

 #SHAREorg



Application Development With Data Studio

Tony Leung

IBM

February 4, 2013

13087

leungtk@us.ibm.com

Insert
Custom
Session
QR if
Desired.

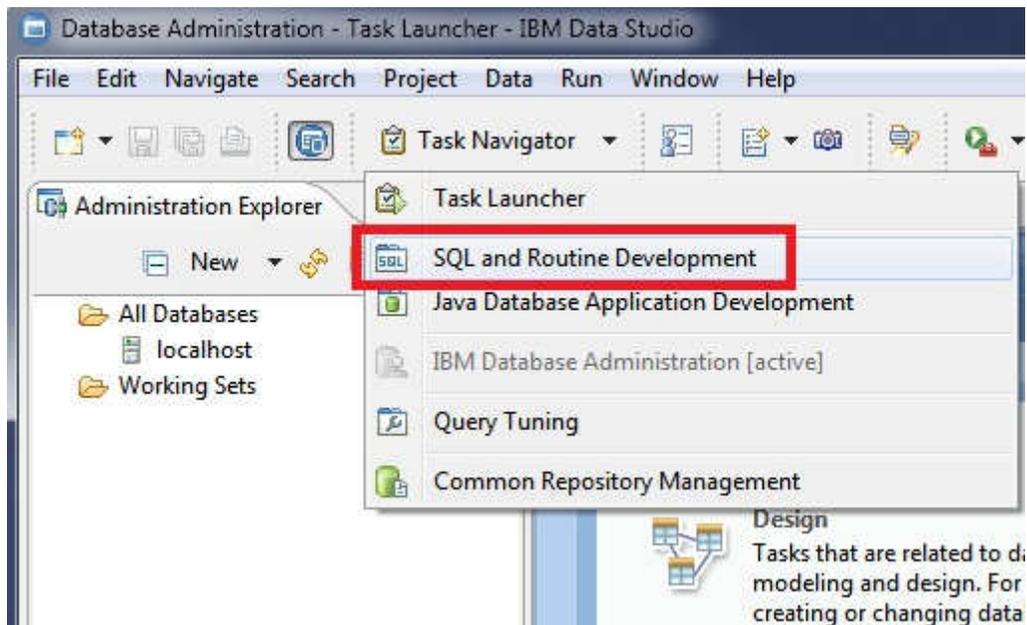


Developing Application

- Application Development
 - Stored Procedures
 - Java Applications
- Query Tuner Integration
 - SQL Formatting
 - Query Tuner Invocation in Context

