Bases de Datos Avanzadas

Práctica no. 4:

Definición y creación de una base de datos orientada a documentos XML

**Número de Práctica: 4**

**Nombre de la Práctica: Definición y creación de una base de datos orientada a documentos XML**

REQUISITOS PREVIOS

Para la realización de la práctica se deberá tener instalado en la computadora el software:

* Oracle Server 11g Express Edition (XE)
* Oracle SQL Developer
* [Archivo xml de empleados](https://drive.google.com/file/d/1SGLYyuvWrTlTW_6Uh_W31O779oUkzQRD/view?usp=sharing)

MARCO TEÓRICO

**Almacenamiento y consulta de documentos XML**

Para almacenar información en formato XML en una tabla Oracle, se tienen las opciones de crear atributos CLOB (Character Large Object) o de tipo XMLType. Aunque ambos son válidos, si se desea utilizarlos como contenedores y la gestión de información se realizará desde una aplicación externa, puede ser mejor opción crear un atributo del tipo CLOB. Los atributos CLOB guardarán contenido XML básicamente como texto plano. En cambio, si se desea trabajar con el contenido de estos atributos directamente desde la base de datos, puede resultar más cómodo hacerlo con atributos del tipo XMLType, ya que el contenido almacenado está preparado y listo para ser utilizado.

Considere las siguientes definiciones:

**CREATE TABLE** alumnosXMLType

( id number **PRIMARY KEY**,

alumno **XMLType**

);

**CREATE TABLE** alumnosClob

( id number **PRIMARY KEY**,

alumno CLOB

);

Para insertar datos en ambas tablas se usa:

**INSERT INTO** alumnosXMLType

**VALUES**(1, **XMLType**('<?xml version="1.0"?>

<ALUMNO>

<NOMBRE>Pedro</NOMBRE>

<APELLIDOS>Díaz</APELLIDOS>

<DIRECCION>

<CALLE>Jacarandas 34</CALLE>

<COLONIA>Juan Escutia</COLONIA>

<ESTADO>Veracruz</ESTADO>

</DIRECCION>

<CURSO>1</CURSO>

</ALUMNO>'));

/

**INSERT INTO** alumnosClob

**VALUES**(1, '<?xml version="1.0"?>

<ALUMNO>

<NOMBRE>Pedro</NOMBRE>

<APELLIDOS>Díaz</APELLIDOS>

<DIRECCION>

<CALLE>Jacarandas 34</CALLE>

<COLONIA>Juan Escutia</COLONIA>

<ESTADO>Veracruz</ESTADO>

</DIRECCION>

<CURSO>1</CURSO>

</ALUMNO>');

Aunque el almacenamiento es prácticamente el mismo, el uso de *XMLType* vigila la buena formación del documento XML, así como las capacidades de búsqueda y procesamiento del documento dentro del propio manejador de bases de datos.

**SELECT** a.alumno.**extract**('/ALUMNO/NOMBRE/text()')

**FROM** alumnosXMLType a

**WHERE** id=1;

DESARROLLO DE LA PRÁCTICA:

a) Abra una ventana de la Línea de Comandos SQL de Oracle, y escriba las siguientes sentencias:

1. CONNECT / AS SYSDBA;
2. CREATE USER bdxml IDENTIFIED BY bdxml;
3. GRANT CONNECT, RESOURCE TO bdxml;
4. GRANT CREATE ANY DIRECTORY TO bdxml;
5. GRANT DROP ANY DIRECTORY TO bdxml;
6. DISCONNECT;
7. CONNECT bdxml/bdxml;
8. ALTER SESSION SET NLS\_TERRITORY='MEXICO';
9. CREATE TABLE personas (

RFC CHAR(13) PRIMARY KEY,

Nombre VARCHAR(30) NOT NULL,

Apellidos VARCHAR(30) NOT NULL,

CV XMLType);

