IBM Data Studio debugger and IBM DB2 for i

Graphically debug your SQL stored procedures and functions

Kent Milligan August 29, 2013

IBM® Data Studio provides a graphical debugger that can be used with IBM DB2® for i SQL and Java™ stored procedures as well as SQL user-defined scalar functions. This powerful debugger can help you more quickly identify and resolve issues with your SQL code. This article shows how to configure and use the Data Studio debugger with DB2 for i procedural objects.

Overview

Starting with the IBM DB2 for i V5R4 release, IBM i developers have had the ability to graphically debug SQL and Java stored procedures. This capability is not known by many IBM i developers. These graphical capabilities are provided by the IBM Data Studio product (formerly known as IBM DB2 Developers Workbench and IBM DB2 Development Center). IBM Data Studio is a visual tool that supports the entire IBM DB2 family across all IBM platforms for the rapid development of DB2 business objects. The latest version of Data Studio (4.1.0) also adds support for DB2 for i SQL user-defined functions. This new support requires a recent version of the IBM i 6.1 or IBM i 7.1 Database Group PTF to be installed on your system.

Data Studio overview

IBM Data Studio is a graphical development tool that is based on the Eclipse technology. Data Studio supports a wide variety of features for the application developer from browsing and editing data in tables to creating and running SQL statements. This Data Studio website provides a complete listing of the Data Studio features that are supported for DB2 for i databases.

IBM i developers can download Data Studio from the Data Studio download website.

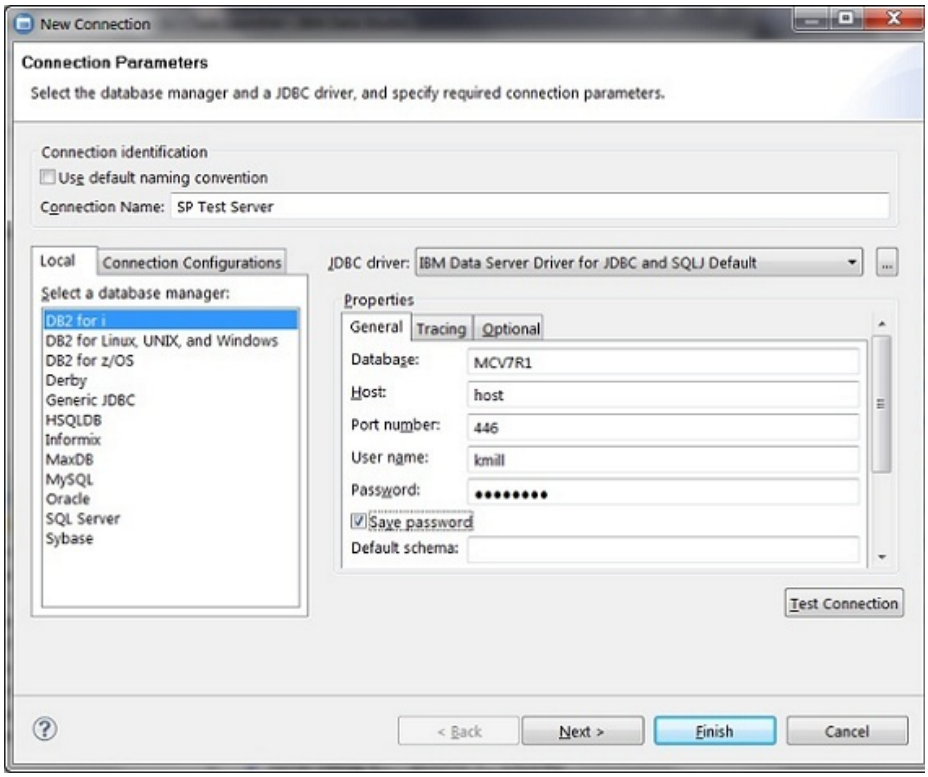
Data Studio connection setup

Setting up the connection properly is the key setup step for using the debug capabilities of Data Studio. The following steps guide you through the connection configuration process.

1. When you first launch Data Studio, you need to establish a new database connection for your development project. Connections are added in the Data Source Explorer view which is

normally located in the bottom left corner of the window. Adding a new database connection is done using the interface shown in Figure 1.

