FirstSQL® 100% Java Object-Relational Database Management System

A First Class J2EE ORDBMS -

FirstSQL/J Enterprise Server is a robust, ultra-high performance Object-relational SQL database system written entirely in Java and to support SQL database standards.

FirstSQL/J Enterprise Server is focused on J2EE standards and enterprise server applications that use complex Java object models and the relational database model together to address business needs. FirstSQL/J Enterprise Server provides a scalable and sophisticated relational database with advanced OLTP and SQL functionality that includes comprehensive manipulation of serialized Java objects and their methods using standard SQL.

With Advanced Features -

FirstSQL/J not only supports the execution of methods from unmodified Java classes cataloged in the database, but SQL commands can reference both instance and static methods. Static methods also serve as stored procedures.

Performance gains are substantial running with the optional Main Memory mode.In Main Memory mode, the database stores all data (tables, objects, indexes) in memory for ultra fast direct access and manipulation. Both traditional Disk access mode and Main Memory modes are provided and persistent data is always written to permanent storage ensuring complete recovery.

The Enterprise Server edition also supports J2EE Connector Architecture (JCA), Distributed Transactions (XA/JTA), On-line Backup, and Replication with Active/Standby Fault Recovery support for Continuous Availability.

On The Most Important Platforms - Anywhere.

FirstSQL/J is built on a component model and can be developed and deployed on any computing platform that supports a 1.1 JDK, or greater, J2EE or J2SE compliant JVM. FirstSQL/J provides a scalable and robust multi-threaded database engine that installs and executes the same across any 32-bit or 64-bit platform, regardless of operating system.

Enterprise Class Database - Designed for J2EE class Servers

Written in 100% Java - Available for any platform with a JDK 1.1, or greater, JVM

Scalable In-Memory and Disk access Mode - Real-time performance, optimized query engine, and full data persistence

SQL92 Intermediate Level - Supports sophisticated SQL Standard with JDBC access

Object-oriented - Internal object type-to-SQL data type mapping provides full serialized Java Object persistance with comprehensive SQL manipulation

High Speed Transactions - Full Transaction processing, XA/JTA support, and transaction reliability - 4 transaction isolation levels

Small Dynamic Footprint - Component Model allows configurable footprint

Sophisticated SQL - Unique and powerful SQL features not available in other databases

Administrative Tools - Comprehensive Utilities and Tools

Replication and Fault Tolerance - High availability, clustering, and On-Line Backup

Main Memory Operation

The FirstSQL/J engine offers several modes of operation at startup. In-memory mode provides substantial performance gains without sacrificing data persistence and reliability.

In default mode, the DBMS uses disk for storage of data during the run and maintains it on disk between runs. The other modes, **memory** and **flash**, both maintain the database in available memory during a run. The In-memory modes yield much higher performance in database access and modification.

The **memory** mode of operation uses disk for persistent storage of data changes during runs. The disk format is the same for **disk** and **memory** modes. During a run, **memory** mode reads tables into memory when first referenced and maintains them there for the remainder of the run. For persistent data and recovery the **memory** mode writes a roll forward journal of changed data to disk.

