

The move to store images in the database

By Marcel Kratochvil, Piction.

As the business imperative grows for companies to start managing and then publishing their digital image assets, the issue about where those image assets are stored is raised. The initial choice is to store them in a disk file system. This can be seen as the quickest and simplest approach. Another, better alternative should be looked at and that is the move to store those images in an Oracle 10g database.

A database is not normally considered to be an ideal repository for multimedia. Historically they are known to have had issues with performance with image retrieval. There has also been a noticeable lack of support through third party tools, leaving any images in the database well and truly locked in. This was seen in the older Oracle7 release and the use of *long* fields.

In the past five years, with changes in database technology and improvements in disk performance and storage, the rules have changed and it now makes business sense to use the Oracle 10g database to store and manage all of an organisations digital assets.

The following are the strengths an Oracle 10g database can offer over traditional file system storage.

Manageability

Images stored in the database can be directly linked with metadata. In the one transaction an image can be manipulated, a thumbnail of that image created and all associated metadata modified. Related information is kept

in sync. If an image is stored in a file system, then it is possible for external processes to delete or modify that image, causing the image itself to either become orphaned or lose synchronicity with its corresponding relational data. Another common issue is web quality images losing their associated thumbnails, meaning web page displays become broken.

Oracle *interMedia*, which extends control over images, allows images to be manipulated inside the database. They can be resized, copied, converted and rotated. This simplifies management of them and allows the one programming environment. (PL/SQL or Java)

Security

If all images are stored in a directory then fine grained control is not possible. That is, it is not possible to restrict access of the images to individual users. Once a user can gain access to an image in the directory they can access all of them (assuming Digest Authentication).

By storing an image in the database fine grained security becomes possible. Access to an image can be restricted to individual users and it also becomes possible to achieve the following.

- Attach a timeout to accessing the image.
- Include check in/check out capabilities.
- Audit who accessed the image and when.
- Offer image exclusivity. (One user accesses an image for a set period of time during which no one else can access it.)

Backup/Recovery

The one backup program that is used to backup the database will also backup the images. This simplifies the backup process. In the event of failure the whole database can be recovered to the last committed transaction.

The traditional behaviour for backing up a file system is to back it up daily or weekly. This means in the event of failure the file system on recovery will be out of sync with the database by at least one day.

So by having the images in the database only one backup program is required and in the event of failure only one recovery procedure is needed.

Another advantage comes when using some of the more advanced database features such as standby databases and replication. Images are automatically replicated if the advanced replication option is used, and for disaster recovery situations, image data is automatically transferred to a standby database.

Extensibility

An image stored in the database can be indexed. If an image is a document it can be thematically searched and gists can be extracted from it.

An image can be converted from one format to another. Metadata can be extracted from it. It can be copied, resized and the image quality controlled.

Flexibility

When it comes to managing and controlling the images in the database, the Oracle 10g database offers the greatest in flexibility. Sets of images

can be deleted, updated or copied as easy as it is to write a query.

Images can be linked together and metadata can be easily attached to them. All data related to an image or set of images can logically co-exist.

These add flexibility which gives a DBA and Developer greater control over managing and working with the images.

Addressing the concerns

Performance

“Isn’t it slower to retrieve an image from the database compared to a file system?”

This might have been true ten years ago on older disk systems, but with improvements in disk technology this issue has subsequently disappeared. Tests have shown that it is just as fast to retrieve an image from a database as it is from a disk file system.

In addition, by using optional caching technology, it is possible to cache frequently access images thus improving the time to retrieve them.

Database Size

“Doesn’t it take more storage to put in an image in the database compared to a file system?”

Yes it does. The storage format used in the database adds extra overheads to manage locking and to reserve storage for growth. In addition, Oracle puts indexes on images to improve the time it takes to retrieve and manipulate them.

Though there is extra storage required it is not significant in the overall storage requirements. When tens of thousands of images are stored in the

database, the extra storage required is dwarfed by the overall size of the images. It is also fair to say that disk is cheaper than it once was, and when it comes to database management, the strategy now is to sacrifice storage for performance. When dealing with relational data it is now common practice to add additional indexes and use locally managed tablespaces to improve performance. These extra features come at an additional storage cost.

So increasing the storage requirements in the database by storing image data in it, only ensures that the image data is retrieved optimally and consistently.

Complexity

“Isn't it more complicated and time consuming having to put images into the database and retrieve them compared to a file system?”

This was exactly the same argument used fifteen years ago when relational databases first appeared. But at that time the argument was concerned with storing data in what was known as flat files versus putting it into a relational database. Time has shown that the overheads of putting data into a relational database offer more benefits and ultimately greater control than when storing it in a flat file. The same argument can be applied to images. Yes there is some programming overhead to put them in, but the advantages gained from having them in the database (as shown above) is greater than when not having them in the database.

So when it comes to storing and managing those digital assets, keep in mind that ultimately it is easier, safer and better to keep them stored in a database.

About Piction

Piction has for the last four years successfully used the Oracle 10g database to store and manage digital images. Piction allows an organisation to web enable their digital image assets and then search and sell those assets. In addition to images, Piction also works with documents, audio and video. Piction offers true dynamic, flexibility in its web based interface and works with photo laboratories, museums, galleries, educational institutions, government organisations and any commercial organisation needing to web enable their digital images.

<http://www.piction.com>

About the Author

Marcel Kratochvil is CTO for Piction and is based in Canberra, Australia. Marcel has been using the Oracle database for the past fifteen years and originally began using Oracle with V5 on MVS and VMS. In addition to the development of Piction, Marcel is also known to run DBA training courses covering administration, proactive database management and performance tuning. He is also an Oracle 10g Beta Tester and does tuning trouble shooting for Government Departments and commercial organisations.

marcel@piction.com