

# **Representación y Recuperación de Imágenes Médicas en Bases de Datos Objeto-Relacionales**

**Carlos E. Alvez**

*Facultad de Ciencias de la Administración - Universidad Nacional de Entre Ríos*

**Aldo R. Vecchietti**

*INGAR - UTN – Facultad Regional Santa Fe*

## **Resumen**

*En este trabajo se presenta la implementación del almacenamiento y recuperación de imágenes médicas en ORACLE 10g release 2, que es un motor Objeto-Relacional de Base de Datos. Con el objeto de mostrar el estudio realizado, se presentó la definición del modelo lógico de la base de datos, los procedimientos, métodos y operadores utilizados para el almacenamiento y recuperación de estas imágenes. Para realizar la verificación de esta implementación se utilizaron imágenes DICOM correspondientes a Resonancia Magnética transversal de cerebro. Las capacidades provistas por la Base de Datos Objeto-Relacional empleada, facilitan la implementación de un servidor de imágenes médicas y potencian el uso y extensión del mismo.*

**Palabras clave:** Bases de Datos Objeto-Relacionales, SQL:2003, imágenes médicas, DICOM

## **1 Introducción**

El almacenamiento y recuperación de imágenes va adquiriendo cada vez mayor importancia y relevancia en aplicaciones informáticas de la medicina, impulsado por la continua adquisición de equipos de diagnóstico por imágenes por parte de clínicas y hospitales. Además, desde hace unos años existe DICOM [1] (**D**igital **I**maging and **C**ommunications in **M**edicine), que es un estándar reconocido mundialmente para el intercambio de imágenes médicas, que se generó para el manejo, almacenamiento, impresión y transmisión tales imágenes. Si bien DICOM fue pensado para el intercambio de ficheros entre dos entidades que tengan capacidad de recibir imágenes y datos de pacientes en ese formato, dado el crecimiento de tales imágenes es importante contar con una Base de Datos para que las mismas se almacenen y recuperen de manera eficiente y segura. Cuando se almacenan imágenes en una Base de Datos es importante contar con la con información textual que la soporta, de modo tal que se facilite el diseño y recuperación de imágenes en base a su contenido de bajo y de alto nivel. En este trabajo, se analizan diversos mecanismos de representación de imágenes médicas en una Base de Datos Objeto-Relacional (BDOR) que responde al estándar SQL:2003 [2]. El objetivo del trabajo es analizar diversos mecanismos que brinda esta tecnología, y que facilitan el diseño, la formulación de consultas y la recuperación de la información. En el análisis se emplea la Base de Datos Oracle 10g que es un Sistema de Gestión de Bases de Datos Objeto-Relacionales (SGBDOR) y su extensión *InterMedia* [3] para el soporte de datos multimedia e imágenes DICOM. El selección de Oracle 10g para este estudio obedece a que en la actualidad es uno de los SGBDOR mas avanzados y que más concuerda con el estándar SQL:2003; además provee un mecanismo para el almacenamiento de imágenes DICOM. El análisis que se realiza en este trabajo estará centrado en las alternativas de representación de las imágenes médicas en el SGBDOR Oracle10g, empleando los mecanismos más

adecuados de almacenamiento, que faciliten el diseño, la formulación de consultas y la recuperación de la información.

## **2. Información a Almacenar**

El almacenamiento de imágenes en una BD multimedia presenta un desafío muy grande al momento de decidir respecto de la información a almacenar ya que debe permitir encontrar imágenes en base a la descripción de su contenido [4], como también de la información descriptiva de estos datos mediales. El empleo de metadatos para describir el contenido de la información medial es el medio mas usado.

El contenido se puede describir utilizando diferentes niveles de abstracción, por ejemplo, de una imagen se puede guardar información sobre el contenido físico de la misma (color, textura, forma, etc.) que permita recuperar información mediante la utilización de medidas de similitud. Esta información se puede catalogar como de bajo nivel de abstracción. Además, sobre la misma imagen se puede guardar información semántica, que permite obtener por ejemplo, “*resonancia magnética de cerebro ...*” la cual se puede catalogar como un nivel alto de abstracción

Cuanto mayor es el nivel de abstracción, mayor es la dificultad para realizar un procesamiento automático, fundamentalmente porque la introducción y almacenamiento del contenido semántico de los datos no es posible resumir en palabras, mensajes o características de una imagen, ya que las mismas pueden significar cosas diferentes para distintos observadores; para ello es necesario contar con una sistematización de la misma o definición por medio de una ontología.