The **flash** mode of operation uses no disk access at all. It maintains the database entirely in memory. In **flash** mode all data is transient.
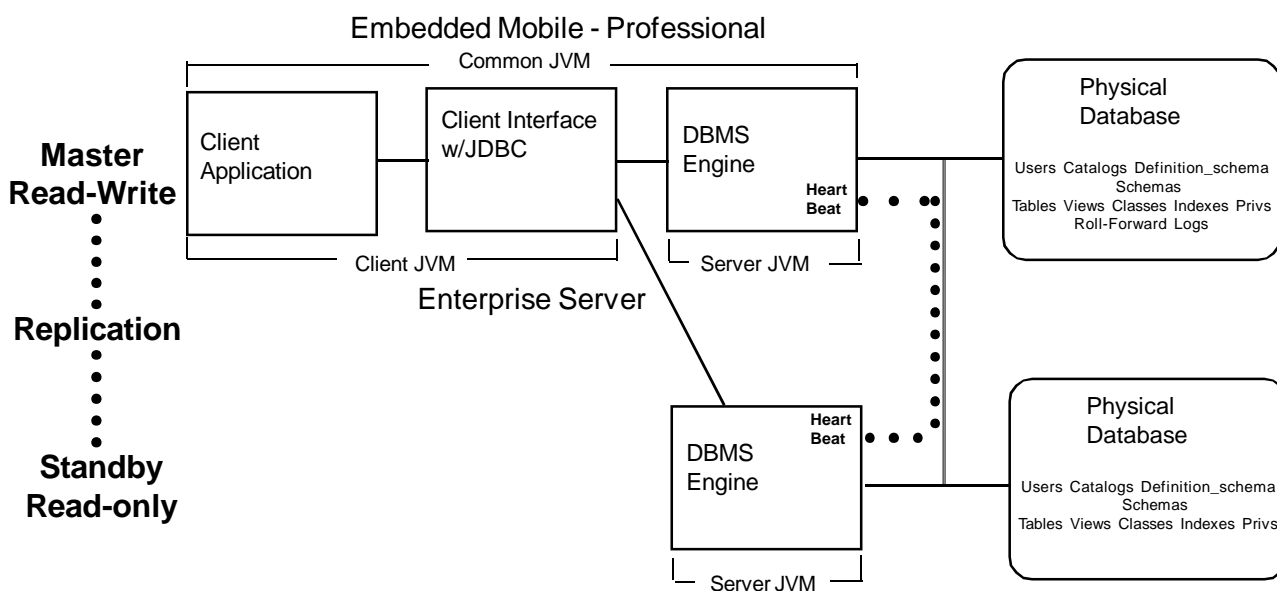
On-line Backup, Replication and Fault Recovery

FirstSQL/J provides several facilities for replicating live data during a DBMS runs. Fault tolerant configurations can be enhanced with the Arbiter API mechanism for greater failover control.

On-line Backup — live replication data by creating a snapshot of the current database and writing a roll forward journal during DBMS runs. Restore returns the database to its latest state by restoring the snapshot data and then rolling forward all changes made since the snapshot.

Replication Server — a secondary server that runs in conjunction with the primary server. The Master server sends all committed changes to the replication servers, which maintain a *mirror* of the database state of the Master server.

Standby Server — a replication server that supports live recovery (failover) from failure of the Master server. When a standby server detects the primary server has failed, it automatically switches to Master server mode allowing connected clients to continue transaction processing. The Arbiter API provides additional failover control.

Enterprise Server allows N-Way Replication Servers and a Standby Server

The FirstSQL/J query engine is optimized for applications using Prepared Statements. This optimization provides a performance advantage for a wide range of applications and platforms, from servers to embedded systems and mobile devices requiring zero administration:

Computationally intensive – Transactional, Analytic Modeling, Data Analysis, XML Manipulation, Personalization, Business Rules, Work Flow, Supply Chain Management.
Heavy user load – OLTP, CRM, Call Management, e-Business, etc.
Strict response or throughput requirements – Real-time Fault Tolerant Systems, Soft Switches, Intelligent Routers, Web Application and Mobility Servers, Process Control, Monitoring, SCADA, etc.

Tools and Utilities
Both GUI and command-line equivalents are provided:

Ijdbc and IjdbcWin — Interactive SQL tool for working with statements and results. A detailed tree view of all database elements provides additional developer productivity. Includes **Import/Export,** a utility for bulk loading data into and out of FirstSQL/J databases. IjdbcWin is also available with Embedded Mobile for smaller screen sizes.

GUI Administrator — full package of server administrative capabilities in one application, including API level server startup, shutdown, and server session control.