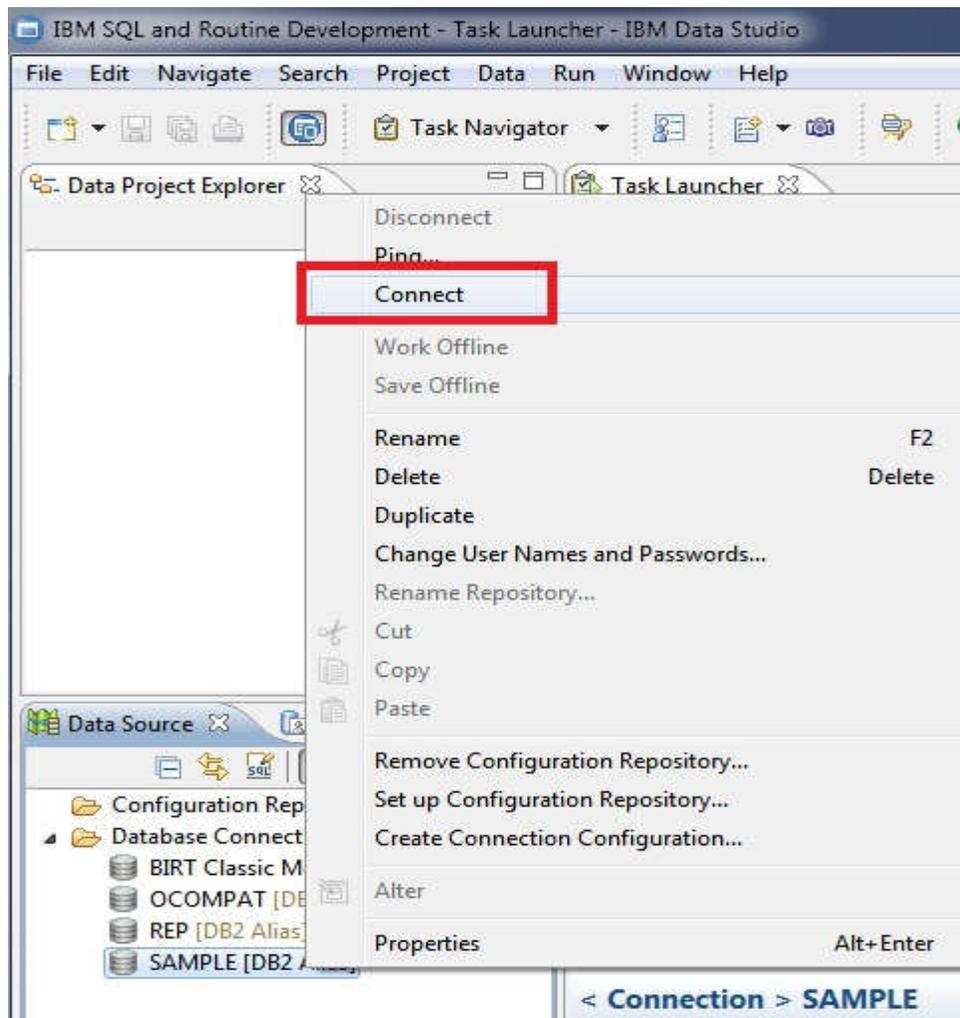
Developing Stored Procedures

Launch Data Studio and switch to Routine Development



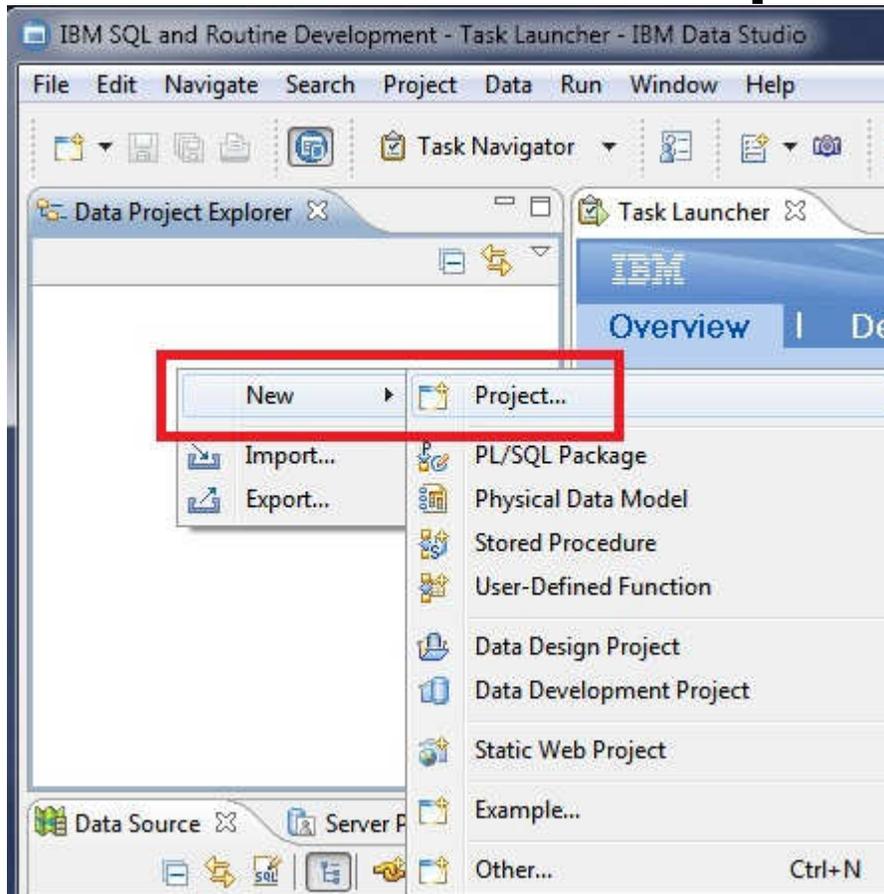
- Data Studio has different perspectives for developers and administrators
- Switching to Routine Development perspective gives you the views necessary for routine development

Connect to the database



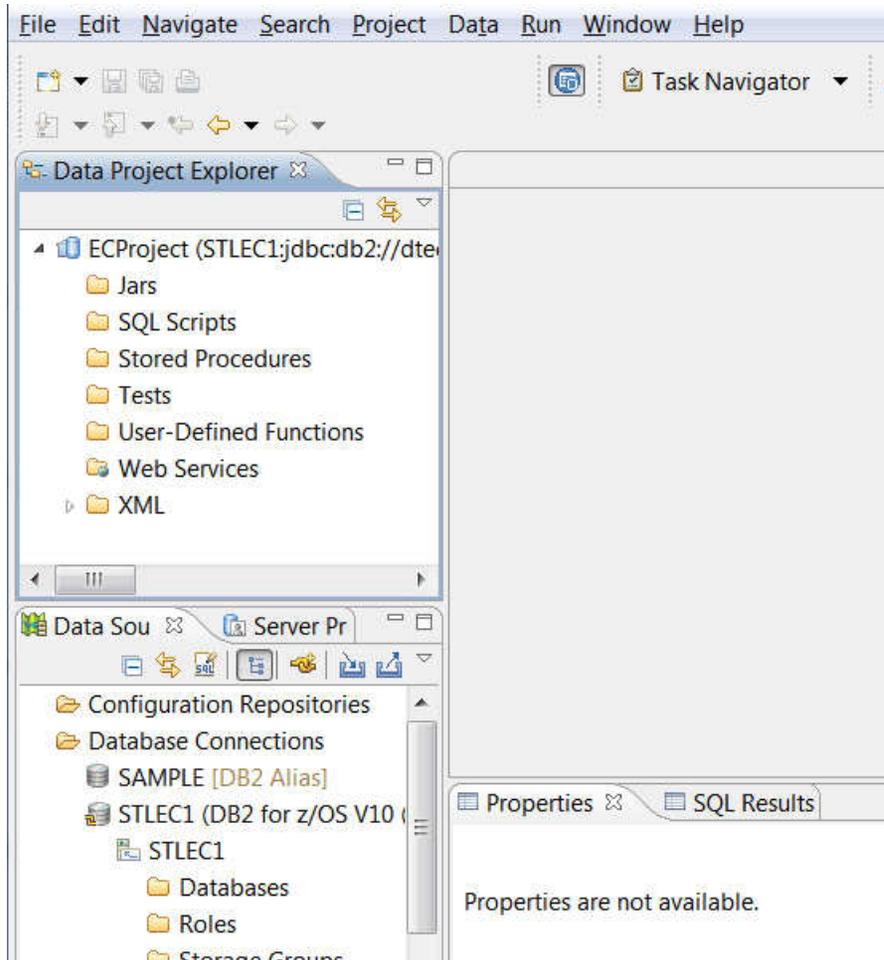
- Connecting to the database allows Data Studio to detect the version of the database
- Data Studio uses the database version in the connection profile to determine the capability of the database
- A database alias does not contain version information unless it has been used to connect to the database at least once

Create a Data Development Project



- Create a Data Development Project to hold your source
- Users can also edit/run/debug existing routines from Data Source Explorer or Administration Explorer in Data Studio
- Create a project if you want to keep track of your routines without a connection or you want source control integration

Data Development Project for DB2 Z



- Data Development Project has different folders depending on database platforms

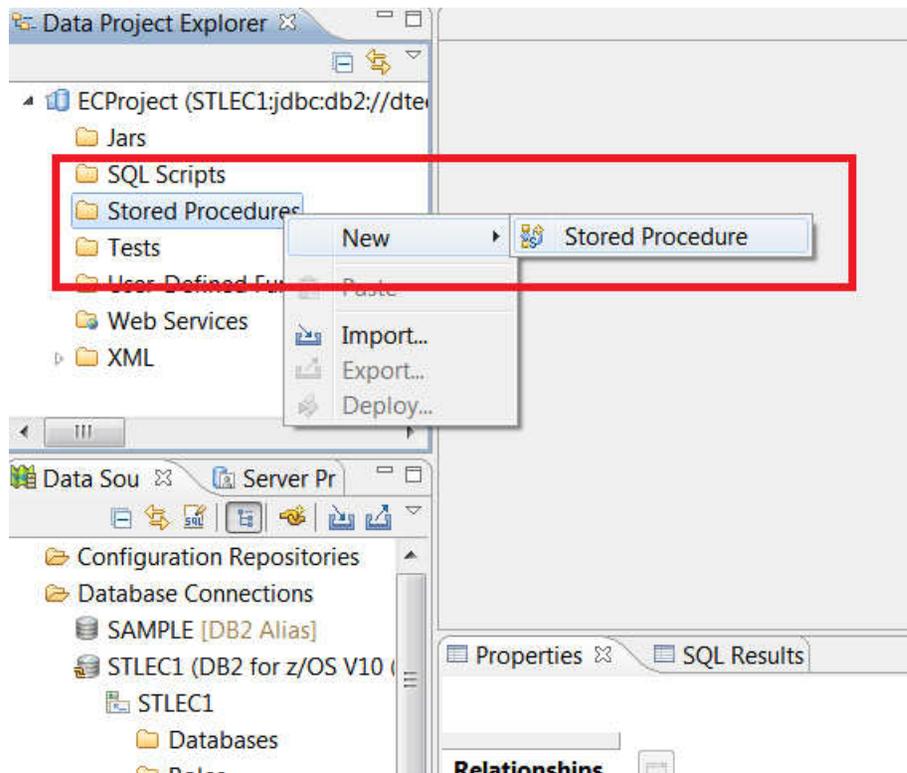
Data Development Project Folders

- Jars
 - Deploy Jar to DB2
- SQL Scripts
 - Develop SQLs using Query builder or Query editor
- Stored Procedures
 - Develop and Debug Stored Procedures
- Tests
 - Develop regression test cases for routines