1. INSERT INTO personas VALUES ('PELJ900304JJ6', 'Juan', 'Pérez López', XMLType('<curriculum\_vitae><empleos><empleo><empresa>Fábricas de Cartón </empresa><periodo>junio/2011-enero/2017</periodo></empleo> <empleo><empresa>Comercializadora Internacional </empresa><periodo>marzo/2017-febrero/2020</periodo></empleo></empleos><formacion><secundaria>Heroes de Nacozari</secundaria><medio\_superior><escuela>Escuela preparatoria No.5 </escuela><comprobante>certificado</comprobante></medio\_superior></formacion></curriculum\_vitae>'));
2. INSERT INTO personas VALUES ('CACM871109LI8', 'Martha', 'Carbajal Carbajal', XMLType('<curriculum\_vitae><empleos><empleo><empresa>Comercializadora de Ropa del Centro </empresa><periodo>abril/2010-enero/2011</periodo></empleo> <empleo><empresa>Refacciones para Autobuses S.A.</empresa><periodo>febrero/2011-febrero/2012</periodo></empleo></empleos><formacion><secundaria>Independencia</secundaria><medio\_superior><escuela>Vocacional No. 3 </escuela><comprobante>certificado</comprobante></medio\_superior><superior><escuela>Escuela Internacional de Comercio C.V.</escuela><comprobante>titulo profesional</comprobante></superior></formacion></curriculum\_vitae>'));
3. INSERT INTO personas VALUES ('CALJ851211K9O', 'Juan Manuel', 'Camacho López', XMLType('<curriculum\_vitae><empleos><empleo><empresa>Banca de Desarrollo Empresarial </empresa><periodo>diciembre/2005-mayo/2009</periodo></empleo> <empleo><empresa>Asesores Fiscales S.A.</empresa><periodo>julio/2009-marzo/2012</periodo></empleo></empleos><formacion><secundaria>Marruecos</secundaria><medio\_superior><escuela>Escuela Comercial y Contable </escuela><comprobante>certificado</comprobante></medio\_superior></formacion></curriculum\_vitae>'));
4. INSERT INTO personas VALUES ('MEGR910508PY3', 'Rodrigo', 'Medina García', XMLType('<curriculum\_vitae><empleos><empleo><empresa>Armadora de Motores Nacionales </empresa><periodo>abril/2010-junio/2012</periodo></empleo></empleos><formacion><secundaria>Vasco de Quiroga</secundaria></formacion></curriculum\_vitae>'));
5. COMMIT;
6. SELECT \* FROM personas;

Texto

Descripción generada automáticamente

1. SELECT XMLColattval(p.apellidos) FROM personas p;

Texto

Descripción generada automáticamente

1. SELECT p.nombre || ' ' || p.apellidos AS persona, EXTRACT(p.cv, '//empresa').getstringval() AS RESULT FROM personas p WHERE p.nombre LIKE '%Martha%';

Texto

Descripción generada automáticamente

1. SELECT p.nombre || ' ' || p.apellidos AS persona, EXTRACTVALUE(p.cv,'//empresa[contains(.,"S.A.")]') AS RESULT FROM personas p;

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

1. SELECT p.nombre || ' ' || p.apellidos AS persona, EXTRACTVALUE(p.cv, '//empresa[contains(.,"S.A.")]') AS RESULT FROM personas p WHERE EXISTSNODE(p.cv, '//empresa[contains(.,"S.A.")]') = 1;

Texto

Descripción generada automáticamente

1. SELECT p.nombre || ' ' || p.apellidos AS persona, EXTRACT(p.cv, '//medio\_superior/escuela/text() | //superior/escuela/text()').getstringval() AS RESULT FROM personas p WHERE EXISTSNODE(p.cv, '//medio\_superior/comprobante | //superior/comprobante') = 1;

Imagen en blanco y negro

Descripción generada automáticamente con confianza media

1. SELECT p.nombre || ' ' || p.apellidos AS persona, EXTRACT(p.cv, '//empleos/empleo/empresa/text()').getstringval() AS RESULT FROM personas p WHERE EXISTSNODE(p.cv, '//empleos[count(empleo)>1]') = 1;

Texto

Descripción generada automáticamente

1. SELECT p.rfc, p.nombre || ' ' || p.apellidos AS persona, EXTRACT(p.cv, '//comprobante').getstringval() AS RESULT FROM personas p WHERE p.nombre LIKE '%Martha%';

Texto

Descripción generada automáticamente con confianza baja

1. UPDATE personas p SET p.cv = updateXML(p.cv, '//medio\_superior/comprobante', '<comprobante>diploma</comprobante>') WHERE p.rfc = 'CACM871109LI8';
2. SELECT p.rfc, p.nombre || ' ' || p.apellidos AS persona, EXTRACT(p.cv, '//comprobante').getstringval() AS RESULT FROM personas p WHERE p.nombre LIKE '%Martha%';

