

DB2-åtkomst från WebSphere

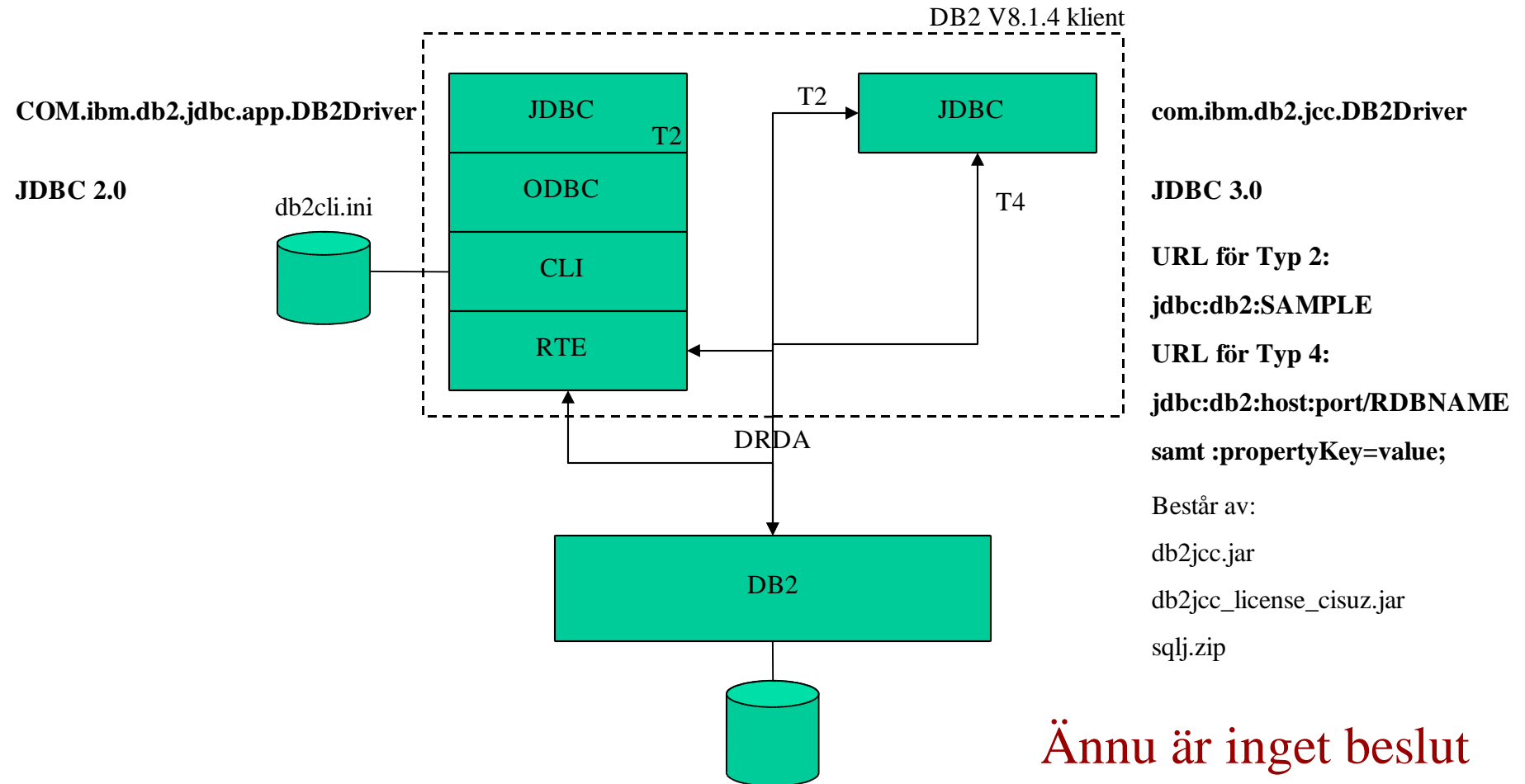
Idéer och tankar inför etablerandet av
”best practices” vid användning av DB2
i WebSphere-miljö

Lennart Henäng

Agenda

- JCC - ny design och konfiguration (T2/T4)
- Autenticering och behörighet
- Spårbarhet och debitering
- Prestandamätning
- Loggning, klient och server
- Felhantering
- Trace
- SQLJ vs JDBC

JCC - ny design och konfiguration (T2/T4)

