Технология сервлетов СУБД для доступа к данным БД

Обновление ноябрем 2007 года

Владимир Пржиялковский
Преподаватель технологий Oracle
prz@yandex.ru,
www.ccas.ru/prz

Люди часто совершают старые ошибки, но при этом ссылаются на новые обстоятельствами.

А. Эйнштейн.

Введение

Эта статья является продолжением ранее опубликованных статей «*XML DB – новое измерение в организации данных в Oracle», «Что дает репозитарий XML DB и как с ним работать»* и «*Как зарегистрировать схему XML в XML DB и как этим воспользоваться*».

Репозитарий XML DB обеспечивает возможность доступа к данным БД на основе технологии сервлетов HTTP/HTTPS. Конфигурацию сервлетов можно наблюдать и устанавливать в ресурсе-файле */xdbconfig.xml*.

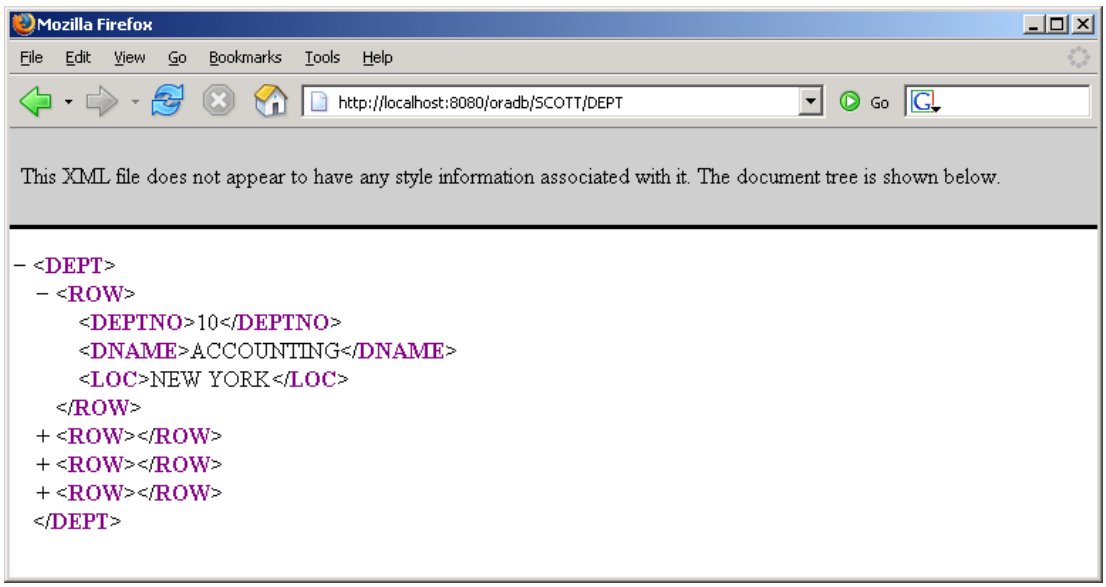
Создавать сервлеты «внутри БД» можно на C и на Java, однако для первого случая фирма дает в распоряжение пользователям только готовый для употребления сервлет, а во втором – правила самостоятельного построения.

Если говорить о реализации на Java, то это часть возможностей по организации работы СУБД с разнохарактерными программными элементами на Java, имевшихся в Oracle в версиях с 8.1 по 9.0, но выполненных на иной техлогической основе.

Встроенный сервлет DBURIServlet

Сервлет DBURIServlet написан на C и готов для употребления при установленном репозитарии по адресу */**oradb**/схема_БД/имя_таблицы*[...]. Тем не менее он параметризован, и поэтому представляет интерес для конечных потребителей БД.

Пример обращения:



Правила формирования адреса соответствуют правилам типа DBURITYPE.

Упражнение. Проверить работу сервлета DBURIServlet на следующих обращениях:

http://localhost:8080/oradb/SCOTT/DEPT/ROW[DEPTNO='30']
http://localhost:8080/oradb/SCOTT/DEPT/ROW[DEPTNO='30']/DNAME/text()
http://localhost:8080/oradb/SCOTT/EMPXML_SCHEMA_VIEW/ROW/employee[@empno='7499']

Сервлет DBURIServlet имеет параметры:

rowsettag – для смены имени корневого элемента, например:
http://localhost:8080/oradb/SCOTT/DEPT?rowsettag=Departments
contenttype – для указания типа MIME, например:
http://localhost:8080/oradb/SCOTT/DEPT?contenttype=text/plain
transform – для преобразования текста XML средствами XSLT.