Imagen que contiene Interfaz de usuario gráfica

Descripción generada automáticamente

1. UPDATE personas p SET p.cv = deleteXML(p.cv, '//medio\_superior/comprobante["diploma"]') WHERE p.rfc = 'CACM871109LI8';
2. SELECT p.rfc, p.nombre || ' ' || p.apellidos AS persona, EXTRACT(p.cv, '//comprobante').getstringval() AS RESULT FROM personas p WHERE p.nombre LIKE '%Martha%';

Texto

Descripción generada automáticamente

b) Descargue el archivo *empleados.xml* siguiendo las indicaciones del profesor, y cópielos en la carpeta **c:\repositorio** (debe crear la carpeta antes). Ejecute las siguientes sentencias, describa lo que realiza cada una de ellas y muestre el resultado obtenido de las marcadas en color amarillo. Pueden ser capturas de pantallas, vigilando que sean legibles de los resultados obtenidos, o el texto generado por la consola de comandos SQL o la herramienta SQL Developer.

1. CREATE TABLE empleados\_xml OF XMLType;
2. CREATE OR REPLACE DIRECTORY REPOSITORIO AS 'c:\repositorio';
3. INSERT INTO empleados\_xml VALUES (XMLType(BFILENAME('REPOSITORIO', 'empleados.xml'), NLS\_CHARSET\_ID('AL32UTF16')));
4. SET PAGESIZE 500;
5. SET LINESIZE 300;
6. SET LONG 2000;
7. COLUMN RESULT FORMAT A300;
8. SELECT \* FROM empleados\_xml;

Texto

Descripción generada automáticamenteTexto

Descripción generada automáticamente

1. SELECT XMLRoot(e.OBJECT\_VALUE, VERSION '1.0', STANDALONE YES) FROM empleados\_xml e;

Texto

Descripción generada automáticamente

1. SELECT EXTRACT(e.OBJECT\_VALUE, '/empleados/empleado/paterno').getstringval() AS res FROM empleados\_xml e;

Interfaz de usuario gráfica

Descripción generada automáticamente

1. SELECT EXTRACTVALUE(e.OBJECT\_VALUE, '/empleados/empleado[@id=101]/paterno') AS res FROM empleados\_xml e;

Texto

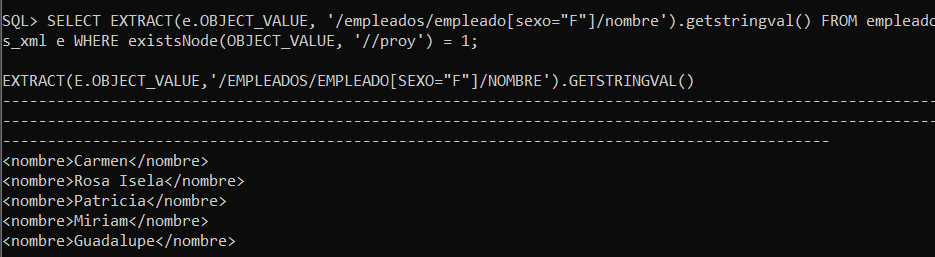
Descripción generada automáticamente

1. SELECT EXTRACT(e.OBJECT\_VALUE, '/empleados/empleado[sexo="F"]/nombre').getstringval() FROM empleados\_xml e;

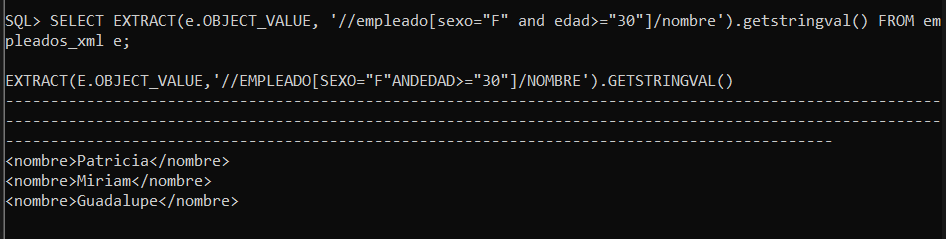
Texto

Descripción generada automáticamente

1. SELECT EXTRACT(e.OBJECT\_VALUE, '/empleados/empleado[sexo="F"]/nombre').getstringval() FROM empleados\_xml e WHERE existsNode(OBJECT\_VALUE, '//proy') = 1;