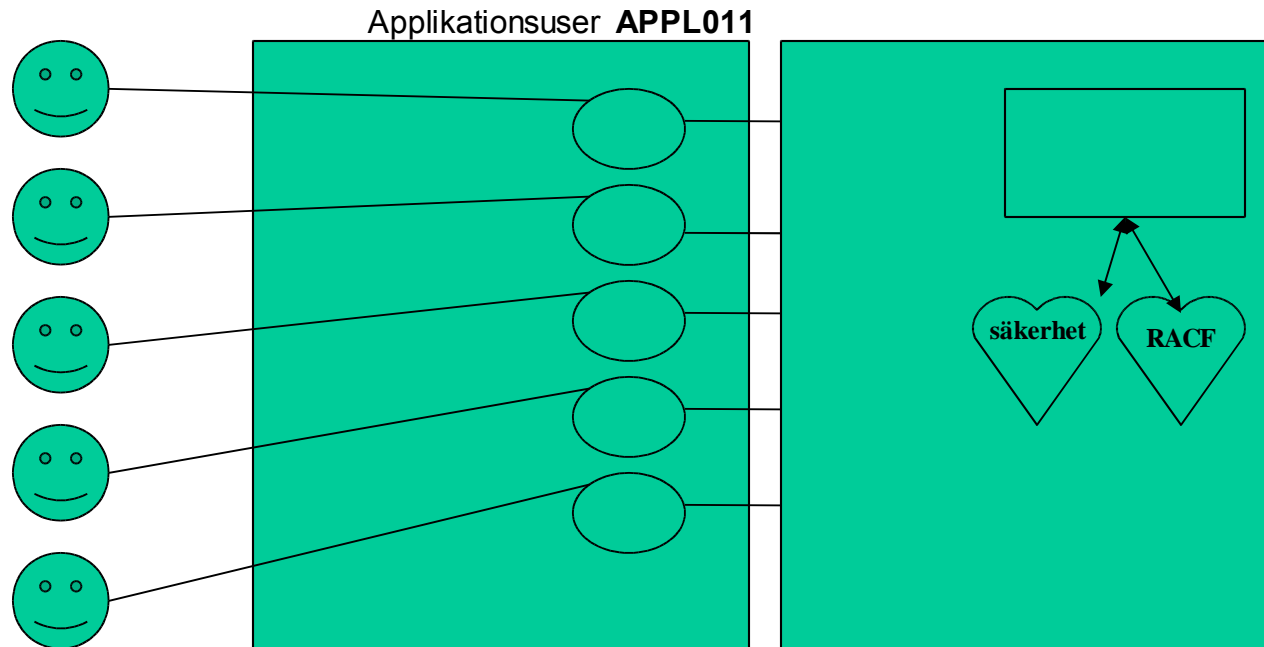


Ännu är inget beslut taget om T2 eller T4!

propertyKeys - några exempel

- driverType
- user
- password
- currentPackageSet
- currentSchema
- currentSQLID
- retrieveMessagesFromServerOnGetMessage
- clientUser, clientWorkstation...

Autentisering och behörighet



Connection Pooling (endast T2)

Thread Pooling

Autenticering och behörighet

- Applikationsuser konfigureras i DataSource
- Applikationen autenticerar användaren
 - Interna användare via anrop till RACF
 - Externa användare via anrop till säkerhet
- Behörighet kontrolleras mot RACF
- Egna resurser sätts upp i RACF
- Grupper ges READ eller UPDATE behörighet
- Användare kopplas till rätt grupp

Test av autenticering och behörighet

- Anrop till Stored Procedure AUTHENTICATE
 - #sql {CALL AUTHENTICATE(:in user, :in passwd, :in application, :out outText, :out statusCode)};
- Assemblerprogram anropar RACF
 - RACROUTE REQUEST=VERIFY
 - Bygger ACEE baserat på userid och password
 - RACROUTE REQUEST=AUTH
 - Frågar RACF om behörighet till resurs
- Stored Procedure svarar med statusCode
 - 0 eller 8 plus meddelande ev SQLSTATE
- Anrop tar cirka 90 ms varav 60 ms i DB2

OSA-adaptorn används inte i nuläget!