Data Development Project (contd)

- User-Defined Functions
 - Develop and Debug User-Defined Functions
- Web Services
 - Create Web Services wrapper for routines or SQL Scripts
- XML
 - Edit and deploy XML Schema

Create new Stored Procedure



Data Studio groups routines according to their types

- Stored Procedures
- User-Defined Functions
- SQL Scripts

Create Routines from Templates

New Stored Procedure

Name, Language, and Template

Specify a name and language for the new stored procedure. You can choose a template to use as the framework appears in the Preview window. Click Finish to open the editor.

Name: PROCEDURE1

Language: SQL

Java Package: com.leungtk.leungtk

Jar Name: DS_20121221201906

Select a template

Template	Description
Custom: (External) You supply the SQL, return a result ...	You specify the SQL to execute and the values are re
Custom: (Native) You supply the SQL, return a result set	You specify the SQL to execute and the values are re
Deploy & Run: (External) IN/OUT parameters	Returns the count of rows from SYSIBM.SYSTABLES t
Deploy & Run: (External) Return a result set	Opens a cursor and retrieves values from SYSIBM.SY
Deploy & Run: (Native) IN/OUT parameters	Returns the count of rows from SYSIBM.SYSTABLES t

Preview:

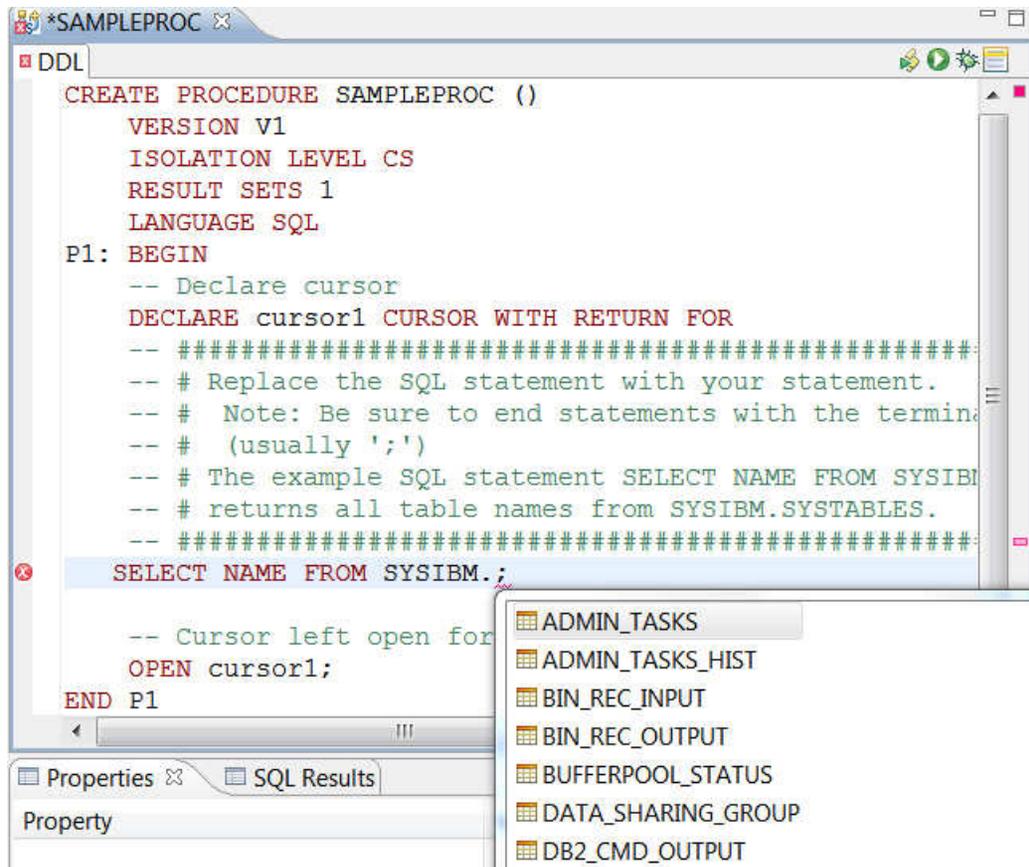
Template Details DDL

Custom: (External) You supply the SQL, return a result set

You specify the SQL to execute and the values are returned.

- Users can define their own templates according to database platforms and routine types
- Templates can be used to provide coding standards and hints/tips for developers when creating routines from scratch

