Ashpool - XML Database Users Guide

Rob Rohan

Revision History
Revision 3rc3 2003-04-20
Draft

Table of Contents

Introduction 1
Starting Ashpool Basic 1
Starting Ashpool Advanced 2
DBMS Commands 2
Scripts 3
Running Scripts 3
Writing Scripts 3
Using the JDBC Driver 4
Using Ashpool in Java 4
Using Ashpool with SQuirreL 5
Using Ashpool with ColdFusionMX 6

Introduction

...TBD...

Starting Ashpool Basic

To start Aspool do the following:

Procedure 1. Starting Ashpool

1. If not done already, download the JRE 1.4 (or greater), and install it.
2. Download Ashpool.jar and note the directory where the file is saved. For example, `C:\ashpool`
3. Open a command window, and navigate to the directory where `Ashpool.jar` is located.
   `C:\>`**cd ashpool**
4. Create a sub directory in the directory where the `Ashpool.jar` is located. For our example we'll call it datas-"tore" -
   `C:\ashpool>`**mkdir datastore**
5. Start Ashpool by typing the following at the command line
   `C:\ashpool>`**java -jar Ashpool.jar datastore**

6. If all went well, you sould now see the ashpool prompt - `Ashpool~#`.
   You can add, create, and query XML documents in the `datastore`.

Starting Ashpool Advanced

...TBD... **java -jar Ashpool.jar** [directory] [scriptfile]
If you are really going to put Ashpool to the test, try giving it more memory to play with using the command:
java -Xmx128m -jar Ashpool.jar [directory]
or increase the 128 to a higher number to give Ashpool more memory.

DBMS Commands

The following is a table of the DBMS built in commands.

> **Note**
> The commands that start with *SELECT* can be used outside of the DBMS - by JDBC connections for example

Table 1. Ashpool DBMS Commands

SQL Statements;	Runs SQL statements. **Note** Statements will not run until a ; char is entered.
help;	Displays a quick reference of commands
quit;	Exits the DBMS, or terminates a script
sys;	Displays current system environment. Memory usage, xslt processor, etc.
gc;	Force the JVM to try to dispose of old objects (Garbage Collection).
done;	Put the DBMS into interactive mode. This command should be at the end of any scripts that are run during an interactive session.
! [command];	Execute a system command. Mostly used in scripts, but can be used during an interactive session.
@ [file name];	Loads and runs a script during an interactive session.
echo [string];	echo text to standard out
-- [comments];	Ignored line. Used to comment scripts.
select test;	Sends a simple pre-made result. Used to test connections.

select tables;	Shows a list of tables in the datastore.
select types;	Show all currently supported data types.
select columns [table]	Shows column information about the requested table.

Scripts

Scripts can be very helpful when trying to write complicated, or repetitive SQL statments. Ashpool let's you write two kinds of scripts - those that run in interactive mode and those that are standalone.

Running Scripts

The difference between standalone and interactive scripts is almost nill, save one item. Scripts that run in interactive mode *must* end with a **done;** command, and those that run stand alone should end with a **quit;** command.

done; sets control back to the envoker while **quit;** exits Ashpool altogether. Once a script has been written and saved, the script can be envoked by typing **@** [script name]; at the Ashpool prompt. The following are some script envoking examples.

Example 1.

`Ashpool#` **@s;**
or with a path
`Ashpool#` **@C:\myscript.ash;**
non interactive
`C:\ashpool>` **java -jar Ashpool.jar datastore/ breaktable.ash**

Writing Scripts

Ashpool script syntax is for the most part SQL. All of the commands available in interactive mode are available in scripts. The following is a short example of an interactive script.

Example 2.

```
-- fake script that does nothing;
-- this is used while in Ashpool;
-- for fun garbage collect first;
gc;
-- get a temporary table of something;
insert into ~list
select
    firstname,
    email,
    logflag,
    userid
from contact_tbl
join log_tbl on itemid = userid;
-- use the system to make a copy of the file;
!cp datastore/~list.xml /home/dude/report.xml
-- another for a hoot;
```

```
insert into ~half
select
    firstname,
    email
from ~list
where logflag = 'true';
drop table ~list;
-- don't forget done or you'll be sad;
done;
```

The fake script creates a temporary table out of some XML documents (tables) and then copies the results out of the datastore. It then creates a smaller temporary table from the original and then returns control to the DBMS.

