallelReader	**60**	**75**
Talend	138	88
PDI	433	223
PDI parallel	410	90

RESULTS COMMENTS

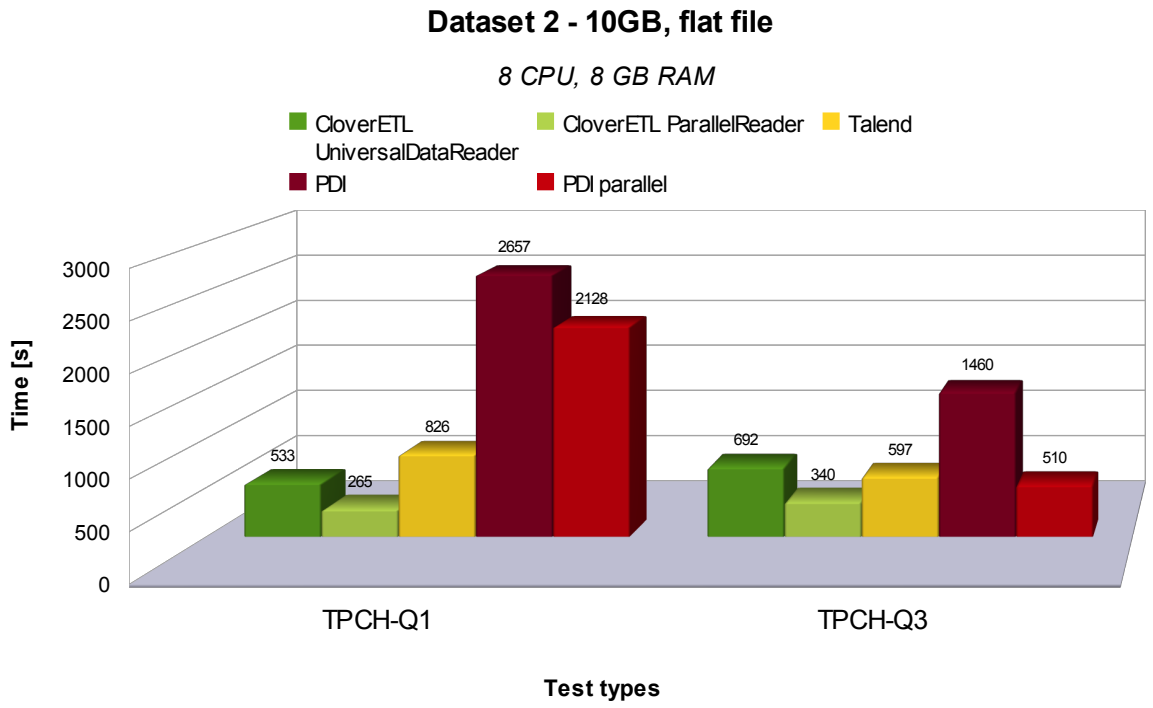
CloverETL, Talend and PDI tests were run using Java 1.6 with 1,5GB heap memory and Java in 'server' mode (java parameters: -Xms256m -Xmx1536m -server). All tools were run with multithread execution support.

PDI parallel reads the data with "Parallel reading" feature switched on. But it doesn't bring significant performance improvement in TPCH-Q1 test because PDI is very limited by necessity of sorting data before aggregation.

The source files contained the following number of records:

- lineitem - 6,001,215 records, 724 MB
- customers - 15,000 records, 23.2 MB
- orders - 1,500,000 records, 163 MB

Dataset 2 result on System B, data stored in flat file



Dataset 2 on System B	TPCH-Q1 [s]	TPCH-Q3 [s]
CloverETL UniversalDataReader	**533**	**692**
CloverETL ParallelDataReader	**265**	**340**
Talend	826	597
PDI	2657	1460
PDI parallel	2128	510

RESULTS COMMENTS

CloverETL, Talend and PDI were run using Java version 1.6 with 1,5GB heap memory in 'server' mode (java parameters: -Xms256m -Xmx1536m -server). All tools were run with multithread execution support.

PDI parallel reads the data with "Parallel reading" feature switched on. But it doesn't bring significant performance improvement in TPCH-Q1 test because PDI is very limited by necessity of sorting data before aggregation.

