I have a query

```
SELECT  JOB_ID,
  PHYSICAL_DATASET,
  LOGICAL_DATASET,
  FACT_BREAK,
  ENTERPRISE_ID,
  BREAK_ID,
  SOURCE_TIMESTAMP,
  DATAFEED_BREAK,
  ETL_TRANSACTION,
  TARGET_STATE,
  Dataset_Type,
  Target_Dimension,
  Target_Category,
  --Dimension_Context,
  Context_ID,
  Reference_Composite,
  Dimension_ID,
  Control_Key,
  Placeholder_Flag,
  Unknown_Flag,
  Orphan_Flag,
  --SOURCE_COLUMN,
  --SOURCE_VALUE,
  CONTEXT_KEY_01,
  CONTEXT_KEY_02,
  CONTEXT_KEY_03,
  CONTEXT_KEY_04,
  CONTEXT_KEY_05,
  CONTEXT_KEY_06,
  CONTEXT_KEY_07,
  CONTEXT_KEY_08,
  CONTEXT_KEY_09,
  CONTEXT_KEY_10
FROM (
SELECT  JOB_ID,
  PHYSICAL_DATASET,
  LOGICAL_DATASET,
  FACT_BREAK,
  ENTERPRISE_ID,
  BREAK_ID,
  SOURCE_TIMESTAMP,
  DATAFEED_BREAK,
  ETL_TRANSACTION,
  TARGET_STATE,
  Dataset_Type,
  Target_Dimension,
  Target_Category,
  Dimension_Context,
  Context_ID,
  Reference_Composite,
  Dimension_ID,
  Control_Key,
  Placeholder_Flag,
  Unknown_Flag,
```

```
 Orphan_Flag,
 --SOURCE_COLUMN,
 SOURCE_VALUE
FROM ..zSource1_Reference ) x
PIVOT
 (
 MIN(SOURCE_VALUE)
 FOR X.Dimension_Context IN (
        Context_Key_01,
        Context_Key_02,
        Context_Key_03,
        Context_Key_04,
        Context_Key_05,
        Context_Key_06,
        Context_Key_07,
        Context_Key_08,
        Context_Key_09,
        Context_Key_10
       )
 ) p
```

That works perfectly in Talend SQL Builder but not in the actual Job:
Talend SQL Builder:

Pic of job:

Actual Code:

```
// ============================================================================
//
// Copyright (c) 2005-2009, Talend Inc.
//
// This source code has been automatically generated by Talend Open Studio
// / JobDesigner (CodeGenerator version 4.1.0.M3_r46036).
// You can find more information about Talend products at www.talend.com.
// You may distribute this code under the terms of the GNU LGPL license
// (http://www.gnu.org/licenses/lgpl.html).
//
// ============================================================================
package poc_star.xxxxxxtest_0_1;
import routines.DataOperation;
import routines.Mathematical;
import routines.Numeric;
import routines.Relational;
import routines.StringHandling;
import routines.TalendDataGenerator;
import routines.TalendDate;
import routines.TalendString;
import routines.Forum6748Routine;
import routines.system.*;
import java.text.ParseException;
import java.text.SimpleDateFormat;
import java.util.Date;
import java.util.List;
import java.math.BigDecimal;
```

```
import java.io.ByteArrayOutputStream;
import java.io.ByteArrayInputStream;
import java.io.DataInputStream;
import java.io.DataOutputStream;
import java.io.ObjectOutputStream;
import java.io.ObjectInputStream;
import java.io.IOException;
import java.util.Comparator;
/**
 * Job: xxxxxxtest Purpose: <br>
 * Description: <br>
 *
 * @author davidag1@uci.edu
 * @version 4.1.0.M3_r46036
 * @status
 */
public class xxxxxxtest {
 public final Object obj = new Object();
 // for transmiting parameters purpose
 private Object valueObject = null;
 public Object getValueObject() {
  return this.valueObject;
 }
 public void setValueObject(Object valueObject) {
  this.valueObject = valueObject;
 }