Figure 1. Connection setup window



2. Identify the type of DB2 server where your procedure resides. In this example, the blue highlight bar (shown on the left side of the window) indicates the selection of DB2 for i.
3. Select the JDBC driver on the middle right side of the window. This is an important step because the default driver, AS/400 Toolbox for Java, does not work if your goal is to use the Data Studio debugger. The Data Studio product and Toolbox JDBC driver work together well for editing and creating DB2 for i SQL objects, such as stored procedures, but not for stored procedure debugging. Instead, you must use the IBM Data Server Driver for JDBC and SQL (also known as the DB2 JCC or DB2 Universal JDBC driver) when debugging stored procedures. The IBM Data Server Driver for JDBC is also a Type-4 JDBC driver (as is the Toolbox JDBC driver). In contrast, the Data Server Driver for JDBC driver is not included with the IBM i operating systems, but it is included with the Data Studio download along with a DB2 Connect license that enables access of DB2 for i servers from Data Studio.
4. Identify the DB2 for i database server. You need to set the database entry field with the local database name on your server. You can obtain this database name by running the IBM i Display Database Relational Database Entries (DSPDBRDIRE) command and using the name of the *LOCAL entry. The host is the TCP/IP host name of the IBM i server that is being accessed. In this example, the local database and TCP/IP host name are the same.

Stored procedure setup

The Data Studio debugger requires your stored procedure to be created with a special attributes to make the procedure eligible to debug. You add these attributes to the procedure with the `ALLOW DEBUG MODE` clause or by using the `CURRENT DEBUG MODE` special register. The `*SQL DBGVIEW` option is not compatible with the Data Studio debugger. In fact, a syntax error is returned if a procedure specifies both the `ALLOW DEBUG MODE` clause and the `DBGVIEW` option.

Listing 1 contains an example of an SQL procedure that uses the `ALLOW DEBUG MODE` clause.

Listing 1. SQL procedure with ALLOW DEBUG MODE clause

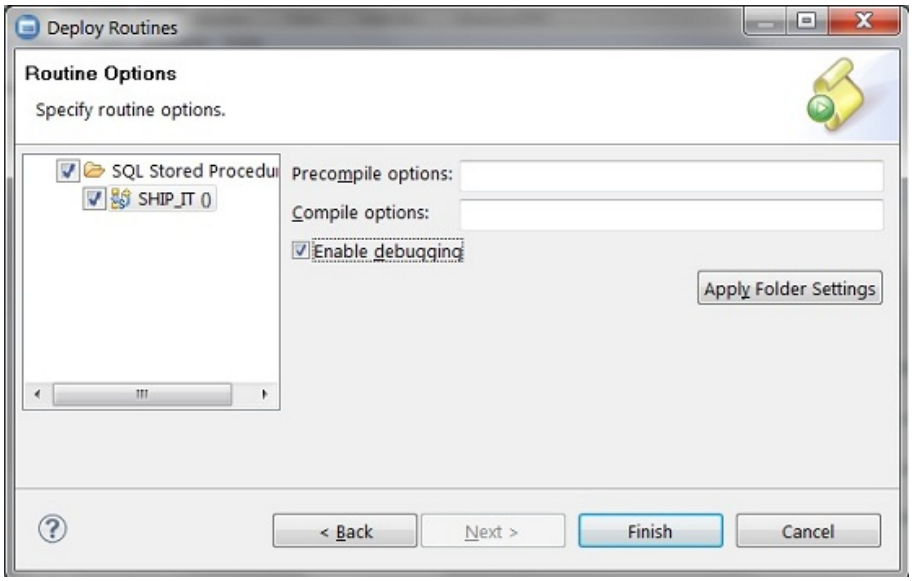
```
CREATE PROCEDURE myschema.ship_it(IN ordnum INTEGER, IN ordtype CHAR(1),
IN ordweight dec(3,2))
LANGUAGE SQL
ALLOW DEBUG MODE
sp: BEGIN
DECLARE ratecalc DECIMAL(5,2);

/* Check for international order */
IF ordtype='I' THEN
SET ratecalc = ordweight * 5.50;
INSERT INTO wwshipments VALUES(ordnum,ordweight,ratecalc);
ELSE
SET ratecalc = ordweight * 1.75;
INSERT INTO shipments values(ordnum,ordweight,ratecalc);
END IF;

END
```

