

the “default” to the Parameter. These parameters are outside the scope of the custom table.

[0277] In the StoredProcedure Scenario, it is known if the default keyword will be applied to a parameter that does not have a default declared, and thus, a compile time error can be issued notifying the user that there is an unbound parameter that has no default.

[0278] In the Inline SQL scenario, there is no capability to validate the fact that more parameters are declared in the command text so the user will receive a runtime error that there are unbound parameters in the command.

[0279] Additional Parameter Guidelines for CUD

[0280] Insert Commands—when binding values to input parameters on InsertCommands, the CP must respect the setting of ForceNullOnInsert for the Map to the current CustomTable.

[0281] Update Commands—when binding values to the parameters in the command we will always bind all values even if they are unchanged rather than attempting to execute the procedure with the “default” keyword or null specified for that parameter.

[0282] Parameter Scoping in Inline Scenarios—if multiple instances are batched for execution on a particular Command, and the Command is Inline SQL, CP must account for the fact that there will be overlapping parameter names in the Command text.

[0283] CustomTable Narrower than CUD Result Set

[0284] In this scenario, the additional columns are simply ignored as they are in the Query case.

[0285] CustomTable Wider than CUD Result Set

[0286] In this case, only those values to the target domain for which has been returned in the CUD Command’s result set, are bound back.

SCENARIO EXAMPLES

[0287] Simple Condition

[0288] RSD file:

```
<Database Name="Northwind">
  <Schema Name="dbo">
    <CustomTables>
      <CustomTable Name="CustomerYTD">
        <BasedOn Name="Customers"
          AutoCommands="true"/>
        <Condition Column="ContactTitle"
          Value="Owner"/>
        <Condition Column="City"
          Value="Mexico D.F."/>
      </CustomTable>
```

-continued

```
</CustomTables>
</Schema>
</Database>
```

[0289] SingleComplexMapping

[0290] Suppose the user has an attribute on the Customer class that is Year-to-Date (YTD) sales for that customer. This must be calculated, as it is not stored on the Customer table. The user writes a parameterless UDF that returns all the customers with their YTD sales totals.

[0291] Using an ExtendedColumns feature, the Customers table is used as the BasedOn structure and then the salesYTD column is appended to the columns inherited from the Customers table. The QueryCommand then simply references the UDF (udf\_CustomerWithYTD) which has been custom-built for mapping such that the columns returned in the UDFs resultset are named identically to the columns on the Customer table and the salesYTD column specified in the ExtendedColumns. This naming takes advantage of default column binding and precludes the need for explicit bindings. Since a UDF can be used as a subselect, predicates from OPath are supported as if this was an Auto Command type.

[0292] RSD file:

```
<Database Name="Northwind">
  <Schema Name="dbo">
    <CustomTables>
      <CustomTable Name="CustomerYTD">
        <BasedOn Name="Customers"
          AutoCommands="true"/>
        <ExtendedColumns>
          <Column Name="salesYTD"
            SqlType="Integer"/>
        </ExtendedColumns>
        <QueryCommand>
          <CommandReference
            Name="udf_CustomerWithYTD"/>
        </QueryCommand>
        <CustomKey
          Name="pk_Customers_logical" >
          <ColumnRef
            Name="customerid" />
        </CustomKey>
      </CustomTable>
    </CustomTables>
  </Schema>
</Database>
```

[0293] Mapping File

[0294] The mapping for this scenario looks like the following:

```
<m:MappingSchema
  xmlns:m="http://schemas.microsoft.com/data/2002/09/28/mapping">
  <m:DataSources>
    <m:DataSource Type="SQL Server" Direction="Source">
      <m:Schema Location="Northwind.rsd"/>
```