solidDB In-Memory Database

Executive summary

The solidDB® relational in-memory database platform delivers the extreme speed, availability and adaptability that organizations need for mission-critical environments while helping to control operating costs.

solidDB is a fully featured, relational, in-memory database that delivers extreme speed and availability to meet the performance and reliability demands of real-time applications.

By keeping critical data in memory, rather than on disk, solidDB can perform significantly faster than conventional databases. It helps applications achieve throughput of hundreds of thousands to millions of transactions per second with response times measured in microseconds.

Beyond game-changing performance, solidDB also provides built-in data availability features that help sustain uptime, prevent data loss and accelerate recovery. Additionally, solidDB supports administrators with the flexibility to tailor the software to precise application needs and features designed to simplify deployment and administration, helping drive down the total cost of ownership (TCO).

Given the advantage os solidDB, it’s no surprise that market leaders such as Cisco, HP, Alcatel, Nokia and Siemens rely on it for their mission-critical applications. Across the globe, there millions of deployments of solidDB in telecommunications networks, enterprise applications and embedded software and systems.

Extreme speed

The solidDB in-memory database architecture helps organizations meet the performance demands of real-time applications that require extreme speed and predictable response times. HotStandby configuration options enable administrators to maximize performance by effectively balancing loads.

Achieving extreme speed with in-memory database technology

The solidDB in-memory database was designed to ensure that all data is rapidly accessible. As the name implies, it keeps data entirely in main memory rather than on disk, making data access extremely fast. solidDB also uses data structures and access methods specifically created for storing, searching and processing data in main memory, with efficient concurrency control mechanisms.

The design of solidDB provides important performance advantages over conventional, disk-based databases. First and foremost, solidDB eliminates the need to transfer data blocks from disk to main memory — any data requested by the application is already in main memory. Even when disk-based databases cache all data in main memory, solidDB is still faster because it uses data structures and access methods that are optimized for retrieving data from main memory.

Highlights: solidDB 7.0

Extreme speed

solidDB delivers extreme speed by keeping data in main memory at all times rather than on disk. Applications can take advantage of its capability through standard ODBC, JDBC and SQL interfaces.

Extreme availability

Maintaining two copies of data synchronized at all times helps solidDB provide extreme availability. In case of system failure, applications can recover access to solidDB in less than a second without loss of data.

Low cost

By simplifying deployment and administration, solidDB helps control operational costs. solidDB streamlines application migration from other enterprise data servers. In addition, it can be embedded directly into applications and run virtually unattended to deliver low TCO.

Adaptability

solidDB is easily adaptable to diverse applications and deployment needs. It scales horizontally with two-node shared memory access to increase performance and reduce response times.

Organizations can further enhance performance and reduce response times by linking multiple applications to a dynamic library comprised of the full solidDB server code. Linking applications directly with the server code means that the application-level requests can be executed within the same address space. Applications can access the memory-resident data without the overhead incurred by multiple context switches on each interaction between the application and the server. With the solidDB shared memory access (SMA), you can bring data closer to the application to use the fastest and most efficient data access paradigms—yielding more results faster.

solidDB is also designed to take full advantage of the large memory capacities offered by 64-bit computers, as well as the processing scalability provided by multicore, multiprocessor architectures. solidDB access methods implement efficient concurrency control mechanisms to help ensure transactional integrity across large numbers of concurrent operations.

For data that does not need the extreme speed of an in-memory database, solidDB also offers disk-based tables, and manages them with performance typical of conventional disk-based databases. Applications can access both in-memory and disk-based tables transparently, and use them in the same transactions.

Enhancing performance with HotStandby configurations

Organizations can boost performance in a number of ways by using a HotStandby configuration. Read operations are load balanced across primary and HotStandby solidDB instances transparently to the application. To take advantage of load balancing, an application need only use solidDB ODBC or JDBC drivers and maintain a single logical connection to both the primary and HotStandby solidDB instances. Write transactions are automatically directed to the primary solidDB node, and read transactions are either directed to the HotStandby solidDB node alone or load balanced across primary and HotStandby solidDB instances. Because it takes advantage of the multithreaded computation power of both primary and standby database instances, load balancing can improve performance by up to 100 percent.