The `CURRENT DEBUG MODE` special register provides a different alternative if you do not want to include debug clauses in your CREATE PROCEDURE statements. With this approach, you can drop the `ALLOW DEBUG MODE` clause from the CREATE PROCEDURE statement, as long as the special register is set to the ALLOW value prior to the procedure being created. Here is an example of setting the debug mode special register: `SET CURRENT DEBUG MODE=ALLOW` Setting the special register to ALLOW means that any routine created after this assignment is eligible to be debugged by the Data Studio debugger. When using the Data Studio to create a procedure, you can also enable the debug mode by selecting the **Enable Debugging** option on the Deploy Routines interface, as shown in Figure 2.

Figure 2. Deploy Routines window



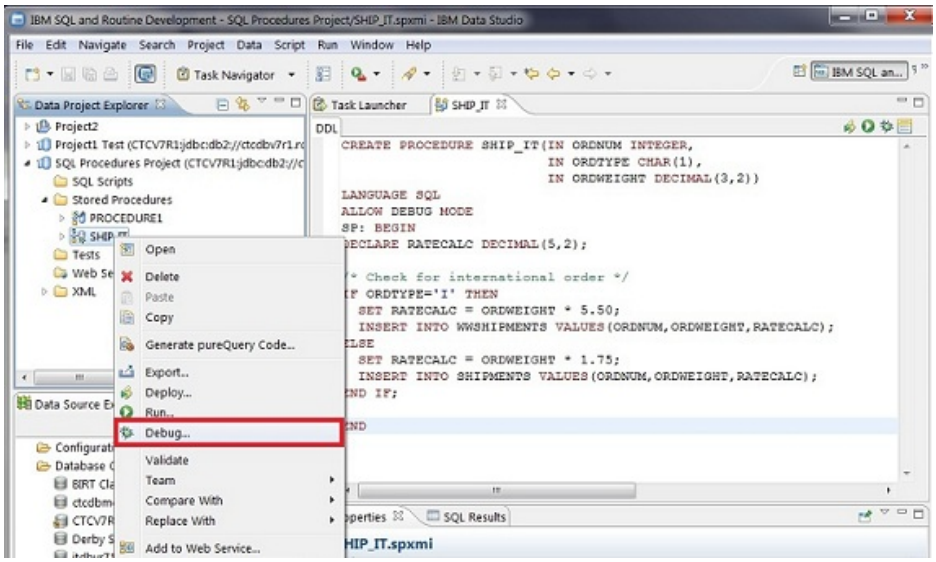
Selecting the **Enable debugging** check box causes Data Studio to set the debug mode special register to a value of ALLOW before creating the stored procedure on the selected server. This ensures that a debug view of the stored procedure exists on the server even if the ALLOW DEBUG MODE clause is missing from the stored procedure source code.

Starting the debugger

Before launching the debugger, you need to find your SQL stored procedure in the Data Project Explorer window in the upper-left corner (as shown in Figure 3). Double-clicking the procedure will result in the source code being displayed in the IBM SQL and Routine Development perspective on the right-hand side. This perspective allows you to edit, create, and run the SQL procedure.

