DB2-Java Performance Analysis & Tuning Considerations
Hugh Smith (smithhj@us.ibm.com)
IBM Silicon Valley Laboratory
Agenda

- Java JDBC/SQLJ processing overview
- Overview of performance analysis tools
- Java performance analysis steps
- Java performance problem areas and program tuning considerations
Disclaimer

• The information contained in this document has not been submitted to any formal IBM review and is distributed on an "As Is" basis without any warranty either express or implied. The use of this information is a customer responsibility.

• The measurement results presented here were run in a controlled laboratory environment using specific workloads. While the information here has been reviewed by IBM personnel for accuracy, there is no guarantee that the same or similar results will be obtained elsewhere. Performance results depend upon the workload and the environment. Customers attempting to adapt this data to their own environments do so at their own risk.

• In addition, the materials in this document may be subject to enhancements or Programming Temporary Fixes (PTFs) subsequent to the level used in this study.
Configurations

Response time is slow - no throughput !!!!

What do I do? Where do I start?

Java App. is either Web environment or stand-alone Java application
Local execution on z/OS via RRSAF or CAF

Sample configuration:
WebSphere on OS/390 or z/OS
Java application in USS

Available IBM JDBC/SQLJ driver on OS/390 and z/OS

JDBC 1.2, DB2 V6, V7 (CLASSPATH:db2sqljclasses.zip)
JDBC 2.0, DB2 V7 (CLASSPATH:db2j2classes.zip)
(Universal Driver for SQLJ and JDBC T2,
DB2 V7, V8 (CLASSPATH:db2jcc.jar) - avail. end of 2003)
Connecting via RRSAF/CAF

<table>
<thead>
<tr>
<th>USS or WAS</th>
<th>JDBC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Connect</td>
<td></td>
</tr>
<tr>
<td>Prepare</td>
<td></td>
</tr>
<tr>
<td>Open</td>
<td></td>
</tr>
<tr>
<td>Fetches</td>
<td></td>
</tr>
<tr>
<td>Close Cursor &amp; Commit</td>
<td></td>
</tr>
</tbody>
</table>

CLASS 1

CLASS 2
Types of DB2 Connectivity

- Distributed access to DB2 z/OS via DDF
  - Sample Configuration:
    WebSphere on workstation - DB2 Connect
    Java application using JCC Type 4 driver
  - Available IBM JDBC/SQLJ driver
    JDBC .app (CLASSPATH:db2java.zip)
    JDBC .net (CLASSPATH:db2java.zip)
    Universal Driver for SQLJ and JDBC T4, DB2 V8
    Universal Driver for SQLJ and JDBC T2, DB2 V8 FP2
      (CLASSPATH:db2jcc.jar)
Connecting via DDF - Client

Java Appl.

getConnection
prepareStatement
executeQuery
next
close
commit

JDBC/CLI  -  Client Comm or
Univ.Dr.T2  -  Client Comm or
Univ.Dr. T4

Connect
Prepare &
Open Cursor
Get Next Query Block
Next Block Returns
Close Cursor
Commit
Connecting via DDF - DB2 z/OS

Connect
Prepare & Open Cursor
Close Cursor & Commit

CLASS 1
CLASS 2

SQL
Fetches
Types of DB2 Connectivity ...

- Distributed access to DB2 workstation
  - Sample Configuration:
    WebSphere on workstation
    Java application using JCC Type 4 driver
  - Available IBM JDBC/SQLJ driver
    JDBC .app/.net (CLASSPATH:db2java.zip)
    Universal Driver for SQLJ and JDBC T4, DB2 V8
    Universal Driver for SQLJ and JDBC T2, DB2 V8 FP2
    (CLASSPATH:db2jcc.jar)
Connecting to DB2 Workstation

- Connect
- Prepare/
- Execute
- Next
- Block
- CloseCursor
- Commit
- Build client block of rows
- Build client block of rows
- Client Comm
- Server
Analysis Tools

JCC Monitor API/Trace
Java Profile Interface

Network Analyzer

DB2 Connect Gateway

Java App.
JCC T4

DRDA

DB2 for OS/390 and z/OS
JDBC/SQLJ T2 trace or
JCC Monitor API/Trace
Java Profile Interface

Java App.

