Speedment

How Speedment™ Coexists With a DB2 RDBMS

For Accelerated SQL Database Applications

by Dan Lawesson

October 25, 2017

Why Read This Paper?

Relational databases like IBM AS/400 based DB2 are superior when it comes to handling complicated querying and analysis of data. It is also a very robust and safe environment to store the data. However large databases especially the ones used for evolving applications with increasing demands often run into scalability problems with slow applications as a result.

The purpose of this paper is to describe how Speedment coexists with a relational database such as DB2 in order to avoid a risky migration to a non-SQL solution and still achieve the extreme speed of an in-memory database.

1. DB2 Application Acceleration

Speedment allows database applications to run orders of magnitude faster without upgrading the server hardware. All legacy tools will still work and there is no need for data model migration.

Speedment leverages an in-memory data store to accelerate data access by orders of magnitude. Since the memory store works with read only snapshots of the underlying data and uses standard SQL to access the database, it coexists without any friction with existing DB2 applications both in terms of consistency and interface. While this paper focuses on DB2 databases, all points made in the following are applicable to any relational database since Speedment uses standard JDBC for database interaction.

Since Speedment generates the code used to access the database from the database metadata, the application is guaranteed to build on bug free data access code that automatically adapts when the database data model changes. This means that when an evolving database model creates a mismatch with existing application code the problem will be found even before testing. Since the Speedment part of the application will automatically remain up to date with the data model it seamlessly coexists with a legacy application being in constant development.

For more details about Speedment, please see “Speedment - the Java Stream ORM White Paper”.

2. Iterative piecewise integration - No migration

Since Speedment coexists nicely with the legacy system it is possible to adopt an iterative integration approach. This allows migration of the legacy system to be postponed or even completely avoided.

2.1 Implementation step-by-step

Upgrading a legacy database system can be a daunting task. With large numbers of tables in the database and many lines of code both in backend and frontend upgrading it all to a more modern data access framework may incur both high costs and considerable risk. Instead of embarking on a disruptive migration project, we advocate a piecewise implementation. Speedment is designed to facilitate such a process. Speedment coexists nicely with the legacy business logic. It will instantly deliver value to the selected piece of the solution whereas the rest remains unaffected.

A typical approach for a Speedment integration project is to single out a particular database query and its associated business logic code. By replacing only that particular part of the data access logic, the system will leverage benefits of Speedment for the parts affected without making any change to the rest of the legacy system.

In general, adding a new data source to an existing system may introduce inconsistencies, but when the Speedment in-JVM-memory datastore is used as a read-only store, the acceleration does not add any new ways of writing data. Therefore, the underlying relational database remains the source of truth for all queries.

Speedment provides two different APIs. Speedment and Speedment Enterprise uses a Java Streams API, whereas the product named Ext Speeder leverages this API to auto generate a backend that has a full REST API that can be used by the frontend directly. When using the Java API, Speedment will coexist with legacy backend code since both use JDBC. For the Ext Speeder case, the auto generated backend provides a REST API to the parts of the data model that is integrated. The Speedment and legacy REST endpoints can coexist, allowing the frontend client to use both simultaneously. Thus, the client may keep using the legacy system for data from some tables of the database, while using the accelerated Speedment endpoint for others.

2.2 Separate Deployment

While the section above describes how a partial Speedment migration will coexist with a legacy DB2 system with focus on the software handling the database, in some cases it may be required to be even more conservative in terms of legacy backend impact. Particular useful for the REST API case provided by the Ext Speeder tool, Speedment supports introducing a dedicated server while leaving the legacy backend solution untouched.

Since piecewise migration to Ext Speeder means introducing a separate REST endpoint for Speedment accelerated data access, this endpoint may be served from a dedicated server. The Speedment server then appears as a separate backend machine, accessing the relational database using standard JDBC connections.

While such a deployment means that the relational database now feeds two different back ends with data, the total number of requests is considerably smaller since the Ext Speeder backend only periodically updates its datastore contents while continuously providing the frontend with data from the datastore.

Speedment, Inc.
470 Ramona Street
Palo Alto 94301 CA,
USA
+1 650 387 4069
info@speedment.com