



# Java DB: The Multi-Tier Database System

**Dag H. Wanvik**

**Staff Engineer**

Database Technology Group

Sun Microsystems, Inc.



# Java™ DB

- Sun's supported distribution of Apache Derby
  - All development done in the Apache Derby community
- Complete relational database engine
- 100% Java technology
- Bundled in Sun Java Development Kit (JDK™) 6 and Project GlassFish™
- Supported by NetBeans™ software, Sun Java Studio Enterprise software, Eclipse
- **The multi-tier database** for Java applications

# Java™ DB Characteristics

- Complete Multi-User relational database engine
- Embeddable and client/server database
- Easy to use, zero maintenance
- Small footprint (2MB)
- Standards-based [Java DataBase Connectivity (JDBC™) 4.0, JSR169/CDC 1.1, SQL92/99/2003/XML, DRDA]
- Compact, secure, mature and robust
- 100% Java technology (write once, run anywhere)

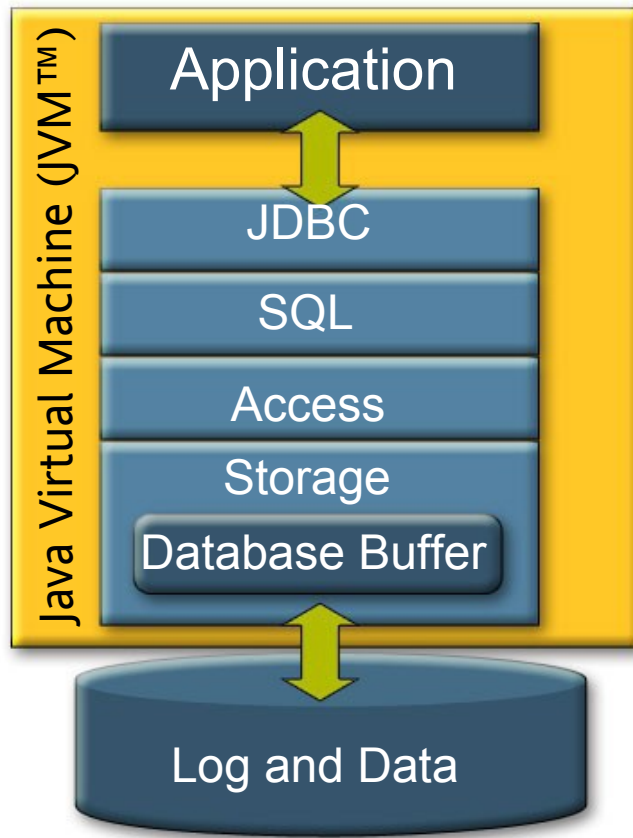
# Complete Relational Engine

- Multi-user, row-level locking, transactions, isolation levels, deadlock detection, crash recovery
- Fully ACID compliant
- Complete SQL Engine including:
  - views, triggers, stored procedures, functions
  - Foreign keys, check constraints, cost based optimizer
- Data caching, statement caching, write ahead logging, group commit
- Online backup/restore
- Database encryption, client authentication, GRANT/REVOKE

# Embeddable Multi-User Database

- Database engine may run in application's virtual machine
  - No additional process
  - Database requests are method calls within the Java Virtual Machine (JVM™)
- Startup and shutdown controlled by application
- Just one Java Archive (JAR) file
- Invisible to the user
- Easy to use, zero maintenance
- Can run as an embeddable database server

# Embeddable Multi-User Database



- Include **derby.jar** in your classpath
- Boot the Java DB engine<sup>1)</sup>:  

```
Class.forName (
    "org.apache.derby.jdbc.
    EmbeddedDriver");
```
- Create a new database  

