

Database Kit Tips and Examples

This document contains tips and examples that can help developers solve specific problems they may encounter when using the Database Kit. Some understanding of the Database Kit is required, particularly with regard to the classes defined by the Kit and the layers into which the classes fall. Primary documentation of the Database Kit is provided on-line with NEXTSTEP Release 3 in **/NextLibrary/Documentation/NextDev/GeneralRef/04_DatabaseKit** and **/NextLibrary/Documentation/NextDev/Concepts/DatabaseKit**. Additional notes can be found in **/NextLibrary/Documentation/NextDev/ReleaseNotes/DBKit.rtf**.

Frequently Asked Questions

Database Kit Programming

Q: I put together a Database Kit program using InterfaceBuilder and it works fine in

InterfaceBuilder's test mode. When I compile this program and try to run it independently, an attention panel opens and indicates: "Cannot load Adaptor: Oracle Adaptor." Why?

Q: I build a Database Kit application and add **libdbkit_s.a** to the Libraries directory in ProjectBuilder. Why do I still get the following error message in the console:

```
Error loading /NextLibrary/Adaptors/SybaseAdaptor.adaptor/SybaseAdaptor  
rld(): Undefined symbols:
```

In GDB, the undefined symbols are reported as `_sys_nerr`, `_sys_errlist`, and `_send`.

A: In order for a Database Kit application to work, adaptor code must be made available to it. There are two ways to achieve this:

- 1) Hard-link an adaptor into your application. To do this, simply drag the adaptor you want onto the Libraries icon in ProjectBuilder's Files window. The adaptors provided by NeXT are located in **/NextLibrary/Adaptors**.

2) Dynamically load the adaptor code. To do this, you need to create a file in your project directory called **Makefile.preamble**, which contains the following line:

```
OTHER_LDFLAGS = -u libdbkit_s -u libNeXT_s -u libsys_s
```

Q: What are the pros and cons of dynamically loading adaptor code?

A:

Pros

You can write generic tools, without knowing which database they will be run against. Your application gets the latest and greatest version of adaptors, in case NeXT revises them.

Cons

Dynamically loading an adaptor slows down program startup (often by several seconds)

Q: How can I get hold of the DBBinder used by my application's DBDatabase object?

A: You can't. If you want to use DBBinders in your application, you have to create them yourself. See **/NextDeveloper/Examples/DatabaseKit/Binder** for some usage of a DBBinder.

Q: How can I view the SQL queries generated in my program?

A: You can trace the actual SQL queries generated by the Database Kit by having an object becoming the DBDatabase object's delegate. The delegate object must implement the **db:willEvaluateString:usingBinder:** method. For more details, see **Controller.m** in the **/NextDeveloper/Examples/DatabaseKit/AddressBook** example.

Q: What are the pros and cons of using one DBModule vs. two to implement insert/save in the case of a one-to-many relationship?

A: In case of 2 DBModules (one for the master table and one for the detail table), you don't have to find the detail fetchgroup programmatically. You can insert a new record into the detail module by simply connecting a button to the detail module **insertNewRecord:** method. When saving, you need to save the master module first, then save the detail module. See the

/NextDeveloper/Examples/DatabaseKit/OracleDemo for an example of two modules.

In case of a single DBModule (representing the master table), you need to find the detail fetchgroup programmatically in order to insert a new record. However, you need to do only *one* save operation on the module, and it saves all changes from the root fetchgroup as well as the detail fetchgroup(s). The module save operation groups both fetchgroup saves as a single transaction.

The Database Kit and Interface Builder

Q: How can I synchronize a master DBModule with a detail DBModule ?

A: Just drag and drop the one-to-many relationship in the master DBModule onto the icon of the detail DBModule in the nib file. See

/NextDeveloper/Examples/DatabaseKit/OracleDemo for more details on how to synchronize two modules.

Q: How do I connect a property of my table to a PopUpList object? I've tried connecting an attribute of my table to the PopUpList but nothing happens. The PopUpList still shows item 1, item 2, etc.

A: PopUpList connections can be made only to a field "x.y.z" where x is an entity, y is a one-to-one relationship, and z is a leaf-node field that is either a number or a string. For example, if you connect the attribute "name" from your Employee table to the PopUp, it only shows item1, item 2, etc. However, if you connect "Employee.Department.name", it shows all the names in the department.

Q: Which Application Kit objects are supported by the Database Kit?

A: The Database Kit contains various subclasses of DBAssociation designed to connect DBFetchGroups to certain Application Kit objects. Those Application Kit objects are:

TextField

Slider

Matrix of Cells or FormCells

NXBrowser

Text Object in a ScrollView

Button connected to a PopUp menu

Note:

Radio buttons and switches are not supported. See the RadioAssociation MiniExample for a workaround (radio buttons only).

Oracle Adaptor

Q: The model that I built with DBModeler shows only the views and tables that I own. How can I access all the views and tables in an Oracle database?