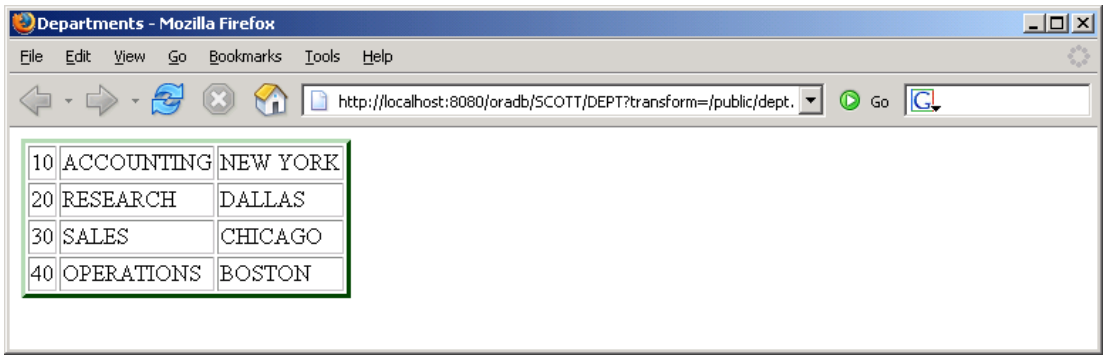
Зарегистрируем в качестве ресурса */public/dept.xsl* следующий текст:

```
<?xml version="1.0"?>
<xsl:stylesheet version="1.0"
 xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
>
<xsl:output method="html" />

<xsl:template match="ROW">
  <tr>
  <td><xsl:value-of select="DEPTNO"/></td>
  <td><xsl:value-of select="DNAME"/></td>
  <td><xsl:value-of select="LOC"/></td>
  </tr>
</xsl:template>

<xsl:template match="DEPT">
<html><head></head><body><title>Departments</title>
<table border="3" bordercolor="green">

  <xsl:apply-templates select="ROW"/>

</table>
</body></html>
</xsl:template>

</xsl:stylesheet>
```

Теперь обращение к XML DB по адресу:

http://localhost:8080/oradb/SCOTT/DEPT?transform=/public/dept.xsl&contenttype=text/html

даст следующий результат:



Текст с определением преобразования XSLT можно разместить и в таблице БД, обратившись за ним опять-таки через DBURITYPE примерно так:

transform=/oradb/SCOTT/XSLTDEFS/ROW[TAB='DEPT']/DEFINITION/text()

Упражнение. Создать таблицу XSLTDEFS с определением преобразования XSLT для таблицы DEPT (столбец TAB для имени таблицы и столбец DEFINITION для текста преобразования) и указать сервлету взять данные преобразования оттуда, а не из ресурса XML DB. Указание: поле DEFINITION определить типом VARCHAR2.

Прочие свойства сервлета приведены в документации по XML DB.

Создание сервлета на Java

Oracle XML DB (версии 9.2) поддерживает Java Servlet API версии 2.2 с некоторыми ограничениями, и с возможностью дополнительно установить сервлет поддержки JSP.

Подготовим текст сервлета в файле *XMLDBServlet.java*:

```
import java.io.PrintWriter;
import java.io.IOException;
import javax.servlet.GenericServlet;
import javax.servlet.ServletRequest;
import javax.servlet.ServletResponse;
import javax.servlet.ServletException;

public class XMLDBServlet extends GenericServlet {

  public void service (
              ServletRequest  request
            , ServletResponse response
  )
  throws ServletException, IOException {

    String s = request.getParameter ( "rex" );

    response.setContentType ( "text/xml" );

    PrintWriter out = response.getWriter ( );
    out.println ( "<?xml version=\"1.0\"?>" );
    out.println ( "<html><head>" );
    out.println ( "<title>My XMLDBServlet servlet demo</title>" );
    out.println ( "</head><body>" );
    out.println ( "<h2>Ave, " + s + " !</h2>" );
    out.println ( "</body></html>" );
    out.close ( );
```

```
   }
}
```

Пример ради общности рассматривает употребление суперкласса `GenericServlet`, а не его наследника `HttpServlet`.

Загрузим сервлет в БД одним из возможных способов:

```
>loadjava -grant public -u scott/tiger -r XMLDBServlet.java
```

Проверкой легко убедиться, что СУБД не только загрузит в БД исходный текст, но и получит из него класс.

Чтобы сервлет мог вызываться извне, сведения о нем требуется занести в ресурс-файл */xdbconfig.xml* в репозитарии XML DB. Это файл с «объектно-реляционным» хранением, и подправить его можно либо через WebDAV (системами, обеспечивающими такую правку), либо обычными функциями UPDATEXML и прочими. (Именно этот файл не допускает удаления из репозитария, поэтому выгрузить его, подправить и загрузить заново невозможно).