DB2 for Unix and Windows
JDBC/SQLJ T2 or
JCC T2

Java App.

ddcstrc
DB2 CC PerfMon

DB2 CLI Trace
JDBC/SQLJ T2
ddcstrc
Java Profile Interface

DB2 CLI Trace or
JCC Monitor API/Trace
Java Profile Interface

DB2 System Monitor

JDBC/SQLJ T2 or
JCC T2
Using DB2 Statistics

- Not really used for analysis but verifying DB2 is behaving as expected
- If JDBC is used, check dynamic SQL statement block
- If distributed access, check DDF blocks

### GLOBAL DDF ACTIVITY

<table>
<thead>
<tr>
<th>Activity</th>
<th>Quantity</th>
</tr>
</thead>
<tbody>
<tr>
<td>DBAT QUEUED-MAXIMUM ACTIVE</td>
<td>0.00</td>
</tr>
<tr>
<td>CONV. DEALLOC-MAX. CONNECTED</td>
<td>0.00</td>
</tr>
<tr>
<td>COLD START CONN.</td>
<td>0.00</td>
</tr>
<tr>
<td>WARM START CONN.</td>
<td>0.00</td>
</tr>
<tr>
<td>RESYNCHRONI. ZATI ON ATTEMPTED</td>
<td>0.00</td>
</tr>
<tr>
<td>RESYNCHRONI. ZATI ON SUCCEEDED</td>
<td>0.00</td>
</tr>
<tr>
<td>CUR TYPE 1 I NACTI VE DBATS</td>
<td>0.00</td>
</tr>
<tr>
<td>TYPE 1 I NACTI VE DBATS HWM</td>
<td>1.00</td>
</tr>
<tr>
<td>TYPE 1 CONNECTI ONS TERM NAT</td>
<td>0.00</td>
</tr>
<tr>
<td>CUR TYPE 2 I NACTI VE DBATS</td>
<td>17.39</td>
</tr>
<tr>
<td>TYPE 2 I NACTI VE DBATS HWM</td>
<td>42.00</td>
</tr>
<tr>
<td>ACC QUEUED TYPE 2 I NACT THR</td>
<td>6301.00</td>
</tr>
<tr>
<td>CUR QUEUED TYPE 2 I NACT THR</td>
<td>0.00</td>
</tr>
<tr>
<td>QUEUED TYPE 2 I NACT THR HWM</td>
<td>17.00</td>
</tr>
<tr>
<td>CURRENT ACTI VE DBATS</td>
<td>31.00</td>
</tr>
<tr>
<td>ACTI VE DBATS HWM</td>
<td>20.00</td>
</tr>
<tr>
<td>TOTAL DBATS HWM</td>
<td>42.00</td>
</tr>
<tr>
<td>CURRENT DBATS NOT IN USE</td>
<td>6.39</td>
</tr>
<tr>
<td>DBATS NOT IN USE HWM</td>
<td>30.00</td>
</tr>
<tr>
<td>DBATS CREATED</td>
<td>0.00</td>
</tr>
<tr>
<td>POOL DBATS REUSED</td>
<td>12602.00</td>
</tr>
</tbody>
</table>

### DYNAM C SQL STMT

<table>
<thead>
<tr>
<th>Activity</th>
<th>Quantity</th>
</tr>
</thead>
<tbody>
<tr>
<td>PREPARE REQUESTS</td>
<td>480.05</td>
</tr>
<tr>
<td>FULL PREPARES</td>
<td>1.37</td>
</tr>
<tr>
<td>SHORT PREPARES</td>
<td>478.68</td>
</tr>
<tr>
<td>GLOBAL CACHE HIT RATIO (%)</td>
<td>99.71</td>
</tr>
</tbody>
</table>

### DRDA REMOTE LOCs

<table>
<thead>
<tr>
<th>Activity</th>
<th>Sent</th>
<th>Received</th>
</tr>
</thead>
<tbody>
<tr>
<td>TRANSACTIONS</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>CONVERSATI ONS</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>CONVERSATI ONS QUEUED</td>
<td>0.00</td>
<td></td>
</tr>
<tr>
<td>SQL STATEMENTS</td>
<td>1459.2K</td>
<td></td>
</tr>
<tr>
<td>SINGLE PHASE COMMITS</td>
<td>442.7K</td>
<td></td>
</tr>
<tr>
<td>SINGLE PHASE ROLLBACKS</td>
<td>0.00</td>
<td>0.00</td>
</tr>
</tbody>
</table>

