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

Christian Shay
Product Manager, Oracle
Program Agenda

- PL/SQL Development Lifecycle in VS
- Using PL/SQL with ODP.NET
  - Introduction
  - PL/SQL Data Types Mapping in .NET
  - Anonymous PL/SQL Blocks
  - Using Oracle Supplied PL/SQL Packages in .NET
  - VARRAYs and Nested Tables
- PL/SQL Debugging in Visual Studio
- Next Steps
SQL and PL/SQL Development Lifecycle

- Create Users, and Roles and grant privileges to them
  - User and Role Designers
  - Grant and Revoke Privileges Wizard
- Create Schema Objects, PL/SQL procedures, functions, packages
  - Oracle Wizards
  - Query Window – Ad Hoc SQL
  - Run SQL*Plus Scripts for existing scripts
  - Import Table Wizard
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
SQL and PL/SQL Development Lifecycle

- Edit SQL and PL/SQL Scripts
  - Oracle SQL Editor – file based
  - Oracle PL/SQL Editor – database based

- Tune SQL
  - Oracle Performance Analyzer
  - SQL Tuning Advisor
SQL and PL/SQL Development Lifecycle

- 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
- Deploy
Oracle’s .NET Products

- **Oracle Developer Tools for Visual Studio**

- **Oracle Data Provider for .NET (ODP.NET)**
  - ADO.NET compliant data provider
  - Native access to Oracle database
  - Utilize advanced Oracle Database features
    - RAC, performance, security, data types, XML, etc.

- Both available for free download:
PL/SQL Lifecycle
Program Agenda

- PL/SQL Development Lifecycle in VS
- Using PL/SQL with ODP.NET
  - Introduction
  - PL/SQL Data Types Mapping in .NET
  - Anonymous PL/SQL Blocks
  - Using Oracle Supplied PL/SQL Packages in .NET
  - VARRAYs and Nested Tables
- PL/SQL Debugging in Visual Studio
- Next Steps
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)

- ODP.NET Types vs. .NET types
  - OracleParameter.DbType
  - OracleParameter.OracleDbType
  - .NET DataSet can store ODP.NET types
    - OracleDataAdapter.ReturnProviderSpecificTypes = true
Hello PL/SQL
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
DEMONSTRATION

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

```
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
VARRAYs and NESTED TABLES

- Supported as of ODAC 11g
  - Use Custom Class Code Generation wizard
  - Check out code samples in directory
    <OH>\odp.net\samples\4.x\UDT
VARRAYs and NESTED TABLES

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;
Program Agenda

- PL/SQL Development Lifecycle in VS
- Using PL/SQL with ODP.NET
  - Introduction
  - PL/SQL Data Types Mapping in .NET
  - Anonymous PL/SQL Blocks
  - Using Oracle Supplied PL/SQL Packages in .NET
  - VARRAYs and Nested Tables
- PL/SQL Debugging in Visual Studio
- Next Steps
Oracle PL/SQL Debugging Architecture

Oracle Developer Tools for Visual Studio

Visual Studio Environment

Oracle 9.2 or later

PL/SQL Debugging Engine

Does all the work of debugging

Connect user/pass

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

PL/SQL Debugging Engine

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 doc
  - “Debugging Setup Checklist”
- See ODT release notes for known issues
  - `<ORACLE_HOME>\odt\doc\readme.html`
PL/SQL Debugging Configuration

- GRANT debug privileges as SYSDBA
  - 9.2 or later: GRANT DEBUG ANY PROCEDURE TO username, and
  - 10g or later also requires: 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 Oracle Explorer
PL/SQL Debugging Configuration

- New requirement in 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 (e.g., ASP.NET app)
- 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
Advanced Debugging
DBMS_DEBUG_JDWP Package

- Allows you to pick and choose when debugging is turned on
- Enable External Application Debugging
- Provide port number and IP address
Advanced Debugging
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
  - Call SP from external application
Next Steps
Upcoming .NET Sessions

Wednesday Sept 25

- ALM with Visual Studio: SQL and PL/SQL Development, Source Control, and Deployment
  - 3:30 PM - 4:30 PM, Marriott Marquis - Golden Gate C2

- Oracle Database 12c on Windows
  - 5:00 PM – 6:00 PM, Moscone South - 104
Upcoming .NET Sessions
Thursday Sept 26

- Hands-on Lab: Building .NET Applications with Oracle
  - 11:00 AM - 1:30 PM, Marriott Marquis - Salon 3/4

- Oracle and .NET: Best Practices for Performance and Deployment
  - 2:00 PM - 3:00 PM, Marriott Marquis - Golden Gate C2
Visit .NET Experts at the Demogrounds
Monday through Wednesday

- .NET Development for Oracle Database 12c
  - Moscone South, Left Rear in Oracle Database Section
  - Booth # SL-044
Additional Oracle .NET Resources

- OTN .NET Developer Center
  - [http://otn.oracle.com/dotnet](http://otn.oracle.com/dotnet)
- Twitter
  - @OracleDOTNET
- YouTube
  - [http://www.youtube.com/user/OracleDOTNETTeam](http://www.youtube.com/user/OracleDOTNETTeam)
- For more questions
  - [christian.shay@oracle.com](mailto:christian.shay@oracle.com)
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