*eXtreme*SQL™

Standards-based programming interface for e*Xtreme*DB, the main memory embedded database.

Product Datasheet

"e*Xtreme*SQL adds a familiar interface to technology that provides radically improved performance, compared to existing data management solutions in these markets"

-- SySoft Company

Overview

McObject's *eXtreme*SQLÊ is a high-performance implementation of the SQL database programming language for the *eXtreme*DB in-memory database. With *eXtreme*SQL, McObject targets the real-time enterprise software market by greatly simplifying programming with *eXtreme*DB for corporate developers using SQL. The new interface strengthens *eXtreme*DBôs appeal for application development in fields such as banking and securities trading, where real-time responsiveness is a must and SQL is the dominant database language.

Built on the unsurpassed performance of *eXtreme*DB, and a RAM-based SQL optimizer, *eXtreme*SQL delivers blazingly fast processing of dynamic SQL queries.

Benefits of using *eXtreme*SQL include…

Co-exists with native *eXtreme*DB. Use *eXtreme*SQL alongside the *eXtreme*DB API in the same application. Where maximum performance is critical, the *eXtreme*DB API canô t be beat. When a higher level of access would be beneficial, such as retrieving data from multiple tables or performing aggregation, then *eXtreme*SQL is advantageous.

Broad coverage of the SQL-89 standard. *eXtreme*SQL implements much of the ANSI SQL-89 specification.

Extensions to exploit *eXtreme*DB features and data types. In addition, *eXtreme*SQL implements *eXtreme*DB-specific extensions including support for structures, arrays and vectors, as well as query optimizations based on specific *eXtreme*DB capabilities.

Compatible with all *eXtreme*DB editions. *eXtreme*SQL is fully compatible and interoperable with *eXtreme*DB Standard Edition, High Availability Edition, and Transaction Logging Edition. There is no need to worry about the migration path within the *eXtreme*DB product family

No client/server inter-process communications. Like *eXtreme*DB, *eXtreme*SQL is embedded in the application, not deployed as a separate process. This eliminates client/server inter-process communication round-trips from the execution path, resulting in breakthrough performance.

Interactive SQL utility. *eXtreme*SQL comes with an interactive SQL program, *eXQL*, that can be used to test SQL statements independently from application programs. Full source code for the utility is provided so that it also serves as a useful example of a full *eXtreme*SQL implementation. In addition to executing SQL statements, *eXQL* also supports *eXtreme*DB-specific commands, such as "report" and "save <file>".

eXQL can also be employed as a batch processing utility by redirecting input from a text file containing *eXtreme*SQL statements.

*eXtreme*SQL Query Optimization

Creating the optimal plan for execution of SQL statements is a very complex and challenging task. SQL optimizers analyze SQL queries sent to the database and select the best search strategies for accessing the database. Query optimization greatly depends on data distribution. Often, optimizers take samples and use statistics provided by the database engine, and collect statistical information themselves, to calculate the cost of candidate execution plans. Therefore, building optimal plans requires is a very CPU intensive operation and inherently unpredictable; the amount of time spent in the optimizer varies from query to query and execution plans can change from one invocation to another as the distribution of data changes.

For real-time and embedded systems where predictable performance is key, a rule-based optimizer such as is used by *eXtreme*SQL is more appropriate. In addition, the *eXtreme*SQL optimizer makes it possible for applications to specify their own execution plan. For example, the optimizer never reorders tables in the query: the joins are performed in the sequence the tables were specified in query Some of the other key rules that are used for query optimization include:

- If possible, an index is used
- Each table is assigned an index representing its position in the FROM list
- The search predicate is divided into the set of conjuncts and the conjuncts are sorted. Therefore the expressions accessing the tables with smaller indices are checked first.
- The execution of subqueries is optimized by checking the dependencies of the subquery expression. The results of the subquery are saved and recalculated only if the subquery expression refers to fields from the enclosing scope.

McObject LLC
22525 SE 64th Place
Suite 302
Issaquah, WA 98027
Phone:+1 425 888 8505
Fax: +1 425 888 8508
http://www.mcobject.com