Spårbarhet

Idag:

```
DSNV401I = DISPLAY THREAD REPORT FOLLOWS -
DSNV424I = INACTIVE THREADS -
NAME      ST A   REQ ID          AUTHID   PLAN      ASID TOKEN
SERVER    R2      0 java          APPL011  DISTSERV 00A8   17
V437-WORKSTATION=app1p, USERID=app1011,
        APPLICATION NAME=java
V445-SEFOLK00.U007I001.BB563C56EB75=17 ACCESSING DATA FOR
<U007I001>:U007I001
```

Spårbarhet

Idag:

```
DSNV401I = DISPLAY THREAD REPORT FOLLOWS -
DSNV424I = INACTIVE THREADS -
NAME      ST A   REQ ID          AUTHID   PLAN      ASID TOKEN
SERVER    R2     0 java              APPL011  DISTSERV 00A8   17
V437-WORKSTATION=app1p, USERID=app1011,
        APPLICATION NAME=java
V445-SEFOLK00.U007I001.BB563C56EB75=17 ACCESSING DATA FOR
<U007I001>:U007I001
```

Förslag:

```
DSNV401I = DISPLAY THREAD REPORT FOLLOWS -
DSNV402I = ACTIVE THREADS -
NAME      ST A   REQ ID          AUTHID   PLAN      ASID TOKEN
SERVER    RA *   3 db2jccmain     APPL011  DISTSERV 0208   54
V437-WORKSTATION=193.44.158.105, USERID=19730415-8652,
        APPLICATION NAME=S70D016
V445-GA0135E3.G504.00F8EDFF2E26=54 ACCESSING DATA FOR 10.1.53.227
```

API för spårbarhet

```
DB2Connection con = (DB2Connection)DriverManager.getConnection( url, userID, password );
...
// clientUser maximum 16 characters for DB2 for z/OS
String clientUser = "19730415-8652";
// wkStn maximum 18 characters for DB2 for z/OS
String wkStn = "193.44.158.105";
// applInfo maximum 32 characters for DB2 for z/OS
String applInfo = "S70D016";
// acclInfo maximum 200 characters for DB2 for z/OS
String acclInfo = "1234";
...
// Set client information
con.setDB2ClientUser( clientUser );
con.setDB2ClientWorkstation( wkStn );
con.setDB2ClientApplicationInformation( applInfo);
con.setDB2ClientAccountingInformation( acclInfo );

// Generated correlator (luwid)

System.out.println("Correlator: " +con.getDB2Correlator
());

//will print:

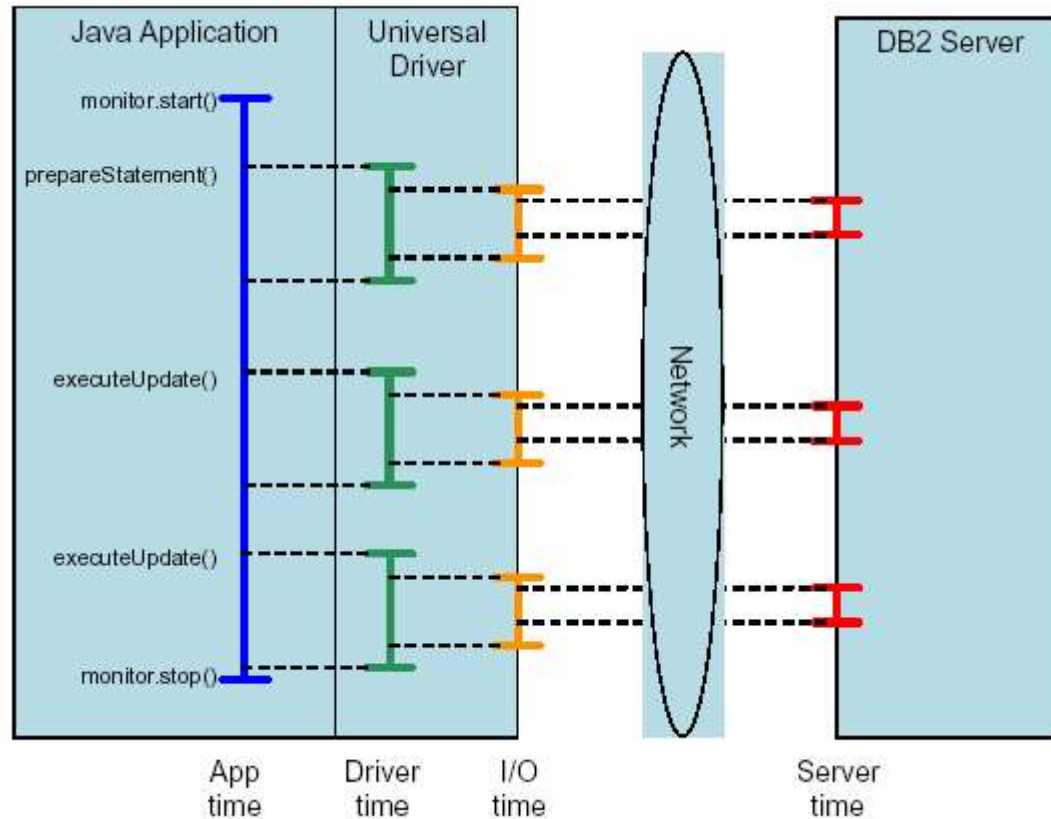
//Correlator: GA0135E3.G504.00F8EDFF2E26
```

```
// Create an application trace (four last apps)
con.setDB2ClientApplicationInformation(applInfo + "
" + getDB2ClientApplicationInformation());
```

