**XML y XQuery. Prácticas**

**Práctica 1**. Consideremos un archivo de XML que almacena algunos datos del sistema solar de acuerdo con el siguiente formato DTD:

```
<!ELEMENT planetas(planeta)+>
<!ELEMENT planeta(nombre,diametro,masa,satelites?)>
<!ELEMENT satelites(satelite)+>
<!ELEMENT satelite(nombre)>
```

diseña las siguientes consultas XQuery:

1. Muestra un listado de todos los nombres de planetas cuya masa sea mayor que 1 con el siguiente formato:

```
<nombre>Jupiter</nombre>
<nombre>Saturno</nombre>
<nombre>Neptuno</nombre>
```

2. Muestra un listado de todos los nombres de planetas y los nombres de sus satélites con el siguiente formato:

```
<resultado>
    <plan>Tierra</plan>
    <nom>Luna</nom>
</resultado>
<resultado>
    <plan>Marte</plan>
    <nom>Fobos</nom>
    <nom>Deimos</nom>
</resultado>
```

**Solución**:

```
for $x in doc("datos.xml")/planetas/planeta
return <resultado>
           <plan>
           {data($x/nombre)}
           </plan>
           {
               for $y in $x/satelites/satelite
               return <nom>{data($y/nombre)}</nom>
           }
       </resultado>
```

**Práctica 2**. Consideremos una base de datos sobre alojamientos vacacionales representada mediante un fichero XML con información acorde a la siguiente especificación:

```
<!ELEMENT alojamientos(alojamiento)+>
<!ELEMENT alojamiento(cif,direccion,telefono,habitaciones?)>
<!ELEMENT habitaciones(habitacion)+>
<!ELEMENT habitacion (precio,estado)>
<!ELEMENT precio (#PCDATA)> <!--Precio de la habitación-->
<!ATTLIST alojamiento tipo (hotel|hostal|casarural) #required>
<!ATTLIST habitacion id CDATA #required>
<!ATTLIST estado est CDATA #required>
```

diseña las siguientes consultas XQuery:

1. El nombre de cada hotel con su teléfono.
2. Nombre del hotel y número de habitaciones.
3. Para cada hotel, el precio mínimo de sus habitaciones y máximo.
4. El nombre y el teléfono de los hoteles con habitaciones libres.

Solución:

```
for $h in doc("datos.xml")//alojamiento
where $h[@tipo="hotel"]
return <hotel>
{$h/nombre}
{$h/telefono}
</hotel>

for $h in doc("datos.xml")//alojamiento
let $numhabitaciones:=count($h/habitaciones/habitacion)
where $h[@tipo="hotel"]
return <hotel>
{$h/nombre}
<numero-habitaciones>
{$numhabitaciones}
</numero-habitaciones>
</hotel>

for $h in doc("datos.xml")//alojamiento
let $preciomin:=min($h/habitaciones/habitacion)
let $preciomax:=max($h/habitaciones/habitacion)
where $h[@tipo="hotel"]
return <hotel>
{$h/nombre}
<precios>
<minimo>
{$preciomin}
</minimo>
<maximo>
{$preciomax}
</maximo>
</precios>
</hotel>

for $a in doc("datos.xml")//alojamiento
where some $h in $a//estado
satisfies $h/@id="libre"
return <hotel>
{$a/nombre}
{$a/telefono}
</hotel>
```

**Práctica 3**. Consideremos una base de datos sobre información de una línea aérea representada mediante un fichero XML cuyo formato es acorde a la siguiente especificación:

```
<!ELEMENT lineas(linea)+>
<!ELEMENT linea(numero,asientos)>
```

```
<!ELEMENT asientos(asiento)+>
<!ELEMENT numero (#PCDATA)> <!--Número del vuelo-->
<!ELEMENT asiento (#PCDATA)> <!--Ver última línea-->
<!ATTLIST linea origen CDATA #required>
<!ATTLIST linea destino CDATA #required>
<!ATTLIST asiento estado (libre|ocupado) #required>
```

diseña las siguientes consultas XQuery:

1. Todos los números de vuelos con origen *Madrid* según el formato:

```
<vuelo-Madrid>
    <num>1234 </num>
    <num>453 </num>
    ...
</vuelo-Madrid>
```

2. El número de los asientos de cada vuelo con origen *Madrid* según el formato:

```
<vuelo-Madrid>
    <num>1234 </num>
    <asientos>59</asientos>
    <num>453 </num>
    <asientos>29</asientos>
    ...
</vuelo-Madrid>
```

3. El número de todos los vuelos con plazas libres según el formato:

```
<vuelo>
    <num>1234 </num>
    <num>453 </num>
    ...
</vuelo>
```

4. Todas las plazas libres por número de vuelo según el formato:

```
<vuelo>
<num>1234</num>
    <plazas-libres>
        <asiento> 2 </asiento>
        <asiento> 7 </asiento>
        <asiento> 15 </asiento>
        ...
    </plazas-libres>
</vuelo>
```