MaintGUI — utilities for maintaining physical databases, Maint and MaintGui utilities provide capability to:
- copy a physical database,
- backup a physical database, including on-line backup,
- restore a physical database from backup,
- build a new physical database including; image, Standby, Replica, Backup, Read-only.
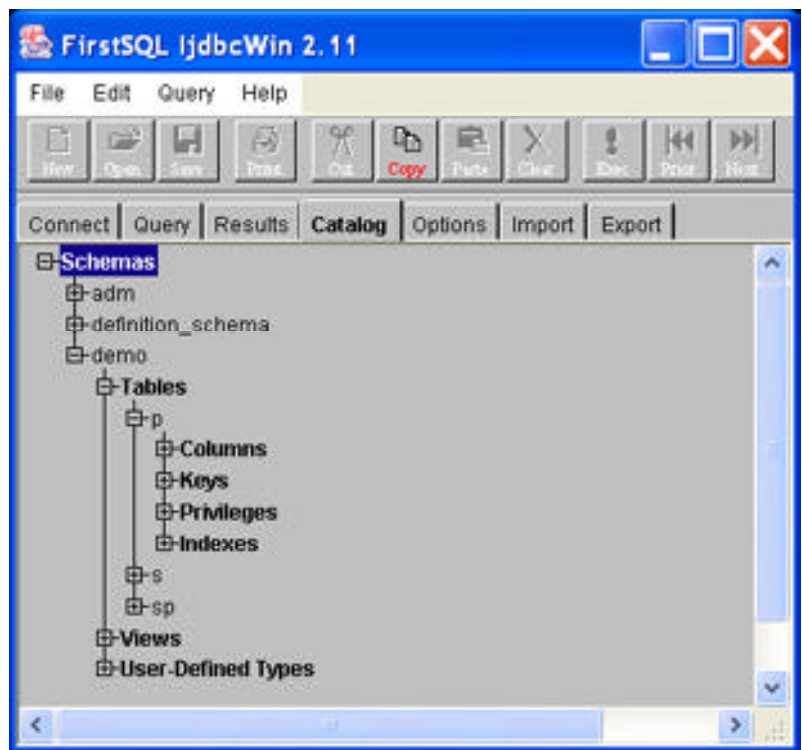
API's
> **Build** — create databases & tables from within application.
> **Arbitor** — full failover control over daisy-chained Standby Replication servers and more.
> **Administrative** — provides the following APIs:
> Admin Dialog - send commands to the server
> Admin Push - recieve server log messages
> Admin Immediate - direct communication for server startup.
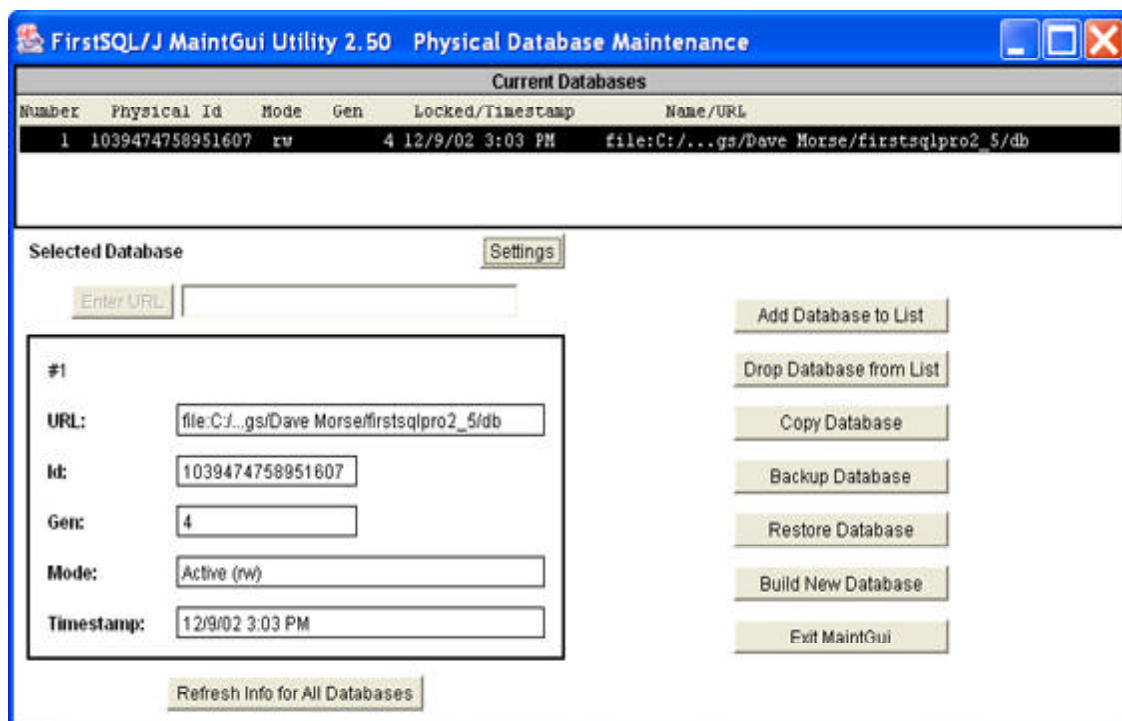> **Import/Export** — low level load/unload control

Monitor — dynamic charting and stats of 27 elements and activities.

RemoteLogger — log management tool to manage and store server logs remotely or locally with configurable file size, number of backup files, and file location. Uses Log4J.

