Polyhedra Database Systems
Polyhedra and Polyhedra DBMS

Polyhedra® is a family of compact, lightweight, SQL relational database management systems (RDBMS) optimized for embedded systems applications. Built on an ironclad clientserver architecture, Polyhedra® in-memory database system (IMDB) and Polyhedra® FlashLite provide a secure data repository for embedded systems, mobile devices and server-based applications demanding the utmost in performance and reliability.

Polyhedra® In-Memory Database for High-Availability Systems

Polyhedra® is the industry's leading in-memory relational database management system. First released in 1993, Polyhedra is now used at the core of some of the worlds most advanced and important data-intensive 3G wireless, IP network infrastructure, and industrial control systems. Enea Polyhedra's memory-resident design and active technology can boost performance by orders of magnitude compared to traditional disk-based RDBMSs and available in 32-bit and 64-bit versions. Polyhedra also provides high-availability features such as transaction journaling, hot standby support, dynamic schema update and inter-version compatibility that make it the RDBMS of choice for equipment makers and service providers who require '5 nines' or better uptime.

In-memory performance

High performance and low latency are critical for real-time systems. Enea Polyhedra's memory resident, event driven design enables it to deliver ACID compliant, millisecond-level performance (even on slow machines) and service requests that would other wise be unworkable in a real-time environment. Polyhedra"s active query mechanism further enhances performance by not ifying client applications immediately after a relevant data change.

Polyhedra DBMS Database for Mobile and Diskless Applications

Polyhedra DBMS is a compact !ash-based SQL data base system optimized for low-power device such as feature phones and diskless systems such as home gateways and set-top boxes. Built on the same rock solid client-server foundation as the in-memory Polyhedra RDBMS, Enea Polyhedra DBMS provides full ACID compliance while greatly reducing system cost and power consumption.

Minimal memory consumption

In portable applications, RAM usage must be kept low – not only to reduce the bill of materials, but also to minimize power consumption. By allowing data to be stored in !ash memory rather than RAM, Polyhedra DBMS greatly reduces RAM requirements, while still delivering the functionality of a full-featured RDBMS.

A typical Polyhedra DBMS configuration requires less than one megabyte of RAM for code and working space (including code, stack, heap, and a cache page). In fact, because the code is ROMable, and the cache size is controllable, designers can reduce RAM usage to as little as 200 kbytes if necessary.

Core Polyhedra Technology

All Polyhedra DBMS products, including Polyhedra and Enea Polyhedra DBMS, share the same code base as well as most key features and benefits. Common features include:

PRODUCT FEATURES

Polyhedra In-Memory Database

- Memory resident, real-time performance
- Active SQL relational database
- Continuous availability
- Active query technology: no need to poll
- Available on many platforms
- Heterogeneous client-server architecture
- Extended ODBC API for C/C++ clients
- Portable JDBC driver for Java clients
- Fully transactional and ACID-compliant

Continuous availability, fault tolerant

The database systems used in next generation wireless and broadband network infrastructure equipment require a high level of availability. Polyhedra enhances availability by reducing susceptibility to single points of failure, providing fault-tolerant mechanisms that ensure continuous client operation and transaction integrity.

Polyhedra uses a hot-standby mechanism to maximize availability. The standby database is continually informed about changes to the master database. This enables the standby to

ENEA

FACTS

ACID-Compliant Databases

ACID is an abbreviation for Atomic, Consistent, Isolated and Durable. All of these transactional properties are vital for a reliable database system.

Atomic Transactions are all-or-nothing – the system never does just part of what is asked.

Consistent Transactions take the system from one consistent state to another – the system rejects any transactions that break the rules.

Isolated Transactions operate independently, as though fully serialized, with intermediate, mid-transaction states hidden.

Durable If a database indicates that a transaction has successfully completed, the application can rely on the changes being preserved even if there is a system failure.

take over immediately when it receives notification of a failure to the master. In addition, the client libraries can automatically switch over to the new master when they lose touch with the old master. Very little application code is required to take advantage of these high-availability features.

Figure 1 shows a typical fault-tolerant configuration. When a failover occurs, user applications automatically reconnect from one server to another in a seamless fashion, without the need for special coding in the application. A journal logging mechanism ensures that critical data survives major failures (such as power outage) that require a full restart of the entire system. Replica servers may also be configured, allowing the off-loading of frequent or complex queries to other machines.

