*eXtreme*DB®

Hybrid in-memory/on-disk database system for maximum performance and data durability.

"eXtremeDB is well-designed to solve the unique development challenges of data management for real-time embedded software."

—Schneider Electric

*eXtreme*DB, the real-time embedded database for devices that are eXtremely innovative

Overview

In-memory database systems (IMDSs) offer superior performance and the possibility of very small RAM, CPU and storage demands. IMDSs boost speed by eliminating file system I/O, multiple data copies, and redundant processes, such as caching. This streamlined design can also dramatically reduce system footprint.

In contrast, on-disk databases cache frequently requested data in memory, for faster access, but write database inserts, updates and deletes through the cache to persistent storage. Byte-for-byte, disk storage can cost less than memory, and require less physical space: RAM chips can't yet approach the density of a micro-drive, for instance. So, for small form-factor devices with large storage needs, such "spinning memory" can be better.

*eXtreme*DB provides the best of both worlds, marrying in-memory database technology with the traditional disk-based database system. The result is a hybrid database for resource-constrained and high-performance systems that affords developers the ultimate in flexibility.

McObject's *eXtreme*DB

Since its introduction, McObject's *eXtreme*DB has set the standard for small footprint, in-memory embedded database systems, offering benefits including:

- **Tiny code size** of approximately 200K or less
- Blazing speed: **micro-second transactions** even on modest hardware
- C/C++ developers benefit from a **type-safe, intuitive API** with extensive checking to speed development
- Optional **SQL** and **XML** interfaces
- **Java Native Interface (JNI)** affords Java developers the ease of working with "plain old Java objects" (POJOs)
- **High Availability Edition**, with asynchronous (1-safe) or synchronous (2-safe) replication, for applications requiring complete fault tolerance
- **Available source code**, for porting to new platforms and highest degree of control over development
- **64-bit edition** scales beyond 1TB in-memory data
- **Multi-version concurrency control (MVCC)** transaction manager and advanced memory management fully leverage multi-threaded, multi-core systems

*eXtreme*DB: Best of Both Worlds

*eXtreme*DB enables the developer to combine in-memory *and* on-disk paradigms in a single database system. Specifying that data will be stored in memory (transient), or on disk (persistent), requires a simple database schema declaration, as shown below.

```
transient class classname {
      [fields]
};

persistent class classname {
      [fields]
};
```

The resulting system retains in-memory strengths (speed, footprint, etc.), yet leverages the potential cost savings and durability of an on-disk database.

Key On-Disk Database Features

*eXtreme*DB's on-disk features are uniquely configurable, including:

- Three transaction logging policies – Undo, Redo and No Logging – to meet the target system's footprint, performance and durability needs
- Synchronous or asynchronous transaction logging
- Developers can specify the maximum database size, which is especially important when the 'disk' is actually a flash memory file system
- Database cache can be saved and re-used across sessions – for example, so a user can resume some activity when a device is switched back on
- The database can exist in one file, to simplify maintenance, limit I/O and reduce size
- Logical Database Devices feature can spread a database across multiple disks, including in a RAID, with the database striped across RAID disks
- Or, pages can be written simultaneously to multiple RAID disks for perpetual backup

With these tools, the developer fine-tunes the database according to the speed, footprint and other requirements of the target system. *eXtreme*DB puts the developer in charge.

McObject LLC
33309 1st Way South
Suite A-208
Federal Way, WA 98003

Phone:+1 425 888 8505
info@mcobject.com
www.mcobject.com

Highly efficient indexing

For transient classes, rather than storing duplicate data, *eXtreme*DB's diverse indexes contain only a reference to data, minimizing memory requirements. Supported indexes include:

- Hash indexes for exact match searches
- Tree indexes for pattern match, range retrieval and sorting
- R-tree indexes for geospatial searches
- KD-tree for spatial and Query-By-Example (QBE)
- Patricia trie indexes for network, telecom
- Object-identifier references, for direct access
- Custom indexes

Additional Features

*eXtreme*DB's many extras help developers and application end-users get the most from the database.

- **HTML database browser/editor**. Retrieve database and class statistics and lists of classes; generate schema in the form of a Data Definition Language (DDL) file
- **XML Extensions**. Generates interfaces to create or update an object in the database from the content of an XML document, export an object as an XML document, and to generate an XML schema
- **Remote procedure call mechanism (MCORPC)**. Framework enables remote processes to read/update an *eXtreme*DB in-memory or persistent database
- **Database calculator**. Collect information needed to choose ideal page size and to optimize schema designs, storage layout and performance
- **Pattern search**. Use wildcards to search tree index entries for single and multiple character matches

Supported Platforms

Embedded Platforms:

- Linux (Various distributions)
- Windows 2000 and later including RT
- Apple iOS and MacOS X
- QNX 6.x
- SUN Solaris
- HP Inc HP-UX
- IBM AIX
- LynxOS
- eCos
- WindRiver Linux and VxWorks
- Green HillsSoftware INTEGRITY
- GNU CygWin and MinGW32
- And more
- Bare bones board (no operating system required)

Development environments

- GNU toolchain (gcc 2.96 and higher)
- Tornado 2.0 and 2.2 (GNU and Diab compilers)
- QNX Momentics IDE (C, C++, Embedded C++)
- Elipse
- XCode
- GreenHills Multi
- Microsoft Visual Studio (C/C++, .NET)

Server and Desktop Platforms:

- Solaris
- HP-UX
- Linux distributions
- Windows
- MacOS X
- QNX 6x
- AIX

Database Specifications

Maximum objects per database	
32-bit:	2^32
64-bit:	2^64
Maximum classes per database:	65,535
Maximum indexes per database:	65,535
Maximum fields per class:	65,535
Maximum fields per index:	65,535
Maximum elements per vector:	65,535
Code Size:	As little as 200K
Maximum database connections:	configurable
Maximum open databases:	configurable

Supported Data types

- 1, 2, 4, 8-byte signed/unsigned integers
- float, double
- date, time
- char (fixed length)
- string (variable length)
- rect(angle)
- Unicode
- boolean (array of bits)
- enum
- fixed-size array
- variable-length vector
- structs (embedded to any depth)
- autoid (auto-increment)
- user-defined object-id and references

McObject LLC
33309 1st Way South
Suite A-208
Federal Way, WA 98003
Phone:+1 425 888 8505
info@mcobject.com
www.mcobject.com