Documentation — HTML includes an SQL Tutorial, Quick Start, User Reference, JavaDocs, and complete listing of SQL Exception codes by:
- SQL State code number for type of exception
- Text message describing exception
- FirstSQL/J specific error number

Java Objects in the Database

FirstSQL has solved the missmatch between relational database and OO Java capabilities in addition to simplifying the use of both together. The flexibility of cataloging Java Classes in the database provides many advantages:

- **Java classes are made persistent without O/R mapping tools or special treatment of Java classes**
- **Supports simplified SQL access to persistent data, objects, and methods**
- **Supports complex object relationships and unlimited flexibility of information storage**

The **CREATE CLASS** command catalogs a Java Class into the database. No special treatment is required and O/R data type mapping is performed internally. Once cataloged, the Java Class has multiple uses, for:

User Defined Functions – the static methods of the class can be called as functions and stored procedures in SQL.

Java Stored Procedures – the static methods of the class serve as stored procedures. The class methods have direct access to the current database connection and can return multiple and scrollable resultsets.

Data Wrappers – instances of the class can be used by the client to pass data to and from the ORDBMS.

Column Data Types – table columns can use the class as their data type. Values for the columns in table rows are then instances of the class. SQL statements construct instance values for the column with the NEW operator and utilize the object column values by calling their methods.

Java capabilities enhance the power of SQL. Java methods can be called anywhere in SQL commands, for example, SELECT List, WHERE Clause, SET Clause.

Java methods use JDBC classes for database access. Database clients using JDBC can retrieve and send database objects and they can then execute the retrieved objects in the client JVM environment.

In defining object columns, the class-name replaces the data type:

```
CREATE CLASS Money FROM 'Money' ;
CREATE TABLE sales_orders
(ord_id int,
 cust_id int REFERENCES customers,
 ord_date date,
 ord_amt Money,
 PRIMARY KEY(ord_id, cust_id)
);
```

Execution Sandbox Flexibility

The internal database methods execute in a special sandbox that restricts access to portions of the standard API. The list of built-in classes that are accessible from database methods can be expanded. This powerful capability enhances the flexibility and range of FirstSQL/J applications. Examples from expanding internal methods can include new classes for communication with external devices, databases, email, and XML data storage.

Feature Support Overview

Primary Keys (Entity Integrity)
Foreign Keys (Referential Integrity)
- Cascaded Update and Delete
- Self-referencing and Cross-referencing Foreign Keys
Transactions
- Standard 4 transaction isolation modes
- Distributed Transactions (XA)
- Full Transaction Support (Commit, Rollback, Recovery)
- Row Level Locking
- AutoCommit on/off
Privileges – table and column access
SQL Support – SQL92 Intermediate Level
- Complete Subqueries
- Outer Joins
- GROUP BY, HAVING, LIKE, Set Functions (SUM, AVG, MAX, …),
- UNION
- Views (with GROUP BY, UNION), WITH CHECK OPTION
- Stored Procedures can return multiple resultsets
- UDFs (User Defined Functions)
- CONNECTBY - Recursive queries on hierarchical data
- Max.rows - Maximum query rows
Data Types
- String – CHAR, VARCHAR (CHAR and VARCHAR can be used for binary data with no loss of information and no overhead.)
- Numeric – TINYINT, SMALLINT, INT, BIGINT, DECIMAL,
- FLOAT, DOUBLE
- Date-Time – DATE, TIME W/TIMEZONE, TIMESTAMP w/TIMEZONE, YEAR-MONTH Interval, DAY-TIME Interval
- Complex Java Objects – based on unmodified Java Classes and Methods - Java types internally mapped to SQL data types
- BLOB, CLOB (Unicode)
Functions, plus full set of Math Functions
CASE, CAST, and COALESCE Operators
Advanced *NULL* processing - two types of Nulls
Quoted Names – Special characters in table, column, user, schema, class, … names using "

JDBC 2.0/3.0
- Supports SQL DML, DCL and DDL
- Supports J2EE Connector Architecture (JCA) v1.0, 1.5
- XA Processing - Distributed Transactions
- Connection Pooling
- Scrollable Result Sets
- Full Escape Processing — {d '2002-02-28'}
- JNDI (Java Naming and Directory Services) support for using important APIs including LDAP, RMI, EJB, JMS, and CORBA.

Operating on Database Objects in SQL allows objects to be instantiated from Java classes. These operations are:
- assign them to database columns,
- call their methods, and
- pass them to database methods.
FirstSQL/J provides additional operations on database objects:
- conversion - CAST operator
- type testing - INHERITS operator