

Getting the Rtf file from the Database without Accessing Oracle Applications.

In XMLP all the files such as .xml, .rtf , PDF etc are stored in the BLOB column in xdo_lobs table.

This Document briefs you on the different ways of retrieving the Rtf file from the Database. There are 2 ways of doing this.

- 1) Retrieving and Storing a BLOB column in a file using UTL_FILE Package.
- 2) Retrieving and Storing a BLOB column in a file using Java Stored Procedure.

Retrieving and Storing a BLOB column in a file using UTL_FILE Package:

We can retrieve only the RTF files from the Template Manager but using BLOB column we can retrieve even the .xsl and files.

The following function shows how to retrieve the files stored in the blob column.

Step 1: Store the BLOB column FILE_DATA of xdo_lobs table in a variable of type BLOB.

```
select file_data
into xmlstr
from xdo_lobs
where file_name='PO_STANDARD_PO.rtf';
```

Step 2: Store the BLOB data in the variable in a file using UTL_FILE package.

```
--to write blob column data to a file
DECLARE
v_file utl_file.file_type;
line_buf RAW(32767);
maxbufsize BINARY_INTEGER := 32767;
amount BINARY_INTEGER;
offset varchar2(40);
xmlstr BLOB;
dir varchar2(50) := 'ATTACHMENTS';
filename varchar2(50) := 'PO_STANDARD_PO.rtf';
bsize number;
BEGIN
v_file := utl_file.fopen(dir,filename,'w',maxbufsize);
select file_data
into xmlstr
from xdo_lobs
where file_name='PO_STANDARD_PO.rtf';
```

```
bsize := dbms_lob.getLength(xmlstr);
dbms_output.put_line(bsize);
amount := maxbufsize;
offset := 1;
while (amount >= maxbufsize)
loop
  DBMS_LOB.read(xmlstr, amount, offset, line_buf);
  offset := offset + amount;
  utl_file.put_raw(v_file, line_buf);
  utl_file.fflush(v_file);
end loop;
--close context
utl_file.fclose(v_file);
END;
/
```

Retrieving and Storing a BLOB column in a file using Java Stored Procedure:

Another way is to use a Java Stored Procedure to retrieve the BLOB column.

Step1: Load the java stored procedure into the database

CREATE OR REPLACE JAVA SOURCE NAMED "BlobHandler" AS

```
import java.lang.*;
import java.sql.*;
import oracle.sql.*;
import java.io.*;

public class BlobHandler
{

  public static void ExportBlob(String myFile, BLOB myBlob) throws
  Exception
  {
    // Bind the image object to the database object
    // Open streams for the output file and the blob
    File binaryFile = new File(myFile);
    FileOutputStream outputStream = new FileOutputStream(binaryFile);
    InputStream inputStream = myBlob.getBinaryStream();

    // Get the optimum buffer size and use this to create the read/write buffer
    int size = myBlob.getBufferSize();
    byte[] buffer = new byte[size];
    int length = -1;

    // Transfer the data
    while ((length = inputStream.read(buffer)) != -1)
```

```
{
  outputStream.write(buffer, 0, length);
  outputStream.flush();
}

// Close everything down
inStream.close();
outStream.close();
}
};
/
```

Step2: Compile the loaded java stored procedure code

```
ALTER java source "BlobHandler" compile;
show errors java source "BlobHandler"
```

Step3: Create a PL/SQL procedure which maps the java stored procedure

```
CREATE OR REPLACE PROCEDURE ExportBlob (p_file IN VARCHAR2,
                                         p_blob IN BLOB)
AS LANGUAGE JAVA
NAME 'BlobHandler.ExportBlob(java.lang.String, oracle.sql.BLOB)';
/
```

Step4: Grant to scott in order to write to os files

```
EXEC Dbms_Java.Grant_Permission( 'APPS', 'java.io.FilePermission', '*', 'read ,write, execute,
delete');
```

Step5: Execute the Java Stored procedure

```
declare
  xmlstr blob;
begin
  select file_data
  into xmlstr
  from xdo_lobs
  where file_name='onepage.rtf';
  ExportBlob ('/data/tst1/attachment/sample.rtf', xmlstr);
end;
/
```