Además de disponer de la descripción relacionada con el contenido, también es necesario incluir otro tipo de información descriptiva, como es la información sobre su creación, formato utilizado, vistas o cortes empleados, links hacia otro material relacionado, etc.

Contado con estos metadatos que describen el contenido multimedia, se pueden indexar los mismos para un eficiente acceso a la información multimedia por contenido físico y/o semántico.

El primer beneficio de utilizar recuperación basada en contenido es reducir el esfuerzo en tiempo requerido para obtener una información basada en imagen. Con una adición frecuente y actualización de imágenes en base de datos enormes, no es práctico requerir entradas manuales de datos de todos los atributos que pueden necesitarse para consultas, y la recuperación basada en contenido proporciona un aumento en la flexibilidad y valores prácticos. Esto es además útil proveyendo la habilidad de consultar sobre atributos tales como textura o forma que son difíciles de representar utilizando palabras clave.

## **3. ORACLE 10g y ORACLE *inter*MEDIA**

Los SGBDOR han evolucionado en gran medida desde sus primeras implementaciones, y en la actualidad, han alcanzado un grado de estabilidad y madurez que permiten abordar aplicaciones multimedia que hace algunos años no era posible.

La tecnología Objeto Relacional nos permite:

- Conservar ciertas características de las BD relacionales, como realizar consultas declarativas (SQL).

- Aprovechar las características de OO de los SGBDOR, que dan la posibilidad de establecer nuevos tipos de datos “object types”, para adaptar los servicios provistos por el servidor de BD a una aplicación o dominio específico.

El uso del SGBDOR Oracle 10g en este trabajo, se debe a que su extensión *interMedia*, permite la gestión de contenido multimedia de manera integrada con otra información de la aplicación

La utilización de Oracle10g y en particular de *interMedia* permite el almacenamiento, recuperación y administración de:

- BLOB (Binary Large Objects) que se almacenan localmente en la Base de Datos y contienen los datos multimediales (imágenes en nuestro caso).
- BFILE que son objetos grandes almacenados en archivos específicos del sistema operativo.
- URLs que contienen datos mediales almacenados en servidores http.

En nuestro caso se analizará sólo la primera opción, basados en consideraciones transaccionales y de seguridad en el manejo de las imágenes, las otras formas si bien pueden llegar a contar con “cierto” control transaccional (sobre todo los links a archivos y páginas Web), al estar soportadas en archivos del sistema operativo o en servidores http no cuentan con un control transaccional y de seguridad pleno.

La utilización de BLOB permite la selección de tres formas diferentes de almacenar una imagen:

- Como tipo de dato BLOB plano que puede estar incluido en un tipo definido por el usuario (UDT: User Define Type).
- Como un tipo *ORDImage* que es un tipo objeto “object type” predefinido en ORACLE *InterMedia* para el almacenamiento de imágenes, que además incluye un conjunto de métodos para su manipulación. Para el caso del almacenamiento de imágenes DICOM se emplean este tipo de objetos, esto está disponible a partir de ORACLE 10g release 2.
- Como un tipo *StillImage* [5] que es un también un tipo objeto predefinido en ORACLE *InterMedia* pero que en este caso, a diferencia del anterior, responde al estándar SQL/MM de SQL:2003 para imágenes quietas, también posee un conjunto de métodos para su manipulación.

En este caso, y como se mostrará más adelante, se emplearán los objetos *ORDImage* en lugar de los *StillImage* del estándar SQL:2003. La razón fundamental es que los *ORDImage* a diferencia de los *StillImage* permiten el almacenamiento de imágenes DICOM y el procesamiento de sus metadatos.

