

Aula Prática

GDI CC - 2018.2

XML (Aula 1)

Roteiro

- XMLType
- SYS_XMLGEN
- DBMS_XMLGEN
- DTD
- XML SCHEMA
- Exercícios

XMLType

- Tipo pré-definido para criar, extrair, indexar e validar dados XML com XML Schema
 - Pode ser aplicado a Coluna ou Tabela
- Quando usar XMLType ?
 - Armazenar e recuperar XML como um todo
 - Consultas em elementos XML
 - Utilizar a funcionalidade XPath
 - Preparar para futuras atualizações

Manipulando XMLType

- Tabela com atributo XML

```
CREATE TABLE tb_produto(  
  identificacao NUMBER(5),  
  descricao SYS.XMLTYPE,  
  nome VARCHAR2(35)  
);
```

- Inserindo

```
INSERT INTO tb_produto (identificacao, descricao,  
  nome) VALUES (1001,  
  sys.XMLType.createXML('<Produto num="100">  
  <Tipo>Madeira</Tipo> </Produto>'), 'Porta');
```

Manipulando XMLType

- Consulta

```
SELECT  
p.descricao.extract('/Produto/Tipo/text()').getStringVal()  
"Tipos" FROM tb_produto P;
```

- Modificar Dados

```
UPDATE tb_produto SET descricao =  
Sys.XMLType.createXML('<Produto num="100">  
<Tipo>Aluminio</Tipo> </Produto>') WHERE identificacao =  
1001;
```

SYS_XMLGEN

- Função SYS_XMLGEN ()
 - Usada para gerar XML dentro de consultas SQL
 - Mistura elementos de XML com SQL
 - Retorna um tipo XMLType
- Consulta :

```
SELECT SYS_XMLGEN(atributo) as XML FROM tabela;
```

Pacote DBMS_XMLGEN

- Cria documentos XML a partir de consultas SQL
- Uso:
- Permitir output na interface de caracteres

```
Set serveroutput on;
```

- Definir um bloco :
 - Declarar variável para criar um contexto
 - Variavel contexto
 - Tipo(pacote)
 - Método do pacote

```
ctx dbms_xmlgen.ctxhandle;
```

Pacote DBMS_XMLGEN

- Declarar variável do tipo CLOB para armazenar o arquivo XML gerado

```
result clob;
```

- Criar um novo contexto com a consulta SQL apropriada
 - Variavel contexto
 - Consulta SQL para gerar resposta em documento XML

```
ctx := dbms_xmlgen.newContext('select * from carro');
```


Pacote DBMS_XMLGEN

- Personalizar a tag raiz do documento
 - Método do Pacote
 - Personalização da raiz

```
DBMS_XMLGEN.setRowset(ctx, 'Locadora');
```

- Personalizar a tag de entidade
 - Método do Pacote
 - Personalização da linha

```
DBMS_XMLGEN.setRowTag(ctx, 'CARRO');
```

Pacote DBMS_XMLGEN

- Gerar um valor CLOB como resultado

```
result := dbms_xmlgen.getXML(ctx);
```

- Dar saída do resultado
 - Variável do tipo clob

```
dbms_output.put_line(result);
```

- Fechar o contexto

```
dbms_xmlgen.closeContext(ctx);
```

Pacote DBMS_XMLGEN

- Exemplo Completo:

```
set serveroutput on;
declare
ctx dbms_xmlgen.ctxhandle;
result clob;
begin
ctx := dbms_xmlgen.newContext('select * from carro');
DBMS_XMLGEN.setRowsetTag(ctx, 'LOCADORA');
DBMS_XMLGEN.setRowTag(ctx, 'CARRO');
result := dbms_xmlgen.getXML(ctx);
dbms_output.put_line(result);
dbms_xmlgen.closeContext(ctx);
end;
/
```

DTD

- Definem a estrutura de um documento XML (Onde estão especificados e quais elementos e atributos são permitidos).
- **DTD interno:** contida no documento e visível apenas dentro dele.
- **DTD externo:** está contida em um arquivo com extensão '.dtd' e é chamado pelo documento XML .

DTD

- Declarando:

```
<!DOCTYPE elementoRaiz [  
[conteúdo]  
>
```

DTD

- DTD Externa

```
<?xml version = '1.0' encoding = 'utf-8'?>  
<!DOCTYPE nome_raiz SYSTEM “nomeArquivo.dtd”  
[“http://.../nomeArquivo.dtd”] >
```

DTD - Elementos

<!ELEMENT nomeElemento (conteúdo) >

- Conteúdo de um elemento:
 - EMPTY: Vazio
 - #PCDATA: Caracteres analisáveis
- Conectores :
 - Sequência (,): representa o 'e'.
 - Escolha (|): representa o 'ou'.

DTD - Elementos

- Controle de frequência:
 - Elemento opcional sem repetição (nenhuma ou uma vez) : ?.
 - Elemento opcional com repetição (nenhuma, uma ou várias vezes): *.
 - Elemento obrigatório (uma ou mais vezes): +.

