

HowToGetResultsetFromSybaseStoredProcedure

How-to call Sybase stored procedures returning Resultset with ESQL tags in XSP

What you will get from this page

You'll learn how to get resultsets from Sybase stored procedures using ESQL tags in XSP pages. Cocoon User documentation [explains](#) how to do it using esql:call tag. Here's another (imho, simpler) way for Sybase database. Additionally, Sybase Connection setup is discussed briefly.

Your basic skills

- Basic Cocoon configuration: cocoon.xconf, web.xml
- Resin 2.x or other servlet container
- Basic XSP and ESQL
- Basic T-SQL for Sybase

Technical prerequisites

- Cocoon 2.1.4 deployed on Resin 2.x or other servlet container.
- Sybase jConnect 5.5 JDBC drivers (jconn2.jar) are configured in pools for Cocoon.
- Sybase 12.5 RDBMS.

Links to other information sources

- [Cocoon ESQL Documentation](#)
 - [Resin 2.1.12](#) distribution
 - [Sybase jConnect 4.5 and 5.5 JDBC driver](#)
-

esql:query tag

In order to retrieve a resultset from a stored procedure, use the esql:query tag as shown below:

```
<esql:query>

exec my_stored_proc
<esql:parameter><xsp:expr>"public"</xsp:expr></esql:parameter>

</esql:query>
```

That's it!

Now I'll explain this in more details.

Setup Sybase database and datasource

Install Cocoon on your Servlet container following instructions [here](#). I am using Resin 2.1.12.

Copy jConn2.jar into your /WEB-INF/lib directory of a deployed Cocoon application.

Configure web.xml to pre-load the JDBC driver class as per instructions <http://cocoon.apache.org/2.1/developing/datasources.html#Installing+the+Driver>. Sybase driver Class name is: com.sybase.jdbc2.jdbc.SybDriver. Note, that I had a problem when there were more than one driver in load-class parameter. So, my web.xml has only Sybase driver and looks like this:

```
<init-param>

<param-name>load-class</param-name>

<param-value>

<!-- For Sybase Driver: -->

com.sybase.jdbc2.jdbc.SybDriver

</param-value>

</init-param>
```

Now, you have to configure a datasource in the cocoon.xconf file following instructions [here](#). Sybase JDBC url looks like this: jdbc:sybase:Tds:host:port/database?user=username?password=password. It's the longest url variant, usually user and password are set separately.

Here's my datasource:

```
<jdbc logger="argyn.syb" name="syb">
<pool-controller max="10" min="5"/>
<dburl>jdbc:sybase:Tds:localhost:5001/test_db</dburl>BR
<user>argyn</user>
<password>crap</password>
</jdbc>
```

In order to test our setup, create a stored procedure like shown below:

```
CREATE PROCEDURE my_stored_proc
@NAME varchar(30)=null
AS
BEGIN
select * from sysusers where name=@NAME
END
```

Sample XSP page

Let's create our test XSP page. If you installed default Cocoon.war built by command "build webapp", then you should have a directory with sample code at \samples\databases\xsp\ . Create here one more file called test.xsp. Here's what you should put inside:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<xsp:page language="java"
xmlns:xsp="http://apache.org/xsp"BR
xmlns:esql="http://apache.org/cocoon/SQL/v2">BR
<page>
<title>Argyn's sample</title>
<content>
<esql:connection>
<esql:pool>syb</esql:pool>
<para>
System user info below:
</para>
<esql:execute-query>
<esql:query>
exec my_stored_proc
<esql:parameter><xsp:expr>"public"</xsp:expr></esql:parameter>
</esql:query>
<esql:results>
<esql:row-results>
<esql:get-columns/>
</esql:row-results>
</esql:results>
</esql:execute-query>
```

<hr/>

</esql:connection>

</content>

</page>

</xsp:page>

The trick is that if you call a stored procedure with "exec *stp_name* ?,..." syntax, then you can use esql:query tag just as you would do it with a any "select * from..." query. This works for Sybase.

Now, if you hit the page with <http://localhost:8080/cocoon/samples/databases/xsp/test> (port number and cocoon context may be different in your setup), you should see something like this:

Argyn's sample

System user info below:

-2 0 0 public

page metadata

- AUTHOR: [Argyn](#)
- AUTHOR-CONTACT: argyn@cox.net