HotStandby can also increase performance by optimizing transaction logging. To balance performance and durability, solidDB transaction logging offers multiple configuration options. With strict logging, transactions are logged as soon as they are committed, synchronously. With relaxed logging, solidDB defers writing to the transaction log, asynchronously, increasing performance without sacrificing data safety. When a failure occurs, the remaining solidDB instance automatically shifts to synchronous logging to maintain data durability. In addition, a snapshot-consistent checkpoint function enables administrators to turn off transaction logging to help accelerate application performance in situations where it is sufficient to provide recoverability only to the last checkpoint.

As of 7.0, HotStandby configurations scale to multiprocessor environments, which helps improve performance. The standby (secondary) server can process write operations using multiple threads, which minimizes or eliminates the need for the primary server to wait for operations to be carried over to the secondary server.

Extreme availability

solidDB is designed to provide the extreme data availability required by business-critical applications with data persistence and recoverability capabilities that help to prevent data loss and accelerate data recovery. By providing both synchronous and asynchronous HotStandby configurations, solidDB also helps organizations find the right balance between data safety, application throughput and recovery time.

Maintaining data persistence and helping to ensure recoverability

While keeping all data accessible in main memory, all the time, solidDB also writes updated data to disk to help ensure data recoverability. The solidDB checkpoint function copies committed transactions from main memory to database files on disk. If the server fails between checkpoints, solidDB helps to make sure that the disk has a consistent snapshot of the data by writing committed transactions to a transaction log between checkpoints. After a system crash, solidDB automatically recovers transactions committed since the last checkpoint to the most recent committed state using this transaction log and roll-forward recovery.

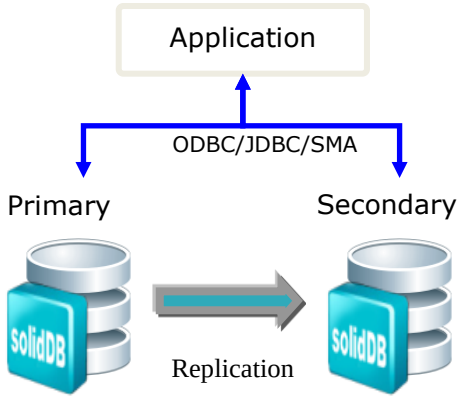

Figure 1: A two-node HotStandby configuration of solidDB provides data security and workload balancing capabilities transparently to applications through standard drivers.

Achieving rapid failover and preventing planned downtime

Using a two-node HotStandby configuration, organizations can maintain two copies of data, synchronized at all times between the solidDB nodes. In case of system failure, failover between solidDB nodes occurs in less than a second (see Figure 1).

Administrators can also use the solidDB HotStandby node to conduct maintenance tasks—such as rolling upgrades, reporting and backups—without taking the primary solidDB node offline. Just as with a system failure, a planned switchover between the pair of solidDB nodes can be completed in less than a second, with no data loss. The switchover is transparent to the application—solidDB ODBC and JDBC drivers maintain the database connection and session attributes.

solidDB 7.0 introduces transparent connections to SMA configurations. Full read/write loads can now be run against both the HotStandby primary and secondary servers of the two-node SMA setup, facilitating active-active configurations that help sustain availability.

Balancing data safety, application throughput and recovery time

solidDB offers several high-availability configuration options that specify how primary and secondary database servers are synchronized. At runtime, applications can query the configurations and modify them for individual sessions and transactions. These capabilities enable organizations to balance data safety, application throughput and recovery time with unprecedented flexibility.

Synchronous HotStandby configuration

In the synchronous replication configuration, transactions are committed at the primary database only after they are acknowledged by the secondary one.

Three different settings for synchronous replication are available:

- The secondary database acknowledges the commit record as soon as it receives the request from the primary database.
- The secondary database acknowledges the commit record after processing the transaction request it receives from the primary database, but before writing to the log. • The secondary database acknowledges the commit record after the transaction log is fully committed at the secondary database.