### Blocking: rows are put into blocks which are then sent out in messages
Using DB2 Accounting

- Processing "in DB2" (Class 2) should be the same regardless of connection type

- Time in DB2 is:
  - Local Access:
    - CL.2 non-nested ET +
    - CL.1 SP, UDF, trigger ET
  - Distr. Access:
    - CL 2 non-nested ET +
    - CL.1 SP, UDF, trigger ET +
    - Nonnested (CL.1 CPU - CL.2 CPU)

- Time outside DB2 is
  - Total CL.1 ET - time in DB2
    (previous calculation)
  - Total CL.1 ET also includes
    idle time when thread is reused
  - Local Access: CPU spent in app.
    Nonnested (CL.1 CPU - CL.2 CPU)

- Accounting Records are created at
  thread deallocation or at DBAT inactive

<table>
<thead>
<tr>
<th>Average</th>
<th>APPL (CL.1)</th>
<th>DB2 (CL.2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ELAPSED TIME</td>
<td>7: 22.69662</td>
<td>5.287841</td>
</tr>
<tr>
<td>NONNESTED</td>
<td>7: 20.65807</td>
<td>4.456104</td>
</tr>
<tr>
<td>STORED PROC</td>
<td>2.027677</td>
<td>0.820862</td>
</tr>
<tr>
<td>CPU TIME</td>
<td>3.270153</td>
<td>3.270153</td>
</tr>
<tr>
<td>AGENT</td>
<td>3.270153</td>
<td>3.270153</td>
</tr>
<tr>
<td>NONNESTED</td>
<td>2.027677</td>
<td>0.820862</td>
</tr>
<tr>
<td>STORED PROC</td>
<td>2.027677</td>
<td>0.820862</td>
</tr>
<tr>
<td>UDF</td>
<td>0.000000</td>
<td>0.000000</td>
</tr>
<tr>
<td>TRIGGER</td>
<td>0.010875</td>
<td>0.010875</td>
</tr>
<tr>
<td>CPU TIME</td>
<td>3.270153</td>
<td>3.270153</td>
</tr>
<tr>
<td>AGENT</td>
<td>3.270153</td>
<td>3.270153</td>
</tr>
<tr>
<td>NONNESTED</td>
<td>2.434445</td>
<td>1.551221</td>
</tr>
<tr>
<td>STORED PROC</td>
<td>0.835235</td>
<td>0.588131</td>
</tr>
<tr>
<td>CPU TIME</td>
<td>3.270153</td>
<td>3.270153</td>
</tr>
<tr>
<td>AGENT</td>
<td>3.270153</td>
<td>3.270153</td>
</tr>
<tr>
<td>NONNESTED</td>
<td>2.434445</td>
<td>1.551221</td>
</tr>
<tr>
<td>STORED PROC</td>
<td>0.835235</td>
<td>0.588131</td>
</tr>
<tr>
<td>UDF</td>
<td>0.000472</td>
<td>0.000472</td>
</tr>
<tr>
<td>TRIGGER</td>
<td>0.000000</td>
<td>0.000000</td>
</tr>
<tr>
<td>PAR. TASKS</td>
<td>0.000472</td>
<td>0.000472</td>
</tr>
<tr>
<td>SUSPEND TIME</td>
<td>N/A</td>
<td>2.433828</td>
</tr>
<tr>
<td>AGENT</td>
<td>N/A</td>
<td>2.433828</td>
</tr>
<tr>
<td>PAR. TASKS</td>
<td>N/A</td>
<td>0.000000</td>
</tr>
</tbody>
</table>
Using DB2 Accounting ...