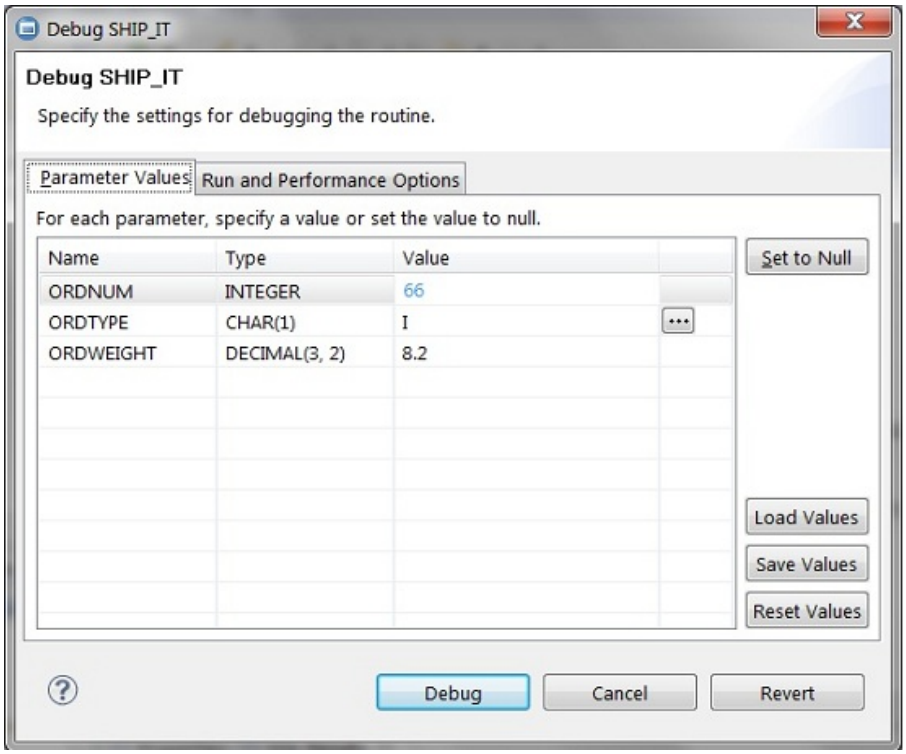
As highlighted in Figure 3, the easiest way to launch the debugger is by right-clicking the stored procedure in the Data Project Explorer window and clicking **Debug**.

Figure 3. Start debug interface



After the debug task is started for a stored procedure, the window (as shown in Figure 4) is displayed so that you can specify the debug settings for this routine invocation. The primary settings that can be changed on this interface are the values of the input parameters that will be passed on the call of the stored procedure. The **Run and Performance Options** tab has a setting that enables the changes made to the database to be automatically committed.

Figure 4. Debug routine settings window



If you want to change the input parameter values, click in the **Value** column for the parameter. From this interface, you can retrieve values from a file that you want to test, so that you do not have to manually enter these values each time you need to run and debug a specific test.

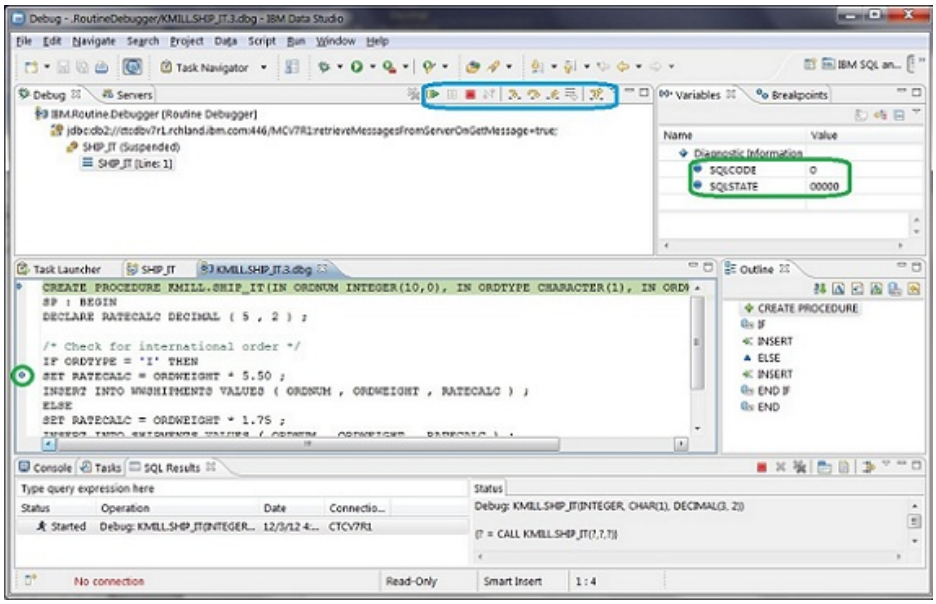
With the configuration now in place, the debugger is started by clicking **Debug**. After clicking Debug, Data Studio automatically switches to the Debug perspective that is displayed in Figure 5.