These settings can have an impact on the overall recovery time and the performance of the application. Recovery time is much shorter when all transactions committed at the primary database are also committed at the secondary database. In case of a system failure of the primary database, the secondary database has no recovery to perform, and can immediately take over the role of the primary database.

Asynchronous HotStandby configuration

Using an asynchronous HotStandby configuration can help organizations avoid the performance impact of synchronous operation. When using the asynchronous replication configuration, a COMMIT command is generated as soon as the transaction is committed at the primary server. Because the application does not wait for an acknowledgement from the secondary database server, this configuration incurs no extra latency at the primary database server.

solidDB 7.0: Features overview

Persistent, in-memory relational database

- SQL-92 entry-level support, with selected features of SQL-98 and SQL 2003
- Full ACID support
- Multiple table types: persistent, transient, temporary; disk-based and in-memory
- Concurrency control with multiple transaction isolation levels
- Snapshot-consistent checkpointing, transaction logging and automatic roll-forward recovery
- Advanced, cost-based query optimizer
- Multithreaded database engine that leverages multiprocessor and multicore architectures
- Tables, views, indices and sequences
- Dynamically controlled integrity constraints including primary and foreign keys
- Programmable through SQL and C-language–based stored procedures, triggers and user configurable events
- Configurable security through user and role privileges
- Full Unicode support for all character data, with external encoding separately selectable for each application
- Tools for diagnostics and performance monitoring
- Tools for bulk-loading and exporting data
- Built-in scheduler to run online backups, network backups, status reports and other administrative tasks
- Transaction log reader
- Persistent audit trail functionality

High availability and load balancing

- HotStandby configuration with configurable replication and logging settings
- Transparent connectivity that allows applications to maintain one logical connection to both primary and HotStandby instances and configurable load-balancing of read operations across primary and HotStandby solidDB instances
- High-availability controller that monitors health and state of solidDB instances and performs failovers
- Built-in, dynamically queriable and programmatically configurable high-availability state machine that simplifies integration with third-party high-availability managers

Flexible deployment models

- Small footprint and hybrid in-memory disk solution
- Ability to be deployed as a server process to enable access by applications over the network
- Ability to be linked with Java or C/C++ applications that are deployed on the same server as solidDB to boost performance and to make solidDB invisible for embedded deployments

Synchronous and asynchronous replication

- Advanced replication: Built-in, bidirectional, publish-andsubscribe–based asynchronous replication between solidDB instances with data consistency; supports replication of full databases or replication by tables or columns or rows
- solidDB High Availability (HotStandby): Push-based synchronous or asynchronous replication to propagate all data changes in the primary to the secondary

Low Cost

solidDB can help organizations drive down TCO by simplifying management, avoiding outages, offering cost-effective hardware options and enabling exceptional scalability.

Streamlining Migration

solidDB 7.0 introduces several enhancements that can help streamline the migration of applications from other enterprise data servers to run on top of solidDB. Support for user-defined stored SQL functions and external, C-language–based stored functions and procedures reduce the work required for migration. Updates to the SQL statement and keywords further simplify migration by improving support for syntax conventions used in other databases.

Simplifying management and avoiding costly outages

solidDB can be embedded directly into applications and controlled by the application. In addition, it can be run virtually unattended, helping organizations reduce ongoing administrative costs.

The high-availability capabilities of solidDB also help organizations avoid unexpected expenses. IT groups can reduce the administrative costs of planned and unplanned outages, as well as the costs of lost productivity or revenues resulting from those outages.

Cost-effective deployment and scaling

solidDB runs on commodity hardware as well as best-of-breed systems. Organizations can select from a variety of proven solutions to find the right balance between cost and other factors.

When organizations are ready to scale, the in-memory database technology of solidDB can take advantage of the large memory sizes offered by 64-bit computers as well as the processing scalability provided by multiprocessor and multicore architectures. solidDB can scale vertically, cost-effectively accommodating growing data volumes and more concurrent connections while maintaining microsecond response times and high transaction throughput.

For more information

To learn more about solidDB, please contact your Unicom sales representative, or visit: http://unicomsi.com/soliddb