

Taurus DataBridger Evaluation at Avmed Health Plans

Turbo-Image to Oracle Migration

1. Definitions

- a. GB — Gigabyte
- b. GigE — Gigabit Ethernet
- c. Mb/s — Megabit per second
- d. Gb/s — Gigabit per second
- e. MHz — MegaHertz
- f. HBA — Host bus adapter
- g. SAN — Storage Area Network

2. Infrastructure

- a. Source Machine
HP e3000 N4000-400-55 running MPE/iX v7.0, four 550 MHz CPU's, 6 GB memory, 10 F/W SCSI channels attached to an EMC Symmetrix 3830 Storage Array and one 100 Mb/s network interface adapter
- b. Target machine
HP RP7410 n-partition running HP_UX 11i v1 (B11.11), four 875 MHz CPU's, 8 GB memory, two 2 Gb/s HBA's, two 2 Gb/s SAN switches, EMC Clariion CX600 SAN and one 1 Gb/s network interface adapter
- c. Network
100BaseT full-duplex connection from the HP e3000 to a Cisco Catalyst 6500 switch and GigE full duplex from the RP7410 to the same switch

3. Test Environment

The source machine was running the end of our normal batch production schedule when we started the migration. We had four Suprtool processes running against the MPE warehouse server for over an hour. After that, the source machine was essentially dedicated to the warehouse server process. We re-scheduled the next batch cycle to allow more time with minimum contention but we allowed on-line inquiry and updates to proceed normally.

The target machine was dedicated to the DataBridger process. I tried, somewhat successfully, to keep four processing threads running at all times.

Taurus DataBridger Evaluation at Avmed Health Plans

Turbo-Image to Oracle Migration

4. Noteworthy Events

I replaced the conv_all.sh and ~/master_script shell scripts. Conv_all.sh runs the entire migration process single thread. Several of the master_script shell scripts had syntactical errors that precluded them from running. There was an incorrectly named table row in los_table_m.sh that caused that script to abort. I removed all scripts for empty datasets and for the entire ITS and Workflow databases. We left all indices active during the migration. We disabled the foreign key constraints on all tables. Sqldr automatically disables triggers.

We incorrectly specified the storage for several table spaces. This prevented ten of the major tables from loading completely on the first attempt.

5. Measurements

I corrected los_table_m.sh and resumed processing that thread from the point of failure. I did a second run to migrate the ten tables that aborted the first time. I included the time wasted by these errors in the throughput statistic.

a. Run 1 August 1, 2004

i. Thread 1

1. Bytes transferred 18,617,798,298
2. Elapsed time 8.06 hours
3. Transfer rate 2,309,900,503 bytes per hour

ii. Thread 2

1. Bytes transferred 20,523,321,252
2. Elapsed time 8.57 hours
3. Transfer rate 2,394,786,611 bytes per hour

iii. Thread 3

1. Bytes transferred 13,217,810,588
2. Elapsed time 7.98 hours
3. Transfer rate 1,656,367,242 bytes per hour

iv. Thread 4

1. Bytes transferred 12,403,169,196
2. Elapsed time 3.28 hours
3. Transfer rate 3,781,454,023 bytes per hour

Taurus DataBridger Evaluation at Avmed Health Plans

Turbo-Image to Oracle Migration

b. Run 2 August 3, 2004

i. Thread 1

1. Bytes transferred 12,764,856,614
2. Elapsed time 4.71 hours
3. Transfer rate 2,710,160,640 bytes per hour

ii. Thread 2

1. Bytes transferred 9,964,031,888
2. Elapsed time 4.26 hours
3. Transfer rate 2,338,974,622 bytes per hour

iii. Thread 3

1. Bytes transferred 3,854,352,198
2. Elapsed time 1.44 hours
3. Transfer rate 2,676,633,471 bytes per hour

iv. Thread 4

1. Bytes transferred 18,381,316,046
2. Elapsed time 5.28 hours
3. Transfer rate 3,481,309,857 bytes per hour

c. Aggregate times

i. Thread 1

1. Bytes transferred 31,382,654,912
2. Elapsed time 12.77 hours
3. Transfer rate 2,457,583,209 bytes per hour

ii. Thread 2

1. Bytes transferred 30,487,353,140
2. Elapsed time 12.83 hours
3. Transfer rate 2,376,358,016 bytes per hour

iii. Thread 3

1. Bytes transferred 17,072,162,786
2. Elapsed time 9.42 hours
3. Transfer rate 1,813,186,985 bytes per hour

iv. Thread 4

1. Bytes transferred 30,784,485,242
2. Elapsed time 8.56 hours
3. Transfer rate 3,597,135,476 bytes per hour

Taurus DataBridger Evaluation at Avmed Health Plans

Turbo-Image to Oracle Migration

- d. Reinstate foreign key constraints.
We enabled all foreign keys in six minutes using a customer written script.
- e. Enable primary key triggers and calculate next value.
We ran this process on all tables in about six seconds using a customer written script.

6. Observations and conclusions.

This was a preliminary test of the product aimed at determining how long the migration would take. Amisys/Syntertech, LLC supplied the original scripts. We expected to encounter some problems and we did. Take note that the problems were with the customer/third party developed scripts not with the migration tool. This test demonstrated that we could easily migrate and validate the data in a single day with our data volume and the current condition of the data. We had no records rejected by *sqlldr*.

The Taurus DataBridger product works well. The MPE version of Warehouse acts as a server reading data from the individual datasets, and writing it directly to a named pipe. The Unix version reads the pipe applying any data correction rules to the record and passes it directly to *sqlldr*. There is no intermediate storage required or used.

It puts a small load on the MPE system. Using Lund's MetaView monitoring tool, I measured a maximum aggregate one-minute CPU load of slightly over 16% for all four concurrent threads. The average load was less than 11% for four concurrent threads.

I believe that we can improve on this performance. Obviously, we will not have the space or syntactical errors again.

By more evenly distributing the number of datasets per thread, we can spread and balance the processing load. During run 1, Thread 3 had 90 datasets totaling over 13.2 billion bytes and it ran more than twice as long as thread 4 which had about 12.4 billion bytes in 37 datasets.

By experimenting further, we can determine an optimal number of threads and the dataset sequence within each thread. Four threads may be too few or too many. We may want to start some threads with tiny datasets and work up to the jumbos and start others with jumbos and work down to the tiny datasets. During the first run, each thread started with the smallest datasets and worked up to the multi-billion byte ones.