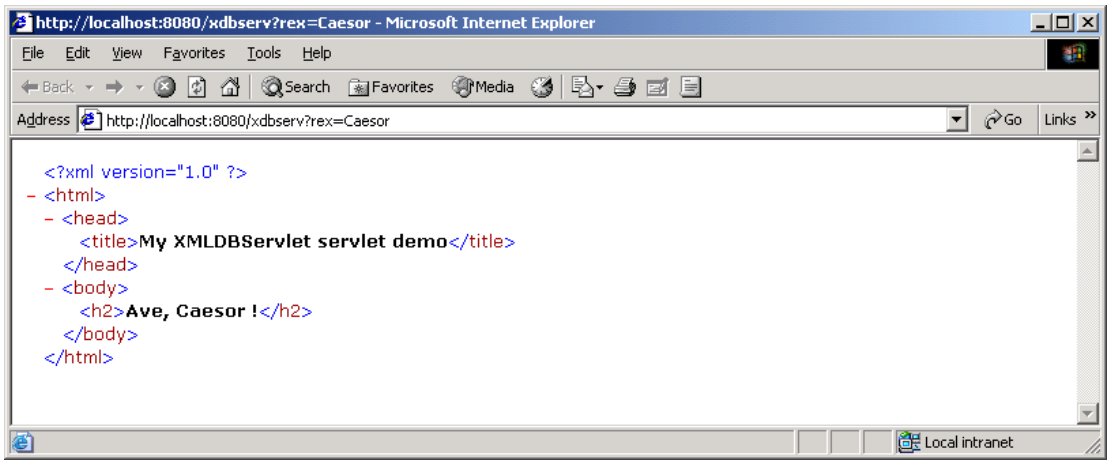
Однако при желании, для удобства правки файла */xdbconfig.xml* можно использовать специально созданые для этого подпрограммы пакета DBMS_XDB. Выполним в SQL*Plus от имени пользователя XDB:

```
DECLARE
xval XMLTYPE;

xnode1 XMLTYPE := XMLTYPE (
'
<servlet xmlns="http://xmlns.oracle.com/xdb/xdbconfig.xsd">
  <servlet-name>XMLDBServletPrimer</servlet-name>
  <servlet-language>Java</servlet-language>
  <display-name>Oracle XML DB Servlet Primer</display-name>
  <servlet-class>XMLDBServlet</servlet-class>
  <servlet-schema>SCOTT</servlet-schema>
</servlet>
');

xnode2 XMLTYPE := XMLTYPE (
'
<servlet-mapping xmlns="http://xmlns.oracle.com/xdb/xdbconfig.xsd">
  <servlet-pattern>/xdbserv</servlet-pattern>
  <servlet-name>XMLDBServletPrimer</servlet-name>
</servlet-mapping>
');

BEGIN
SELECT DBMS_XDB.CFG_GET ( ) INTO xval FROM dual;

SELECT
 INSERTCHILDXML (
   xval, '/xdbconfig//servlet-list', 'servlet', xnode1
 )
INTO xval FROM dual
;
SELECT
 INSERTCHILDXML (
   xval, '/xdbconfig//servlet-mappings', 'servlet-mapping', xnode2
 )
INTO xval FROM dual
;

DBMS_XDB.CFG_UPDATE ( xval );
COMMIT;
END;
/
```

Теперь обращение по адресу:

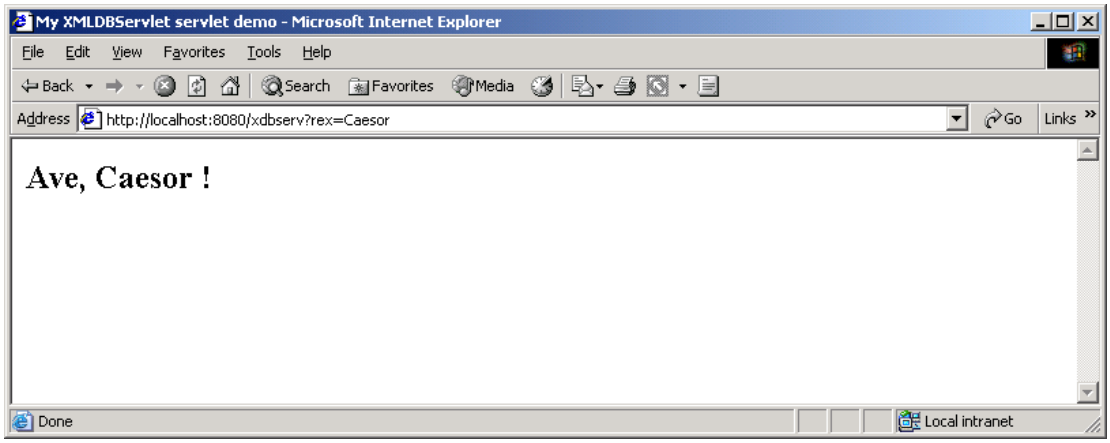
http://localhost:8080/xdbserv?rex=Caesor

даст результат:



Выдача данных в формате HTML и обращение к БД делается как обычно.

Например, для перехода к HTML в *данном* случае даже необязательно пользоваться классом `HttpServlet`. Достаточно в описании сервлета заменить `text/xml` на `text/html` и изъять строку `out.println ( "<?xml version=\"1.0\"?>" );`. Перетранслировав сервлет, получим по тому же адресу:



Обращение в тексте сервлета к данным БД делается как обычно для Java (через JDBC), и оставляется для самостоятельного упражнения.