Oracle *inter*Media Image Quick Start

Relational Interface

Introduction

Oracle *inter*Media ("*inter*Media") is a feature that enables Oracle Database to store, manage, and retrieve images, audio, video, or other heterogeneous media data in an integrated fashion with other enterprise information. Oracle *inter*Media extends Oracle Database reliability, availability, and data management to multimedia content in traditional, Internet, electronic commerce, and media-rich applications.

This article provides simple PL/SQL examples that upload, store, manipulate, and export image data inside a database using Oracle *inter*Media's relational interface and a table with a BLOB column. Some common pitfalls are also highlighted. We assume only Oracle Database release 9*i* or later with Oracle *inter*Media installed (the default configuration provided by Oracle Universal Installer).

The following examples will show how to store images within the database in BLOB columns so that the image data is stored in database tablespaces. Oracle *inter*Media image also supports BFILEs (pointers to files that reside on the filesystem), but this article will not demonstrate the use of BFILEs. Note that BFILEs are read only so they can only be used as the **source** for image processing operations (i.e. you can process from a BFILE but you can't process into a BFILE).

NOTE: While the relational interface is a standard and supported part of Oracle *inter*Media in Oracle Database 9*i* and 10*g* (OTN download for 8*i*), the recommended way for storing media in Oracle is the *inter*Media object interface. The *inter*Media object types are recommended because they are self-describing and easy for applications and other Oracle tools (e.g. BC4J, Code Wizard, and many others) to understand. If you use BLOBs in your application instead of *inter*Media object types, the knowledge of what is in the BLOB must be hard coded into the application.

NOTE: Access to an administrative account is required in order to grant the necessary file system privileges. In the following examples, you should change the command `connect / as sysdba` to the appropriate `connect username/password as sysdba` for your site. The following examples also connect to the database using `connect scott/tiger`, which you should change to an actual user name and password on your system. You should also modify the definition of `IMGDIR` to point at the directory where you have downloaded the three sample image files `goats.gif`, `flowers.jpg`, and `dummy.dcm`.

Creating a Table with an Image BLOB column

First, we create a simple table with six columns: a numeric identifier (`id`), image width (`width`), image height (`height`), the size of the image data (`contentLength`), the mime-type of the image (`mimeType`), and a Binary Large OBject "BLOB" to hold the image itself (`image_blob`).

```
connect scott/tiger
create table image_blob_table (id            number primary key,
                               width         integer,
                               height        integer,
                               contentLength integer,
                               mimeType      varchar2(20),
                               image_blob    BLOB);
```

Importing Images into the Database

This section shows how to bring images from the file system into the newly created `image_blob_table`. Note that all *inter*Media procedures are defined in the ORDSYS schema.

1. Create a *directory* object within the database that points to the file system directory that contains the sample image files. This is the directory where you saved the image files included with this quickstart.

```
connect / as sysdba
create or replace directory imagedir as '/home/alamb/quickstart/';
-- For Windows:
-- create or replace directory imagedir as 'c:\quickstart';
grant read on directory imagedir to scott;
```

2. Create a PL/SQL procedure `image_blob_import()` that inserts a new row into `image_blob_table` and then imports the image data into the newly created BLOB locator.

```
connect scott/tiger
create or replace procedure image_blob_import(dest_id number, filename varchar2) is
  img_blob BLOB;
  ctx raw(64) := null;
begin
   delete from image_blob_table where id = dest_id;
   insert into image_blob_table (id, image_blob) values (dest_id, empty_blob())
     returning image_blob into img_blob;

   ORDSYS.ORDImage.importFrom(ctx, img_blob, 'file', 'IMAGEDIR', filename);
   update image_blob_table set image_blob=img_blob where id=dest_id;
end;
/
```

3. Call the newly created procedure to import 2 sample image files.

```
call image_blob_import(1,'flowers.jpg');
call image_blob_import(2,'goats.gif');
```

NOTE: The directory object is named `IMAGEDIR` (in uppercase letters) even if it was created with upper or lower case letters. Thus the command `ORDSYS.ORDImage.importFrom(ctx, img_blob, 'file', 'imagedir', filename);` **will not work** and the following error is returned.

```
ORA-22285: non-existent directory or file for FILEOPEN operation error.
```

