Eloquence Training

What’s new in Eloquence B.08.00

© 2010 Marxmeier Software AG

Overview

- Released December 2008
- Supported until November 2013

- Supports 32-bit and 64-bit platforms
  - HP-UX Itanium and PA-RISC
  - Linux x86, x64 and Itanium
  - Windows x86 and x64

- Installed in a separate location and may be used concurrently with previous Eloquence versions on HP-UX and Linux

Technological enhancements

- Implements new thread model for Eloquence database server
- Provides base for future enhancements

- Adapt Eloquence technology to newer HW and OS capabilities
  - Larger number of CPU cores
  - CPU speed increases are more moderate
  - Larger memory sizes
  - OS level threading improvements

- Additional platform support

Functional enhancements

- Implements major functional and scalability improvements in the database server
- Performance enhancements
- Security and monitoring improvements
- Support for case insensitive indexes
- Enhances a broad range of Eloquence components
- Some functional enhancements are backported to B.07.10

Database Scalability

- Address needs of large customers that need additional performance headroom
  - previous Eloquence was limited to 4000 concurrent connections
  - practical limit was lower depending on activity, independent of hardware
  - dedicated database cache was limited to 1 GB (plus OS shared cache)

- Scalability improvements
  - make use of available CPU resources (improved multi-threading)
  - improved scalability on large memory configurations
  - release 64 bit versions
  - optimizing concurrency

- Improves throughput on contemporary hardware

Eloquence thread model

Eloquence B.07.xx versions use its own threads implementation

- Designed when OS threading support was limited
- Low overhead
  - minimum overhead on locking, context switches, memory efficient
- Two types of threads
  - internal threads (used for application tasks)
  - OS interface threads (uses separate process on HP-UX, kernel threads on Linux and Windows), used for I/O

Eloquence thread model

Eloquence B.08.00 uses OS native threads

- Improves scalability and latency (depending on workload)
- Improves utilization of modern hardware
- Additional system resources needed

Performance enhancements

Various performance enhancements to improve scalability, such as

- Reduce lock contention during cache-miss handling
- Improved concurrency with client-side caching
- Revised lock scheduler scalability for competing locks
- More efficient forward-log format for index and meta data

Database replication

- Replicates entire server instance, <u>not</u> individual database
- Replication is unidirectional (master / slave)
- Replication is asynchronous but close to real time
- Slave server(s) are are read-only, write access may be redirected to master
- Replicates server transactions, <u>not</u> IMAGE calls
- Requires little maintenance
- Optional feature, requires separate license key

Replication use cases

- Hot standby server for disaster failover
- Load sharing by moving reporting to slave
- Incremental remote backup
- „time snapshots“ on slave for fallback or reporting
- Distributing central data to branch offices