To make this script non-interactive, simply change **done;** to **quit;** and run it from outside the DBMS by running a command like
$ **java -jar Ashpool.jar** [datastore] [scriptname]

Here is a helpful tip to writing scripts. While you are in Ashpool you can envoke an external editor to create a script. For example, on windows you could issue the following command:
`Ashpool~#` **!notepad** r.txt;
After writing the script and saving, you could then run the script in Ashpool by running the command:
`Ashpool~#` **@r.txt**;

Using the JDBC Driver

Ashpool has a built in JDBC driver that can be used to connect to Ashpool from any JDBC compliant program. Please note, Ashpool's JDBC driver - like Ashpool - is not commerical grade software yet. However, it is still a very powerful and useful program.

Refer to the documentation for whatever program you are trying to connect to to get the specifics of how to setup JDBC drivers. The following information is what is commonly needed:
Driver: **com.rohanclan.ashpool.jdbc.Driver**
URL: **jdbc:ashpool://file://** [path]

Example 3.

URI on windows could look like
jdbc:ashpool://file://C:\thisdir\thatdir\myxmldir
and on Unix the URI could look like
jdbc:ashpool://file:///thisdir/thatdir/myxmldir

Some programs offer a verify feature to check to see if database connections are valid. Ashpool might fail these checks even though the connection is working. I encourage you to try to query Ashpool even if the program's verify feature says it is invalid.

The key to using Ashpool's JDBC driver is in putting Ashpool.jar in the classpath of the application that will use Ashpool. Ashpool is not (yet) a server so when it runs it will be running in the same space as the program. This is important to note because Ashpool will be limited by the memory supplied to the calling program.

> **Note**
> Ashpool.jar *must* be in the classpath of the program that will be calling it.

Using Ashpool in Java

Using Ashpool in a Java program can be quite powerful. Saving and updating complex configuation files is a snap when using Ashpool. Imagine being able to say **select * from globals** and having all your programs display needs, or **update lookandfeel set lnf = 'windows'** to save settings all while keeping all the benifits of XML. Aspool also opens up the door for 3rd form normalized configuration (or other types) of files.

Enough yapping. Setting up Aspool is the same as setting up any JDBC driver. The following is an example program that should get you started.

Example 4. JDBC Test Program

```
import java.sql.*;
import javax.sql.*;
public class JdbcTest {
  public void testSource(String url){
   try{
     Driver driver = (Driver)Class.forName("com.rohanclan.ashpool.jdbc.Driver").newInstance();
     DriverManager.registerDriver(driver);
     Connection conn = DriverManager.getConnection(url,null);
     Statement stmt = conn.createStatement();
     java.sql.ResultSet rs;
     rs = stmt.executeQuery("select tables;");
     while(rs.next()){
      System.out.println(rs.getString(3));
     }
   }catch(Exception e){
    System.err.println("testSource: " + e.toString());
    e.printStackTrace(System.err);
   }
  }
  public static void main(String args[]){
   JdbcTest boot = new JdbcTest();
   if(args.length < 1){
    System.out.println("Usage: JdbcTest <constring>");
    System.exit(0);
   }
   boot.testSource(args[0]);
  }
}
```

Assuming C:\myxml is a directory with xml documents (on a windows system), one could test the above program with the command: **java -cp Ashpoo.jar;./ JdbcTest jdbc:ashpool:file//C:\myxml** The program should list the names of any files in the directory with an xml extension.

Using Ashpool with SQuirreL

As stated above the most important part of getting Ashpool to work with another program is to get Ashpool into the class path of the program that is going to use it. So, first we'll setup SQuirreL to see Ashpool then we will use the Ashpool driver.

