You should be able to switch to the code screen and hit ctrl+F to bring up a find box. You can edit the code, but you can search it. The code you copied and pasted isn't all of it for the tMSSqlOutput. There is a lot more. Here is an example I quickly knocked up. This isn't configured for use, but just dumped onto a job and configured to carry out an update. You will notice there is a section of code like this....

```
String update_tMSSqlOutput_1 = "UPDATE ["
      + tableName_tMSSqlOutput_1
      + "] SET [newColumn] = ?,[newColumn1] = ?,[newColumn2] = ?,[newColumn3] = ?,[newColumn4] = ? WHERE [newColumn3] = ? AND [newColumn4] = ?";
```

I've highlighted in bold what you should search for in your job. I assume you are using Talend 6.something? If so this code will be there if you have your component configured for an update.

```
public void tMSSqlOutput_1Process(
   final java.util.Map<String, Object> globalMap)
   throws TalendException {
  globalMap.put("tMSSqlOutput_1_SUBPROCESS_STATE", 0);

  final boolean execStat = this.execStat;

  String iterateId = "";

  String currentComponent = "";
  java.util.Map<String, Object> resourceMap = new java.util.HashMap<String, Object>();

  try {
String currentMethodName = new java.lang.Exception()
  .getStackTrace()[0].getMethodName();
boolean resumeIt = currentMethodName.equals(resumeEntryMethodName);
if (resumeEntryMethodName == null || resumeIt || globalResumeTicket) {// start
                  // the
                  // resume
 globalResumeTicket = true;

 /**
  * [tMSSqlOutput_1 begin ] start
  */

 ok_Hash.put("tMSSqlOutput_1", false);
 start_Hash.put("tMSSqlOutput_1", System.currentTimeMillis());

 currentComponent = "tMSSqlOutput_1";

 int tos_count_tMSSqlOutput_1 = 0;

 if (log.isDebugEnabled())
  log.debug("tMSSqlOutput_1 - " + ("Start to work."));
 class BytesLimit65535_tMSSqlOutput_1 {
  public void limitLog4jByte() throws Exception {

   StringBuilder log4jParamters_tMSSqlOutput_1 = new StringBuilder();
   log4jParamters_tMSSqlOutput_1.append("Parameters:");
   log4jParamters_tMSSqlOutput_1
```

```
  .append("USE_EXISTING_CONNECTION" + " = "
    + "false");
log4jParamters_tMSSqlOutput_1.append(" | ");
log4jParamters_tMSSqlOutput_1.append("DRIVER" + " = "
  + "JTDS");
log4jParamters_tMSSqlOutput_1.append(" | ");
log4jParamters_tMSSqlOutput_1.append("HOST" + " = "
  + "\"\"");
log4jParamters_tMSSqlOutput_1.append(" | ");
log4jParamters_tMSSqlOutput_1.append("PORT" + " = "
  + "\"1433\"");
log4jParamters_tMSSqlOutput_1.append(" | ");
log4jParamters_tMSSqlOutput_1.append("DB_SCHEMA"
  + " = " + "\"\"");
log4jParamters_tMSSqlOutput_1.append(" | ");
log4jParamters_tMSSqlOutput_1.append("DBNAME" + " = "
  + "\"\"");
log4jParamters_tMSSqlOutput_1.append(" | ");
log4jParamters_tMSSqlOutput_1.append("USER" + " = "
  + "\"\"");