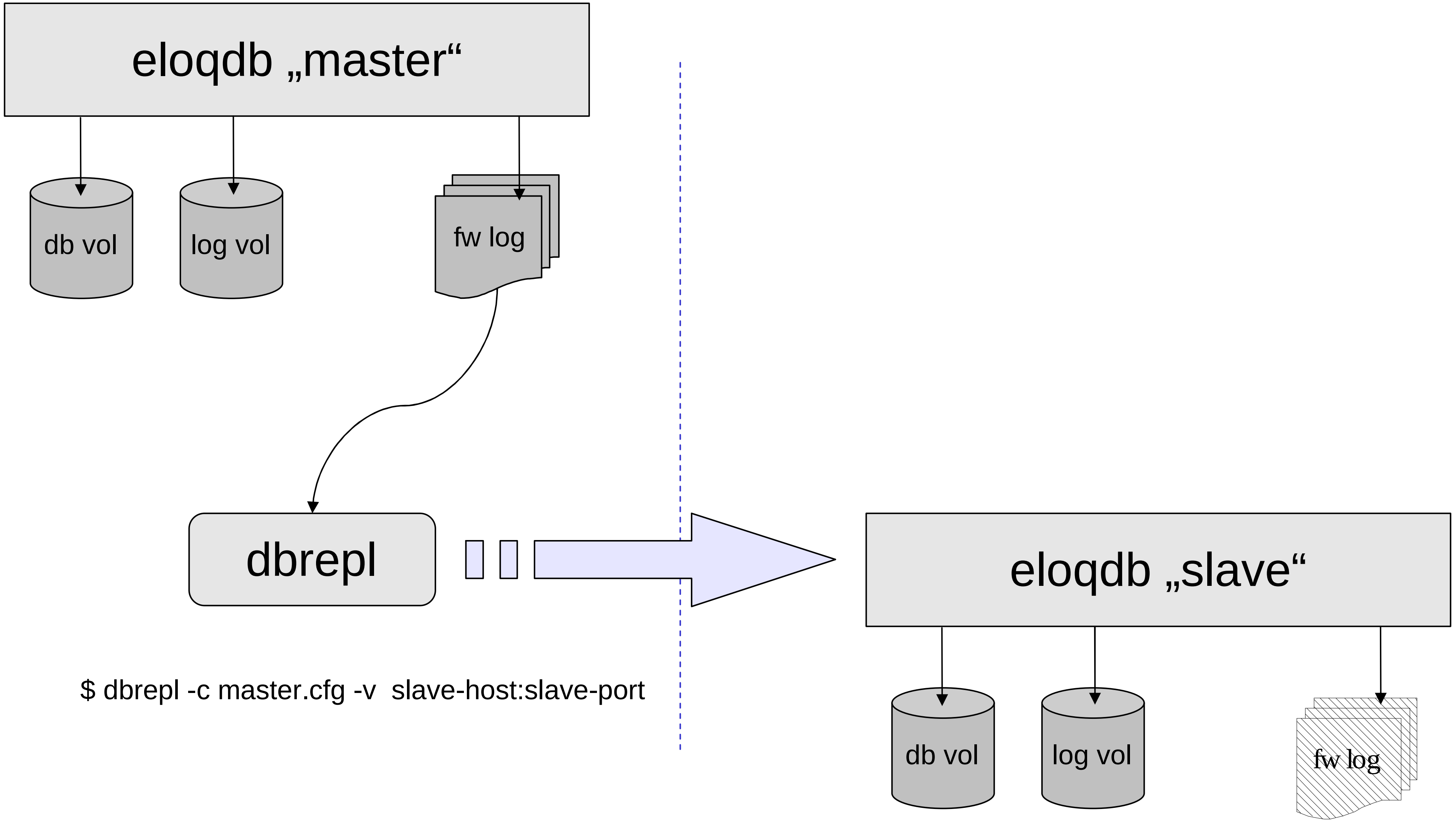
Database Replication

- Master and slave server must have compatible architectures
- Master and slave server may reside on the same or different machines
- Requires fw logs on master server

- dbrepl utility reads fw logs and interacts with slave server
  - Contacts slave server to obtain replication status
  - Follows fw logs on master and replicates to slave

- “dbctl replication status” may be used to obtain replication progress/status from master and slave server

Database replication



```
$ dbrepl -c master.cfg -v  slave-host:slave-port
```

Additional recovery options

- Point-in-time recovery
  - recover from backup to a specific point in time
  - requires previous backup and fw log files since then

- Incremental recovery
  - recovery may continue from last point
  - server may not be started in between

Database monitoring

- Additional options for server performance logging
  - Dynamically enable/disable performance logging via dbctl
  - Simplified integration with monitoring framework

- Logging of session performance information
  - session-specific performance monitoring
  - Disk accesses, DB calls and elapsed time
  - Optionally log information at a specified frequency in addition to session completion
  - Dynamically enable/disable logging via dbctl
  - Useful to analyze application performance problems
  - Also available interactively through http status

- Enhancements to http status pages

Session Performance Log

- Logging of session activity and performance information
  - Disk accesses, DB calls and elapsed time
  - Optionally log information at a specified frequency in addition to session completion
  - Also available interactively through http status
  - Useful to analyze application performance problems

- Configured in the config file
  - [server] SessionStatFile specifies log file
  - [server] SessionStatMode specifies log frequency
    - 0 = off, 1 = at end of session, > 1 also log at specified interval (sec)

- May be configured dynamically with dbctl
  - dbctl sessionstatfile [FileName|DISABLED]
  - dbctl sessionstatmode [mode]

fwutil library

- Programmatic access to db transactions
  - easy way to monitor database changes from custom program
  - incremental and asynchronous, typically close to real time
  - fwutil library performs complex work and isolates utility from any internals
  - Uses fw logs and audit information
- Example uses
  - Implement custom actions on database changes
  - Data extraction and reporting (data warehouse)

fwutil library

- Custom program needs to be written in C and linked with the fwutil library
- Program passes control to the fwutil library
- fwutil library extracts information from fw logs and invokes customer defined callback functions
- fwutil library works incrementally, saving its progress in a status file

- Example programs are available
  - fwtest.c - print some information on database transactions
  - fwsql.c - convert database changes to SQL like syntax

More information

Detailed information is available on the Eloquence web site

http://eloquence.marxmeier.com

Get in contact

info@marxmeier.com