Populating height, width, contentLength and mimeType in image_blob_table

Once the image data has been imported from the file system into `image_blob_table`, the database does not know what the binary bytes in the `image_blob` BLOB column represent. Since the *inter*Media object interface is not being used, the application itself must contain knowledge that the BLOB column holds image data and must manage and associate image metadata explicitly. In the following example, we show how to use the `ORDSYS.ORDImage.getProperties()` procedure to extract the images' properties and update the metadata columns appropriately.

```
connect scott/tiger
create or replace procedure image_blob_getproperties is
  unused_attributes        CLOB;
  img_mimeType             varchar2(32);
  img_width                integer;
  img_height               integer;
  img_contentLength        integer;
  unused_fileFormat        varchar2(32);
  unused_contentFormat     varchar2(32);
  unused_compressionFormat varchar2(32);
begin
 for rec in (select id, image_blob from image_blob_table where mimeType is null) loop
     ORDSYS.ORDImage.getProperties(rec.image_blob,
                                   unused_attributes,
                                   img_mimeType,
                                   img_width,
                                   img_height,
                                   unused_fileFormat,
                                   unused_compressionFormat,
                                   unused_contentFormat,
                                   img_contentLength);
     update image_blob_table
       set width=img_width,
           height=img_height,
           contentLength=img_contentLength,
           mimeType = img_mimeType
       where id=rec.id;
  end loop;
  commit;
end;
/

call image_blob_getProperties();
```

NOTE: If the image data that is in the `image_blob` column is not one of *inter*Media's supported formats (for example JPEG2000) the following error is returned.

```
ORA-29400: data cartridge error
IMG-00705: unsupported or corrupted input format
```

In these cases, the application must set the metadata columns appropriately using external information.

Selecting and Viewing Image Properties

Once the metadata columns in `image_blob_table` have been populated, we can view the metadata by selecting the non-BLOB columns from `image_blob_table`.

```
connect scott/tiger
select id, height, width, mimeType, contentLength from image_blob_table;
```

The selected values are:

```
        ID     HEIGHT      WIDTH MIMETYPE             CONTENTLENGTH
---------- ---------- ---------- -------------------- -------------
         1        600        800 image/jpeg                  66580
         2        375        500 image/gif                  189337
```

Creating Thumbnails and Changing Formats

We next illustrate some image processing operations that can be invoked within the database. To generate a thumbnail from an existing image, the programmer describes the desired properties of the new image. For example, the following description generates a JPEG thumbnail image of size 75x100 pixels:
`'fileformat=jfif fixedscale=75 100'.`

NOTE: Some three-letter image file extensions and the corresponding *inter*Media `fileformat` are as follows.

Extension	`fileformat`
.jpg	JFIF (9*i*, 10*g*), JPEG (10*g*)
.gif	GIFF(9*i*, 10*g*), GIF (10*g*)
.tif, .tiff	TIFF
.png	PNGF

The following example defines `image_blob_processCopy()` that adds a new row to `image_blob_table` with identifier `dest_id` and creates a new image in the row's BLOB using the image from the source row and processing it with the specified command string.

```
connect scott/tiger
create or replace procedure image_blob_processCopy(source_id number, dest_id number, verb varchar2)
is
    src_blob BLOB;
    dst_blob BLOB;
begin
  delete from image_blob_table where id = dest_id;
  insert into image_blob_table (id, image_blob)
    values (dest_id, empty_blob());

  select image_blob into src_blob
    from image_blob_table
    where id = source_id;
  select image_blob into dst_blob
    from image_blob_table
    where id = dest_id for update;

  ORDSYS.ORDImage.processCopy(src_blob, verb, dst_blob);
  update image_blob_table set image_blob = dst_blob where id = dest_id;
end;
/
```

```
-- Scale flowers.jpg to 10% into row with id=3
call image_blob_processcopy(1,3,'scale=.1');

-- convert goats.gif to grayscale jpeg thumbnail into row with id=4
call image_blob_processcopy(2,4,'fileformat=jfif contentformat=8bitgray maxscale=100 100');

-- update the metadata for the newly created image rows
call image_blob_getProperties();

-- admire our handiwork
select id, height, width, mimeType, contentLength from image_blob_table;
```