 private final static String defaultCharset = java.nio.charset.Charset
   .defaultCharset().name();
private final static String utf8Charset = "UTF-8";
// create and load default properties
private java.util.Properties defaultProps = new java.util.Properties();
// create application properties with default
public class ContextProperties extends java.util.Properties {
 public ContextProperties(java.util.Properties properties) {
  super(properties);
 }
 public ContextProperties() {
  super();
 }
 public void synchronizeContext() {
 }
}
private ContextProperties context = new ContextProperties();
public ContextProperties getContext() {
 return this.context;
}
private final String jobVersion = "0.1";
private final String jobName = "xxxxxxtest";
private final String projectName = "POC_STAR";
public Integer errorCode = null;
private String currentComponent = "";
private final java.util.Map<String, Long> start_Hash = new java.util.HashMap<String, Long>();
private final java.util.Map<String, Long> end_Hash = new java.util.HashMap<String, Long>();
private final java.util.Map<String, Boolean> ok_Hash = new java.util.HashMap<String, Boolean>();
private final java.util.Map<String, Object> globalMap = new java.util.HashMap<String, Object>();
public final java.util.List<String[]> globalBuffer = new java.util.ArrayList<String[]>();
```

```
private final java.io.ByteArrayOutputStream baos = new java.io.ByteArrayOutputStream();
private final java.io.PrintStream errorMessagePS = new java.io.PrintStream(
  new java.io.BufferedOutputStream(baos));
public String getExceptionStackTrace() {
 if ("failure".equals(this.getStatus())) {
  errorMessagePS.flush();
  return baos.toString();
 }
 return null;
}
private Exception exception = null;
public Exception getException() {
 if ("failure".equals(this.getStatus())) {
  return this.exception;
 }
 return null;
}
private class TalendException extends Exception {
 private java.util.Map<String, Object> globalMap = null;
 private Exception e = null;
 private String currentComponent = null;
 private TalendException(Exception e, String errorComponent,
   final java.util.Map<String, Object> globalMap) {
  this.currentComponent = errorComponent;
  this.globalMap = globalMap;
  this.e = e;
 }
 @Override
public void printStackTrace() {
 if (!(e instanceof TalendException || e instanceof TDieException)) {
  globalMap.put(currentComponent + "_ERROR_MESSAGE", e
    .getMessage());
  System.err
    .println("Exception in component " + currentComponent);
 }
 if (!(e instanceof TDieException)) {
  if (e instanceof TalendException) {
   e.printStackTrace();
  } else {
   e.printStackTrace();
   e.printStackTrace(errorMessagePS);
   xxxxxxtest.this.exception = e;
  }
 }
 if (!(e instanceof TalendException)) {
  try {
   for (java.lang.reflect.Method m : this.getClass()
     .getEnclosingClass().getMethods()) {
    if (m.getName().compareTo(currentComponent + "_error") == 0) {
     m.invoke(xxxxxxtest.this, new Object[] { e,
       currentComponent, globalMap });
     break;
    }
   }
   if (!(e instanceof TDieException)) {
   }
```

```
   } catch (java.lang.SecurityException e) {
    this.e.printStackTrace();
   } catch (java.lang.IllegalArgumentException e) {