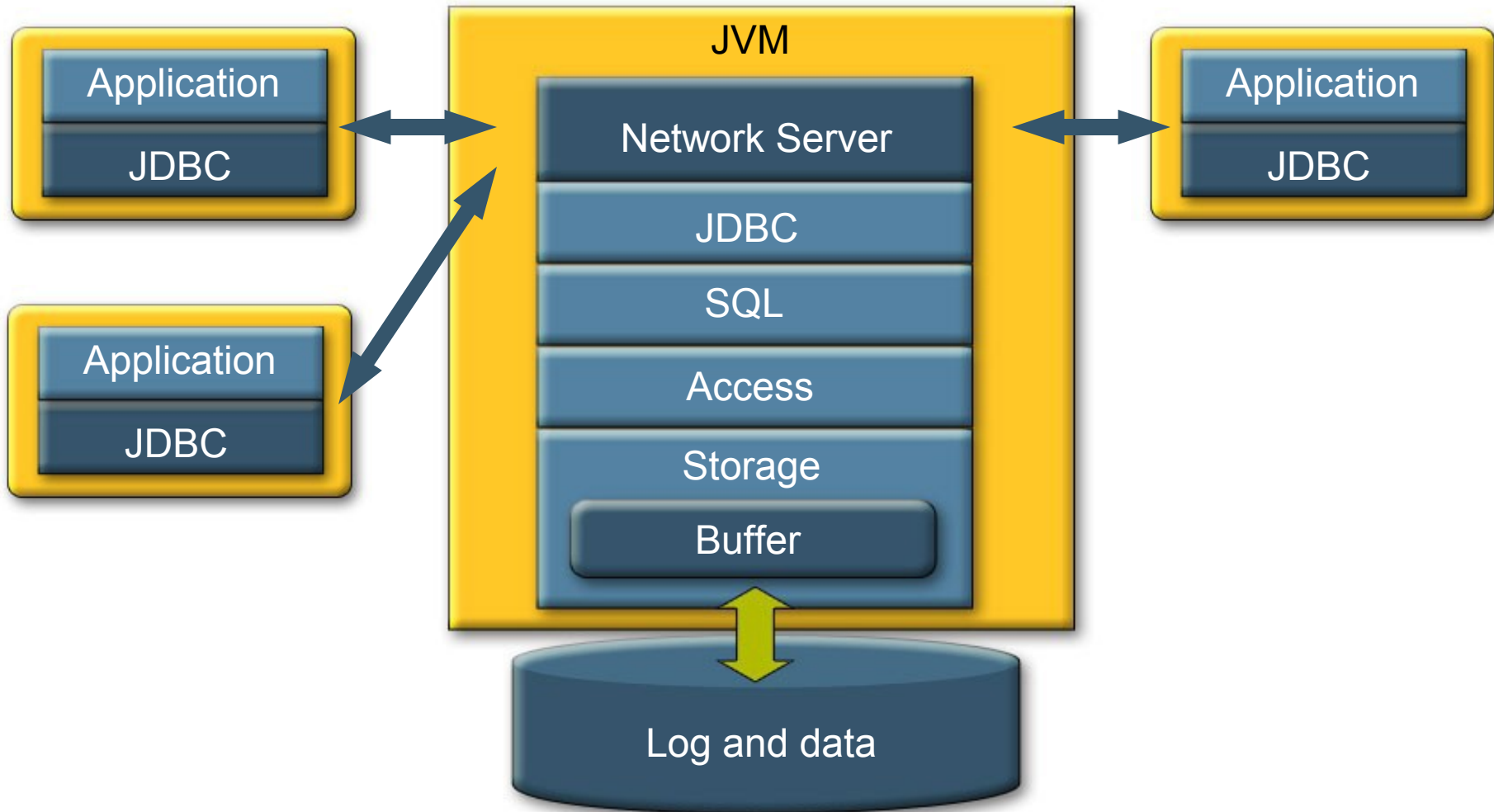
```
Connection conn =
    DriverManager.getConnection (
    "jdbc:derby:dbName; " +
    "create=true");
```

1) Optional when running with JDK version 6

# Standalone Database Server

- Database engine can run in a client-server configuration
  - Standalone server
  - Server runtime management tool
- Secure and support for various network connection mechanisms
- Easy to use, ~zero maintenance
- Can also run embedded in other server frameworks

# Standalone Database Server



# Java Platform Stored Procedures & Functions

```
CREATE FUNCTION SEND_MAIL(  
    TO_ADDRESS VARCHAR(320),  
    SUBJECT VARCHAR(320),  
    BODY VARCHAR(32000)) RETURNS INT  
LANGUAGE JAVA PARAMETER STYLE JAVA NO SQL  
EXTERNAL NAME 'testing.MailTest.endSMTP_F';
```

# Java Platform Stored Procedures & Functions

```
CREATE FUNCTION SEND_MAIL(  
    TO_ADDRESS VARCHAR(320),  
    SUBJECT VARCHAR(320),  
    BODY VARCHAR(32000)) RETURNS INT  
LANGUAGE JAVA PARAMETER STYLE JAVA NO SQL  
EXTERNAL NAME 'testing.MailTest.endSMTP_F';
```

*-- Send a Welcome e-mail when new customers are added.*

```
CREATE TRIGGER WELCOME_CUSTOMER  
    AFTER INSERT ON CUSTOMER REFERENCING new_table AS newtab  
    FOR EACH STATEMENT MODE DB2SQL  
    SELECT SEND_MAIL(C.EMAIL, 'Welcome to AcmeWidgets',M.EMAIL_TEXT)  
    FROM newtab C, MAILINGS M  
    WHERE C.TYPE = M.CUST_TYPE AND M.OFFER_TYPE = 'welcome';
```