A: By default, DBModeler loads only those tables and views owned by the current user. However, you can extend the list of users for whom tables and views should be loaded. To do

so, add the following to the defaults database—for example, to show the tables and views owned by the users David, Joe, Tony:

```
localhost> dwrite OracleAdaptor OracleTableOwners "'DAVID','JOE','TONY'"
```

The user names have to be quoted and the current user name has to be included as well. In order to load system tables, you can simply add the users 'SYSTEM' and 'SYS'. You can modify or change your list by using **dremove**, then **dwrite**. See the UNIX man pages for more information on **dread**, **dremove**, and **dwrite**. Note that this answer does not cover issues about access rights to system tables. Please refer to the Oracle manuals for further information.

Q: What should I do on my NEXTSTEP system in order to use my Oracle server with the Database Kit?

A: You need to do the following:
On the server side:

- 1) Check if oraserv is running.
- 2) Check if you have a file oratab under /etc. The oratab file has entries of the following format:

`$ORACLE_SID:$ORACLE_HOME:<N|Y>`

The entries in that file must match the entries you give to DBModeler. See item#4d. Specify "Y" so that the server can be started with dbstart at system boot up time.

On the client side:

- 3) Look at your /etc/services file:
 - a) Check with the Oracle DBA to verify that the database is using socket 1525 (this is the default). Enter the following into /etc/services:

`orasrv 1525/tcp oracle`

b) Niload the Oracle services. Be sure to check with the Oracle DBA to verify that the database is using socket 1525 (this is the default). If you want your entire network of NeXT machines to see the server, niload the entry in the root NetInfo domain on the master NetInfo server.

```
localhost> su  
localhost# niload services / </etc/services
```

- 4) To access an Oracle server over TCP/IP.
 - a) Launch /NextDeveloper/Apps/DBModeler.app
 - b) Select the OracleAdaptor from the Preferences menu item.
 - c) Choose "New Model."
 - d) For the Oracle login panel, enter the appropriate information:

server id: <whatever ORACLE_SID is set to>

host: <server machine name>

login: <user login>

password: <user password>

e) Save the model in **~/Library/Databases, /LocalLibrary/Databases, /NextLibrary/Databases** or **/usr/local/Databases**. These are the default search paths that InterfaceBuilder uses to load models into the DatabaseKit palette.

f) From InterfaceBuilder, drag a DBModule palette (the lower left icon in the DBKit palette view) into the IB Objects suitcase. This operation creates an instance of DBModule. You can look at your model via the DBModule inspector and start creating your DBKit application.

Some Known Problems

Sorting Records

Sorting cannot be set from InterfaceBuilder. To specify a sort order, you use the DBRecordList method **addRetrieveOrder:(DBRetrieveOrder)*anOrder* for:(id<DBProperties>)*aProperty*;**

A problem is that when you reset the DBRecordList using the **setProperties:** method, it also resets the sort order. Both DBFetchGroup and DBModule send a **setProperties:** message before fetching. So, if you try to interpose in the fetch with a custom object, set up the sort order, then fetch, the sort order won't be set anymore. The way to get around this is to be the DBFetchGroup's delegate and respond to the **fetchGroupWillFetch:** message, and to set up the sort order on the sender's DBRecordList.

See the [/NextDeveloper/Examples/DatabaseKit/TableView](#) which changes the sort order every time you move the columns of the DBTableView.

Sybase Adaptor

- The adaptor doesn't infer relationships from the database.
- The adaptor doesn't read in system tables.

Oracle Adaptor

- The adaptor doesn't handle sequence numbers.
- The adaptor doesn't read relationships from the database.
- The adaptor doesn't handle catalog of databases.

DBModeler

- DBModeler doesn't import relationship information from the database into the dbmodels.

Keys

- You cannot update the primary key of your record. The only way is to delete the old record, do a save operation, and insert the new record with the modified key.

Qualifiers

- In general, all expressions in the qualifier need to be attributes or relationships coming from the qualifier's root entity. Only one-to-one relationship attributes can be built into the qualifier. There is a known bug however with fetching with a qualifier built from one-to-one relationship attributes. As a workaround, you need to invoke the DBFetchGroup method **addExpression:** or create an expression implicitly using a connection to the File Owner. This bug has been fixed in Release 3.2.

- To qualify records fetched from a detail DBFetchGroup in case of a one-to-many relationship, you must use custom code. See the [**/NextDeveloper/Examples/DatabaseKit/AssociationSybase**](#) or [**AssociationOracle**](#) for more details.