Debitering

- Ordinarie DB2 Accounting Trace
- Införs i ordinarie debiteringsrutiner

Prestandamätning "end-to-end"



API för prestandamätning

- Kräver JVM 1.4.1 från IBM
 - annars SQLException på
 - getCoreDriverTimeMicros
 - getNetworkIOTimeMicros
- Kräver DRDA-anlutning
 - annars 0 i getServerTimeMicros

```
...  
// do some performance measuring  
DB2SystemMonitor monitor = con.getDB2SystemMonitor(); /* I already have a DB2Connection */  
monitor.enable(true);  
// monitor.start(DB2SystemMonitor.ACCUMULATE_TIMES);  
monitor.start(DB2SystemMonitor.RESET_TIMES);  
// Call the stored procedure by invoking this method  
callMyStoredProc(outMedian, ctx);  
// and stop the monitor  
monitor.stop();
```

Test av API för prestandamätning

```
// The Core Driver and Network IO times can only be gotten with an IBM JVM
// The Server time always shows as 0 when running with a local DB2
// This is verified with DB2 development (Curt Cotner) 2003-11-03
System.out.println("Application Time: " + monitor.getApplicationTimeMillis() + " ms");
try {
    System.out.println("Core Driver Time: " + monitor.getCoreDriverTimeMicros() + " us");
} catch (Exception x)
{
    System.out.println("JVM does not have accurate timer support");
    System.out.println("Called method was: getCoreDriverTimeMicros");
}
try {
    System.out.println("Network IO Time: " + monitor.getNetworkIOTimeMicros() + " us");
} catch (Exception x)
{
    System.out.println("JVM does not have accurate timer support");
    System.out.println("Called method was: getNetworkIOTimeMicros");
}
System.out.println("Server Time: " + monitor.getServerTimeMicros() + " us");
System.out.println("If above time is 0, then you are probably running with a local DB2");
```

Output vid prestandamätning

Run as a Java app

Application Time: **611 ms**

JVM does not have accurate timer support

Called method was: getCoreDriverTimeMicros

JVM does not have accurate timer support

Called method was: getNetworkIOTimeMicros

Server Time: **57939 us**

If above time is 0, then you are probably running with a local DB2

Run within WSAD

Application Time: **109 ms**

Core Driver Time: 100526 us

Network IO Time: 52867 us

Server Time: **38183 us**

If above time is 0, then you are probably running with a local DB2

Loggning, klient och server

- Klient

- Authid, Correlation, Application, Workstation, RealUser, Time, AppTime, DriverTime, NetworkIOTime, ServerTime

- ...

- Server

- Authid, Correlation (?), Application (V8), Workstation (V8), RealUser (V8), Time

- ...

?

Felhantering

```
try
{
  #sql {CALL AUTHENTICATE(:in user, :in passwd, :in application, :out outText, :out statusCode)};
}
catch (SQLException e)
{
  if (e instanceof DB2Diagnosable) {
    DB2Sqlca sqlca = ((DB2Diagnosable) e).getSqlca();
    System.err.println("DSNT408I SQLCODE = " + sqlca.getSqlCode() + ", " + sqlca.getMessage());
    System.err.println("DSNT418I SQLSTATE = " + sqlca.getSqlState() + " SQLSTATE RETURN CODE");
    // System.err.println("SQLERRM : " + sqlca.getMessage());
    // System.err.println("SQLCODE : " + sqlca.getSqlCode());
    // System.err.println("SQLSTATE: " + sqlca.getSqlState());
    // String[] tokens = sqlca.getSqlErrmcTokens();
    // if (tokens != null) {
    //   for (int i = 0; i < tokens.length; i++) {
    //     System.err.println("ERRMC[" + i + "]: " + tokens[i]);
    //     System.err.println("SQLWARN: " + new String(sqlca.getSqlWarn()));
    //   }
    // }
  }
}
```

```
DSNT408I SQLCODE = -471, INVOCATION OF FUNCTION OR PROCEDURE ALHE03.AUTHENTICATE
FAILED DUE TO REASON 00E79001
DSNT418I SQLSTATE = 55023 SQLSTATE RETURN CODE
```