Routine Editor

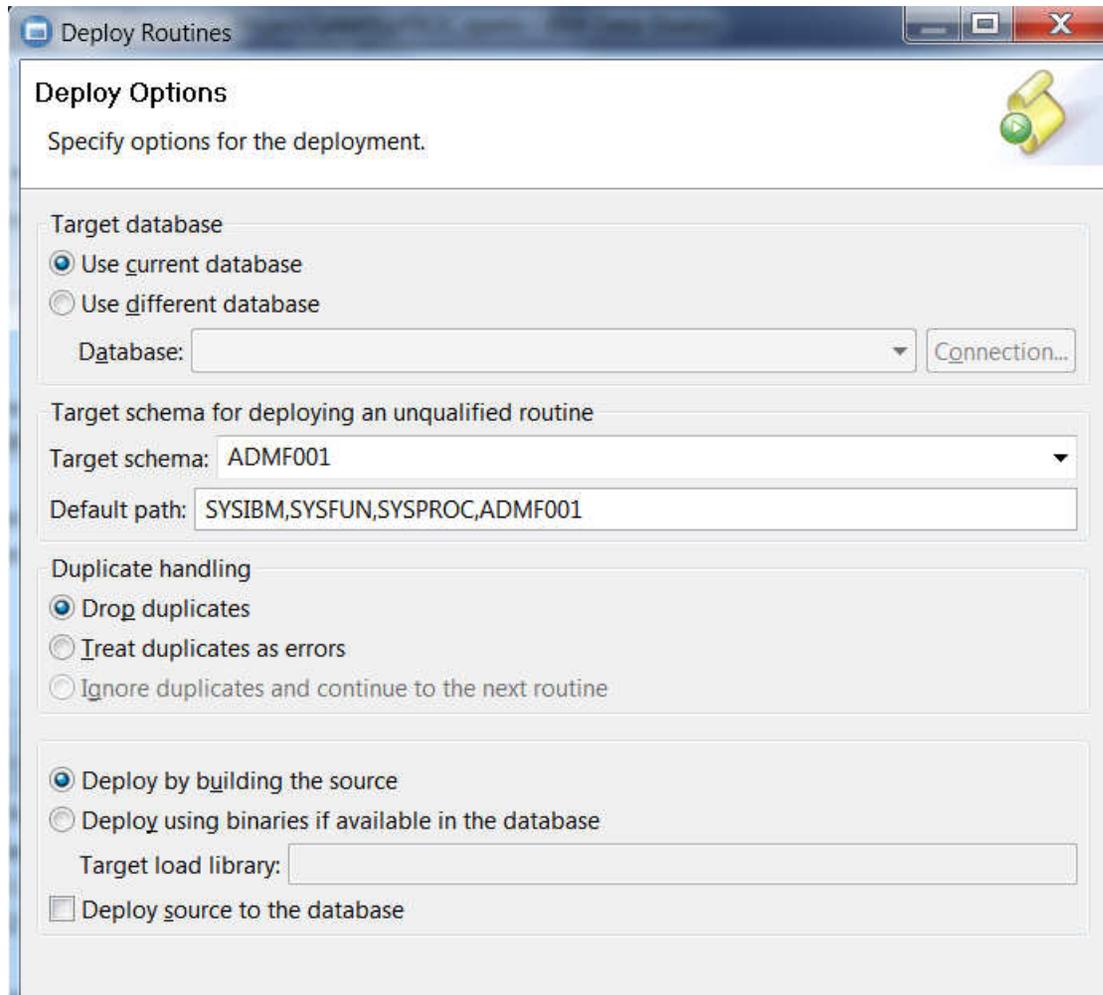


```
CREATE PROCEDURE SAMPLEPROC ()
  VERSION V1
  ISOLATION LEVEL CS
  RESULT SETS 1
  LANGUAGE SQL
P1: BEGIN
  -- Declare cursor
  DECLARE cursor1 CURSOR WITH RETURN FOR
  -- *****
  -- # Replace the SQL statement with your statement.
  -- # Note: Be sure to end statements with the termin
  -- # (usually ';'')
  -- # The example SQL statement SELECT NAME FROM SYSIBM
  -- # returns all table names from SYSIBM.SYSTABLES.
  -- *****
  SELECT NAME FROM SYSIBM.;
  -- Cursor left open for
  OPEN cursor1;
END P1
```

ADMIN_TASKS
ADMIN_TASKS_HIST
BIN_REC_INPUT
BIN_REC_OUTPUT
BUFFERPOOL_STATUS
DATA_SHARING_GROUP
DB2_CMD_OUTPUT

- Provides syntax check on SQL statements
- Provides content assist to complete SQL Object references

Deploy Routine to Server

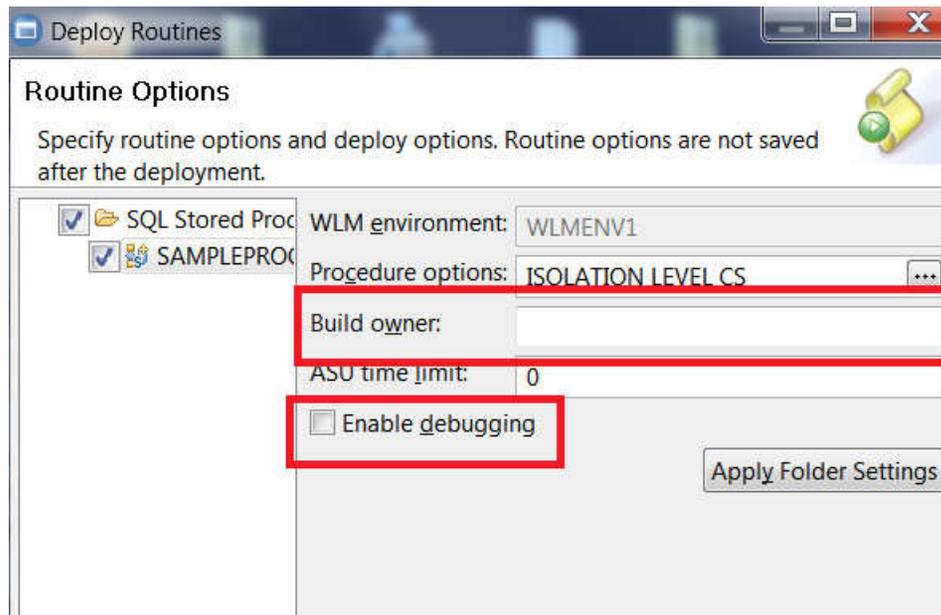


The screenshot shows a window titled "Deploy Routines" with a "Deploy Options" section. The options are as follows:

- Target database:**
 - Use current database
 - Use different database
 - Database: [] [Connection...]
- Target schema for deploying an unqualified routine:**
 - Target schema: ADMF001 []
 - Default path: SYSIBM,SYSFUN,SYSPROC,ADMF001
- Duplicate handling:**
 - Drop duplicates
 - Treat duplicates as errors
 - Ignore duplicates and continue to the next routine
- Deployment method:**
 - Deploy by building the source
 - Deploy using binaries if available in the database
 - Target load library: []
 - Deploy source to the database