# Java Platform SQL Function

```
public static int sendSMTP_F
    (String toAddress, String subject, String content)
{
    recipient = new InternetAddress(toAddress);
    ...

    msg = new MimeMessage(session);
    msg.setFrom(from);
    msg.setSubject(subject);
    msg.setText(content);
    msg.addRecipient(Message.RecipientType.TO, recipient);
    javax.mail.Transport.send(msg);
    return 0;
}
```

See <http://wiki.apache.org/db-derby/SendEmailRoutine>

# Java DB Performance

- Java DB performs well
  - Comparable to other major Open Source RDBMS
  - Can support a lot of users
- Java is not slow!
  - Just In Time (JIT) compilation of interpreted byte code into native machine code
- Multi-Threaded, Multi-User, Row-level locking
- Embedded mode removes network overhead

# Performance Hints (1)

- Use (and reuse) prepared statements
  - Not: “SELECT a FROM t WHERE b=4”
  - But: “SELECT a FROM t WHERE b=?”
- Put DB log and data on separate disks
  - Specify the `logDevice` attribute when creating DB
- Tune page cache size (default 4 MB)
  - `derby.storage.pageCacheSize`
-

## Performance Hints (2)

- Use indexes to avoid table scans
  - Check query plans  
(`derby.language.logQueryPlan=true`)
- Use `RunTimeStatistics`:  
`syscs_util.syscs_set_runtimestatistics(1)`  
`syscs_util.syscs_get_runtimestatistics()`
- Use embedded if possible
- Use lower isolation level if applicable
- Java 6 is faster

See also JavaOne<sup>SM</sup> 2007: TS-45170

# Java DB: The Multi-Tier DB

- **Client tier**
  - Embeddable in rich client applications
  - Local client store for web 2.0 applications (demo)
  - Read-only DB in JAR file
  - Java DB on a memory stick
- **Middle tier**
  - Embeddable middle tier database store (in GlassFish)
  - Can act as front-end cache for back-end enterprise DB
- **Back-end tier**
  - Standalone departmental client/server database

# Java DB in the Client Tier

- Embeddable into rich Java client applications
- Local client store for web 2.0 applications
  - Web Client Store Service (demo)
- Read-only DB in JAR file
- Java DB on a memory stick

# Java DB in the Middle Tier

- Embeddable middle tier database store (in GlassFish)
- Can act as front-end cache for back-end enterprise DB
  - Sequoia for load balancing, replication & fail-over
  - Daffodil Replicator for synchronization
- The only multi-user embeddable RDBMS
  - No network overhead
  - Data locality advantage
- Can act as a persistent cache for middleware

# Java DB in the Back-End Tier

- Standalone departmental client/server database
  - Runs in its own JVM
- Can run in embedded server mode
  - As part of some Java framework
- Network server management tools & API
  - NetworkServerControl scripts & API
- Supports for SSL & XA
- Various network authentication mechanisms
- DRDA Level 7 compliant

# Data Synchronization

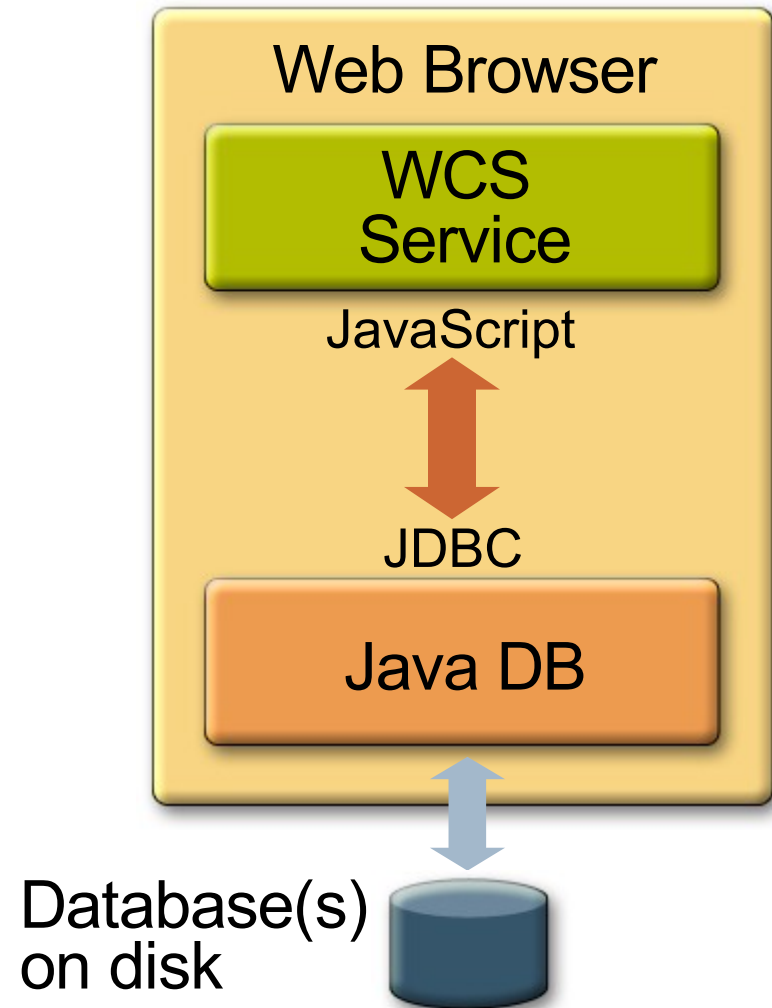
- Not always required
  - Depends on the application
- Conflicts resolution is the biggest problem
- At the application level
  - Zimbra desktop
  - Offline Derby Google Calendar (demo part of Derby)
- Database level
  - Daffodil Replicator w/ Java DB  
<http://sourceforge.net/projects/daffodilreplica/>