Distributed, scalable client-server Architecture

Polyhedra's client-server architecture enhances data integrity and resilience by separating data from the applications that use it, thereby protecting the memory used by the database software from accidental modification. Enea Polyhedra's heterogeneous client/server architecture also enhances flexibility, enabling distributed applications to seamlessly access Polyhedra databases residing on any combination of hardware platforms. This !flexibility enables Polyhedra database implementations to scale linearly, from a single node to large clusters and networks. The ability to remotely access the database also simplifies system development and testing, even when the system is intended for standalone use.

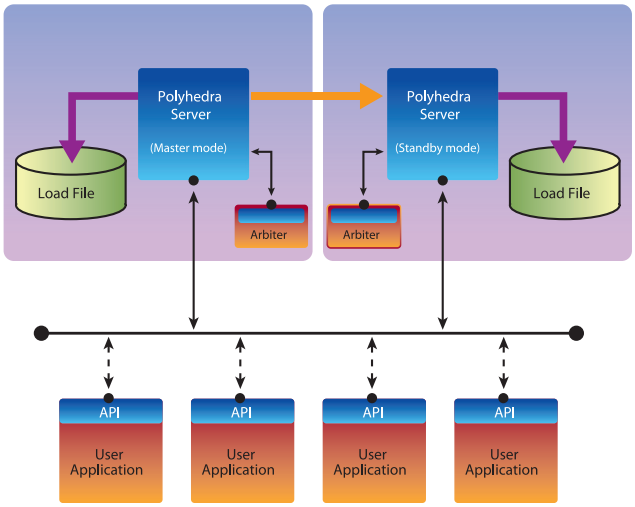

Figure 1. The Enea Polyhedra® database management system in a fault tolerant coniguration.

PRODUCT FEATURES

Polyhedra FlashLite DBMS

- Standards-based: SQL, ODBC and JDBC
- Designed for both NOR and NAND • ash
- As little as 200 Kbytes RAM
- Fully transactional and ACID compliant
- Client-server architecture, so application failure does not corrupt the database
- Multi-platform, with cross-platform interoperability
- Continuous availability
- Active queries avoid the need for polling
- Trigger language for embedding business logic in the database
- Database • les are not tied to any particular location or machine. Thus, they can be copied for backup purposes, o• ine analysis, or to send to another machine.

Polyhedra's heterogeneous design frees client programmers from compatibility concerns such as differing endianisms. At the same time, support for standard protocols like TCP/IP and platform specific transports such as Enea OSE® messaging enables programmers to tackle the most complex distributed applications without compromising performance.

Active technologies boost performance and simplify development

Polyhedra's active queries boost performance by enabling developers to query data using standard SQL select statements, and then be kept up to date if the query causes a data change. When a data change occurs, the server automatically transmits a "delta" message to the client. This automatic notification eliminates the need for clients to poll the server or reissue queries to determine what has changed.

Polyhedra provides another active technology that lets programmers associate "behavior" with data, behavior

ENEA

that can be triggered by changes to the data. This active technology simplifies the development of client applications and enhances robustness by enabling developers to set up active, triggered, database-resident code, which enables application-level data integrity rules to be built right into the database. Embedding "business logic" in this way improves the overall "correctness" of the information in the database. It also simplifies application development by centralizing consistency checks and enabling the database to handle knock on changes without the need for external application code.

Faster development, standard client apis

Polyhedra's portability and crossplatform support greatly enhance flexibility during development and testing. Polyhedra also simplifies application design by offering standard APIs that make it easy to interface client applications with Polyhedra databases. These standard APIs enable users of data-aware objects to combine event driven applications with a high-performance, event-driven SQL database.

- For C/C++ developers, Polyhedra provides an ODBC-compliant API with extensions for event-driven programs using active queries, as well as a proprietary API that uses a callback model. Polyhedra® provides a native ODBC support library for each supported platform. The library does not require an ODBC driver manager.

- For Java developers, Polyhedra® provides a Type 4 portable JDBC driver, which supports all the features of Polyhedra, including active queries. Additional support for Windows programmers includes an ODBC driver, a DDE driver and an OLE/DB provider, which gives C++ and .NET applications a variety of ways to access Polyhedra databases.