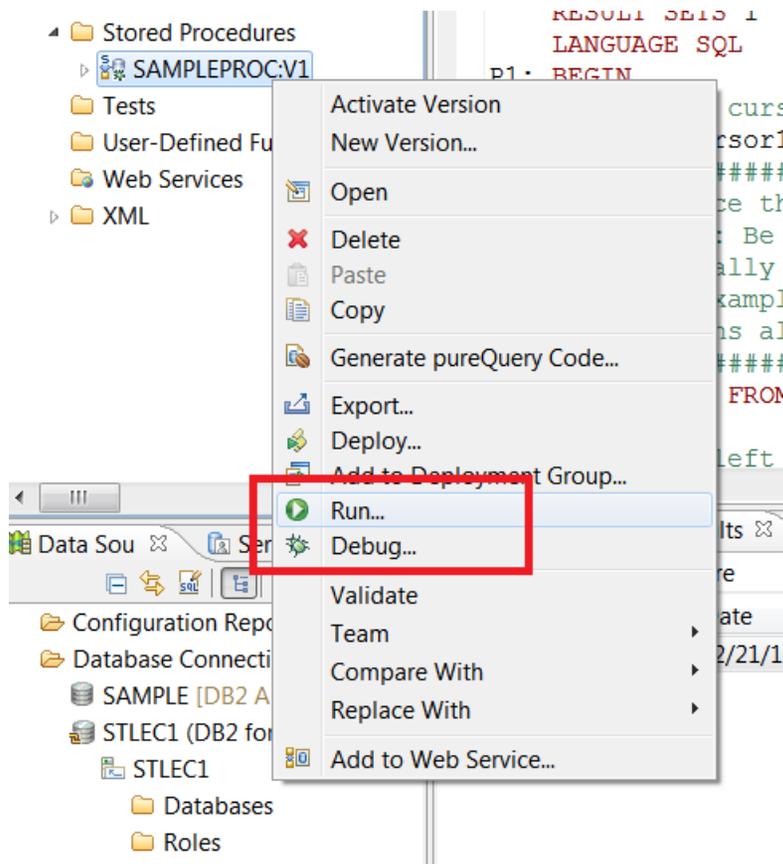
- Automatically generate
Alter routine statements
if routine already exists
on server

Support for setting CURRENT SQLID



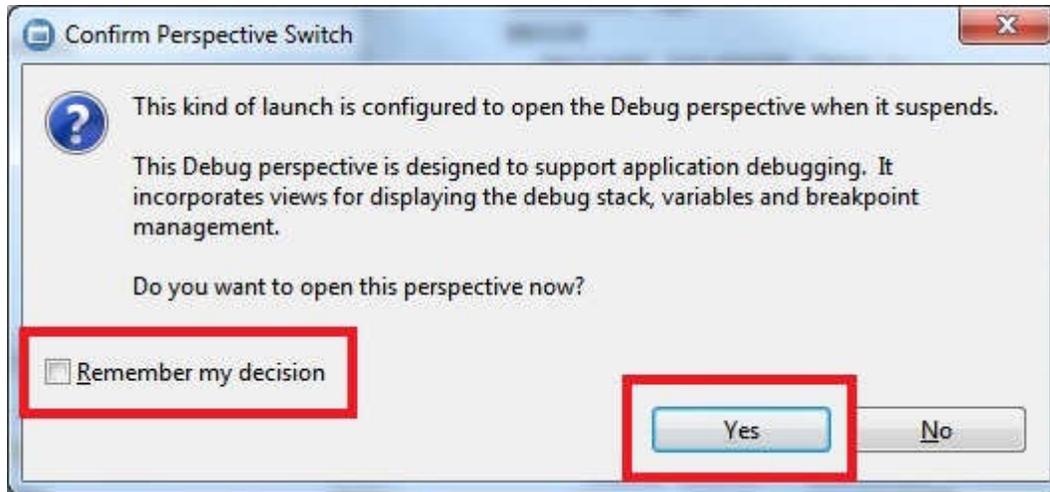
- Ability to set current sqlid during routine creation
- Ability to compile routine in Debug mode

Run or Debug



- Use popup menu on the routine to run or debug the routine

Launch the Debugger



- Remember to switch to the debug perspective
If you do not switch perspective, the debug views may be hidden

Developing Java Stored Procedure

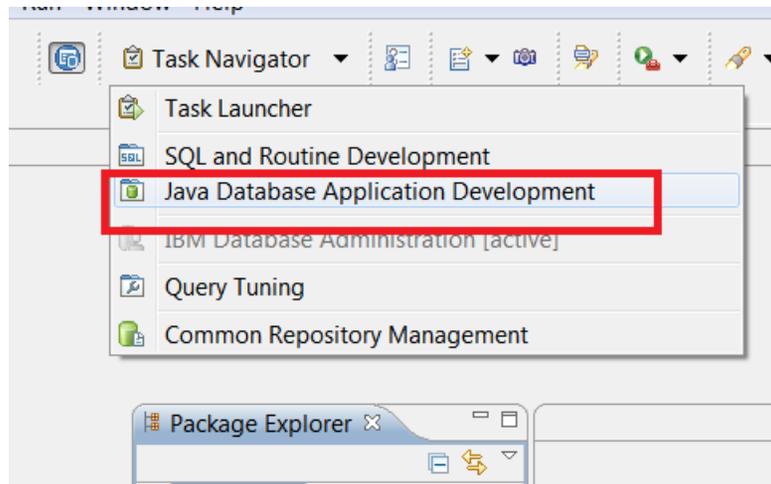
Edit DDL and Java Source

```
SELECPROC ✕
DDL Java
CREATE PROCEDURE SELECPROC ( )
  RESULT SETS 1
  EXTERNAL NAME 'DS_20130204152508:com.leungtk
  LANGUAGE Java
  PARAMETER STYLE JAVA
  NOT DETERMINISTIC
  COLLID DSNJDBC
  WLM ENVIRONMENT WLMENVJU
```

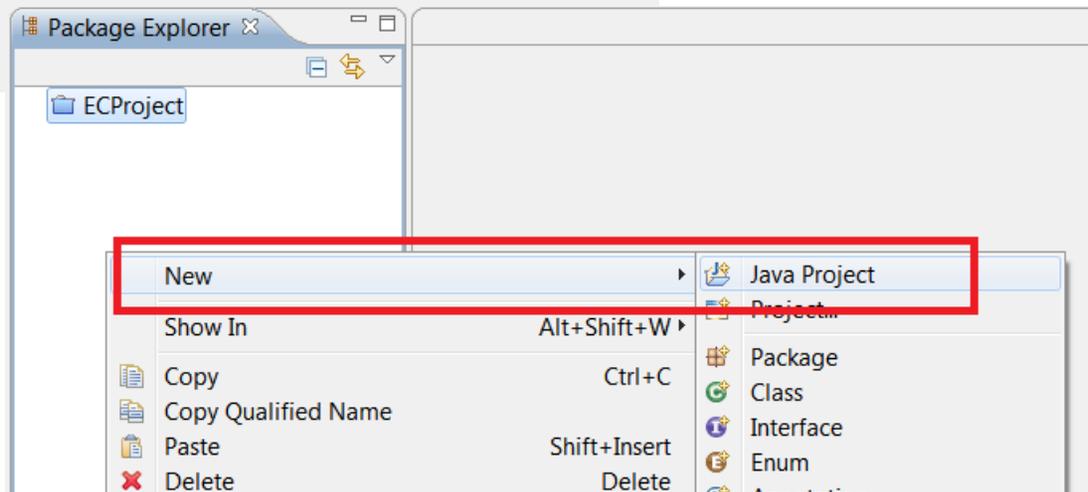
- Developer can edit both the DDL and the Java Source concurrently
- Routine Editor checks for consistency between Java source and DDL