#### **4. Modelo Conceptual.**

Las imágenes médicas por lo general están asociadas con un paciente y su historia clínica, por lo tanto parece apropiado emplear el modelo de referencia propuesto por HL7 [6] que se puede ver en la Fig. 1:

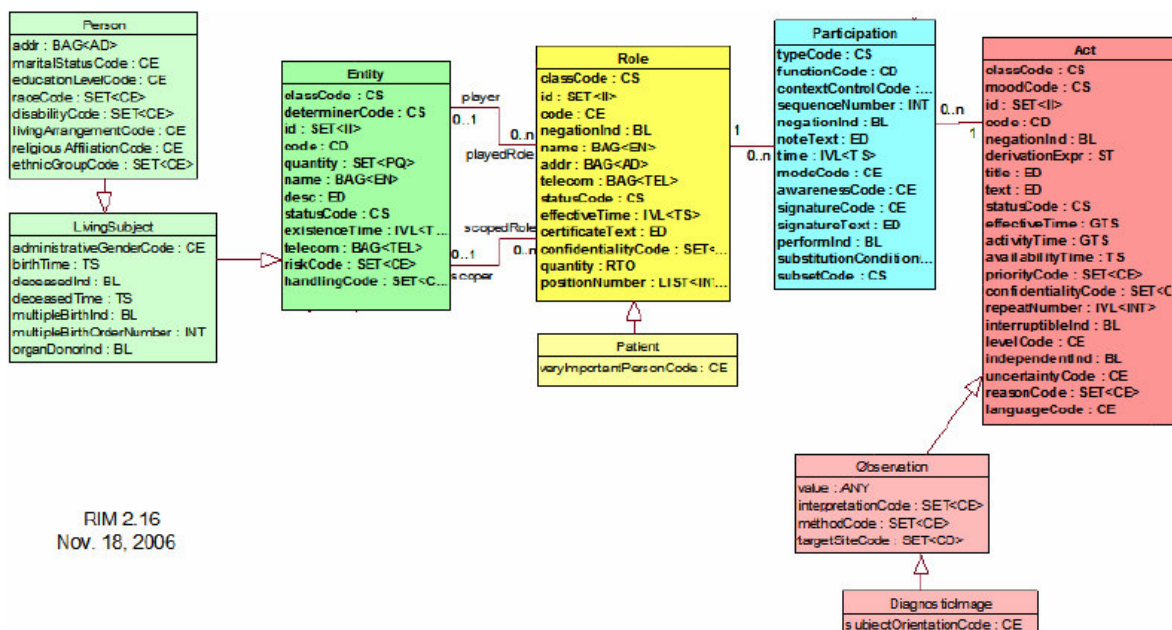


Figura 1: Extracto del Reference Information Model (RIM) de HL7.

Si bien las BDOR permiten una implementación completa del RIM de HL7, ya que es posible crear cada uno de los tipos definidos en el modelo y generar las relaciones de herencia, asociación, agregación y composición de manera natural [7], para la implementación de las imágenes en este trabajo inicial, se empleó un modelo (Fig. 2) muy simplificado del extracto del modelo de referencia de HL7 (Fig. 1), en el cual la clase *Paciente* engloba a *Person*, *LivingSubject*, *Entity*, *Role* y *Patient*; y *Diagnóstico-Imagen* engloba a *Participation*, *Act*, *Observation* y *DiagnosticImage*. Una asociación de cardinalidad 1:N se estableció entre ambas clases.