# Data Replication

- Not always required
  - Depends on the application
- Several open source initiatives
  - Fault-tolerant, fail-over & load-balancing
  - Support Java DB (Apache Derby)
  - **Sequoia** (formerly C-JDBC)
    - Apache 2.0 license
  - **HA-JDBC**
    - LGPL

# DEMO: Web Client Store Service

- WCS service access via JavaScript technology
- Embedded Java DB
- ACID compliant
- Fast
- Zero administration



# Local Client Service via JavaScript Technology

- Interact with local service directly via JavaScript technology
  - No new syntax or else—All JavaScript technology
- Local service installed as a Java Plug-in software extension
  - Trusted, runs in Java platform Sandbox
  - Automatically installed on client host
  - Service versioning management handling
- [http://java.sun.com/j2se/1.4.2/docs/guide/plugin/developer\\_guide/extensions.html](http://java.sun.com/j2se/1.4.2/docs/guide/plugin/developer_guide/extensions.html)
- LiveConnect to interact transparently with core service implementation in Java technology
  - JavaScript technology to Java technology and vice-versa
  - Browser agnostic

# Demo—Things to Remember

- Ease of deployment over a large user base (e.g. consumer desktops)
- Transparent—embeddable and zero—administration
  - invisible to the end user
- ACID RDBMS—high levels of durability and consistency to prevent data loss
- Ease of upgrade (using Firefox or Java Web Start software)
- Small footprint
- Highly secure to ensure desktop data is safe

# Demo & Use Cases: More Information

- Demo code publicly available at <http://developers.sun.com/javadb/>
- For more information see <http://blogs.sun.com/roller/page/FrancoisOrsini>
- Working with Apache Derby <http://wiki.apache.org/db-derby/WorkingWithDerby>

# Apache Derby Community

- Apache Software License v2.0
- Anyone can contribute
- Active contributors become committers through community vote
- Apache Derby community is
  - growing at fast pace
  - a very active one
  - a great place to learn more about database internals

# Next Release Features (10.4)

- Additional security improvements
- SQL roles
- SQL OLAP functionality
  - e.g. LIMIT()
- More performance Improvements
- Basic replication
- Table functions (VTI)
- JMX management interface
- More info at:
  - <http://wiki.apache.org/db-derby/DerbyTenFourRelease>

# Participate!

- <http://db.apache.org/derby>
  - Download, read docs
- JIRA  
<http://issues.apache.org/jira/browse/DERBY>
  - Report bugs, submit patches
- [derby-user@apache.org](mailto:derby-user@apache.org)
  - Discuss experience, get help, give feedback
- [derby-dev@apache.org](mailto:derby-dev@apache.org)
  - Discuss developer issues

# Derby Integration

- Apache ActiveMQ
- Apache JPA
- Apache Roller
- Apache Cocoon
- Apache Geronimo
- Apache JDO
- Apache Xalan
- Daffodil Replicator
- Data Direct SequeLink
- DB Visual Architect
- Eclipse
- Project Glassfish
- Hibernate
- IBM DB2 Everyplace
- IBM DB2 JDBC Universal Driver
- IBM WebSphere App Server
- ISQL-Viewer
- Java DB
- JBoss
- JPOX
- Jython
- Kodo 3.3.3
- Maven
- NetBeans Software
- Zimbra
- Red Hat Application Server
- AntHill Pro
- Sequoia (C-JDBC)
- SQuirreL SQL
- Sun Java Enterprise System
- Sun Java System Portal Server
- Sun Java Studio software
- Sun Java Platform, Enterprise Edition
- Sun Java System Service Registry
- SUSE Linux 9.3
- Zend core for IBM
- Tomcat



# Java DB: The Multi-Tier Database System

**Dag H. Wanvik**  
[dag.wanvik@sun.com](mailto:dag.wanvik@sun.com)

Database Technology Group  
Sun Microsystems, Inc.