Developing Java Applications

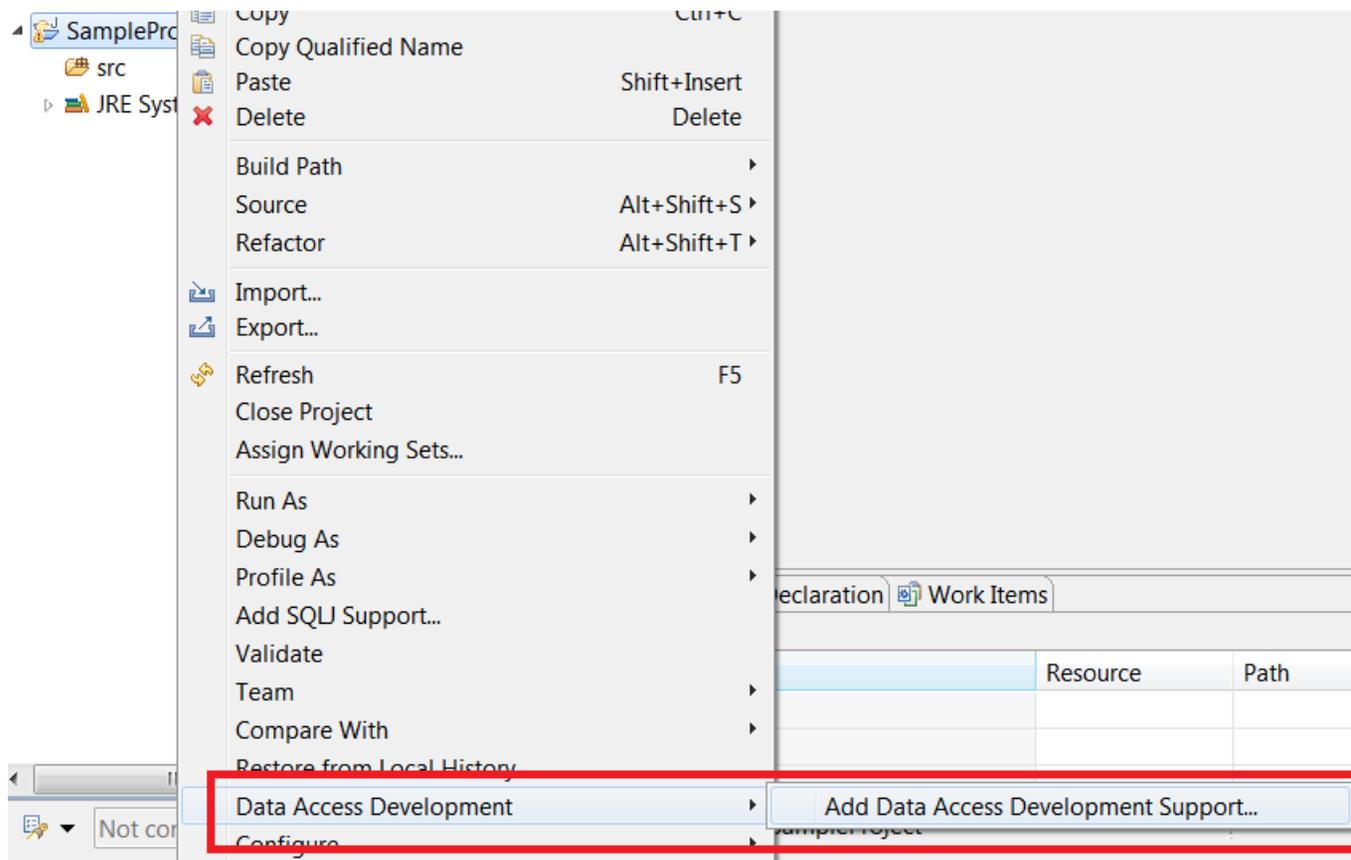
Switch to Java Perspective



Create Java project



Add Data Access Development Support



Data Access Development Support

Add Data Access Development Support

Data Access Development Support

Specify a connection, default values for unqualified SQL objects, and if pureQuery support should be added.

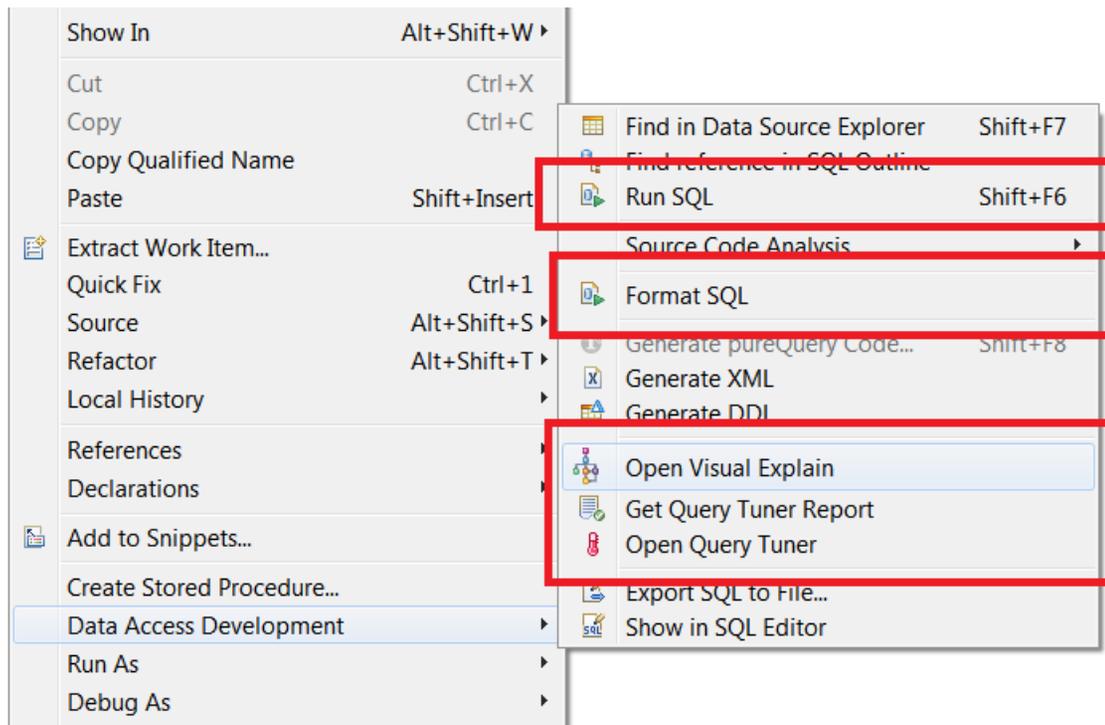
Connection	
STLEC1	
Database name:	STLEC1
Database vendor:	DB2 UDB zSeries
Database version:	V10 (New-Function Mode)
Database user name:	admf001
Database URL:	jdbc:db2://dtec514.vmec.svl.ibm.com:446/STLEC1: retrieveMessagesFromServerOnGetMessage=true;emulateParameterMe
<input checked="" type="checkbox"/>	Add JDBC drivers from the connection to the project's buildpath
Default values for unqualified SQL objects	
Default schema:	ADMF001
<input checked="" type="checkbox"/>	Omit default schema in generated statements
Default path:	SYSIBM,SYSFUN,SYSPROC,ADMF001
pureQuery	
<input checked="" type="checkbox"/>	Using pureQuery in production requires a pureQuery runtime license.
<input type="checkbox"/>	Add pureQuery support to project