Polyhedra® feature comparison chart	Polyhedra 64 IMDB	Polyhedra 32 IMDB	Polyhedra Lite	Polyhedra Flash DBMS
SQL relational database, supporting views; INSERT, UPDATE, DELETE; queries with joins; foreign keys; cascaded deletes	✓	✓	✓	✓
Dynamic table creation, alteration and deletion	✓	✓	✓	✓
	✓	✓	✓	✓
▪ 64-bit, 32-bit, 16-bit and 8-bit signed integers ▪ 64-bit and 32-bit floating point values ▪ Bounded and unbounded character strings (+ UNICODE option) ▪ Unbounded binary objects ▪ Datetime ▪ Boolean	✓	✓	✓	✓
Table inheritance; 'shared' and 'virtual' attributes	✓	✓	✓	✓
Polyhedra domain and array attributes	✓	✓	✓	✗
Ability to mark tables and columns as persistent or transient	✓	✓	✓	✓
Ordered and hash indexes	✓	✓	✓	✓
Client-server architecture, keeping data separate from the applications that use it (and thus safer)	✓	✓	✓	✓
Standards-conformant ODBC and JDBC client libraries	✓	✓	✓	✓
Cross platform support, interworking with other Polyhedra products and other versions of the products	✓	✓	✓	✓
Platform-independent database files	✓	✓	✓	✓
Fully transactional, with ACID operation	✓	✓	✓	✓
Support for fine-grained optimistic and pessimistic locking	✓	✓	✓	✓
Provision for snapshots, for offline storage and analysis	✓	✓	✓	✓
Mechanisms for data durability	**Snapshots plus journal logging**			**Shadow paging**
	✓	✓	✗	✓
Support for read-only replica servers	✓	✓	✗	✗
Active queries, with clients automatically updated	✓	✓	✓	✓
Trigger code, running in-transaction to enforce business logic	✓	✓	✓	✓
Interface to PLCs, RTUs ('the DVI module')	✓	✓	✗	✗
Historian module, to store and retrieve time-series data	✓	✓	✗	✗
Where is the master copy of the data stored?	**RAM**	**RAM**	**RAM**	**File**
Configurable RAM cache	n/a	n/a	n/a	✓
Configurable limit on total file size	n/a	n/a	n/a	✓
Configurable limits on size of individual tables	✗	✗	✗	✓
Configurable limits on total RAM usage	✓	✓	✓	✓
Configurable limit on space used on behalf of any client	✓	✓	✓	✓
Debugging module	✓	✓	✗	✓
Optimistic and pessimistic locking	✓	✓	✓	✓

ENEA

Enea LINX and Enea OSE messaging for enhanced Performance and availability

Enea Polyhedra software components running on Linux can optionally be configured to use Enea® LINX interprocess communication. Enea LINX messages allow clients to access a local or remote server in a transparent fashion, provide higher performance than TCP/IP when working locally, and allow a variety of underlying transport mechanisms to be used to connect to remote servers. They also enhance reliability and availability by facilitating remote process notification for failed connections, which simplifies the implementation of fault-tolerant failover control mechanisms. In a similar way, OSE messaging can be used when running Enea Polyhedra on the Enea OSE RTOS.

Supported Platforms

Polyhedra servers and C/C++ client libraries are supported on the following platforms. The platforms are fully interoperable, enabling clients on any platform to connect to databases on any other platform. This is also truefor applications that combine Enea Polyhedra and Enea Polyhedra FlashLite in the same system.

New platforms are added regularly, so please check our web site for an up-to-date list. Porting services are also available for platforms that are not currently supported.

Polyhedra memory-resident RDBMS Embedded Systems

OSE	PowerPC; Solaris and Linux SoftKernel
VxWorks	PowerPC, x86 and compatibles; Windows VxSim
Embedded Linux	x86 and compatibles, PowerPC, ARM LynxOS x86 and compatibles
Integrity	PowerPC
Windows CE	ARM

Workstations and Servers

Linux	x86 and compatibles, PowerPC
Solaris	SPARC and x86_64 architectures
AIX	Power5
Windows	x86 and compatibles; 2000, XP, 2003

Polyhedra64

Linux	x86_64 architecture
Windows	x86_64 architecture
Solaris	Sparc and x86_64 architectures
AIX	Power5

Polyhedra Fashlite RDBMS

This can be provided on any platform supported by the 32-bit version of the Enea Polyhedra IMDB product.