c-treeACE: The Marriage of SQL and NoSQL

by Robin Bloor
Chief Analyst

The Genesis of the Multimodel Database

The possibility of multimodel databases was never contemplated until fairly recently. Such an idea – a database that could manage all data structures rather than focusing solely on a particular subset of data structures – was, for decades, believed to be impractical. Over the last 25 years, relational databases dominated, despite the fact that they could only accommodate data that was shoehorned into tables. They proved capable of handling most OLTP applications and most data warehouse applications, and where their rigid adherence to tabular data structures led to problems, those issues were circumvented by programming techniques: stored procedures, object-relational mappings and so on.

While relational databases were the dominant database products, they were not the only ones. Some niche database products offer far superior support for recursive data structures, nested tables or graph data. They found use in situations where the performance requirements for managing such data was paramount. Database species have been increasing at a steady pace in recent years. A list of the "newly evolved" includes: XML DBMSs, column stores, document DBMSs, time series DBMSs, graph DBMSs, multivalue DBMSs, RDF DBMSs, event stores and key-value stores.

Common sense dictates that no business will want to house such a diverse variety of product types. Consequently, a "consolidation trend" is emerging, with some database products adopting a multimodel approach – a database that physically stores a wide variety of data and supports multiple data models. One such database is FairCom's c-treeACE.

What Is a Multimodel Database?

Let's first state what we would expect from any transactional database. It should have ACID (atomicity, consistency, isolation, durability) properties to guarantee the reliable processing of transactions and hence support multiple concurrent users. It should store metadata and support one or more query languages that use the metadata. It should provide secure access, enable the backup, import/export and the replication of data, and it should support transaction logging. It should optimize performance. Most databases can reasonably claim to do these things.

What makes a multimodel database different is that it supports a wide variety of data structures rather than just the tables (of relational databases) or the document structures (of document databases). In order to do this, aside from other subtleties, it needs to accommodate multiple, different data models; hence the name multimodel.

At a physical level, all databases use key-value stores as a storage paradigm. Data objects are organized in one way or another and stored on disk or SSD. Then an indexing schema is applied to each collection of data so the data can be retrieved by key. In order to support multiple data models, it is necessary for the physical storage to be more versatile than is required for a traditional relational database, as is the case with c-treeACE. In addition to that, there needs to be metadata support for different types of data models, and the database must be able to retrieve data in accordance with them.

The main IT and business benefit is that you can store much more of your data using just one database product, eliminating the need for multiple database products and the associated skills. If used intelligently, a multimodel database will also reduce the number of ETL programs that need to be written simply because there will be no need to

c-treeACE: The Marriage of SQL and NoSQL

transfer data from one type of database to another.

c-treeACE

The adjacent illustration gives an overview of FairCom's c-treeACE. The database supports SQL access to data as well as a variety of NoSQL access capabilities. This versatility is achieved by the core ACID compliant engine of the database. The database uses a key-value store and can store data at the physical level in a variety of forms. It has a flexible indexing capability that enables it to place multiple distinct indexes on the same data.