    this.e.printStackTrace();
   } catch (java.lang.IllegalAccessException e) {
    this.e.printStackTrace();
   } catch (java.lang.reflect.InvocationTargetException e) {
    this.e.printStackTrace();
   }
  }
 }
}
public void tMSSqlInput_1_error(Exception exception, String errorComponent,
  final java.util.Map<String, Object> globalMap)
  throws TalendException {
 end_Hash.put("tMSSqlInput_1", System.currentTimeMillis());
 tMSSqlInput_1_onSubJobError(exception, errorComponent, globalMap);
}
public void tMSSqlInput_1_onSubJobError(Exception exception,
  String errorComponent, final java.util.Map<String, Object> globalMap)
  throws TalendException {
 resumeUtil.addLog("SYSTEM_LOG", "NODE:" + errorComponent, "", Thread
   .currentThread().getId()
   + "", "FATAL", "", exception.getMessage(), ResumeUtil
   .getExceptionStackTrace(exception), "");
}
public void tMSSqlInput_1Process(
  final java.util.Map<String, Object> globalMap)
  throws TalendException {
globalMap.put("tMSSqlInput_1_SUBPROCESS_STATE", 0);
final boolean execStat = this.execStat;
String iterateId = "";
String currentComponent = "";
try {
 String currentMethodName = new Exception().getStackTrace()
   .getMethodName();
 boolean resumeIt = currentMethodName.equals(resumeEntryMethodName);
 if (resumeEntryMethodName == null || resumeIt || globalResumeTicket) {// start
                   // the
                   // resume
  globalResumeTicket = true;
  /**
   *  start
   */
  ok_Hash.put("tMSSqlInput_1", false);
  start_Hash.put("tMSSqlInput_1", System.currentTimeMillis());
  currentComponent = "tMSSqlInput_1";
  int tos_count_tMSSqlInput_1 = 0;
  int nb_line_tMSSqlInput_1 = 0;
  java.sql.Connection conn_tMSSqlInput_1 = null;
  java.lang.Class.forName("net.sourceforge.jtds.jdbc.Driver");
  String port_tMSSqlInput_1 = "1433";
  String dbname_tMSSqlInput_1 = "UCI_STAR_POC";
  String url_tMSSqlInput_1 = "jdbc:jtds:sqlserver://"
    + "datawhs01.hs.uci.edu";
  if (!"".equals(port_tMSSqlInput_1)) {
```

```
 url_tMSSqlInput_1 += ":" + "1433";
}
if (!"".equals(dbname_tMSSqlInput_1)) {
 url_tMSSqlInput_1 += "//" + "UCI_STAR_POC";
}
url_tMSSqlInput_1 += ";appName=" + projectName + ";" + "";
String dbschema_tMSSqlInput_1 = "";
String dbUser_tMSSqlInput_1 = "dwmgr";
String dbPwd_tMSSqlInput_1 = "dwmgr_pwd";
conn_tMSSqlInput_1 = java.sql.DriverManager.getConnection(
  url_tMSSqlInput_1, dbUser_tMSSqlInput_1,
  dbPwd_tMSSqlInput_1);
java.sql.Statement stmt_tMSSqlInput_1 = conn_tMSSqlInput_1
  .createStatement();
String dbquery_tMSSqlInput_1 = "SELECT  JOB_ID,    PHYSICAL_DATASET,    LOGICAL_DATASET,    FACT_BREAK,    ENTERPRISE_ID,    BREAK_ID,    SOUF

java.sql.ResultSet rs_tMSSqlInput_1 = stmt_tMSSqlInput_1
  .executeQuery(dbquery_tMSSqlInput_1);
java.sql.ResultSetMetaData rsmd_tMSSqlInput_1 = rs_tMSSqlInput_1
  .getMetaData();
int colQtyInRs_tMSSqlInput_1 = rsmd_tMSSqlInput_1
  .getColumnCount();
globalMap.put("tMSSqlInput_1_QUERY", dbquery_tMSSqlInput_1);
String tmpContent_tMSSqlInput_1 = null;
while (rs_tMSSqlInput_1.next()) {
 nb_line_tMSSqlInput_1++;
 /**
  *  stop
  */
  /**
   *  start
   */
   currentComponent = "tMSSqlInput_1";
   tos_count_tMSSqlInput_1++;
   /**
    *  stop