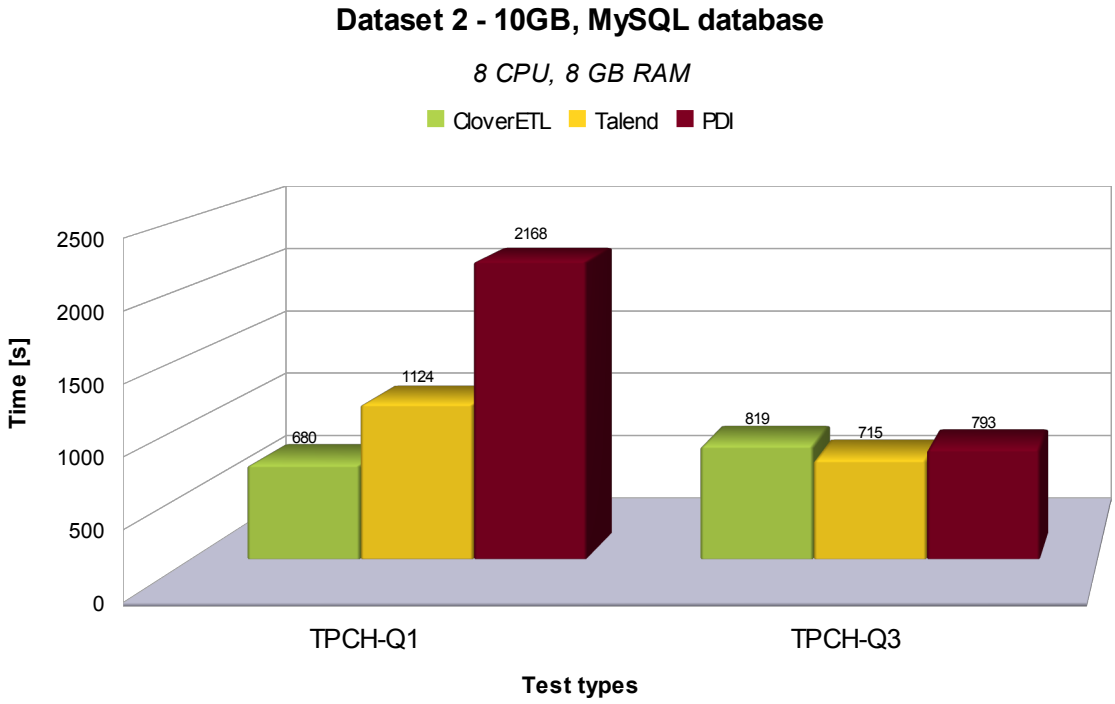
The source files:

- lineitem - 59,986,052 records, 7.24 GB

- customers – 1,500,000 records, 233 MB
- orders - 15,000,000 records, 1.62 GB

Talend was able to use only 1 CPU, even though our server had 8 – despite the fact that we enabled multithreaded setting in Talend's options.

Dataset 2 result on System B, data stored in MySQL database



Dataset 2 on System B	TPCH-Q1 [s]	TPCH-Q3 [s]
CloverETL	**680**	**819**
Talend	1124	715
PDI	2168	793

RESULT COMMENTS

CloverETL, Talend and PDI were run using Java version 1.6 with 1,5GB heap memory in 'server' mode (java parameters: -Xms256m -Xmx1536m -server). All tools were run with multithread execution support.

The source data was the same volume as in the previous test, only loaded into database tables.

For PDI we had to modify MySQL server parameter "net_write_timeout". Without this modification PDI's TPCH-Q3 transformation failed.

SUMMARY

We have reviewed several popular ETL tools. So what can we conclude?

We see an interesting spread of features, options and prices. The situation is no more black and white; there used to be the “Informatica, DataStage and AbInitio” triumvirate and then “the rest”. The choice was relatively easy and it largely depended on your budget to spend on the ETL domain.

These days, when economy is slow and cutting costs is no more an option but rather necessity, it comes very handy you can find a quite powerful and reliable match to the “big three”. When you look at the Total Cost of Ownership and consider not only license but also development and support fees, the choice becomes quite obvious.

APPENDIX A - INDIVIDUAL COMPARISONS

CloverETL vs. Talend

Physical Architecture

Both **Talend** and **CloverETL** use the Eclipse framework for their visual editors of ETL transformations and use JRE for running the transformations. Talend's transformation can be also run as Perl script.