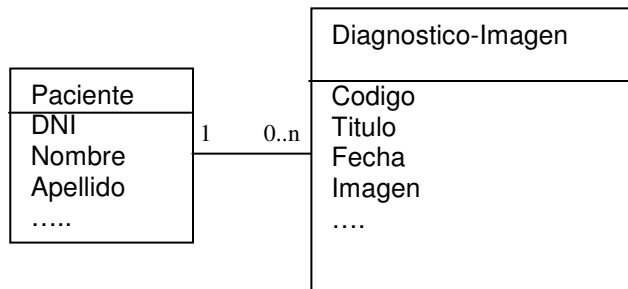


Figura 2. Modelo simplificado para almacenamiento de imágenes

La transformación de este modelo conceptual al modelo lógico de una Base de Datos Objeto-Relacional (BDOR) es directa. Se pueden definir las clases del modelo conceptual como tipos definidos por el usuario (UDTs), para las relaciones de asociación del modelo se pueden emplear referencias entre objetos. La definición del “schema” para ORACLE sería la siguiente:

```
CREATE OR REPLACE TYPE paciente_ob AS OBJECT(
```

```

        DNI          NUMBER(9),
        nombre       VARCHAR2(30),
        apellido     VARCHAR2(30),
        ...
    ) NOT FINAL;

CREATE OR REPLACE TYPE diagnostico_imagen_ob AS OBJECT(
    codigo          NUMBER(6),
    titulo          VARCHAR2(256),
    fecha           DATE,
    imagen          ORDImage,
    caracteristica_imagen ORDImageSignature,
    DicomMD         XMLType;
    ref_paciente REF paciente_ob;
    ...
);

CREATE OR REPLACE TYPE diag_image_tab AS TABLE OF REF
diagnostico_imagen_ob;

ALTER TYPE paciente_ob ADD ATTRIBUTE(
imagenes diag_image_tab) CASCADE;

CREATE TABLE paciente_tab OF paciente_ob (DNI Primary key)
    NESTED TABLE imagenes STORE AS reftoimagenes_tab;

CREATE TABLE diagnostico_imagen_tab OF diagnostico_imagen_ob (codigo
Primary key);

```

*Paciente\_ob* es el UDT definido para representar la información de los pacientes a los que se les hará el diagnóstico por imágenes y *diagnostico\_imagen\_ob* el UDT para las imágenes. En este último se definió el atributo *imagen* del tipo *ORDImage*, *característica\_imagen* que es del tipo *ORDImageSignature* contiene la información de bajo nivel de la imagen como: color, forma, textura y posición de cada uno de estos atributos. La asociación con paciente se implementa como una referencia a objetos paciente en el atributo *ref\_paciente*, que contendrá el OID (Object Identifier) del objeto paciente al que pertenecen las imágenes. Aquí también se ha incluido el atributo *DicomMD* del tipo *XMLType* que se empleará para almacenar metadatos de imágenes DICOM cuyo uso se explicará más adelante. *Diag\_image\_tab* es un tipo colección *multiset* que contiene un “set” de referencias a objetos del tipo *diagnostico\_imagen\_ob*. Cada elemento de este *multiset* contendrá el OID de un objeto imagen correspondiente a ese paciente. Dado que cada paciente puede llegar a tener a lo largo del tiempo un número variable y desconocido de imágenes de un tipo determinado, esta implementación se hace por medio de una colección *multiset*. La sentencia *ALTER TYPE* modifica el UDT de paciente para que incluya este “set” de referencias. Finalmente se crean las tablas de los UDTs definidos previamente, que son las que contendrán los objetos persistentes de la aplicación. Se debe destacar que las BDOR permiten la definición de métodos asociados a los UDTs, lo que permitiría la especialización de los mismos y la inclusión de nuevas aplicaciones. Por otra parte, la evolución de los tipos es una de las más destacables ventajas del empleo de la tecnología Objeto-Relacional, por este mecanismo es posible incorporar nuevos tipos, métodos, atributos, etc., y especializar, incorporar y modificar las aplicaciones existentes. Un

servidor de imágenes con estas características, puede dar respuesta a la creciente demanda del diagnóstico por imágenes de la medicina actual.

## 5. Procedimiento para Insertar Imágenes y sus Características.

Cuando se genera un objeto del tipo *ORDImage* además de soportar la imagen, se extraen diversos metadatos, como por ejemplo:

- Información sobre el almacenamiento del dato, incluyendo el tipo de origen, localización y nombre. Tiempo de actualización del dato y formato.
- *MIME media type* (utilizado en aplicaciones web y correo).
- Alto y ancho de la imagen (para el caso de imágenes), y largo del contenido de la imagen formato y tipo de compresión.
- Matadatos seleccionados de la aplicación luego de la inserción.

Además, las instancias de *ORDImageSignature* contienen métodos para capturar de manera automática las características de bajo nivel de la imagen almacenada en un objeto *ORDImage*, para ello emplea el método *generateSignature()*. Esto permitirá realizar consultas basadas en contenidos intrínsecos como, por ejemplo, en que medida el color y la forma las imágenes almacenadas en la base de datos coinciden con otra en particular.

