PL/SQL Programming for .NET Developers: Tips, Tricks, and Debugging

Christian Shay
Product Manager, .NET Technologies
Oracle
Oracle .NET Customer Advisory Board

- Focus group that provides Oracle input and help to prioritize new features
  - Led by Oracle VP
- Work directly with Oracle Development and PM
- Best for organizations in which Oracle .NET is strategic
- Contact me for details and how to apply
Program Agenda

1. SQL and PL/SQL Development Lifecycle in Visual Studio
2. Using PL/SQL with ODP.NET
3. PL/SQL Debugging
4. Next Steps
SQL and PL/SQL Development Lifecycle in Visual Studio
Oracle’s .NET Products

• Oracle Developer Tools for Visual Studio

• Oracle Data Provider for .NET (ODP.NET)
  – ADO.NET compliant data provider
  – Utilize advanced Oracle Database features
    • RAC, performance, security, data types, XML, etc.

• Both available for free download:
  – http://otn.oracle.com/dotnet/
SQL and PL/SQL Development Lifecycle

• Create Database, Schema Objects, PL/SQL procedures, functions, packages
  – Run SQL*Plus Scripts with existing scripts
  – Oracle Wizards and designers
  – Query Window – Ad Hoc SQL
  – Import Table Wizard
  – Oracle Multitenant
    • Use VS to Clone and Plug in existing PDB or create new one
SQL and PL/SQL Development Lifecycle

• Create SQL and PL/SQL scripts
  – Generate Create Script from existing schema objects
  – Use Schema Compare tool to generate diff script

• Store scripts in source control
  – Oracle Database Project

• Edit SQL and PL/SQL Scripts
  – Oracle SQL Editor – file based
  – Oracle PL/SQL Editor – database based
SQL and PL/SQL Development Lifecycle

- Tune SQL
  - Oracle Performance Analyzer
  - SQL Tuning Advisor

- Create client side .NET code (C#, VB.NET, ASP.NET)
  - Use Oracle Data Provider for .NET to call PL/SQL

- Debug .NET and PL/SQL together
  - PL/SQL Debugger in Visual Studio
SQL and PL/SQL Development Lifecycle

• Deploy
  – New deployments
    • Run SQL scripts to create schema objects
    • Clone and plug in PDBs
  – Updating existing deployments
    • Create and run diff script using output scripts from Schema Compare tool
Using PL/SQL with ODP.NET
Introduction

• Any PL/SQL Call is Supported
  – Stored Procedure
  – Stored Function
  – Package Method
  – Anonymous block
    • Batch SQL support
PL/SQL Data Types Available in .NET

• Data Types
  – PL/SQL Types
  – REF Cursor
  – Associative Array (formerly index-by table)
  – User Defined Types

• ODP.NET Types vs. .NET types
  – OracleParameter.DbType
  – OracleParameter.OracleDbType
Batching SQL and deferring fetching

• You want to execute SQL queries in Stored Procedures and then fetch as needed from the client
• You want to “batch SQL” – multiple SQL statements in one PL/SQL anonymous block
• Solution: Use REF CURSORS and Anonymous PL/SQL
REF Cursors

• Characteristics
  – Pointer to result set on server side
  – Read only
  – Forward only

• Advantages
  – Input REF Cursor parameters
  – Retrieve multiple REF Cursors in a single round trip
REF CURSORS
Passing large amounts of data

• You want to pass in or retrieve large amounts of data in one round trip with best performance possible
• You are using scalar types
• Solution: Use associative arrays
Associative Arrays

• Characteristics
  – Must declare size of array
  – Index key must be sequential
  – Index key must be non-negative integers

• Advantages
  – Pass large amount of data between the DB and .NET in one array
    • Reduces number of parameters
    • Reduces round trips, easier batch processing
Using Associative Arrays in .NET

• Steps to bind an associative array parameter
  – Set OracleParameter.CollectionType to OracleCollectionType.PLSQLAssociativeArray
  – Set OracleParameter.ArrayBindSize for *each* array element
    • Only necessary for variable-length data types
  – Set OracleParameter.Size for number of array elements
Associative Arrays
Anonymous PL/SQL

• Executes multiple SQL statements in a single batch
  – Saves DB round trips
  – Execute as CommandType.Text

• Generate dynamically based on application requirements

```java
string cmdtxt = "BEGIN " +
"OPEN :1 for select * from emp where deptno = 10; " +
"OPEN :2 for select * from dept where deptno = 20; " +
"INSERT INTO DEPT VALUES (50, 'IT', 'SAN FRANCISCO');" +
"END;";
```
Using Pre-Defined PL/SQL Packages

• DB server provides PL/SQL packages to all of Oracle’s key functionality
  — Can be used from ODP.NET, similar to any other PL/SQL call
  — Sample pre-packaged functionality
    • DBMS_AQ
    • DBMS_OLAP
    • DBMS_STREAMS
    • SDO_GEOM
UDTs, VARRAYs and NESTED TABLES