1. SELECT EXTRACT(e.OBJECT\_VALUE, '//empleado[sexo="F" and edad>="30"]/nombre').getstringval() FROM empleados\_xml e;



1. SELECT XMLSerialize(DOCUMENT e.OBJECT\_VALUE AS CLOB) AS XML FROM empleados\_xml e;

Texto

Descripción generada automáticamente

1. DECLARE

res BOOLEAN;

empsxmlstring VARCHAR2(4000):=

'<?xml version="1.0"?><empleados><empleado NSS="777888999" departamento="3"><nombre>Jesús</nombre><apellido>López</apellido><fecha\_nac>1973-04-04</fecha\_nac><direccion>Calle Venus 45, col. Irrigación,Ecatepec,89765</direccion><genero>M</genero><salario>50000</salario><titulo>Ingeniero</titulo><fecha\_ingreso>2000-04-05</fecha\_ingreso></empleado>';

empsxmlstring := empsxmlstring ||

'<empleado NSS="222333444 " Jefe="777888999" departamento="3"><nombre>Guadalupe</nombre><apellido>Oñate</apellido><fecha\_nac>1969-11-24</fecha\_nac><direccion>Av.Revolución 348, col. Fuentes, San Ignacio, 67656</direccion><genero>F</genero><salario>25000</salario><titulo>Licenciado</titulo><fecha\_ingreso>2000-04-07</fecha\_ingreso></empleado>';

empsxmlstring := empsxmlstring ||

'<empleado NSS="444555666" Jefe="111222333" departamento="1"><nombre>Julia</nombre><apellido>Regalado</apellido><fecha\_nac>1975-07-30</fecha\_nac><direccion>Calle cielo 76, col. Independencia, Tepeji, 34256</direccion><genero>F</genero><salario>28000</salario><titulo>Ingeniero</titulo><fecha\_ingreso>2001-06-28</fecha\_ingreso></empleado>';

empsxmlstring := empsxmlstring ||

'<empleado NSS="555666777" Jefe="777888999" departamento="3"><nombre>Mario</nombre><apellido>Medina</apellido><fecha\_nac>1977-10-01</fecha\_nac><direccion>Av.Politécnico 650, col. Profesiones, CDMX, 09765</direccion><genero>M</genero><salario>20000</salario><titulo>Licenciado</titulo><fecha\_ingreso>2100-05-30</fecha\_ingreso></empleado>’;

empsxmlstring := empsxmlstring ||

'<empleado NSS="333444555" departamento="2"><nombre>Rogelio</nombre><apellido>Calzada</apellido><fecha\_nac>1965-03-25</fecha\_nac><direccion>Av.Independencia 123, col. Fuentes, San Ignacio, 56565</direccion><genero>M</genero><salario>39000</salario><titulo>Licenciado</titulo><fecha\_ingreso>2000-04-05</fecha\_ingreso></empleado>';

empsxmlstring := empsxmlstring ||

'<empleado NSS="666777888" Jefe="111222333" departamento="1"><nombre>Bruce</nombre><apellido>Bolaños</apellido><fecha\_nac>1963-08-07</fecha\_nac><direccion>Mar Mediterráneo 56, col. Lomas Lindas, CDMX, 09879</direccion><genero>M</genero><salario>24000</salario><titulo>Licenciado</titulo><fecha\_ingreso>2100-12-17</fecha\_ingreso></empleado>';

empsxmlstring := empsxmlstring ||

'<empleado NSS="999000111" Jefe="333444555" departamento="2"><nombre>Laura</nombre><apellido>Méndez</apellido><fecha\_nac>1982-02-20</fecha\_nac><direccion>Av.Montevideo 98, col. Ampliación Hidalgo, Texcoco, 45679</direccion><genero>F</genero><salario>18000</salario><titulo>Ingeniero</titulo><fecha\_ingreso>2100-12-20</fecha\_ingreso></empleado>';

empsxmlstring := empsxmlstring ||

'<empleado NSS="111222333" departamento="1"><nombre>Sandra</nombre><apellido>Guzmán</apellido><fecha\_nac>1970-11-27</fecha\_nac><direccion>Av.Siempreviva 444, col. San Mateo, Ecatepec, 89740</direccion><genero>F</genero><salario>45000</salario><titulo>Ingeniero</titulo><fecha\_ingreso>2000-04-05</fecha\_ingreso></empleado>';

empsxmlstring := empsxmlstring ||

'<empleado NSS="888999000" Jefe="333444555" departamento="2"><nombre>Guadalupe</nombre><apellido>Hidalgo</apellido><fecha\_nac>1985-03-19</fecha\_nac><direccion>Av.Miguel Hidalgo 56, col. Independencia, Tepeji, 87981</direccion><genero>F</genero><salario>27000</salario><titulo>Ingeniero</titulo><fecha\_ingreso>2001-10-11</fecha\_ingreso></empleado></empleados>';