Trace

- Startas numera ifrån applikationen
- Har ej testat ännu

SQLJ vs JDBC

- SQLJ är lättare att koda
- SQLJ fångar fel tidigare
- SQLJ är snabbare
- SQLJ ger bättre säkerhet
- SQLJ är mer förutsägbart och tillförlitligt
- SQLJ ger bättre prestandauppföljning
- SQLJ har stöd i WSAD 5.1
- SQLJ hanterar inte spontana queries

Vadå enklare?

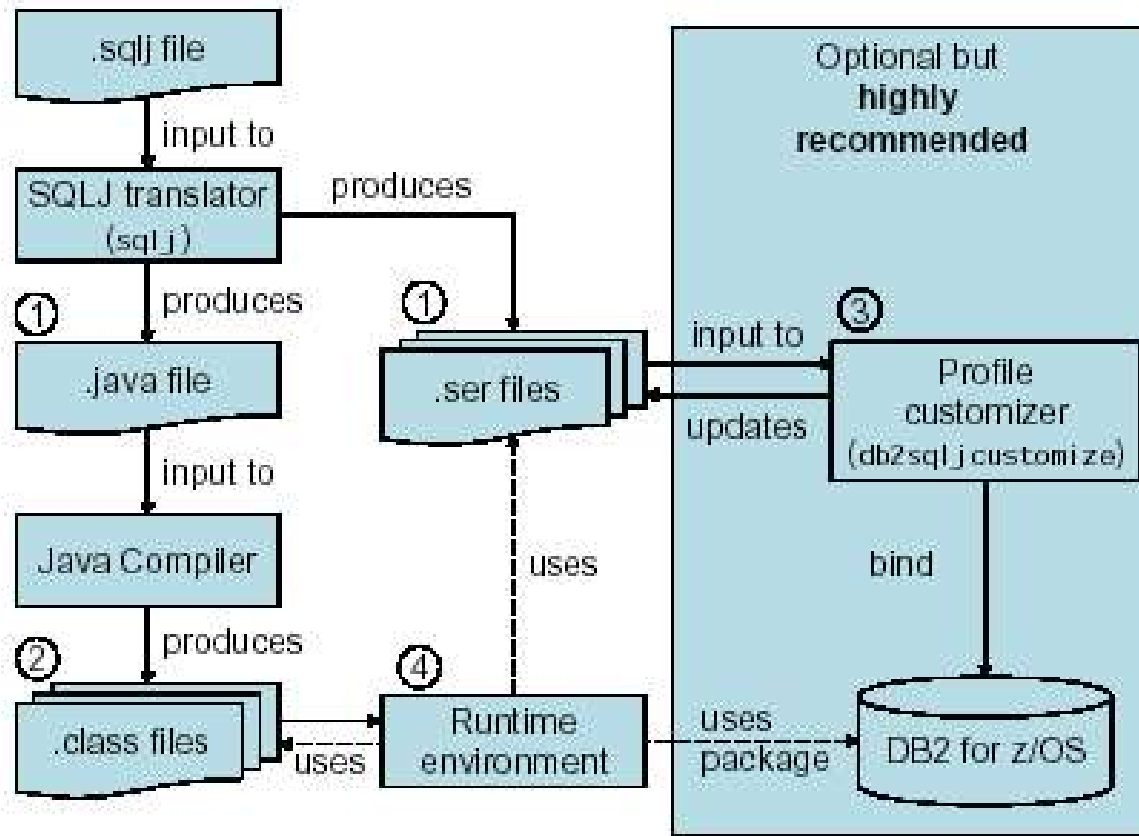
```
PreparedStatement stmt =
    conn.prepareStatement(
        "SELECT MAX(SALARY)
         , AVG(SALARY)"
        + " FROM DSN8710.EMP");
rs = statement.executeQuery();
if (!rs.next()) {
    // Error – no rows found
}
max.salary = rs.getBigDecimal(1);
avgSalary = rs.getBigDecimal(2);
if (rs.next()) {
    // Error – more than one row found
}
rs.close();
stmt.close();
```

Vadå enklare?