Using the debugger

The first line of the stored procedure is highlighted (as shown in Figure 5) when the Data Studio debugger has control of the stored procedure execution. The first time you run the Data Studio debugger on your system, it might take some time for the debugger to gain control, because a job (QSQDBGMGR) has to be started on the IBM i system to support Data Studio debug sessions.

After the Data Studio debugger has control, you can step through the procedure and set breakpoints. You set and clear breakpoints by double-clicking in the left margin. After you have set a breakpoint, a small dot is displayed in the left margin. The breakpoint indicator in Figure 5 is circled for your reference. The top line in the Debug view (highlighted in the Debug tab) is where you can control the execution of the stored procedure. Clicking the green arrow with the yellow bar icon causes the stored procedure to resume execution after a breakpoint has been reached. The icons with yellow arrows provide various step executions (for example, Step Into and Step Over). The Variables view in the upper-right corner (in the Variables tab) is where you can access the value of the procedure's variables. Because Figure 5 displays the initial debug view before the stored procedure starts running, the Variables view contains only two variables, SQLCODE and SQLSTATE, initialized to zero by DB2. As execution of the stored procedure progresses, the Data Studio debugger automatically adds variables to this view.

Figure 5. Debug perspective

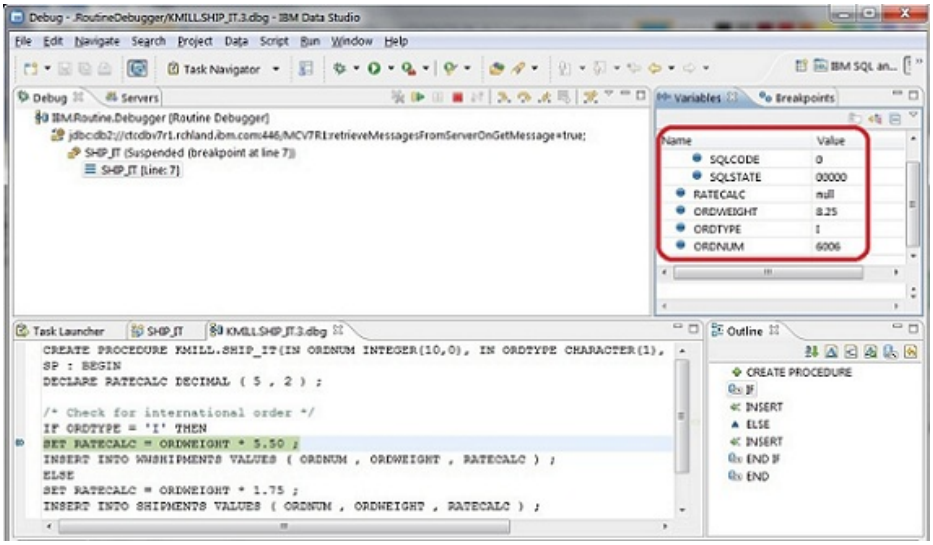


This dynamic nature of the Variables view, which is demonstrated in Figure 6, displays the Debug view after a breakpoint has been reached on the SET statement. Now that a couple of lines of

the stored procedure have run, you can see that the Variables view shows several more variables —including the values of the input parameters of the SQL procedure (ORDNUM, ORDWEIGHT, ORDNUM). You can also use the Variables view to set watches on variables or even change their values—you can access both of these functions by right-clicking the variable name. Monitoring and accessing the variables and parameters within a stored procedure is much easier with the Data Studio debugger, as compared to the IBM i graphical debugger.

Figure 6 also exhibits how the Data Studio debugger highlights the next line of code to be run. You can also deduce that a breakpoint has been reached from the left margin where an arrow points to the small circle that represents a defined breakpoint.

Figure 6. Variable perspective



SQL function considerations

Debugging SQL user-defined scalar functions with the Data Studio debugger is almost identical to debugging stored procedures. The only difference is that the debugger can only be used with user-defined scalar functions that specify the NOT FENCED and DISALLOW PARALLEL attributes. If your SQL function meets this requirement, then simply follow the same debug steps that were covered for an SQL procedure.

With this detailed introduction complete, you should now be ready to speed up your problem determination process for stored procedures by using the IBM Data Studio debugger. Happy debugging!

Related topics

- IBM Data Studio.
- Review the latest DB2 for i enhancements in the DB2 for i Technology Updates wiki.