The preceding example generates the following output.

```
        ID     HEIGHT      WIDTH MIMETYPE              CONTENTLENGTH
---------- ---------- ---------- -------------------- -------------
         1        600        800 image/jpeg                  66580
         2        375        500 image/gif                  189337
         3         60         80 image/jpeg                    1918
         4         75        100 image/jpeg                    2156
```

NOTE: The following error might be returned from `ORDImage.processCopy()`.

```
ORA-29400: data cartridge error
IMG-00703: unable to read image data
ORA-28575: unable to open RPC connection to external procedure agent
```

In Oracle Database release 9*i*, JPEG (and some other less common formats) encoding and decoding requires the external procedure agent (extproc). To fix the preceding error, the Oracle Listener needs to be configured to use extproc. See technical note 198099.1, *Configuration of the External Procedure Call for interMedia* at http://metalink.oracle.com for detailed instructions on setting up extproc. Oracle Database release 10*g* does not require extproc for JPEG encoding and decoding.

If you do not want to change your Oracle Net configuration, try changing the file format to pngf instead as follows.

```
-- Convert to PNG if Extproc is not set up correctly.
call image_blob_processcopy(2,5,'fileformat=pngf contentformat=8bitgray maxscale=100 100');
call image_blob_getproperties();
```

Exporting Images with ORDImage.export()

Exporting image data from the database with *inter*Media's export method requires that the database write to the file system. Writing to the file system requires granting JAVA permissions to your user (scott in the examples) **and** to the ORDSYS* schema as shown in the following example.

```
connect / as sysdba
create or replace directory imagedir as '/home/alamb/quickstart';
-- For Windows:
--create or replace directory imagedir as 'c:\quickstart';
grant read on directory imagedir to scott;

call dbms_java.grant_permission('SCOTT','java.io.FilePermission',
                                '/home/alamb/quickstart/*','WRITE');
call dbms_java.grant_permission('ORDSYS','java.io.FilePermission',
                                '/home/alamb/quickstart/*','WRITE');
-- For Windows:
--call dbms_java.grant_permission('SCOTT','java.io.FilePermission','c:\quickstart\*','WRITE');
--call dbms_java.grant_permission('ORDSYS','java.io.FilePermission','c:\quickstart\*','WRITE');

connect scott/tiger
create or replace procedure image_blob_export (source_id number, filename varchar2) as
  img_blob BLOB;
  ctx raw(64) := null;
begin
  select image_blob into img_blob from image_blob_table where id = source_id;
  ORDSYS.ORDImage.export(ctx, img_blob, 'FILE', 'IMAGEDIR', filename);
end;
/

call image_blob_export(3, 'flowers_thumbnail.jpg');
call image_blob_export(4, 'goats_grayscale.jpg');
```

***NOTE**: For Oracle Database releases 9.2.0.1, 9.2.0.2, and 9.2.0.3 you **must** change ORDSYS in the preceding export example to ORDPLUGINS.

Cleaning Up

To restore your database to its original state, you need to remove all of the objects that were created in this quickstart as shown in the following example.

```
connect / as sysdba
drop directory imagedir;
call dbms_java.revoke_permission('SCOTT','java.io.FilePermission',
                                 '\home\alamb\quickstart\*','WRITE');
call dbms_java.revoke_permission('ORDSYS','java.io.FilePermission',
                                 '\home\alamb\quickstart','WRITE');
-- For Windows
--call dbms_java.revoke_permission('SCOTT','java.io.FilePermission','c:\quickstart\*','WRITE');
--call dbms_java.revoke_permission('ORDSYS','java.io.FilePermission','c:\quickstart\*','WRITE');

connect scott/tiger
drop procedure image_blob_import;
drop procedure image_blob_getproperties;
drop procedure image_blob_processcopy;
drop procedure image_blob_export;
drop table image_blob_table;
```

Conclusion

Using *inter*Media's relational interface, we have shown how to import images into the database, extract image metadata, write SQL queries based on image metadata (width, height, and so on), perform basic image processing, and export images to the file system.

Oracle *inter*Media provides more functionality than is covered in this quickstart. Refer to the following documentation for more information: *Oracle interMedia User's Guide and Reference, Release 9.0.1*, *Oracle interMedia Reference, 10g Release 1 (10.1)*, and *Oracle interMedia User's Guide, 10g Release 1 (10.1).* Additional examples and articles are available on the *inter*Media web page on the Oracle Technology Network at http://www.oracle.com/technology/products/intermedia/index.html.