log4jParamters_tMSSqlOutput_1.append(" | ");
log4jParamters_tMSSqlOutput_1.append("PASS"
  + " = "
  + String.valueOf("f4f7aba1746784ea").substring(
    0, 4) + "...");
log4jParamters_tMSSqlOutput_1.append(" | ");
log4jParamters_tMSSqlOutput_1.append("TABLE" + " = "
  + "\"\"");
log4jParamters_tMSSqlOutput_1.append(" | ");
log4jParamters_tMSSqlOutput_1.append("TABLE_ACTION"
  + " = " + "NONE");
log4jParamters_tMSSqlOutput_1.append(" | ");
log4jParamters_tMSSqlOutput_1.append("IDENTITY_INSERT"
  + " = " + "false");
log4jParamters_tMSSqlOutput_1.append(" | ");
log4jParamters_tMSSqlOutput_1.append("DATA_ACTION"
  + " = " + "UPDATE");
log4jParamters_tMSSqlOutput_1.append(" | ");
log4jParamters_tMSSqlOutput_1
  .append("SPECIFY_DATASOURCE_ALIAS" + " = "
    + "false");
log4jParamters_tMSSqlOutput_1.append(" | ");
log4jParamters_tMSSqlOutput_1.append("DIE_ON_ERROR"
  + " = " + "false");
log4jParamters_tMSSqlOutput_1.append(" | ");
log4jParamters_tMSSqlOutput_1.append("PROPERTIES"
  + " = " + "\"\"");
log4jParamters_tMSSqlOutput_1.append(" | ");
log4jParamters_tMSSqlOutput_1.append("COMMIT_EVERY"
  + " = " + "10000");
log4jParamters_tMSSqlOutput_1.append(" | ");
log4jParamters_tMSSqlOutput_1.append("ADD_COLS" + " = "
  + "[]");
log4jParamters_tMSSqlOutput_1.append(" | ");
log4jParamters_tMSSqlOutput_1
  .append("USE_FIELD_OPTIONS" + " = " + "true");
```

```
log4jParamters_tMSSqlOutput_1.append(" | ");
log4jParamters_tMSSqlOutput_1.append("FIELD_OPTIONS"
  + " = " + "[{UPDATE_KEY=" + ("false")
  + ", DELETE_KEY=" + ("false") + ", UPDATABLE="
  + ("true") + ", INSERTABLE=" + ("true")
  + ", SCHEMA_COLUMN=" + ("newColumn")
  + "}, {UPDATE_KEY=" + ("false")
  + ", DELETE_KEY=" + ("false") + ", UPDATABLE="
  + ("true") + ", INSERTABLE=" + ("true")
  + ", SCHEMA_COLUMN=" + ("newColumn1")
  + "}, {UPDATE_KEY=" + ("false")
  + ", DELETE_KEY=" + ("false") + ", UPDATABLE="
  + ("true") + ", INSERTABLE=" + ("true")
  + ", SCHEMA_COLUMN=" + ("newColumn2")
  + "}, {UPDATE_KEY=" + ("true")
  + ", DELETE_KEY=" + ("false") + ", UPDATABLE="
  + ("true") + ", INSERTABLE=" + ("true")
  + ", SCHEMA_COLUMN=" + ("newColumn3")
  + "}, {UPDATE_KEY=" + ("true")
  + ", DELETE_KEY=" + ("false") + ", UPDATABLE="
  + ("true") + ", INSERTABLE=" + ("true")
  + ", SCHEMA_COLUMN=" + ("newColumn4") + "}]");
log4jParamters_tMSSqlOutput_1.append(" | ");
log4jParamters_tMSSqlOutput_1
  .append("IGNORE_DATE_OUTOF_RANGE" + " = "
    + "false");
log4jParamters_tMSSqlOutput_1.append(" | ");
log4jParamters_tMSSqlOutput_1
    .append("ENABLE_DEBUG_MODE" + " = " + "false");
  log4jParamters_tMSSqlOutput_1.append(" | ");
  log4jParamters_tMSSqlOutput_1
    .append("SUPPORT_NULL_WHERE" + " = " + "false");
  log4jParamters_tMSSqlOutput_1.append(" | ");
  log4jParamters_tMSSqlOutput_1.append("USE_BATCH_SIZE"
    + " = " + "true");
  log4jParamters_tMSSqlOutput_1.append(" | ");