- JDBC executes dynamic SQL
  - check DB2 dynamic statement cache

<table>
<thead>
<tr>
<th>Dynamic SQL Statement</th>
<th>Average</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reoptimization</td>
<td>0.00</td>
<td>0</td>
</tr>
<tr>
<td>Not found in cache</td>
<td>0.07</td>
<td>24</td>
</tr>
<tr>
<td>Found in cache</td>
<td>25.94</td>
<td>8353</td>
</tr>
<tr>
<td>Implicit prepares</td>
<td>0.00</td>
<td>0</td>
</tr>
<tr>
<td>Prepares avoided</td>
<td>0.00</td>
<td>0</td>
</tr>
<tr>
<td>Stmt invalid (MAX)</td>
<td>0.00</td>
<td>0</td>
</tr>
<tr>
<td>Stmt invalid (DDL)</td>
<td>0.00</td>
<td>0</td>
</tr>
</tbody>
</table>

- Distributed access
  - check distributed activity

<table>
<thead>
<tr>
<th>SQL</th>
<th>DML</th>
<th>Average</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Select</td>
<td>1.00</td>
<td>29</td>
<td></td>
</tr>
<tr>
<td>Insert</td>
<td>0.00</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Update</td>
<td>0.00</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Delete</td>
<td>0.00</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Describe</td>
<td>0.00</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Desc. table</td>
<td>0.00</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Prepare</td>
<td>0.00</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Open</td>
<td>1.00</td>
<td>29</td>
<td></td>
</tr>
<tr>
<td>Fetch</td>
<td>15.14</td>
<td>439</td>
<td></td>
</tr>
<tr>
<td>Close</td>
<td>0.00</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

DML - All  17.14  497
JDBC/SQLJ T2 Trace on z/OS

- Activated by environment var settings
  - in `db2sqljjjdbc.properties` file
    - `DB2SQLJ_TRACE_FILENAME=MYTRC.OUT`
    - `DB2SQLJ_TRACE_BUFFERSIZE=4096`

- Generated output
  - `MYTRC.OUT` - binary format
    - must be formatted using `db2sqljtrace` command
  - `MYTRC.OUT.JTRACE` - text format

- Helpful for analyzing
  - method flow
  - time spent in JDBC/SQLJ methods and application
Example of MYTRC.OUT.JTRACE

<2002.08.24 15:52:56.274> <Entry>  <prepareStatement>
<COM.ibm.db2os390.sqlj.jdbc.DB2SQLJConnection@2acf2e20>  <WebSpheret=008cdd90>
--
<p#1=COM.ibm.db2os390.sqlj.jdbc.DB2SQLJConnection@2acf2e20[pCONN=57b36908]>

<2002.08.24 15:52:56.278> <Entry>  <Initialize>
<COM.ibm.db2os390.sqlj.jdbc.DB2SQLJJDBCCursor@1a09ae20>  <WebSphere t=008cdd90>
...

Output of db2sqljtrace flw MYTRC.OUT

DB2SQLJ jdbc_cursor_native_fetch fnc_entry ...
DB2SQLJ jdbc_cursor_native_fetch fnc_data ...
| sqlj_DSNHLI fnc_entry ...
| | sqlj_ctxSetContext fnc_entry ...
| | sqlj_ctxSetContext fnc_data ...
| | sqlj_ctxSetContext fnc_data ...
| | sqlj_ctxSetContext fnc_retcode 0
| sqlj_DSNHLI fnc_data ...
| | sqlj_ctxDiscContextCond fnc_entry ...
| | sqlj_ctxDiscContextCond fnc_data ...
| | sqlj_ctxDiscContextCond fnc_retcode 0
| sqlj_DSNHLI fnc_retcode 0
DB2SQLJ jdbc_cursor_native_fetch fnc_retcode 0
DB2 JDBC/CLI Trace on Unix/Windows

- Activated via [COMMON] section entries of db2cli.ini
  
  ```
  [COMMON]
  TRACE=1
  TRACECOMM=1
  TRACEPATHNAME="a full subdirectory specification"
  JDBCTRACE=1
  
  JDBCTRAFCEPATHNAME="a full subdirectory specification"
  ```

- Generated Output
  - TRACEPATHNAME creates a file for each thread/process in the given directory

- Helpful for analyzing
  - message flow
  - time spent in application, CLI functions and network
Example of CLI trace output

SQLNumResultCols( hStmt=2:1, pcCol=&1343f6ca )
    ---> Time elapsed - +6.330000E-004 seconds

SQLNumResultCols( pcCol=2 )
    <--- SQL_SUCCESS  Time elapsed - +4.500000E-005 seconds

SQLFetch( hStmt=2:1 )
    ---> Time elapsed - +9.600000E-004 seconds

...