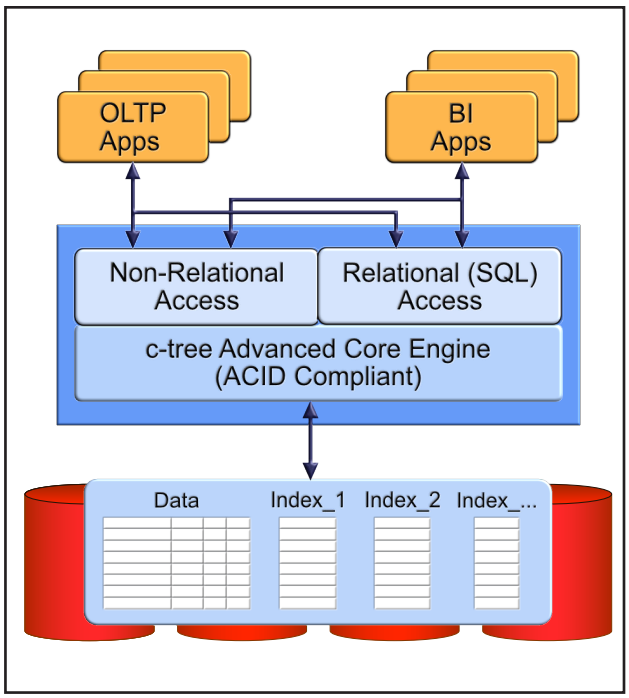

Figure 1. c-treeACE Architecture

Data may be requested using any of the supported schemas, either through the non-relational interface or the SQL interface. Its SQL support is impressive and extensive. It includes SQL User Groups, Table Locks, Table Valued Functions, Recursive Queries, Scrollable ODBC Cursors, ALTER VIEW/INDEX/TABLE and SQL-Only Indexes. When a request for data is made, the database physically retrieves the data and, if required, presents it to the requesting application in a virtual image that satisfies the request. For example, a collection of variable length data held in an "unstructured" NoSQL format might be requested by a SQL statement. The database would gather the information, format it as a relational table and then present it in that form to the application.

In recent releases, c-treeACE has been boosting its functionality. It can currently handle tables/files with multiple record types, physical files as "filesets" and arrays within tables (i.e., subrecords within tables). It has also added (in version 11) a text searching capability that is currently limited to searching character and string data types at the file/record level, but it will be expanded to work as a global database capability.

Performance

FairCom's c-treeACE targets performance, delivering in-memory capabilities, fine cache tuning and deferred indexing. Throughput rates of hundreds of thousands of records per second can be achieved.

The in-memory feature allows for files to remain in memory while open. The cache controls are highly granular, allowing users to specify which files to cache, the proportion of a file to cache and which files to cache on startup. Deferred indexing can be specified on secondary indexes so that index maintenance overhead is carried out in the background. This can enhance performance significantly both for OLTP workloads and for data ingest.

Data compression is supported with developers able to select industry standard compression algorithms. The database's partition rule support enables data to be partitioned into multiple separate files for easy maintenance, while still appearing to the application as a single logical file. A versatile replication capability is also available to

"FairCom's c-treeACE targets performance, delivering in-memory capabilities, fine cache tuning and deferred indexing. Throughput rates of hundreds of thousands of records per second can be achieved."

aide with horizontal scaling.

"From a relational perspective c-treeACE looks very much like a fully functional relational database. The kicker is that the capabilities are multimodel rather than just relational."

Database Capabilities

The c-treeACE multimodel capabilities are built in at the physical level. The database is ACID compliant, irrespective of the mode of data access and update or the specific API that is used. Regarding APIs, it has SQL access via JDBC, ODBC, embedded SQL, interactive SQL, ADO.Net, DBX, PHP and Python. For NoSQL access, it supports Java, C++, VCL and .NET. Integrity related features include time limited data flushing, index cache pages and incremental roll forward/automatic recovery with restore points.

Stored procedures and triggers are supported in Java or any .NET supported languages. From a relational perspective, c-treeACE looks very much like a fully functional relational database. The kicker is that the capabilities are multimodel rather than just relational. From the developer's perspective, what you see is flexibility.

A full set of backup and recovery options are provided, including extensive support for data replication. Replication is implemented via an elegant graphical web-based tool. It is a drag-and-drop interface that allows instances to be connected and the replication activity specified. Peer-to-peer replication and server-to-server file copying are both supported as well.

As such, this database can be used to distribute workloads to data marts for back-end reporting or to separate data warehouse from OLTP workloads. It can be used to configure a real-time hot stand-by database instance. It can be used for database mirroring or other geographical distribution strategies. Even more complex replication topologies are supported as the database provides publish-subcribe capabilities, which are implemented by a general purpose message queue.

The Bottom Line

While you may not be familiar with the name c-treeACE or the database, FairCom is nevertheless a well-established database company. It has a significant footprint both in the ISV market and in the corporate market, counting over 40% of Fortune 100 clients within its customer base. The company is global, with customers in over 100 countries.

Its fundamental technology is mature and proven. FairCom's innovation is in its adoption of a multimodel approach to database, which currently embraces both the relational model and a variety of NoSQL capabilities, employing virtual tables to deliver SQL access to unstructured data. Companies who are currently examining their database options and wish to move to a more versatile database platform should take a look.

Company: FairCom
Location: Columbia, MO

Product: c-treeACE
Multimodel database