  log4jParamters_tMSSqlOutput_1.append("BATCH_SIZE"
    + " = " + "10000");
  log4jParamters_tMSSqlOutput_1.append(" | ");
  if (log.isDebugEnabled())
   log.debug("tMSSqlOutput_1 - "
     + (log4jParamters_tMSSqlOutput_1));
 }
}

new BytesLimit65535_tMSSqlOutput_1().limitLog4jByte();

int nb_line_tMSSqlOutput_1 = 0;
int nb_line_update_tMSSqlOutput_1 = 0;
int nb_line_inserted_tMSSqlOutput_1 = 0;
int nb_line_deleted_tMSSqlOutput_1 = 0;
int nb_line_rejected_tMSSqlOutput_1 = 0;

int deletedCount_tMSSqlOutput_1 = 0;
int updatedCount_tMSSqlOutput_1 = 0;
int insertedCount_tMSSqlOutput_1 = 0;
```

```
int rejectedCount_tMSSqlOutput_1 = 0;
String dbschema_tMSSqlOutput_1 = null;
String tableName_tMSSqlOutput_1 = null;
boolean whetherReject_tMSSqlOutput_1 = false;

java.util.Calendar calendar_tMSSqlOutput_1 = java.util.Calendar
  .getInstance();
long year1_tMSSqlOutput_1 = TalendDate.parseDate("yyyy-MM-dd",
  "0001-01-01").getTime();
long year2_tMSSqlOutput_1 = TalendDate.parseDate("yyyy-MM-dd",
  "1753-01-01").getTime();
long year10000_tMSSqlOutput_1 = TalendDate.parseDate(
  "yyyy-MM-dd HH:mm:ss", "9999-12-31 24:00:00").getTime();
long date_tMSSqlOutput_1;

java.util.Calendar calendar_datetimeoffset_tMSSqlOutput_1 = java.util.Calendar
  .getInstance(java.util.TimeZone.getTimeZone("UTC"));

int updateKeyCount_tMSSqlOutput_1 = 2;
if (updateKeyCount_tMSSqlOutput_1 < 1) {
 throw new RuntimeException(
   "For update, Schema must have a key");
}

java.sql.Connection conn_tMSSqlOutput_1 = null;
String dbUser_tMSSqlOutput_1 = null;
dbschema_tMSSqlOutput_1 = "";
String driverClass_tMSSqlOutput_1 = "net.sourceforge.jtds.jdbc.Driver";

if (log.isDebugEnabled())
 log.debug("tMSSqlOutput_1 - " + ("Driver ClassName: ")
   + (driverClass_tMSSqlOutput_1) + ("."));
java.lang.Class.forName(driverClass_tMSSqlOutput_1);
String port_tMSSqlOutput_1 = "1433";
String dbname_tMSSqlOutput_1 = "";
String url_tMSSqlOutput_1 = "jdbc:jtds:sqlserver://" + "";
if (!"".equals(port_tMSSqlOutput_1)) {
 url_tMSSqlOutput_1 += ":" + "1433";
}
if (!"".equals(dbname_tMSSqlOutput_1)) {
 url_tMSSqlOutput_1 += "//" + "";

}
url_tMSSqlOutput_1 += ";appName=" + projectName + ";" + "";
dbUser_tMSSqlOutput_1 = "";

final String decryptedPassword_tMSSqlOutput_1 = routines.system.PasswordEncryptUtil
  .decryptPassword("f4f7aba1746784ea");

String dbPwd_tMSSqlOutput_1 = decryptedPassword_tMSSqlOutput_1;
if (log.isDebugEnabled())
 log.debug("tMSSqlOutput_1 - "
   + ("Connection attempts to '")
   + (url_tMSSqlOutput_1) + ("' with the username '")
   + (dbUser_tMSSqlOutput_1) + ("'."));
conn_tMSSqlOutput_1 = java.sql.DriverManager.getConnection(
```

```
  url_tMSSqlOutput_1, dbUser_tMSSqlOutput_1,
  dbPwd_tMSSqlOutput_1);
if (log.isDebugEnabled())
 log.debug("tMSSqlOutput_1 - " + ("Connection to '")
   + (url_tMSSqlOutput_1) + ("' has succeeded."));