Output of CLITraceParser

Overall Trace statistics

180 statements in trace.
13.362 seconds total trace time.
3.267 seconds spent for application processing.
10.095 seconds spent for CLI processing.

Network Specific CLI processing time statistics

87 network flows sent to transmit
13821 bytes, requiring a total of
1.065 seconds.
121 network flows received, transmitting
9086 bytes, requiring a total of
8.705 seconds.
IBM DB2 Univ. Driver for SQLJ and JDBC

- Java API for set client information
  - Before could be set as DB2 DataSource property
    
    \[
    \text{ds.setConnectionAttribute}
    \]
  - New Java methods for existing Set Client Information API
    
    \[
    \begin{align*}
    \text{setClientUser(maryela1)} \\
    \text{setClientWorkStation(CL01)} \\
    \text{setClientApplicationInformation(payment)} \\
    \text{setClientAccountingInformation(String)}
    \end{align*}
    \]
  - For all users: provides additional monitoring information
    
    ```
    D71B DIS THREAD(*)
    NAME ST A REQ ID AUTHID PLAN ASI D TOKEN
    SERVER RA * 952 db2jccThread USRT001 DISTSERV 004A 432
    V437 WORKSTATION=CL01, USERID=maryela1,
    APPLI CATI ON NAME=payment
    V445 G91E81C5. G49D. 00F330DF736F=432 ACCESSING DATA FOR 9. 30. 129. 197
    ```

- For DB2 for z/OS: Strings are included in IFC records

  ```
  DB2 PERFORMANCE MONITOR ACCOUNTING REPORT
  ORDER: TRANSACT
  #OCCURS . . . CLASS1 EL. TI ME CLASS2 EL. TI ME
  #DI STRS . . . CLASS1 CPU TI ME CLASS2 CPU TI ME
  -------------------------------
  payment 4 1.000902 0.005561
  4 0.003573 0.002457
  ```
Java API for application monitoring (avail. with FP4)

- DB2SystemMonitor monitor = ((DB2Connection)conn).getDB2SystemMonitor();
- monitor.enable(true);
- monitor.start(com.ibm.db2.jcc.DB2SystemMonitor.RESET_TIMES);
- monitor.stop();
- monitor.getServerTime()
- monitor.getNetworkIOTime()
- monitor.getCoreDriverTime()
- monitor.getApplicationTime()

Prereqs IBM JDK
- non-z/OS JDK 131 SR 5 available now
- z/OS JDK131 next PTF and JDK 141 SR1
Trace activation

- External API of com.ibm.db2.jcc.DB2Connection
- Dynamically turn trace on/off
- Multiple levels of trace detail
- Helpful for analyzing method flows, DRDA buffer
- Calling example

```java
((DB2Connection) con).setJCCLogWriter(
    java.io.PrintWriter logWriter, int tracelevel);
```

Example of trace output

```
[ibm][db2][jcc][Thread: main][Connection@50b9ee8a] setAutoCommit(false) called
[ibm][db2][jcc][Thread: main][Connection@50b9ee8a] prepareStatement(SELECT FKEY FROM WRKTBO1 WHERE (FKEY >= ?) OPTIMIZE FOR 1 ROW) called
[ibm][db2][jcc][Thread: main][Connection@50b9ee8a] prepareStatement() returned PreparedStatement@ee32e8a
[ibm][db2][jcc][Thread: main][PreparedStatement@ee32e8a] setShort (1, 400) called
[ibm][db2][jcc][Thread: main][PreparedStatement@ee32e8a] executeQuery() called
[ibm][db2][jcc][Thread: main][ResultSetMetaData@2b98ae8a] BEGIN
TRACE_RESULT_SET_META_DATA
[ibm][db2][jcc][Thread: main][ResultSetMetaData@2b98ae8a] Result set metadata for statement Statement@136bee8a
[ibm][db2][jcc][Thread: main][ResultSetMetaData@2b98ae8a] Number of result set columns: 12
```
Java Profiler Interface