Para insertar datos en la tabla *diagnostico\_imagen\_tab*, en primer lugar se deben insertar los datos alfanuméricos, y los objetos *imagen* y *caracteristica\_imagen* se deben inicializar, para que en un segundo paso sean instanciados, como se muestra en el siguiente bloque de procedimiento almacenado:

```
connect medico/medico; -- nombre de usuario y contraseña
DECLARE
    cod NUMBER := 11250;
    tit NUMBER := 'titulo del diagnóstico...';
    fech DATE := '11/12/2003';
    ...
    img_aux ORDSYS.ORDImage;
    ctx RAW(64) := NULL;
    imagen_sig ORDSYS.ORDImageSignature;

BEGIN
    -- Inserta una fila e inicializa los valores de los
    -- objetos imagen y caracteristica_imagen.
    insert into diagnostico_imagen_tab
        values (cod, tit, fech,...,
            ORDSYS.ORDImage.init(), ORDSYS.ORDImageSignature.init());
    ...
```

A partir de la ejecución del procedimiento, el registro insertado cuenta con los valores alfanuméricos y con objetos *ORDImage* y *ORDImageSignature* inicializados. En este punto se está en condiciones de importar la imagen a partir del archivo que la contenga. Para esto, se debe obtener el objeto *imagen* inicializado, por medio del método *setSource()* se le especifica el origen de donde se va a importar la imagen y con el método *import()* se carga la misma en el objeto *imagen*.

```

-- Ahora importa la imagen 'imgcerebro11250.dcm' del
-- directorio IMEGENESMEDICAS

    select imagen into img_aux
        from diagnostico_imagen_tab
            where codigo= 11250 for update;

    img_aux.setSource('file', ' IMEGENESMEDICAS ',
        'imgcerebro11250.dcm');
    img_aux.import(ctx);

```

Del código anterior se pueden hacer algunas consideraciones:

- En primer lugar, el origen de la imagen es establecido como *file*, lo que significa que el origen es un archivo ubicado en el sistema de archivos local.
- En segundo lugar, el archivo 'imgcerebro11250.dcm' se debe encontrar en el directorio 'IMEGENESMEDICAS'. Este directorio es un objeto de tipo *directory* definido previamente.

Es importante que los metadatos que describen el contenido físico de la imagen se puedan extraer de manera automática para obtener una mayor precisión, esto se efectúa extrayendo la “signatura” de la imagen y actualizando la fila ya insertada. Como ya fue expuesto, para ello se emplea el atributo *caracteristica\_imagen* del tipo *ORDImageSignature* de donde se pueden extraer la textura, color, y forma de una imagen mediante el método *generateSignature()*. El siguiente fragmento de código representan los pasos descriptos anteriormente:

```

-- Genera la variable imagen_sig:
    imagen_sig := ORDSYS.ORDImageSignature.init();

-- Crea un almacenamiento temporal para el atributo signatura del objeto
-- imagen_sig de tipo Blob
    DBMS_LOB.CREATETEMPORARY(imagen_sig.signature, TRUE);

-- Extrae la signatura de la imagen
    imagen_sig.generateSignature(img_aux);

-- Actualiza el registro insertado anteriormente
    update diagnostico_imagen_tab
        set imageb = img_aux, caracteristica_imagen= imagen_sig
            where codigo = 11250;

    commit;
END;

```

## 6. Recuperación de Imágenes

Como se presentó en las secciones anteriores, las imágenes y metadatos pueden almacenarse en objetos *ORDImage* y además se puede almacenar la “signatura” de una imagen en objetos *ORDImageSignature*.

Los objetos *ORDImage* y *ORDImageSignature* proveen varios métodos y operadores para la recuperación de información por contenido. Por ejemplo, si se desea encontrar una imagen particular del conjunto definido para un determinado paciente o un conjunto de pacientes, se pueden comparar las características de las imágenes utilizando métodos y/o operadores definidos para el objeto del tipo *ORDImageSignature*.