- Specify connection used for SQL Content Assist and Syntax checks

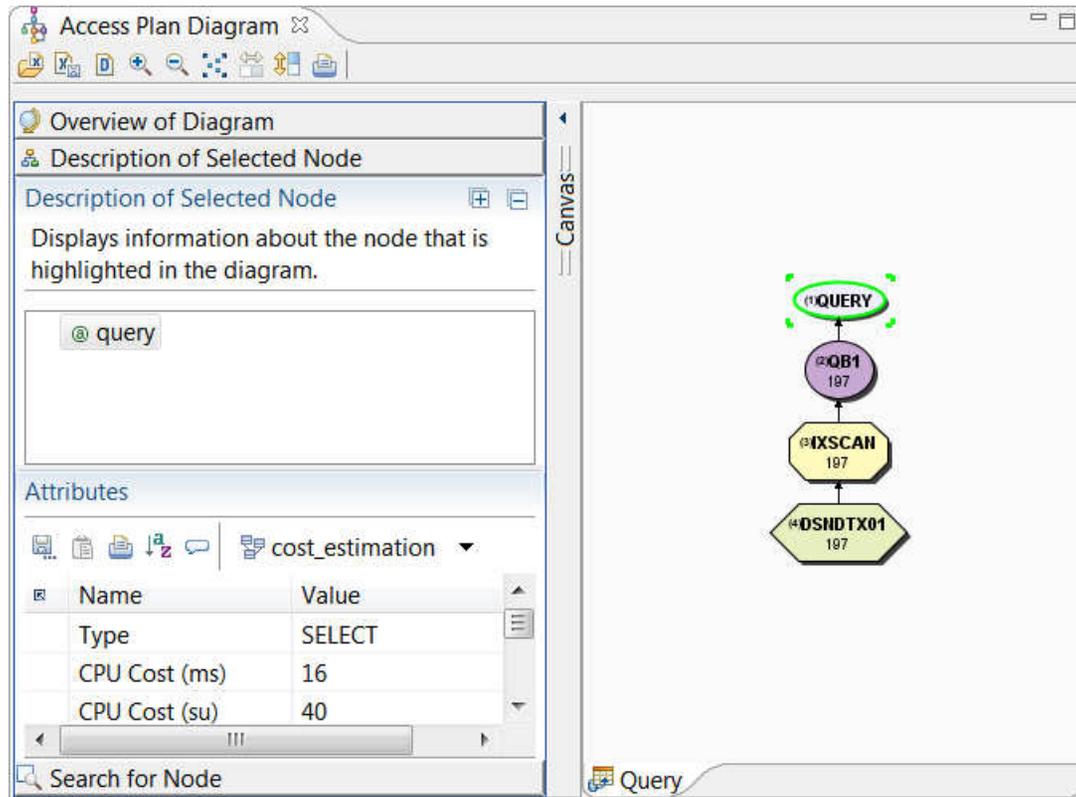
Syntax and Table Reference Validation

```
public class JDBCSTable {  
    private static void run1(Connection connection) throws SQLException {  
        // show non-variable SQL  
        Statement statement = connection.createStatement();  
        ResultSet resultSet =  
        Unable to find table "DUMMYTABLE".  
        executeQuery("SELECT DEPTNO FROM DUMMYTABLE");  
        while (resultSet.next());  
        resultSet.close();  
    }  
}
```

Popup for Highlighted SQL



Visualize Access Plan from SQL



Query Workload Tuner Integration

SQL Formatter Integration

DDL 🔍 ⚙️ 📄

```

CREATE PROCEDURE GETINFO (IN zipcode VARCHAR(255) )
  DYNAMIC RESULT SETS 1
P1: BEGIN
  -- Declare cursor
  DECLARE cursor1 CURSOR WITH RETURN for
  select oadmin.CUSTOMER.c_name, n_regionkey
  from OADMIN.customer, nation, part
  where c_custkey in
  (select SUPPLIER_NO from REVENUE, order where SUPPLIER
  and (c_nationkey = n_nationkey and
  n_name between 'a%' and 'k%' and p_size < 10)
  and not (C_NATIONKEY <> n_nationkey and N_NAME like 'N
  and zip = zipcode
  group by c_name, n_regionkey
  having (SELECT AVG(SALARY) FROM EMPLOYEE) > 60000 an

  -- Cursor left open for client application
  OPEN cursor1;
END P1