- Use ODT Custom Class Code Generation wizard
- Oracle by Example walkthrough of Code Generation Wizard:
  - http://goo.gl/W3OwP6
- Bind the generated classes to input and output parameters
  - See this ODP.NET doc section for binding information:
    - *Oracle User-Defined Types (UDTs) and .NET Custom Types*
- If installed, check out code samples in directory
  <Oracle_Home>\odp.net\samples\4.x\UDT
MyVarrayCustomClass pa = new MyVarrayCustomClass();
pa.Array = new Int32[] { 1, 2, 3, 4 };

pa.StatusArray = new OracleUdtStatus[] {
    OracleUdtStatus.NotNull....
};

param.OracleDbType = OracleDbType.Array;
param.Direction = ParameterDirection.Input;
param.UdtTypeName = "MYVARRAY";
param.Value = pa;
PL/SQL Debugging
Oracle PL/SQL Debugging Architecture

Oracle Developer Tools for Visual Studio

Visual Studio Environment

Connect user/pass

When connect to Oracle we pass:
ORA_DEBUG_JDWP=
host=hostname;port=portnum

Oracle 10.2 or later

PL/SQL Debugging Engine

Does all the work of debugging

Connect user/pass

Database connects back to VS via TCP/IP at hostname and port

VS sends requests to Oracle over TCP/IP connection:

“Step Into Please”
“What are the local variable values?”
“Set Breakpoint here”
PL/SQL Debugging Configuration

• Can be tricky the first time since so many steps
• Check out the PL/SQL Debugging Chapter in ODT online doc
  – “Debugging Setup Checklist”
• Work through the Oracle by Example demo:
  – http://goo.gl/SXvZ8W
PL/SQL Debugging Doc
PL/SQL Debugging Configuration

• GRANT debug privileges as SYSDBA
  – GRANT DEBUG ANY PROCEDURE TO username
  – GRANT DEBUG CONNECT SESSION TO username

• Set port range and IP in Debugging Options page
  – Tools -> Options->Oracle Developer Tools

• Compile PL/SQL units for Debug
  – Via menu in PL/SQL editor or in Server Explorer
  – Server Explorer Icons change color to remind you to recompile later
PL/SQL Debugging Configuration

• New requirement in Oracle Database 12c:
  – SYSDBA must grant ACL access on
    • IP Address
    • Port Range
    • Schema name

• Use new “Grant Debugging Privileges dialog”
  – Right click on Schema name to be granted both debugging roles and the ACL privileges
PL/SQL Debugging Configuration

• Or issue this PL/SQL as SYSDBA:

  • BEGIN
    DBMS_NETWORK_ACL_ADMIN.APPEND_HOST_ACE(
      HOST => '127.0.0.1',
      LOWER_PORT => 65000,
      UPPER_PORT => 65300,
      ACE => XS$ACE_TYPE(PRIVILEGE_LIST => XS$NAME_LIST('jdwp'),
       PRINCIPAL_NAME => 'HR',
       PRINCIPAL_TYPE => XS_ACL.PTYPE_DB));
  END;
Direct Database Debugging

- Debug directly inside the database, no application code
- “Step Into” from Server Explorer
- “Run Debug” from Server Explorer
- Enter parameters manually
  - Not useful with array parameters or complex types
Application Debugging Mode

• Step from .NET code into PL/SQL and back from one instance of Visual Studio
• Useful for client server code (not web apps)
• Check off “Tools -> Oracle Application Debugging”
• ODT automatically starts listener using port in range given in Options page
• **Uncheck** "Enable the Visual Studio hosting process" in the .NET Project Properties Debug tab
Direct and App Debugging
External Application Debugging

- Dedicated VS instance for debugging PL/SQL only
  - Use additional VS instance for any .NET code (eg ASP.NET app)
- PL/SQL called by 10.2 client or later running on ANY platform
- Set ORA_DEBUG_JDWP in client environment
  - SET ORA_DEBUG_JDWP=host=mymachine;port=4444
  - Set in web app environment BEFORE connecting
- Start Listener
  - Tools-> “Start Oracle External Application Debugger”
External PL/SQL Debugging
Using DBMS_DEBUG_JDWP Package

• Allows you to pick and choose when debugging is turned on
  – Good solution when PL/SQL packaged procedure is called often but you only are interested in debugging specific cases

• Enable External Application Debugging
• Provide port number and IP address
Using DBMS_DEBUG_JDWP Package

• Add calls to these PL/SQL Procedures to your SP:
  – DBMS_DEBUG_JDWP.CONNECT_TCP(HOST VARCHAR2, PORT VARCHAR2)
  – DBMS_DEBUG_JDWP.DISCONNECT

• Compile Debug
• Set Breakpoint
• Enable External Application Debugging
• Call SP from external application or middle tier
Next Steps
Oracle .NET Customer Advisory Board

• Focus group that provides Oracle input and help to prioritize new features
  – Led by Oracle VP
• Work directly with Oracle Development and PM
• Best for organizations in which Oracle .NET is strategic
• Contact me for details and how to apply
Additional Oracle .NET Resources

OTN
otn.oracle.com/dotnet

Twitter
twitter.com/OracleDOTNET

YouTube
youtube.com/OracleDOTNETTeam

Email
christian.shay@oracle.com
Safe Harbor Statement

The preceding is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described for Oracle’s products remains at the sole discretion of Oracle.
Hardware and Software
Engineered to Work Together