WorkingWithNameMapping

The default behavior for the RDB DAS is to map database names to SDO names. So, if a CUTSOMER table is read from the database then each row will be represented as a CUSTOMER DataObject. Likewise, if the CUSTOMER tabel has columns (ID, LASTNAME, ADDRESS) then CUSTOMER DataObject instances will have properties (ID, LASTNAME, ADDRESS). Database Table names map to DataObject Type names and database column names map to DataObject property names.

This implicit mapping of database and SDO names woks well but some developers will want explicit name mapping that allows the database and SDO names to vary. One typical example is that the database might have all tables and columns conventionally named in all upercase but a Java developer might want to wrk with SDO Types and properties that are camelcased. So, database names CUSTOMER and LASTNAME might map to SDO names Customer and lastName.

The RDB DAS allows this type of explicit mapping via configuration (usually in the form of a XML config file). The following example illstrates the use of this feature:

```
   DAS das = DAS.FACTORY.createDAS(getConfig("BooksConfigWithAlias.xml"), getConnection());

   Command select = das.getCommand("get book by ID");
   select.setParameter(1, Integer.valueOf(1));

   DataObject root = select.executeQuery();
   String author = root.getString("Book[1]/Writer"));
```

Here is the associated configuration file:

```
   <Config xsi:noNamespaceSchemaLocation="http:///org.apache.tuscany.das.rdb/config.xsd" xmlns:xsi="http://www.
w3.org/2001/XMLSchema-instance">

       <Command name="get all books" SQL="SELECT * FROM BOOK" kind="Select"/>

       <Command name="get Cat in the Hat" SQL="SELECT * FROM BOOK WHERE NAME = 'Cat in the Hat'" kind="Select"/>

       <Command name="get book by ID" SQL="SELECT * FROM BOOK WHERE BOOK.BOOK_ID = ?" kind="Select"/>

       <Table tableName="BOOK" typeName="Book">
           <Column columnName="BOOK_ID" primaryKey="true"/>
           <Column columnName="AUTHOR" propertyName="Writer"/>
           <Column columnName="OCC" collision="true"/>
       </Table>

   </Config>
```

You can see that the Table definition provides "typeName" and "propertyName" attributes to override the defualt use of the all-uppercase database names. You can also see the use of these names in the example Java source.