deptsxmlstring VARCHAR2(4000):=

'<?xml version="1.0"?>

<departamentos><departamento numero="1" Jefe="111222333"><nombre>Sistemas</nombre><fecha\_inicio>2003-03-08</fecha\_inicio><fecha\_fin></fecha\_fin></departamento><departamento numero="2" Jefe="333444555"><nombre>Ventas</nombre><fecha\_inicio>2001-11-11</fecha\_inicio><fecha\_fin></fecha\_fin></departamento><departamento numero="3" Jefe="777888999"><nombre>Administración</nombre><fecha\_inicio>2000-07-03</fecha\_inicio><fecha\_fin></fecha\_fin></departamento></departamentos>';

BEGIN

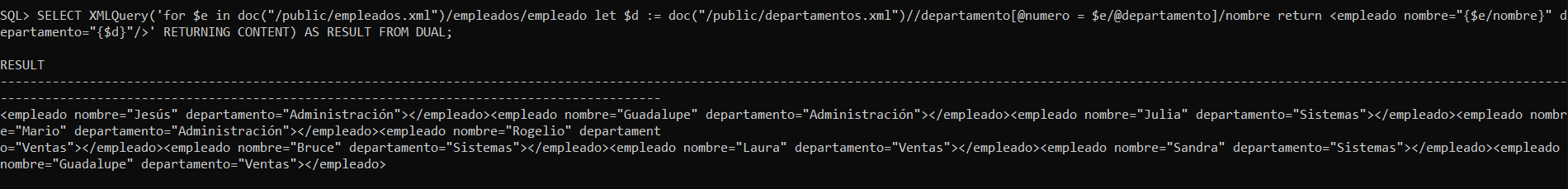
res := DBMS\_XDB.createResource('/public/empleados.xml', empsxmlstring);

res := DBMS\_XDB.createResource('/public/departamentos.xml', deptsxmlstring);

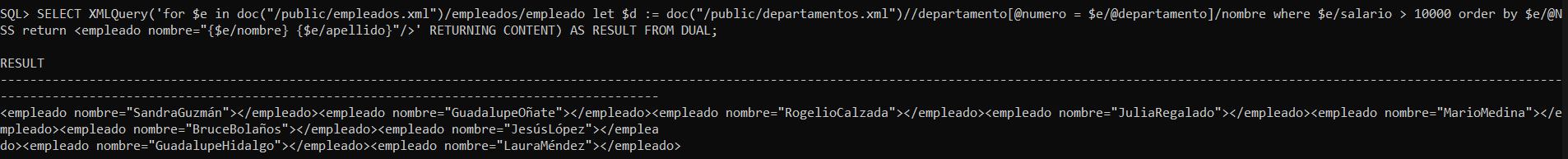
END;

/

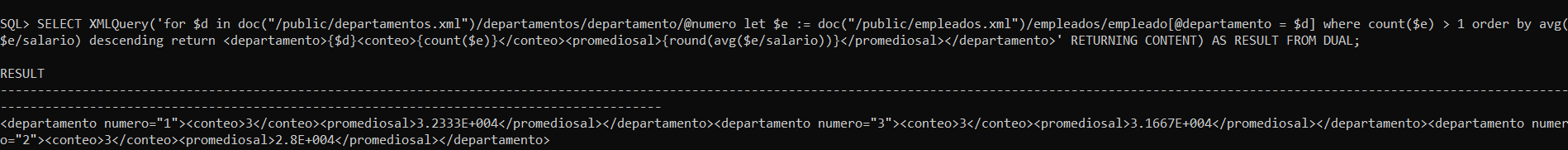
1. COMMIT;
2. SELECT XMLQuery('for $e in doc("/public/empleados.xml")/empleados/empleado let $d := doc("/public/departamentos.xml")//departamento[@numero = $e/@departamento]/nombre return <empleado nombre="{$e/nombre}" departamento="{$d}"/>' RETURNING CONTENT) AS RESULT FROM DUAL;