    */
   /**
    *  start
    */
   currentComponent = "tMSSqlInput_1";
  }
  stmt_tMSSqlInput_1.close();
  conn_tMSSqlInput_1.close();
  globalMap.put("tMSSqlInput_1_NB_LINE", nb_line_tMSSqlInput_1);
  ok_Hash.put("tMSSqlInput_1", true);
  end_Hash.put("tMSSqlInput_1", System.currentTimeMillis());
  /**
   *  stop
   */
 }// end the resume
} catch (Exception e) {
 throw new TalendException(e, currentComponent, globalMap);
} catch (Error error) {
 throw new Error(error);
}
```

```
 globalMap.put("tMSSqlInput_1_SUBPROCESS_STATE", 1);
}
public String resuming_logs_dir_path = null;
public String resuming_checkpoint_path = null;
public String parent_part_launcher = null;
private String resumeEntryMethodName = null;
private boolean globalResumeTicket = false;
public boolean watch = false;
// portStats is null, it means don't execute the statistics
public Integer portStats = null;
public int portTraces = 4334;
public String clientHost;
public String defaultClientHost = "localhost";
public String contextStr = "Default";
public String pid = "0";
public String rootPid = null;
public String fatherPid = null;
public String fatherNode = null;
public long startTime = 0;
public boolean isChildJob = false;
private boolean execStat = true;
private ThreadLocal threadLocal = new ThreadLocal();
{
 java.util.Map threadRunResultMap = new java.util.HashMap();
 threadRunResultMap.put("errorCode", null);
 threadRunResultMap.put("status", "");
 threadLocal.set(threadRunResultMap);
}
private java.util.Properties context_param = new java.util.Properties();
public java.util.Map<String, Object> parentContextMap = new java.util.HashMap<String, Object>();
public String status = "";
public static void main(String[] args) {
 final xxxxxxtest xxxxxxtestClass = new xxxxxxtest();
 int exitCode = xxxxxxtestClass.runJobInTOS(args);
 System.exit(exitCode);
}
public String[][] runJob(String[] args) {
 int exitCode = runJobInTOS(args);
 String[][] bufferValue = new String[][] { { Integer.toString(exitCode) } };
 return bufferValue;
}
public int runJobInTOS(String[] args) {
 String lastStr = "";
 for (String arg : args) {
  if (arg.equalsIgnoreCase("--context_param")) {
   lastStr = arg;
  } else if (lastStr.equals("")) {
   evalParam(arg);
  } else {
   evalParam(lastStr + " " + arg);
   lastStr = "";
  }
 }
 if (clientHost == null) {
  clientHost = defaultClientHost;
 }
```

```
if (pid == null || "0".equals(pid)) {
 pid = TalendString.getAsciiRandomString(6);
}
if (rootPid == null) {
 rootPid = pid;
}
if (fatherPid == null) {
 fatherPid = pid;
} else {
 isChildJob = true;
}
try {
 // call job/subjob with an existing context, like:
 // --context=production. if without this parameter, there will use
 // the default context instead.
 java.io.InputStream inContext = xxxxxxtest.class.getClassLoader()
   .getResourceAsStream(
     "poc_star/xxxxxxtest_0_1/contexts/" + contextStr
       + ".properties");
 if (inContext != null) {
  // defaultProps is in order to keep the original context value
  defaultProps.load(inContext);
  inContext.close();
  context = new ContextProperties(defaultProps);
 } else {
  // print info and job continue to run, for case: context_param
  // is not empty.