resourceMap.put("conn_tMSSqlOutput_1", conn_tMSSqlOutput_1);

conn_tMSSqlOutput_1.setAutoCommit(false);
int commitEvery_tMSSqlOutput_1 = 10000;
int commitCounter_tMSSqlOutput_1 = 0;

if (log.isDebugEnabled())
 log.debug("tMSSqlOutput_1 - "
   + ("Connection is set auto commit to '")
   + (conn_tMSSqlOutput_1.getAutoCommit()) + ("'."));
int batchSize_tMSSqlOutput_1 = 10000;
int batchSizeCounter_tMSSqlOutput_1 = 0;

if (dbschema_tMSSqlOutput_1 == null
  || dbschema_tMSSqlOutput_1.trim().length() == 0) {
 tableName_tMSSqlOutput_1 = "";
} else {
 tableName_tMSSqlOutput_1 = dbschema_tMSSqlOutput_1 + "].["
   + "";
}
int count_tMSSqlOutput_1 = 0;
String update_tMSSqlOutput_1 = "UPDATE ["
  + tableName_tMSSqlOutput_1
  + "] SET [newColumn] = ?,[newColumn1] = ?,[newColumn2] = ?,[newColumn3] = ?,[newColumn4] = ? WHERE [newColumn3] = ? AND [newColumn4] = ?";
java.sql.PreparedStatement pstmt_tMSSqlOutput_1 = conn_tMSSqlOutput_1
  .prepareStatement(update_tMSSqlOutput_1);

/**
 * [tMSSqlOutput_1 begin ] stop
 */

/**
 * [tMSSqlOutput_1 main ] start
 */

currentComponent = "tMSSqlOutput_1";

tos_count_tMSSqlOutput_1++;

/**
 * [tMSSqlOutput_1 main ] stop
 */

/**
 * [tMSSqlOutput_1 end ] start
 */

currentComponent = "tMSSqlOutput_1";
```

```
try {
 int countSum_tMSSqlOutput_1 = 0;
 if (pstmt_tMSSqlOutput_1 != null
   && batchSizeCounter_tMSSqlOutput_1 > 0) {

  if (log.isDebugEnabled())
   log.debug("tMSSqlOutput_1 - " + ("Executing the ")
     + ("UPDATE") + (" batch."));
  for (int countEach_tMSSqlOutput_1 : pstmt_tMSSqlOutput_1
    .executeBatch()) {
   if (countEach_tMSSqlOutput_1 == -2
     || countEach_tMSSqlOutput_1 == -3) {
    break;
   }
   countSum_tMSSqlOutput_1 += countEach_tMSSqlOutput_1;
  }

  if (log.isDebugEnabled())
   log.debug("tMSSqlOutput_1 - " + ("The ")
     + ("UPDATE")
     + (" batch execution has succeeded."));
 }

 updatedCount_tMSSqlOutput_1 += countSum_tMSSqlOutput_1;

} catch (java.sql.BatchUpdateException e) {

 int countSum_tMSSqlOutput_1 = 0;
 for (int countEach_tMSSqlOutput_1 : e.getUpdateCounts()) {
  countSum_tMSSqlOutput_1 += (countEach_tMSSqlOutput_1 < 0 ? 0
    : countEach_tMSSqlOutput_1);
 }

 updatedCount_tMSSqlOutput_1 += countSum_tMSSqlOutput_1;

 log.error("tMSSqlOutput_1 - " + (e.getMessage()));
 System.err.println(e.getMessage());

}
if (pstmt_tMSSqlOutput_1 != null) {

 pstmt_tMSSqlOutput_1.close();

}

if (log.isDebugEnabled())
 log.debug("tMSSqlOutput_1 - "
   + ("Connection starting to commit ")
   + (commitCounter_tMSSqlOutput_1) + (" record(s)."));
conn_tMSSqlOutput_1.commit();

if (log.isDebugEnabled())
 log.debug("tMSSqlOutput_1 - "
   + ("Connection commit has succeeded."));
if (log.isDebugEnabled())
 log.debug("tMSSqlOutput_1 - "
```

```
   + ("Closing the connection to the database."));
conn_tMSSqlOutput_1.close();
if (log.isDebugEnabled())