Para comparar imágenes, se le asigna un peso (weight) a los atributos visuales, entonces *interMedia* calcula una medida de similaridad (score) para ellos, además se debe suministrar un umbral (threshold) para indicar hasta que “score” se tolera en la respuesta.

Si se quiere comparar una imagen externa con las que se encuentran en la Base de Datos se puede emplear el operador *IMGsimilar()* de *ORDImageSignature*. Para ello en primer lugar se debe generar las características de la imagen externa y almacenarlas en un BLOB temporal. Luego se calcula la distancia (score) entre ésta y las distintas imágenes almacenadas en la Base de Datos.

El problema de este operador es que no hace distinciones entre las imágenes que son muy similares (próximas a cero) con aquellas que se acercan al valor umbral. Si la aplicación requiere una distinción más fina sobre la similaridad de las imágenes, es conveniente utilizar el operador *IMGScore()*. Este último retorna la distancia de las imágenes a comparar lo que permite, por ejemplo, ordenar los resultados por distancia.

Para mostrar un ejemplo de cómo utilizar *IMGScore()*, se trabajará con un conjunto de imágenes de resonancia magnética del cerebro de un paciente, que están disponibles en la Web. El operador *IMGScore()* trabaja en combinación con el operador *IMGsimilar()*. Esto lo hace utilizando un número de referencia en común (en este caso 100) que se muestra en **negrita** en el siguiente código:

```
DECLARE
    temp_imagen ORDSYS.ORDIMAGE; -- Obj. almacenara imágenes encontradas
    comp_signatura ORDSYS.ORDIMAGESIGNATURE; -- Signatura a comparar
    comp_imagen ORDSYS.ORDIMAGE; -- Imagen a buscar
    img_score NUMBER; -- Variable que guardará el score

CURSOR obtenerImagenes IS
    SELECT imagen, ORDSYS.IMGScore(100),
        FROM diagnostico_imagenes_tab foto
        WHERE ORDSYS.IMGsimilar(caracteristica_imagen, comp_signatura,
            'color="0.1" texture="0.3" shape="0.3" location="0.3"', 20, 100) = 1;

BEGIN
    comp_imagen :=
        ORDSYS.ORDImage.init('FILE', 'IMAGENESMEDICAS', 'IMG.dcm');
    -- establece las características de la imagen a comparar
    comp_imagen.setProperties();
    -- Inicializa la signatura de la imagen
    comp_signatura := ORDSYS.ORDIMAGESIGNATURE.init();
    -- Crea un almacenamiento temporal para la signatura del
    -- objeto de tipo ORDImageSignature (BLOB)
    DBMS_LOB.CREATETEMPORARY(comp_signatura.signature, TRUE);
    -- Genera la signatura para la imagen a comparar
    comp_signatura.generateSignature(comp_imagen);
```



```

OPEN obtenerImagenes;
LOOP
  FETCH obtenerImagenes INTO temp_imagen, img_score;
  EXIT WHEN obtenerImagenes %NOTFOUND;
  insert into salidas values (temp_imagen.source.srcname,
                              img_score, temp_imagen.source.localdata);
END LOOP;
CLOSE obtenerImagenes;
DBMS_LOB.FREETEMPORARY (comp_signatura.signature);
END;

```

En el código de ejemplo, como las imágenes son en escala de grises, se han empleado los siguientes pesos a las distintas características: 'color="0.1" texture="0.3" shape="0.3" location="0.3". En este caso se le dio menor importancia al color que al resto de las características. Estos pesos pueden variar, por ejemplo, en aquellas que son policromáticas tendrá mas peso el color, que la forma y la posición. También en este ejemplo, se estableció un umbral de similaridad (threshold) igual a 20, con lo cual, se obtendrá como resultado cualquier imagen almacenada con una distancia entre 0 y 20 con “IMG.dcm”.

El propósito del ejemplo que se ejecutó fue el de averiguar si una imagen ya estaba almacenada en la tabla *diagnostico\_imagenes\_tab*, para ello se extrajo el contenido de la imagen a buscar y este se comparó con las imágenes almacenadas.

Las imágenes obtenidas como resultado se guardan en la tabla *salidas*. Los datos son: nombre de la imagen “**srcname**”, la distancia “**img\_score**” y la imagen “**localdata**” (Tabla 1). La imagen con score 0 se corresponde con la imagen buscada, las otras tienen un score mayor a 10.