DTD - Atributos

```
<!ATTLIST nomeElemento nomeAtt tipo valorPadrão >
```

- Alguns Tipos:
 - CDATA : tipo string.
 - ID : identifica de forma única um elemento.
 - IDREF: referencia para atributos ID.
 - ENTITY: funciona como uma Macro.
- Valor Padrão:
 - Valor default: “valDefault”
 - Atributo obrigatório: #REQUIRED.
 - Atributo opcional: #IMPLIED.
 - Atributo com valor Fixo (constante): #FIXED
 - Atributo enumerado: (val1|...|valN) valDefault

DTD Externo - Exemplo

XML

```
<?xml version="1.0"?>

<!DOCTYPE note SYSTEM
"note.dtd">

<note>

  <para>Tove</para>

  <de>Jani</de>

  <titulo>Reminder</titulo>

  <mensagem>Don't forget me this
weekend!</mensagem>

</note>
```

DTD

```
<!ELEMENT note
(para,de,titulo,mensagem)>
<!ELEMENT para(#PCDATA)>
<!ELEMENT de(#PCDATA)>
<!ELEMENT titulo(#PCDATA)>
<!ELEMENT mensagem
(#PCDATA)>
```

XML SCHEMA

- Definem a estrutura de um documento XML e garante a conformidade do documento.
- Melhor suporte para dados (restrição, formato, conversão, criação de tipos a partir de tipos padrões, etc).
- ‘.xs’ ou ‘.xsd’
- Como é chamado no arquivo xml ?

```
<elementoRaiz xmlns:xsi = “url_válida”
```

```
xsi:noNamespaceSchemaLocation= “nomeArquivo.xsd” >
```

XML SCHEMA

- Cabeçalho:

```
<?xml version="1.0" encoding = "utf-8" ?>  
<xsd:schema xmlns:xsd = "url_válida">
```

- Elemento simples (string, decimal, integer, boolean, date, time):

```
<xsd:element name = "nome" [ind_ocorrencia] type =  
"xsd:tipo">
```

XML SCHEMA

- Elemento Complexo (empty, outros elementos, texto, elementos e texto):

```
<xsd:element name = "nome" [indicador_ocorrendia]>  
  
<xsd:complexType>  
  
<xsd:Indicador_ordem>
```

- Atributo

```
<xsd:attribute name = "nome " [indicador_ocorrendia]  
type = "xsd:tipo"/>
```

```
<?xml version="1.0" encoding="UTF-8"?>

<shiporder orderid="889923"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="shiporder.xsd">
  <orderperson>John Smith</orderperson>
  <shipto>
    <name>Ola Nordmann</name>
    <address>Langgt 23</address>
    <city>4000 Stavanger</city>
    <country>Norway</country>
  </shipto>
  <item>
    <title>Empire Burlesque</title>
    <note>Special Edition</note>
    <quantity>1</quantity>
    <price>10.90</price>
  </item>
  <item>
    <title>Hide your heart</title>
    <quantity>1</quantity>
    <price>9.90</price>
  </item>
</shiporder>
```

```
<?xml version="1.0" encoding="UTF-8" ?>  
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">  
...  
</xs:schema>
```

```
<xs:element name="shiporder">
  <xs:complexType>
    <xs:sequence>
      ...
    </xs:sequence>
  </xs:complexType>
</xs:element>
```



```
<xs:element name="orderperson" type="xs:string"/>
```

```
<xs:element name="shipto">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="name" type="xs:string"/>
      <xs:element name="address" type="xs:string"/>
      <xs:element name="city" type="xs:string"/>
      <xs:element name="country" type="xs:string"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

```
<xs:element name="shipto">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="name" type="xs:string"/>
      <xs:element name="address" type="xs:string"/>
      <xs:element name="city" type="xs:string"/>
      <xs:element name="country" type="xs:string"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

```
<xs:element name="item" maxOccurs="unbounded">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="title" type="xs:string"/>
      <xs:element name="note" type="xs:string" minOccurs="0"/>
      <xs:element name="quantity" type="xs:positiveInteger"/>
      <xs:element name="price" type="xs:decimal"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```


Exercício

- Baixe o arquivo XML - Arquivos.rar
 - bit.ly/aulaXml

Exercício 1

- Adicione um campo chamado “curso” ao tipo evento.

Exercício 1 - Resposta

- Adicione um campo chamado “curso” ao tipo evento.

```
ALTER type tp_evento ADD attribute (curso XMLType)  
CASCADE;
```


Exercício 2

- Utilize SYS_XMLGEN para gerar um XML com o nome dos membros que presenciaram o evento de id = 2.

Exercício 2 - Resposta

- Utilize SYS_XMLGEN para gerar um XML com o nome dos membros que presenciaram o evento de id = 2.

```
SELECT SYS_XMLGEN (p.membro.nome) as XML FROM  
tb_presenciar p
```

```
WHERE p.evento.id = 2
```

```
GROUP BY p.membro.nome ;
```

Dúvidas?

Próximas Entregas:

Checklist XML

01/12 !!!