```

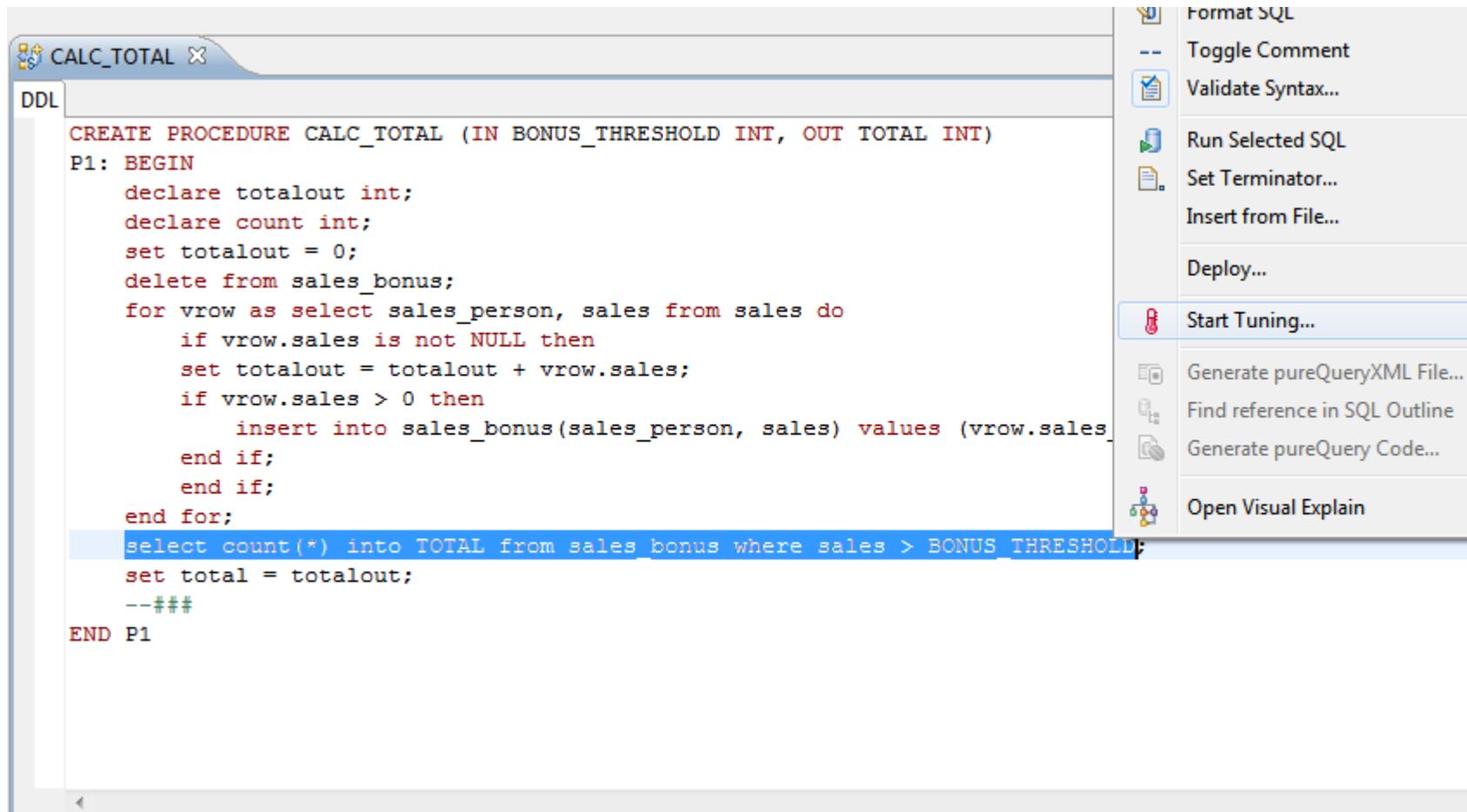
- 🔄 Undo Typing
- ↶ Revert File
- 📄 Save Ctrl+X,
- Show In Alt+S
- Cut C
- Copy
- Paste
- Run As
- Debug As
- Profile As
- Validate
- Team
- Compare With
- Replace With
- Preferences...
- 💡 Content Assist
- 💡 Content Tip
- 📄 Format SQL Ctrl+S
- Toggle Comment

Formatted SQL

```
DDL
CREATE PROCEDURE GETINFO (IN zipcode VARCHAR(255) )
    DYNAMIC RESULT SETS 1
P1: BEGIN
    -- Declare cursor
    DECLARE cursor1 CURSOR WITH RETURN for
    SELECT OADMIN.CUSTOMER.C_NAME
        , OADMIN.NATION.N_REGIONKEY
    FROM OADMIN.CUSTOMER
        , OADMIN.NATION
        , OADMIN.PART
    WHERE OADMIN.CUSTOMER.C_CUSTKEY IN (
        SELECT OADMIN.REVENUE.SUPPLIER_NO
        FROM OADMIN.REVENUE
            , OADMIN.ORDER
        WHERE OADMIN.REVENUE.SUPPLIER_NO = OADMIN.ORDER.C
            AND OADMIN.REVENUE.TOTAL_REVENUE IN (10, 5)
        )
    AND OADMIN.CUSTOMER.C_NATIONKEY = OADMIN.NATION.N_NATIONKEY
    AND OADMIN.NATION.N_NAME BETWEEN 'a%' AND 'k%'
    AND OADMIN.PART.P_SIZE < 10
    AND ( OADMIN.CUSTOMER.C_NATIONKEY = OADMIN.NATION.N_NATIONKEY
        OR OADMIN.NATION.N_NAME NOT LIKE 'NA'
        )
    AND ZIP = ZIPCODE
    GROUP BY OADMIN.CUSTOMER.C_NAME,
        OADMIN.NATION.N_REGIONKEY
    HAVING ( SELECT AVG(OADMIN.EMPLOYEE.SALARY)
        FROM OADMIN.EMPLOYEE
        ) > 60000
    AND OADMIN.NATION.N_REGIONKEY IN (1, 6);

    -- Cursor left open for client application
    OPEN cursor1;
END P1
```

Routine Editor to Query Tuner



The screenshot shows a SQL routine editor window titled "CALC_TOTAL X". The editor contains the following SQL code:

```
DDL
CREATE PROCEDURE CALC_TOTAL (IN BONUS_THRESHOLD INT, OUT TOTAL INT)
P1: BEGIN
  declare totalout int;
  declare count int;
  set totalout = 0;
  delete from sales_bonus;
  for vrow as select sales_person, sales from sales do
    if vrow.sales is not NULL then
      set totalout = totalout + vrow.sales;
      if vrow.sales > 0 then
        insert into sales_bonus(sales_person, sales) values (vrow.sales
      end if;
    end if;
  end for;
  select count(*) into TOTAL from sales_bonus where sales > BONUS_THRESHOLD;
  set total = totalout;
  --###
END P1
```

A context menu is open on the right side of the editor, with the following items:

- Format SQL
- Toggle Comment
- Validate Syntax...
- Run Selected SQL
- Set Terminator...
- Insert from File...
- Deploy...
- Start Tuning...**
- Generate pureQueryXML File...
- Find reference in SQL Outline
- Generate pureQuery Code...
- Open Visual Explain

Query Transformed and Transferred to Query Tuner

 Run Single-Query Advisors And Analysis Tools

Specify EXPLAIN options and runtime environment options for the query. You can optionally create temporary tables using

Database connection: SAMPLE (DB2 for Linux, UNIX, and Windows V9.7.2)

Schema: Description:

Re-EXPLAIN the query

▶ EXPLAIN options and runtime environment options

▼ Query Text - Query 1

```
select count(*) from sales_bonus where sales > CAST ( ? AS INT)
```

Resources

- IBM Data Studio Overview
<http://www.ibm.com/software/products/us/en/data-studio>
- Support Page with links to forums and documentation
<http://www.ibm.com/developerworks/downloads/im/data/support.html>

 #SHAREorg



Application Development With Data Studio

Tony Leung
IBM
February 4, 2013
13087

leungtk@us.ibm.com



Insert
Custom
Session
QR if
Desired.