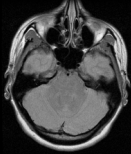
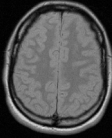
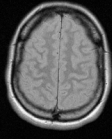

Nombre_imagen	Score	Imagen DCM
imgcerebro11207.dcm	0,0000	
imgcerebro11017.dcm	10,9205	
imgcerebro14218.dcm	10,9713	
imgcerebro11004.dcm	12,2140	

Tabla 1: Resultado de búsqueda por contenido ordenado por distancia (score)

La recuperación de imágenes expuesta en los párrafos anteriores se aplica a cualquier formato de imagen (DCM, JPEG, GIFF, etc.). En el caso de imágenes que responden al estándar DICOM (DCM), que se generan en muchos equipos de diagnóstico para medicina. Estas imágenes, al ser almacenadas en un objeto *ORDImage* los metadatos de las mismas, que fueron generados por el equipo, pueden ser recuperados por el método *getDicomMetadata()* que ORACLE incorporó en su versión 10g release 2. Este método permite extraer información almacenada en la imagen DICOM como ser datos del paciente, tipo de estudio y fecha, datos del equipo, etc. Estos metadatos pueden hacerse disponibles en archivos XML[8] (Extensible Markup Language) que pueden ser útiles no solo para cambiar el formato ('estilo') de cómo se presentan los datos, sino también con propósitos de realizar búsquedas empleando XPATH<sup>1</sup>[9]. Los metadatos DICOM se pueden obtener de manera similar a los metadatos de bajo nivel presentados anteriormente, para este caso se emplea *getDicomMetadata()* que retorna un objeto *XMLType*. Un ejemplo de esto se presenta a continuación:

```
DECLARE
    cod NUMBER(6);
    img ORDSYS.ORDIMAGE;
    dicom_metadata XMLType := NULL;

CURSOR obtenerMetadatos IS
    SELECT cod FROM Diagnostico_Imagen_tab;
BEGIN
    OPEN obtenerMetadatos;
    LOOP
        FETCH obtenerMetadatos INTO cod ;
        EXIT WHEN obtenerMetadatos %NOTFOUND;

        SELECT imagen INTO img
            FROM Diagnostico_Imagen_tab
            WHERE codigo= cod for update;

        -- Obtengo los metadatos de la imagen y la almaceno en un objeto XMLType
        dicom_metadata:=
            img.getDicomMetadata('imageGeneral');

        IF (dicom_metadata is NULL) THEN
            DBMS_OUTPUT.PUT_LINE('dicom metadata is NULL');
        END IF;

        -- Aquí abajo se guardan los metadatos en formato XML en la tabla
        -- Diagnostico_Imagen_tab
        update Diagnostico_Imagen_tab
            set DicomMD = dicom_metadata
            where codigo = cod ;
        commit;

    END LOOP;
    CLOSE obtenerMetadatos;
END;
/
```

---

<sup>1</sup> Lenguaje utilizado para acceder a partes de un documento XML.

Una vez almacenados los metadatos pueden ser consultados y procesados fácilmente por medio de funciones provistas especialmente para tipos *XMLType*.

La ventaja de almacenar los metadatos en formato XML, es que es muy simple de almacenar, pero a la vez como los datos están estructurados, la recuperación por contenido semántico no está limitada. Si los metadatos se almacenaran en esquemas de tablas, las operaciones de almacenamiento dejarán muchos archivos separados llenos de tablas de datos, que tendrán un costo superior[10].

## 7. DISCUSION y CONCLUSIONES

Las clínicas y hospitales están adquiriendo cada vez más dispositivos de imágenes digitales, esto genera la necesidad de una adecuada administración de las mismas, para ello es necesario contar con una Base de Datos que provea a las aplicaciones una administración transaccional de las mismas de manera segura y eficiente, que pueda ser compartida por varias aplicaciones, que tenga control sobre accesos no-autorizados y que además cuente con la posibilidad de backup y recuperación. Estas propiedades no pueden obtenerse si las imágenes son almacenadas en un sistema de archivos.