Procedure 2. Setting up SQuirrel

1. Shutdown SQuirrel if it's open

2. Add Ashpool.jar to the lib sub-directory of SQuirreL. For example, `C:\SQuirreL SQL Client\lib` on a Windows system

3. Startup SQuirrel, and select new driver from the Drivers menu on the top menu bar.

4. Highlight Ashpool.jar in the list of files and press the List Drivers button. In the Name text box type “Ashpool XML” and in the Example URL type **jdbc:ashpool:file://[file]**

5. There should now be an Ashpool XML listing in your drivers box on the SQurriel desktop. To use driver select Alias then New Alias from the menu bar, choose the Ashpool XML driver and fill in the requested information.

Note
Username and password are ignored at this time.

Using Ashpool with ColdFusionMX

To add Ashpool to ColdFusionMX we need to add Ashpool.jar to the list of loaded jars and then setup a datasource.

Note
Ashpool needs JRE 1.4 or higher. ColdFusionMX ships with an older version of the JRE. If you want to use Ashpool, you'll need to upgrade your JRE first.

Procedure 3. Setting up ColdFusionMX

1. Shutdown ColdFusionMX if running

2. Add Ashpool.jar to the cfx directory of ColdFusion. On linux the directory is /opt/coldfusionmx/cfx/java by default. While Ashpool is *not* a custom tag, ptting it in this directory is ensures it will be loaded at startup.

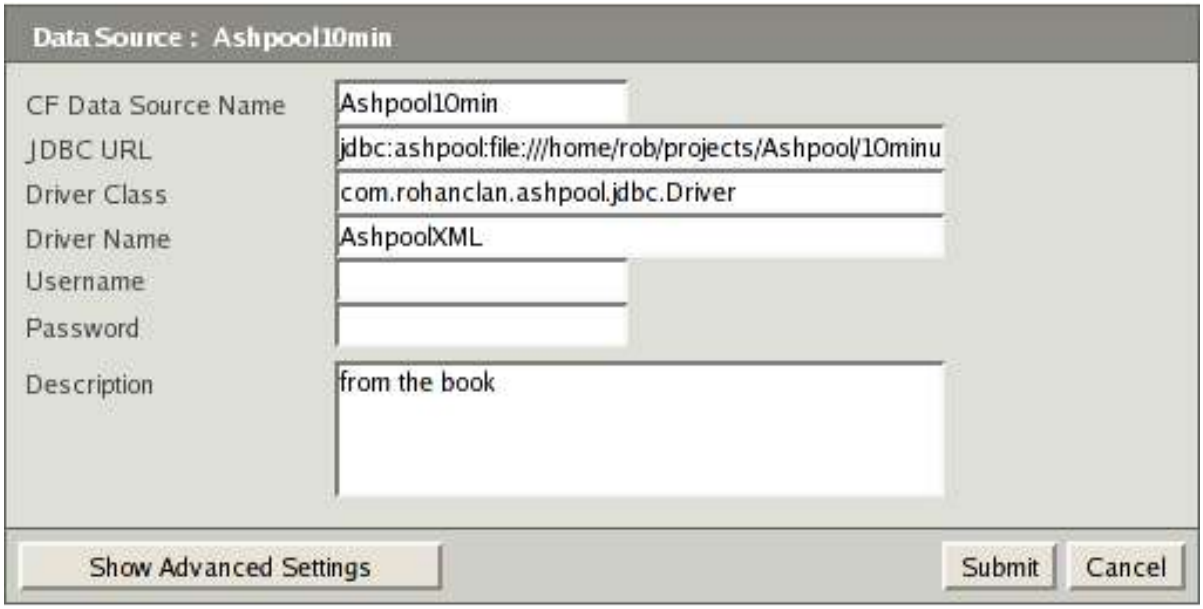
3. Start ColdFusionMX, and log into the /cfide/administrator.

4. Choose Data Sources on the left

5. In the Add New Data Sources box at the top, enter a name for this datasource and choose Other from the drop down list of drivers.

6. Enter the requred information. The JDBC URL should be of the form **jdbc:ashpool:file://[path]** and the driver class should be **com.rohanclan.ashpool.jdbc.Driver**. You can name the Driver anything you like - Aspool XML is fitting.

Note
Username and password are ignored at this time.



7. Click submit. You should now have a usable datasource.

Note

The verify feature will fail no matter what. Simply try to query the datasource with a test page to test connection