- There is a known problem with parenthesizing qualifiers. As a workaround, follow the steps below:

Given the following qualifier:

```
[qual1 initForEntity:e FromDescription:"name = 'SMITH' OR name = 'JONES'"];
```

```
[qual1 addDescription:" AND age < 10 OR age > 20"];
```

This generates an SQL with no parenthesized grouping:

```
"name = 'SMITH' OR name = 'JONES' age < 10 OR age > 20"
```

The workaround is as follows:

```
[qual1 initForEntity:e FromDescription:"name = 'SMITH' OR name = 'JONES'"];  
[qual2 initForEntity:e FromDescription:"(%@) AND (age < 10 OR age > 20)", qual1];
```

or more generally:

```
[qual1 initForEntity:e FromDescription:"name = 'SMITH' OR name = 'JONES'"];  
[qual2 initForEntity:e FromDescription:"age < 10 OR age > 20"];  
[qual3 initForEntity:e FromDescription:"(%@) AND (%@)", qual1, qual2];
```

These qualifiers then generate:

```
"(name = 'SMITH' OR name = 'JONES') AND (age < 10 OR age > 20)"
```

Update

The Database Kit doesn't support modifying columns that are made of derived data (i.e. calculated expressions and the results of a join). For example, the following scenario does not work:

You have a module through which you fetch data for a few properties in that entity, including one property through the join. The fetch works fine, but when you update using **saveChanges:**, you get an attention panel titled "Database" with the message "Error occurred during transaction". The workaround is to add another module and fetch the data previously accessed through the join directly. Then, you'll have no problems resaving.

Example Reference Guide How To's

In order to better utilize the Database Kit Examples and MiniExamples, we have started a

preliminary categorization of their functionalities, which helps you quickly find a topic of interest. To search for all functionalities related to one single example, you can do a search on that particular example name. To search on a particular method name, we strongly recommend that you index the MiniExamples referred in this paper using Digital Librarian. Note that the /NextDeveloper/Examples are already indexed.

Note: The MiniExamples are available from NeXTanswers via anonymous ftp (ftp.next.com) or via email (send mail to nextanswers with the word "help" in the subject).

Attributes and Entities

How to get all the attributes from an entity and the sub-attributes that belong to a one-to-one relationship of that same entity

See /NextDeveloper/Examples/DatabaseKit/TableView (TableViewCell.m, initTableView:)

Formatting

How to display 1 and 0 boolean data in form of strings YES/NO

MiniExample BooleanFormatter(BooleanFormatter.m)

How to subclass DBEditableFormatter to do simple text formatting

See /NextDeveloper/Examples/DatabaseKit/Formatter (DateFormatter.[hm] and DataTFCell.[hm])

DBAssociation

How to set up a new DBAssociation between the master fetchgroup and the detail fetchgroup so that the detail fetchgroup can have a qualifier attached to it

See /NextDeveloper/Examples/DatabaseKit/AssociationOracle for Oracle users and AssociationSybase for Sybase users (QualifiedAssociation.[hm])

How to make a custom view follow the DBCustomAssociation protocol to set/getValue, etc.

See /NextDeveloper/Examples/DatabaseKit/BarChart (ChartOfMatrix.m)

How to connect radio buttons to an integer type attribute of your DBModule and make them

communicate with the DBModule

MiniExample RadioAssociation (RadioAssociation.m)

DBBinder

How to initialize a DBBinder with an Objective-C prototype object

/NextDeveloper/Examples/DatabaseKit/Binder (BinderHandler.m)

How to use a DBBinder to evaluate some arbitrary SQL query

/NextDeveloper/Examples/DatabaseKit/Evaluator (only available with Release 3.2)

DBFetchGroup

How to make a fetch group become the data source of the tableview

See /NextDeveloper/Examples/DatabaseKit/TableView (TableViewController.m, initTableView:)

How to change the sort order using the delegate method fetchGroupWillFetch:

See /NextDeveloper/Examples/DatabaseKit/TableView (method fetchGroupWillFetch: in

TableViewCell.m)

How to insert/update/delete a new record via a DBFetchGroup

/NextDeveloper/Examples/DatabaseKit/PubsDemo (Controller.m,
fetchGroup:didInsertRecordAt:)

DBQualifier

How to build queries using the DBQualifier and select records based on qualifiers

/NextDeveloper/Examples/DatabaseKit/Binder (BinderHandler.m, findByName:)

How to build qualifiers that refer to attributes of a one-to-one relationship

MiniExample QuickApp (Controller.m)

How to do a Fetch with a DBQualifier

/NextDeveloper/Examples/DatabaseKit/PubsDemo (Controller.m, buildSelectQualifier:)

DBRecordList

How to fetch the data from a database using a DBRecordList

See /NextDeveloper/Examples/DatabaseKit/AddressBook(AddressView.m, loadCellsFrom:)

How to add/delete/update a new record to a database via a DBRecordList

See /NextDeveloper/Examples/DatabaseKit/AddressBook (AddressView.m)

How to view all the values inside a record in a DBRecordList using a DBValue

See /NextDeveloper/Examples/DatabaseKit/AddressBook (AddressView.m)

DBTableView

How to set up your own tableview without using Interface Builder

See /NextDeveloper/Examples/DatabaseKit/TableView (TableViewController.m)

NeXT, NeXTSTEP, NeXTanswers, Application Kit, Digital Librarian, and Interface Builder are trademarks of NeXT Computer, Inc. UNIX is a registered trademark of UNIX Systems Laboratories.

QA891

Valid for 3.1, 3.2