**Solución**:

1. Todos los números de vuelos con origen *Madrid*:

```
for $v in doc("datos.xml")//linea
where $v[@origen="Madrid"]
return
<vuelo-Madrid>
<num>{data($v/numero)} </num>
</vuelo-Madrid>
```

2. El número de los asientos de cada vuelo con origen *Madrid*:

```
for $v in doc("datos.xml")//linea
let $numasientos:=count($v/asientos/asiento)
where $v[@origen="Madrid"]
return
<vuelo-Madrid>
<num>{data($v/numero)} </num>
<asientos>{$numasientos}</asientos>
</vuelo-Madrid>
```

3. El número de todos los vuelos con plazas libres:

```
for $v in doc("datos.xml")//linea
where some $h in $v//asiento satisfies $h/@estado="libre"
return
<vuelo>
<num>{data($v/numero)} </num>
</vuelo>
```

4. Todas las plazas libres por número de vuelo:

```
for $v in doc("datos.xml")//linea
return
<vuelo>
<num>{data($v/numero)}</num>
<plazas-libres>
{
for $a in $v//asiento
where $a[@estado="libre"]
return data($a)
}
</plazas-libres>
</vuelo>
```

**Práctica 4**. Una academia lleva la gestión de un curso académico mediante una base de datos con XML con dos archivos:

- `academia.xml` contiene información sobre la asignaturas en las que se ha matriculado cada alumno.
- `asignaturas.xml` contiene información general de las asignaturas.

El formato de los archivos es el siguiente.
`academia.xml`:

```
<!ELEMENT academia (alumno+)>
<!ELEMENT alumno (asignatura+)>
<!ELEMENT asignatura (nota+)>
<!ELEMENT nota (#PCDATA)>
<!ATTLIST alumno nif CDATA #REQUIRED>
<!ATTLIST asignatura cod CDATA #REQUIRED>
```

`asignaturas.xml`:

```
<!ELEMENT asignaturas (asignatura+)>
<!ELEMENT asignatura (nombre,profesor+)>
<!ELEMENT nombre (#PCDATA)>
```

```
<!ELEMENT profesor EMPTY>
<!ATTLIST asignatura cod CDATA #REQUIRED)>
<!ATTLIST profesor nif-prof CDATA #REQUIRED)>
```

Diseña las siguientes consultas:

1. NIF de los alumnos y número de asignaturas en las que se ha matriculado con el siguiente formato:

```
<alumno>
    <nif>0001</nif>
    <num-asig>3</num-asig>
</alumno>
<alumno>
    <nif>0002</nif>
    <num-asig>3</num-asig>
</alumno>
...
```

2. Alumnos que tienen todos exámenes aprobados con el siguiente formato:

```
<alumno-apto>0002</alumno-apto>
<alumno-apto>0004</alumno-apto>
...
```

3. Nota máxima y mínima para aquellos alumnos que han aprobado todo con el siguiente formato:

```
<info>
    <nif>0002</nif>
    <nota-maxima>9</nota-maxima>
    <nota-minima>5</nota-minima>
</info>
<info>
    <nif>0004</nif>
    <nota-maxima>7</nota-maxima>
    <nota-minima>6</nota-minima>
</info>
...
```

4. Listado de profesores por alumno con el siguiente formato:

```
<alumno>
    <nif>0001</nif>
    <profesores>
        <profesor nif-prof="0003"/>
        <profesor nif-prof="0023"/>
        <profesor nif-prof="0003"/>
        <profesor nif-prof="0333"/>
    </profesores>
</alumno>
<alumno>
    <nif>0002</nif>
    <profesores>
        <profesor nif-prof="0003"/>
        <profesor nif-prof="0023"/>
        <profesor nif-prof="0003"/>
        <profesor nif-prof="0401"/>
```

```
        <profesor nif-prof="0402"/>
        <profesor nif-prof="0403"/>
        <profesor nif-prof="0404"/>
    </profesores>
</alumno>
...
```

**Soluciones:**

```
for $x in doc("academia.xml")//alumno
let $num:=count($x/asignatura)
return
<alumno>
        <nif>
            {data($x/@nif)}
        </nif>
        <num-asig>
            {$num}
        </num-asig>
</alumno>
```

```
for $x in doc("academia.xml")//alumno
where every $y in $x//nota satisfies $y>4
return
<alumno-apto>
    {data($x/@nif)}
</alumno-apto>
```

```
for $x in doc("academia.xml")//alumno
where every $y in $x//nota satisfies $y>4
return
<info>
    <nif>{data($x/@nif)}</nif>
    <nota-maxima>{data(max($x//nota))}</nota-maxima>
    <nota-minima>{data(min($x//nota))}</nota-minima>
</info>
```

```
for $x in doc("academia.xml")//alumno
return
<alumno>
    <nif>
        {data($x/@nif)}
    </nif>
    <profesores>
        {
                for $y in $x/asignatura
            let $z:= doc("asignaturas.xml")//asignatura[@cod=$y/@cod]/profesor
            return
            $z
        }
    </profesores>
</alumno>
```