Talend generates code, then compiles and runs it. Any changes to the code aren't reflected in visual graph and have to be maintained manually. Talend's transformation can be run separately without visual transformation editor as jar library.

CloverETL is a metadata based tool, it does not require any code generation in order to run jobs. Changes made to transformations outside visual editor are reflected once loaded back into the designer. The transformations are stored as XML files that are easy to read and adjust. CloverETL's transformations can be run outside CloverETL Designer by open source library CloverETL Engine.

Transformation Development

Talend has a large number of components however a lot of them provide the same function under different name. Moreover some of the components we consider as essential – e.g. persistent lookups – are missing.

CloverETL brings a smaller pallete of components but their functionality is more complex, they beat Talend's equivalents in many aspects. It's also easier to choose suitable component in CloverETL's pallete than in Talend's pallete.

Expressive power

Both tools can use Java as scripting languages. Talend also enables to use JavaScript and Groovy.

CloverETL has a special ETL scripting language – CTL (Clover Transformation Language) which is easy to learn and enables users without programming skills to develop a complex transformation in a short time.

Parallelism

Both products support component and pipeline parallelism to speed up executions.

Our tests show that Talend is not able to efficiently utilize more CPUs to speed up an execution.

CloverETL and Talend both provide enterprise solutions that bring more parallel and performance features.

CloverETL vs. PDI

Physical Architecture

PDI is distributed as a standalone application. **CloverETL Designer** is distributed as both a standalone application and an Eclipse plugin. It allows Eclipse users to add CloverETL to their current Eclipse installation. Both products are written in Java and can be easily moved between different platforms.

Both CloverETL and PDI store transformations in XML files. It is possible to run transformations outside of the visual editor, using only Java libraries (CloverETL Engine and Pentaho's Kettle libraries).

Transformation Development

Even for an experienced ETL developer **PDI** is definitely more difficult to learn than CloverETL. Components often have unexpected names and confusing interface (e.g. calculator component). Many components require sorted input, making graphs more cluttered. Many operations aren't intuitive and the documentation is very poor.

CloverETL Designer's environment resembles that of Informatica or DataStage.

Expressive power

PDI uses JavaScript or Java for calculations/formulas.

CloverETL uses CTL - ETL scripting language. There is an option to use Java as well.

CloverETL has several important ETL components that are missing in PDI - such as persistent lookups or aggregation of unsorted input.

Parallelism

Both tools provide components that can be run in parallel mode. Both can also run in a clustered environment. But in PDI it can produce unexpected results if you don't exactly know how to use it.