 log.debug("tMSSqlOutput_1 - "
   + ("Connection to the database has closed."));
resourceMap.put("finish_tMSSqlOutput_1", true);

nb_line_deleted_tMSSqlOutput_1 = nb_line_deleted_tMSSqlOutput_1
  + deletedCount_tMSSqlOutput_1;
nb_line_update_tMSSqlOutput_1 = nb_line_update_tMSSqlOutput_1
  + updatedCount_tMSSqlOutput_1;
nb_line_inserted_tMSSqlOutput_1 = nb_line_inserted_tMSSqlOutput_1
  + insertedCount_tMSSqlOutput_1;
nb_line_rejected_tMSSqlOutput_1 = nb_line_rejected_tMSSqlOutput_1
  + rejectedCount_tMSSqlOutput_1;

globalMap.put("tMSSqlOutput_1_NB_LINE", nb_line_tMSSqlOutput_1);
globalMap.put("tMSSqlOutput_1_NB_LINE_UPDATED",
  nb_line_update_tMSSqlOutput_1);
globalMap.put("tMSSqlOutput_1_NB_LINE_INSERTED",
  nb_line_inserted_tMSSqlOutput_1);
globalMap.put("tMSSqlOutput_1_NB_LINE_DELETED",
  nb_line_deleted_tMSSqlOutput_1);
globalMap.put("tMSSqlOutput_1_NB_LINE_REJECTED",
  nb_line_rejected_tMSSqlOutput_1);

if (log.isDebugEnabled())
 log.debug("tMSSqlOutput_1 - " + ("Has ") + ("updated")
    + (" ") + (nb_line_update_tMSSqlOutput_1)
    + (" record(s)."));

  if (log.isDebugEnabled())
   log.debug("tMSSqlOutput_1 - " + ("Done."));

  ok_Hash.put("tMSSqlOutput_1", true);
  end_Hash.put("tMSSqlOutput_1", System.currentTimeMillis());

  /**
   * [tMSSqlOutput_1 end ] stop
   */
 }// end the resume

} catch (java.lang.Exception e) {

 if (!(e instanceof TalendException)) {
  log.fatal(currentComponent + " " + e.getMessage(), e);
 }

 TalendException te = new TalendException(e, currentComponent,
   globalMap);

 throw te;
} catch (java.lang.Error error) {

 runStat.stopThreadStat();
```

```
 throw error;
} finally {

 try {

  /**
   * [tMSSqlOutput_1 finally ] start
   */

  currentComponent = "tMSSqlOutput_1";

  if (resourceMap.get("finish_tMSSqlOutput_1") == null) {
   if (resourceMap.get("conn_tMSSqlOutput_1") != null) {
    try {

     if (log.isDebugEnabled())
      log.debug("tMSSqlOutput_1 - "
        + ("Closing the connection to the database."));

     java.sql.Connection ctn_tMSSqlOutput_1 = (java.sql.Connection) resourceMap
       .get("conn_tMSSqlOutput_1");

     ctn_tMSSqlOutput_1.close();

     if (log.isDebugEnabled())
      log.debug("tMSSqlOutput_1 - "
        + ("Connection to the database has closed."));
    } catch (java.sql.SQLException sqlEx_tMSSqlOutput_1) {
     String errorMessage_tMSSqlOutput_1 = "failed to close the connection in tMSSqlOutput_1 :"
       + sqlEx_tMSSqlOutput_1.getMessage();

     log.error("tMSSqlOutput_1 - "
       + (errorMessage_tMSSqlOutput_1));
     System.err.println(errorMessage_tMSSqlOutput_1);
    }
   }
  }

  /**
   * [tMSSqlOutput_1 finally ] stop
   */
 } catch (java.lang.Exception e) {
  // ignore
 } catch (java.lang.Error error) {
  // ignore
 }
 resourceMap = null;
}

globalMap.put("tMSSqlOutput_1_SUBPROCESS_STATE", 1);
}
```