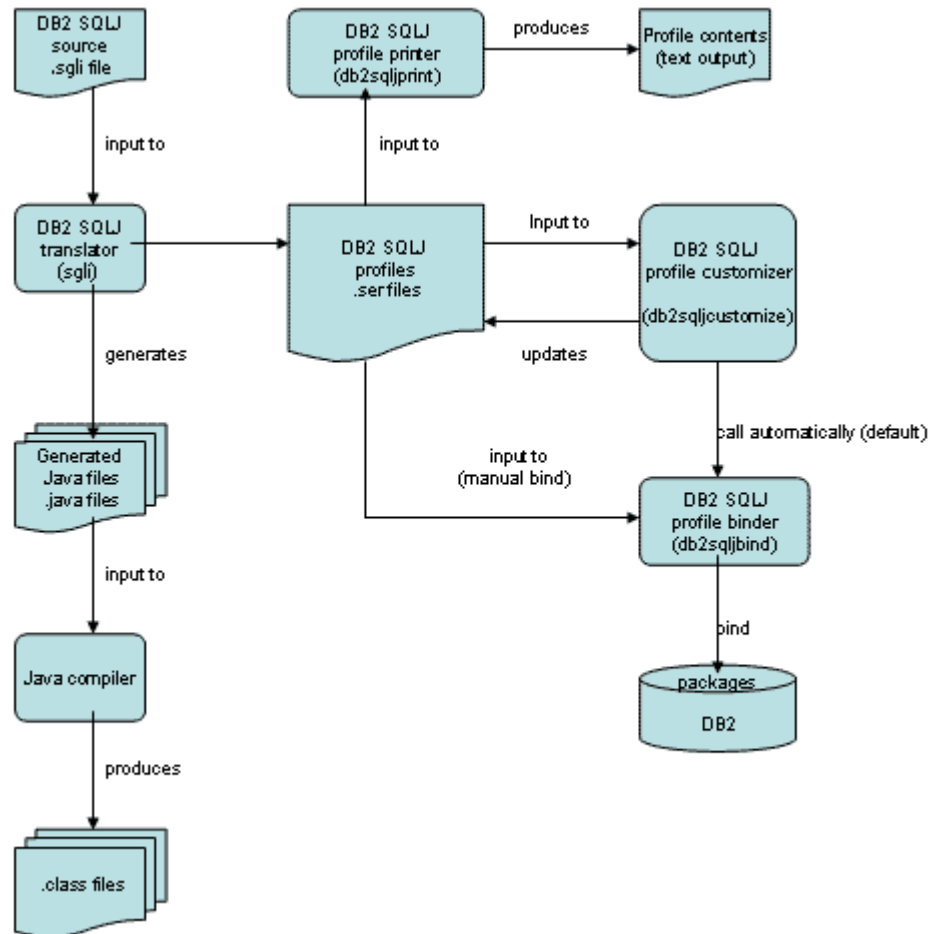
```
PreparedStatement stmt =
    conn.prepareStatement(
        "SELECT MAX(SALARY)
        , AVG(SALARY)"
        + " FROM DSN8710.EMP");
rs = statement.executeQuery();
if (!rs.next()) {
    // Error – no rows found
}
max.salary = rs.getBigDecimal(1);
avgSalary = rs.getBigDecimal(2);
if (rs.next()) {
    // Error – more than one row found
}
rs.close();
stmt.close();
```

```
#sql [ctx] {
    SELECT MAX(SALARY), AVG(SALARY)
    INTO :maxSalary, :avgSalary
    FROM DSN8710.EMP
};
```

SQLJ Program Preparation



SQLJ Program Preparation



Test av SQLJ

sqlj SpSecTE.sqlj



SpSecTE_SJProfile0

SpSecTE.java

SpSecTE.class

db2sqljcustomize

-url jdbc:db2://mvs84.folksam.se:5144/SEFOLKDB2

-user lelle03

-password jättemhemligt

-bindoptions "ISOLATION(CS) PATH(LELLE03) VERSION(TEST)"

-collection S70COLL

-singlepkgname SPSECTE

SpSecTE_SJProfile0

Skapar paketet S70COLL.SPSECTE(TEST) i DB2