En este trabajo se ha presentado la implementación del almacenamiento de imágenes médicas en ORACLE 10g release 2 que es un motor Objeto-Relacional de Base de Datos. Con el objeto de mostrar el estudio realizado, se presentó la definición del modelo lógico de la base de datos, y los métodos y operadores utilizados para el almacenamiento y recuperación de estas imágenes. Para realizar la verificación de esta implementación se utilizaron imágenes DICOM correspondientes a Resonancia Magnética transversal de cerebro.

El empleo de una Base de Datos Objeto-Relacional que cumple con las especificaciones del estándar SQL:2003 presenta ciertas ventajas comparadas con el uso de la tecnología relacional de Base de Datos, por ejemplo una BDOR permite:

- la definición de tipos específicos (UDTs) que permiten ajustar una aplicación para un dominio específico como es el tratamiento de imágenes médicas,
- los tipos pueden evolucionar gradualmente para generar nuevas aplicaciones a partir de las existentes, los mecanismos mas apropiados para esto es la creación de nuevos tipos por medio de los mecanismos de herencia,
- los UDTs pueden incluir métodos específicos desarrollados por el usuario para aplicaciones procesamiento de imágenes para diagnóstico u otro fin. Asimismo estos métodos pueden evolucionar con los tipos fácilmente.

Para este trabajo se empleó ORACLE 10g (release 2) por ser la Base de Datos comercial que cumple con más especificaciones orientadas a objetos del estándar SQL:2003: herencia, polimorfismo, definición de UDTs, tipos referencias, tratamiento de colecciones (arreglos y multiset), etc.; además de la posibilidad de almacenar imágenes DICOM de manera directa en objetos predefinidos *ORDImage* y extraer sus metadatos. Para este caso, los metadatos a su vez se pueden extraer y almacenar en tablas relacionales ó en documentos XML, dando la posibilidad de definir índices para mejorar la eficiencia en la recuperación de imágenes basadas en su contenido; esto será parte de los trabajos futuros a implementar. Por otra parte, los objetos *ORDImage* almacenan sus propios metadatos permitiendo realizar

búsquedas por similitud que puede ser útil tanto para imágenes DICOM como no-DICOM. Las experiencias realizadas en este trabajo para analizar la tecnología nos permite concluir que los objetos y métodos provistos por la BDOR empleada facilitan la implementación de un servidor de imágenes médicas y potencian el uso y extensión del mismo.

## Referencias

- [1] DICOM (Digital Imaging and Communications in Medicine) <http://medical.nema.org/>
- [2] Melton, Jim, ISO/IEC 9075-2:2003 (SQL/Foundation), ISO standard (2003).
- [3] Oracle *interMedia* Reference Manual, 10g Release 2 (10.2), Part No B14297-01, June 2005.
- [4] Harald Kosch, "Enhancement of Processing Efficiency in Multimedia Database Management Systems and Video Servers supported by the Use of Meta-Data" Informatics Colloquium - April 2002 HS-C, University, Klagenfurt.
- [5] ISO/IEC 13249-5:2001, Information technology - Database languages - SQL Multimedia and Application Packages - Part 5: Still Image, International Organization For Standardization, 2001.
- [6] María Fernanda Golobisky y Aldo Vecchietti "Mapping UML Class Diagrams into Object-Relational Schemas". Proceedings of ASSE 2005, vol 1, 65-79, (2005), ISSN 1666-1087.
- [7] HL7 (Health Level 7) <http://www.hl7.org/>
- [8] Extensible Markup Language (XML), <http://www.w3.org/XML/> .
- [9] XPath, W3C Recommendation 16 November 1999, <http://www.w3.org/TR/xpath>.
- [10] M. Döller, "MPEG-7 meets Multimedia Database Systems", Journal of Universal Knowledge Management, pp 18-25, 2006.

Contacto: Carlos E. Alvez Facultad de Ciencias de la Administración - Universidad Nacional de Entre Ríos  
Alvear 1424 – (3200) – Concordia – Entre Ríos [caralv@ai.fcad.uner.edu.ar](mailto:caralv@ai.fcad.uner.edu.ar)