Several tools explore this interface
- hprof (provided with JDK)
- Jinsight
- ...

hprof
- JIT needs to be turned off
- Not usable in production-like environments but good for single Java application analysis
- Calling example
  ```
  java -Xrunhprof:cpu=samples,file=log.txt,depth=4 runit
  ```

Sample Output

```
CPU SAMPLES BEGIN (total = 874) Wed Sep 12 09:58:27 2002
rank self accum count trace method
1 29.41% 29.41% 257 191 java/io/FileOutputStream.writeBytes
2 8.12% 37.53% 71 88 COM/ibm/db2os390/sqlj/jdbc/DB2SQLJDriver.native
3 6.41% 43.94% 56 216 COM/ibm/db2os390/sqlj/jdbc/DB2SQLJConnection.
4 4.81% 48.74% 42 91 java/util/zip/ZipFile.getEntry
5 2.52% 51.26% 22 54 java/lang/ClassLoader.defineClass0
6 1.95% 53.20% 17 187 COM/ibm/db2os390/sqlj/jdbc/DB2SQLJJDBCSection
7 1.72% 54.92% 15 194 COM/ibm/db2os390/sqlj/jdbc/DB2SQLJJDBCCursor.
8 1.37% 56.29% 12 60 java/lang/ClassLoader.findBootstrapClass
9 1.14% 57.44% 10 27 java/io/FileInputStream.readBytes
```
Java Profiler Interface ...

- **Jinsight**
  - Part of WebSphere Studio Application Developer
  - JIT can be turned on
  - Visualizer for performance analysis

- **Sample Output**

![Call Tree: Calls from java/io/PrintStream.println (String): 1 occurrences [Workspace 1]](image-url)
Tools to Analyze Distributed Activity

- **ddcstrc (DRDA trace)**
  - Start on workstation where DRDA comm. to host occurs
  - Not easily used for performance analysis, requires knowledge of DRDA flows

- **z/OS TCP/IP packet trace**
  - Minimal impact to scenario
  - Operational instructions found in "IP Diagnosis Guide"
  - Can be used trace specific IP address/protocol

- **Network Analyzer**
  - Usually no impact on measurement - shows what is on the wire
  - Network switch devices require configuration to use analyzer
  - Location of analyzer trace makes a difference
Java Performance Analysis Steps

1. Understand your configuration

2. Verify DB2 Server processing is satisfactory
   - On z/OS:
     ✓ Use statistics to verify that DB2 is healthy and accounting to ensure that transaction, connection type, plan/program, etc., is behaving correctly
     ✓ Calculate processing time inside and outside of DB2 via accounting
   - On Windows/Unix:
     ✓ Use DB2 System Monitor to verify the correct behavior of DB2
Java Performance Analysis Steps ...

3. Analyze Java (client) application processing
   ✓ Turn on JDBC/SQLJ traces dependent on JDBC/SQLJ driver used
   ✓ Use Java profile tools like Jinsight to analyze behavior of methods in Java application

4. Analyze network performance
   ✓ Check network path via ping or traceroute, tracert, tracerte commands
   ✓ On z/OS: activate TCP/IP packet trace to verify communications stack operation
   ✓ Utilize network analyzer device at appropriate points in network
Java Performance Problem Areas

- **Java Application**
  - No use of JDBC Connection Pooling
  - Mismatch of Java and DB2 data types
  - Usage of String for numbers
  - Retrieval of unused columns (select * )
  - Autocommit(on) - default
  - Transaction isolation REPEATABLE READ or SERIALIZABLE
  - Open cursor SELECT ... FOR UPDATE

- **JDBC**
  - JDBC resources not closed (cursor, statements, connections)
  - No usage of Parameter Markers (select c1, c2 FROM t1 WHERE c3=?)
  - Cursor are defined as hold by default
Java Performance Problem Areas

- **SQLJ**
  - No explicit connection context used
    ```sql
    (#sqlj [cxt1] c1={select c1 FROM t1};
    ```
  - Customization without online checking
  - Usage of named iterator instead of positioned iterator

- **Environment**
  - Old System levels (JDK, JDBC/SQLJ driver)
  - JVM heap to small
  - DB2 Dynamic Statement Cache not active for dynamic SQL