  System.err.println("Could not find the context " + contextStr);
 }
 if (!context_param.isEmpty()) {
  context.putAll(context_param);
 }
} catch (java.io.IOException ie) {
 System.err.println("Could not load context " + contextStr);
 ie.printStackTrace();
}
// get context value from parent directly
if (parentContextMap != null && !parentContextMap.isEmpty()) {
}
// Resume: init the resumeUtil
resumeEntryMethodName = ResumeUtil
  .getResumeEntryMethodName(resuming_checkpoint_path);
resumeUtil = new ResumeUtil(resuming_logs_dir_path, isChildJob, rootPid);
resumeUtil.initCommonInfo(pid, rootPid, fatherPid, projectName,
  jobName, contextStr, jobVersion);
// Resume: jobStart
resumeUtil.addLog("JOB_STARTED", "JOB:" + jobName,
  parent_part_launcher, Thread.currentThread().getId() + "", "",
  "", "", "", resumeUtil.convertToJsonText(context));
long startUsedMemory = Runtime.getRuntime().totalMemory()
  - Runtime.getRuntime().freeMemory();
long endUsedMemory = 0;
long end = 0;
startTime = System.currentTimeMillis();
this.globalResumeTicket = true;// to run tPreJob
this.globalResumeTicket = false;// to run others jobs
```

```
try {
 errorCode = null;
 tMSSqlInput_1Process(globalMap);
 status = "end";
} catch (TalendException e_tMSSqlInput_1) {
 status = "failure";
 e_tMSSqlInput_1.printStackTrace();
 globalMap.put("tMSSqlInput_1_SUBPROCESS_STATE", -1);
} finally {
}
this.globalResumeTicket = true;// to run tPostJob
end = System.currentTimeMillis();
if (watch) {
 System.out.println((end - startTime) + " milliseconds");
}
endUsedMemory = Runtime.getRuntime().totalMemory()
  - Runtime.getRuntime().freeMemory();
if (false) {
 System.out.println((endUsedMemory - startUsedMemory)
   + " bytes memory increase when running : xxxxxxtest");
}
int returnCode = 0;
if (errorCode == null) {
 returnCode = status != null && status.equals("failure") ? 1 : 0;
} else {
 returnCode = errorCode.intValue();
}
resumeUtil.addLog("JOB_ENDED", "JOB:" + jobName, parent_part_launcher,
   Thread.currentThread().getId() + "", "", "" + returnCode, "",
   "", "");
 return returnCode;
}
private void evalParam(String arg) {
 if (arg.startsWith("--resuming_logs_dir_path")) {
  resuming_logs_dir_path = arg.substring(25);
 } else if (arg.startsWith("--resuming_checkpoint_path")) {
  resuming_checkpoint_path = arg.substring(27);
 } else if (arg.startsWith("--parent_part_launcher")) {
  parent_part_launcher = arg.substring(23);
 } else if (arg.startsWith("--watch")) {
  watch = true;
 } else if (arg.startsWith("--stat_port=")) {
  String portStatsStr = arg.substring(12);
  if (portStatsStr != null && !portStatsStr.equals("null")) {
   portStats = Integer.parseInt(portStatsStr);
  }
 } else if (arg.startsWith("--trace_port=")) {
  portTraces = Integer.parseInt(arg.substring(13));
 } else if (arg.startsWith("--client_host=")) {
  clientHost = arg.substring(14);
 } else if (arg.startsWith("--context=")) {
  contextStr = arg.substring(10);
 } else if (arg.startsWith("--father_pid=")) {
  fatherPid = arg.substring(13);
 } else if (arg.startsWith("--root_pid=")) {
  rootPid = arg.substring(11);
```

```
  } else if (arg.startsWith("--father_node=")) {
   fatherNode = arg.substring(14);
  } else if (arg.startsWith("--pid=")) {
   pid = arg.substring(6);
  } else if (arg.startsWith("--context_param")) {
   String keyValue = arg.substring(16);
   int index = -1;
   if (keyValue != null && (index = keyValue.indexOf('=')) > -1) {
    context_param.put(keyValue.substring(0, index), keyValue
      .substring(index + 1));
   }
  }
 }
 public Integer getErrorCode() {
  return errorCode;
 }
 public String getStatus() {
  return status;
 }
 ResumeUtil resumeUtil = null;
}
/************************************************************************************************
 * 18094 characters generated by Talend Open Studio on the June 14, 2011 8:52:21
 * PM PDT
 ************************************************************************************************/
```

Any idea why this fails?
The backend is MSSQL 2008
Thank you,
David