APPENDIX B – ETL TOOLS MAPPINGS

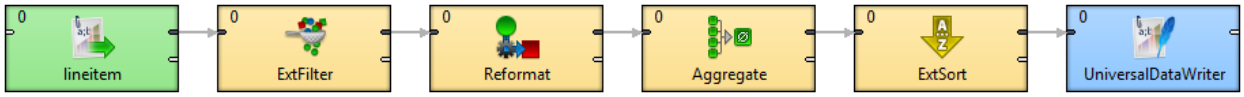


Figure 1 - CloverETL TPC-H Q1

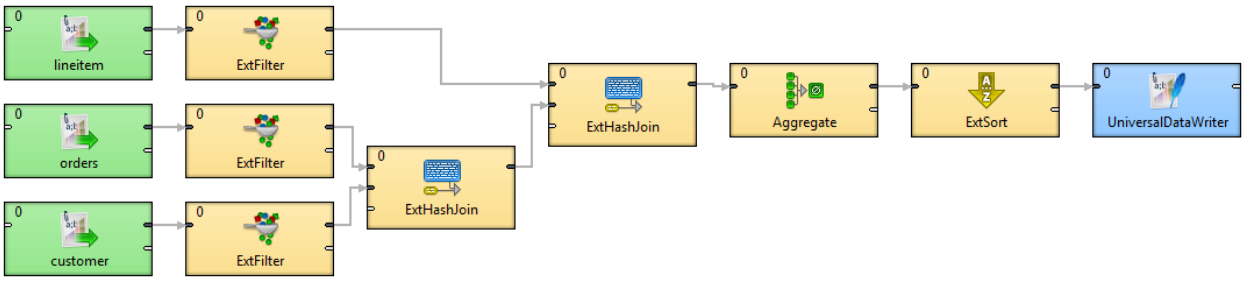


Figure 2 - CloverETL TPC-H Q3

Figure 3 - Talend TPC-H Q1

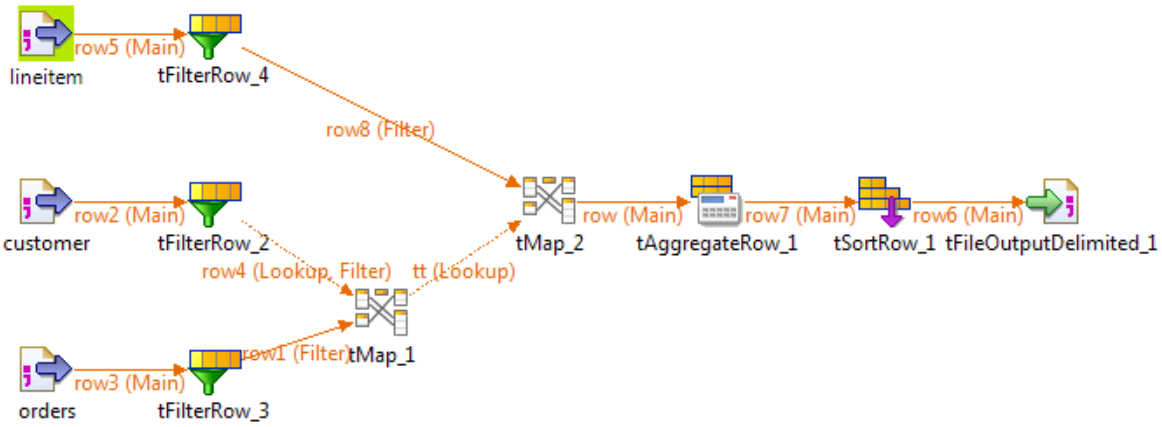


Figure 4 - Talend TPC-H Q3

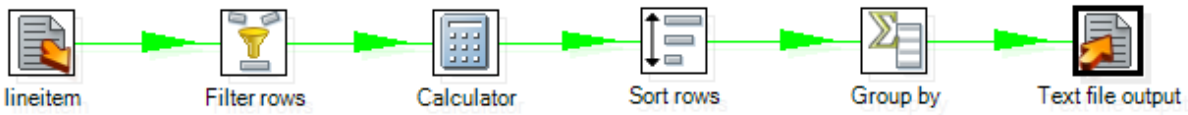


Figure 5 - PDI TPC-H Q1

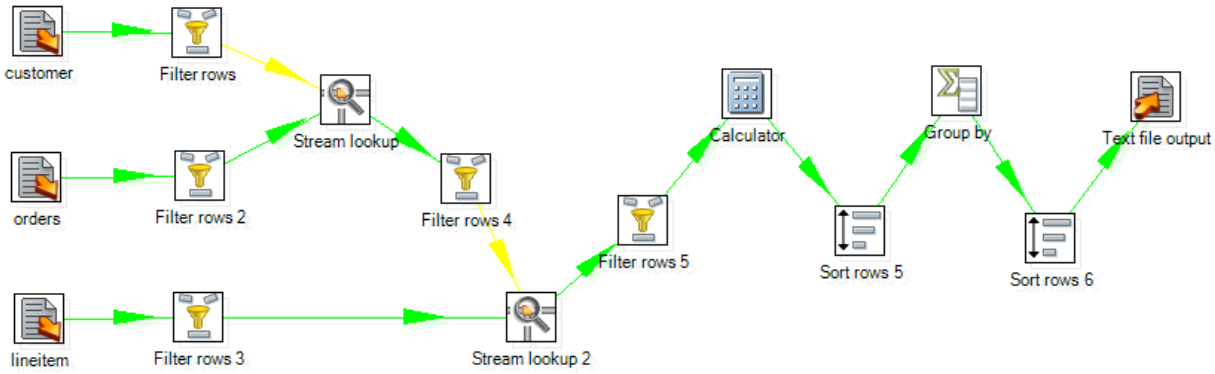


Figure 6 - PDI TPC-H Q3

APPENDIX C - TPC-H TESTS DATA MODEL (ERA)