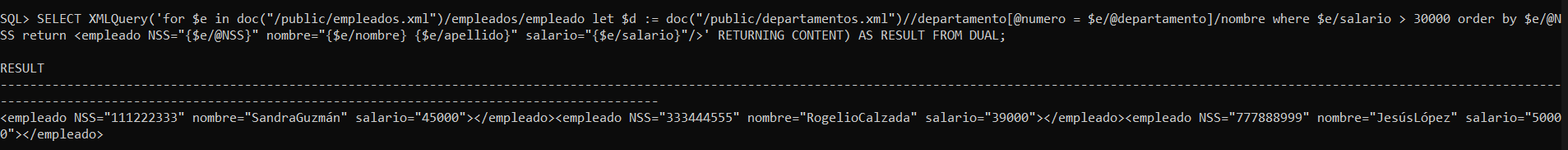
1. SELECT XMLQuery('for $e in doc("/public/empleados.xml")/empleados/empleado let $d := doc("/public/departamentos.xml")//departamento[@numero = $e/@departamento]/nombre where $e/salario > 10000 order by $e/@NSS return <empleado nombre="{$e/nombre} {$e/apellido}"/>' RETURNING CONTENT) AS RESULT FROM DUAL;



1. SELECT XMLQuery('for $d in doc("/public/departamentos.xml")/departamentos/departamento/@numero let $e := doc("/public/empleados.xml")/empleados/empleado[@departamento = $d] where count($e) > 1 order by avg($e/salario) descending return <departamento>{$d}<conteo>{count($e)}</conteo><promediosal>{round(avg($e/salario))}</promediosal></departamento>' RETURNING CONTENT) AS RESULT FROM DUAL;



1. SELECT XMLQuery('for $e in doc("/public/empleados.xml")/empleados/empleado let $d := doc("/public/departamentos.xml")//departamento[@numero = $e/@departamento]/nombre where $e/salario > 30000 order by $e/@NSS return <empleado NSS="{$e/@NSS}" nombre="{$e/nombre} {$e/apellido}" salario="{$e/salario}"/>' RETURNING CONTENT) AS RESULT FROM DUAL;



1. BEGIN
2. DBMS\_XDB.deleteResource('/public/empleados.xml');
3. DBMS\_XDB.deleteResource('/public/departamentos.xml');
4. END;
5. /
6. COMMIT;

Incluya una captura o el texto resultante de la ejecución de las sentencias SELECT marcadas en color resaltado amarillo. No se aceptan archivos externos de resultados fuera del contenido de este documento.

CUESTIONARIO

1.- Comente sobre la definición y uso de documentos XML aplicado sobre una base de datos relacional, con respecto a su almacenamiento y restricciones.

* Es un metalenguaje que permite definir lenguajes de marcas desarrollado por el World Wide Web Consortium (W3C) utilizado para almacenar datos en forma legible.
* Se utilizan bases de datos relacionales, objeto-relacionales o de objetos puras para representar documentos XML.
* Tener un almacenamiento de datos en XML tiene ventajas pues una de ellas es tener un formato correcto y que sean validos los datos y la posibilidad de realizar consultas y actualizaciones detalladas en los datos XML

2.- Indique las diferencias de modelado de una base de datos relacional con respecto a una basada en XML con respecto a la conservación de la integridad referencial.

* Crean modelos lógicos en XML.
* Mapean los modelos al mecanismo de almacenamiento correspondiente.
* Las operaciones con los documentos se realizan en XML.
* Dan un mayor nivel de abstracción al programador.
* Dependencia del esquema.
* Gestionan documentos como colecciones de datos.
* No todas necesitan un esquema para almacenar documentos.
* Problemas de integridad intra-documento.

CONCLUSIONES

Escriba sus conclusiones (no personales) con respecto al desarrollo de esta práctica.

Los documentos XML es una ventaja a la hora de almacenar los datos puesto que es más fácil de tener un formato correcto y gestionar cantidades de documentos grandes, por otro lado, los esquemas se definen como DTD o con XML Schema, además que no utilizan un mecanismo concretro de almacenamiento físico

REFERENCIAS BIBLIOGRÁFICAS (APA, IEEE, ETC.)

1.-Oracle XML DB <http://www.oracle.com/technetwork/database/database-technologies/xmldb/overview/index.html>

2.- XML DB Developer's Guide - <http://docs.oracle.com/database/121/ADXDB/toc.htm>

3.- Oracle XML Functions, <http://psoug.org/reference/xml_functions.html>

4.- Manejando XMLType en Oracle Database 11gR2 a través del componente XDB, <http://www.oracle.com/technetwork/es/articles/sql/xmltype-en-database11g-a-traves-xdb-1931103-esa.html>

5.- Trabajando XML en Oracle <https://blogdeaitor.wordpress.com/2013/02/06/trabajar-con-xml-en-oracle/>