87. How to improve ADF Business Components LOV performance using shared application modules

Abstract:

The performance of static list data used in Oracle ADF applications improves using shared application modules in ADF Business Components. SQL statement for static list will be executed only once per application scope, retrieved static data will be preserved in memory and shared across users, thus avoiding expensive queries being executed for each user session. Oracle JDeveloper 11g R2 (11.1.2) has an additional configuration parameter(jbo.shared.txn) you set to minimize number of open database connections for Shared Application modules.

Author: Andrejus Baranovskis, Red Samurai Consulting
twitter.com/andrejusb
29-JUL-2011
Introduction

This example describes two solutions. First presents usage of shared application modules to reduce SQL execution and optimize data retrieval for static data lists across application users. Second explains how to configure the jbo.shared.txn property with JDeveloper 11g R2 11.1.2.0.0 to minimize the number of opened database connections for shared application modules.

Sample application implements two static lists (Jobs/Departments) and loads these lists into application memory to share between users.
Later in the article we describe how to optimize opened database connections for shared application module instances with jbo.shared.txn property.
Implementing this solution

The sample application that you can download for this article contains three ADF Business Component model projects and one ViewController project.

The **HrCommonModel** and **OrgCommonModel** contain Jobs and Departments LOV’s, both of these projects are packaged into ADF libraries and use different application modules.

The **Model** project imports the generated ADF libraries, one containing the **Jobs LOV** and the other containing the **Departments LOV**.
How to improve ADF Business Components LOV performance using shared application modules

Now you can declare imported AM’s to be shared across application scope, this can be done through Application Module Instances section. For this, open the application model project, Model in this sample, and select the ADF Business Components | Application Module Instances node. With the Application tab selected find and select the HrCommonModule and OrgCommonModule entries.

Next, you use the static list data in the main Model project. For this, instead of using VO instance from the Model project directly, you select a View Object instance from one of the shared application modules and declare it as the list data source.
How to improve ADF Business Components LOV performance using shared application modules

The rest is business as usually for those savvy with model driven List of Value sin ADF BC.

Set the UI Hints to render the List of Values as a Choice List and select the attributes to display in the list.
The choice list as a type is used for a reason! List of Values, like input Text with List of Values, should not be used with shared application modules, because shared data collection should not be filtered as it would impact other user sessions as well.

Now let’s see how it works, first user is loading ADF page with choice list data.

During first load of ADF page for the first user, SQL statements for choice lists are generated and executed in database. One for first choice list – Jobs.

```xml
<ViewObjectImpl> <buildQuery> [804] JobsLovView1>'# q compute
<ViewObjectImpl> <buildQuery> [805] SELECT Jobs.JOB_ID,
Jobs.JOB_TITLE,
Jobs.MIN_SALARY,
Jobs.MAX_SALARY
FROM JOBS Jobs
</ViewObjectImpl> <getPreparedStatement> [806] ViewObject:
```
And another SQL statement, for second choice list – Departments.

```
<ViewObjectImpl> <buildQuery> [1026] DepartmentsView1>
$q computed SQLStmt:
<ViewObjectImpl> <buildQuery> [1027] SELECT Departments.DEPARTMENT_ID,
    Departments.DEPARTMENT_NAME,
    Departments.MANAGER_ID,
    Departments.LOCATION_ID
FROM DEPARTMENTS Departments
<ViewObjectImpl> <getPreparedStatement> [1028] ViewObject: [com.redsamu:
```

When a next user accesses the application, there is no SQL statements generated for fetching the list data, as data is fetched from the memory. This happens, because the LOVs are populated from a shared application module instance, which means that no additional module instance will be created for new users.

This means, static data for all uses will be loaded really fast. Typically enterprise applications tend to have many static data choice lists, so it can be really good performance improvement to use shared application modules for such type of lists.
Controlling the number of database connections

Let’s move on to the second part of this article and look at the new \texttt{jbo.shared.txn} property available in Oracle JDeveloper 11g R2 (11.1.2.0.0).

When you check, how many database connections are opened for the sample application, will count three. The is one connection for each shared Application Module and one for the application itself.
You can however configure the initial number of opened database connections to be 2 (one for main application and another one to be shared between two shared application modules). For this, you use the `jbo.shared.txn` property.

Before configuring `jbo.shared.txn`, have a look into DataBinding.cpx, located in the ViewController project, which has the application ADF Business Component Data Control reference is defined.

```xml
<DataControlUsages>
  <BciDataControl id="HrModuleDataControl" Package="cx.redsaxvw.model"
    FactoryClass="oracle.adf.model.bc3j.DataControlFactoryImpl" SupportsTransactions="true"
    SupportsFindMode="true" SupportsRanges="true" SupportsDesertState="true"
    SupportsSortCollection="true" Configuration="HrModuleLocal" syncMode="Immediate"
    xmlns="http://xmlns.oracle.com/adf/datacontrol"/>
</DataControlUsages>
```

The Data Control reference in ADF usually points to the Local application module configuration (see highlighted text). However, when application modules are defined to be shared, the ADF framework internally will use a Shared configuration. This means, that the `jbo.shared.txn` property must not be set for the Local Application Module configuration, but the Shared.

For this, open the application module configuration in the HrCommonModel project. Select the shared configuration as in the image shown below. Then provide a name value to the `jbo.shared.txn` property, for example: SharedAMGroup. If you use the same `jbo.shared.txn` value in other shared application modules, then, at runtime, ADF Business Components nests all of them under the same transaction so only a single database connection is used.
How to improve ADF Business Components LOV performance using shared application modules

Repeat the same for OrgCommonModel project, specify jbo.shared.txn = SharedAMGroup.

When you run the application with the jbo.shared.txn property applied, only two database connections are being opened for the first application access – one for main application and another one for both shared application modules nested into single group.
How to improve ADF Business Components LOV performance using shared application modules

When a next user accesses the application, only one additional database connection is opened because the shared application module data is already available in memory from where it is reused.

Same behavior is repeated for third user, etc. – only one database connection will be added for main application module.
How to improve ADF Business Components LOV performance using shared application modules

Running the Sample

You can download the Oracle JDeveloper 11g R2 workspace for this example as sample #87 from the ADF Code Corner website:


Configure the database connection used by the workspace to access the Oracle HR schema.

In Oracle JDeveloper 11g R2 11.1.2.0.0, select the main.jsf page in the adfc-config.xml unbounded task flow in the ViewController project and choose Run from the context menu.

You can try to open application URL from different browsers, using different sessions – review generated log messages to see that SQL statements for shared static lists are generated only once, on first access by first user. Using WebLogic Console open dashboard screen to review currently opened database connections.

RELATED DOCUMENTATION

10.2.6 What You May Need to Know About Shared Application Modules and Connection Pooling
http://download.oracle.com/docs/cd/E16162_01/web.1112/e16182/bclookups.htm#BABCEIJF