Ampliación de bases de datos
Grado en Ingeniería informática. Grupos B y C
Examen Primera convocatoria. Curso 2022-2023

Ejercicio 1. (4 puntos). 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>
```

Ejercicio 2. (4 puntos). Una empresa de telefonía tiene una base de datos en datalog formada por una base de datos extensional con el siguiente formato:

```
%usuario(cod_usuario, nombre)
%linea(cod_usuario, num_telefono)
%llamada(cod_llamada, num_telefono, duracion_en_segundos, dia, mes, anio)
```

El coste de cada llamada es el siguiente:

- Llamadas con duración menor que 30 segundos: 2 céntimos cada segundo.
- Llamadas con duración entre 30 y 60 segundos: 1 céntimo cada segundo.
- Llamadas con duración mayor que 60 segundos: 0,5 céntimos cada segundo.

Diseña las siguientes reglas:

1. `factura_linea(L,Cost,M,A)`: `Cost` es el coste de la línea `L` en el mes `M` y en el año `A`. Ejemplo:

```
DES> factura_linea(L,Cost,M,A).

{
  factura_linea(1111,0.63,1,2023),
  factura_linea(1111,0.78,1,2022),
  factura_linea(1111,1.41,2,2023),
  factura_linea(2222,0.505,9,2023),
  factura_linea(2222,0.505,10,2023),
  factura_linea(3333,4.470000000000001,11,2023),
  factura_linea(5555,0.03,11,2022),
  factura_linea(5555,0.505,10,2022),
  factura_linea(6666,0.505,10,2022),
  factura_linea(6666,11.61,11,2022)
}
Info: 10 tuples computed.
```

2. `factura_usuario(Nombre,Cost,M,A)`: `Cost` es el coste de la factura del usuario `Nombre` en el mes `M` y en el año `A`. Ejemplo:

```
DES> factura_usuario(Nombre,Cost,M,A).

{
  factura_usuario(ana,0.505,10,2022),
  factura_usuario(ana,11.639999999999999,11,2022),
  factura_usuario(juan,0.505,9,2023),
  factura_usuario(juan,0.505,10,2023),
  factura_usuario(juan,4.470000000000001,11,2023),
  factura_usuario(pepe,0.63,1,2023),
  factura_usuario(pepe,0.78,1,2022),
  factura_usuario(pepe,1.41,2,2023)
}
Info: 8 tuples computed.
```

3. `linea_max(L)`: `L` es la línea con el mayor número de llamadas. Ejemplo:

```
DES> linea_max(L).

{
```

```
  linea_max(1111)
}
Info: 1 tuple computed.
```

4. Se considera que una llamada que haya durado más del triple que la media es defectuosa. Una línea con al menos una llamada defectuosa se considera también defectuosa. Determina los usuarios con una línea defectuosa: `usuarios_lineas_defectuosas(Nombre,L)` donde `Nombre` es el usuario de la línea defectuosa y `L` el número de la línea. Ejemplo:

```
DES> usuarios_lineas_defectuosas(Nombre,L).

{
  usuarios_lineas_defectuosas(ana,6666),
  usuarios_lineas_defectuosas(juan,3333)
}
Info: 2 tuples computed.
```

Soluciones:

```
coste_llamada(Codigo,L,Coste,M,A):- llamada(Codigo,L,X,_,M,A),X<30,Coste is X*0.02.
coste_llamada(Codigo,L,Coste,M,A):- llamada(Codigo,L,X,_,M,A),X=<30,X<60,Coste is X*0.01.
coste_llamada(Codigo,L,Coste,M,A):- llamada(Codigo,L,X,_,M,A),X>60,Coste is X*0.005.

factura_linea(L,Cost,M,A):-group_by(
        coste_llamada(Cod,L,Coste,M,A),
        [L,M,A],
        Cost=sum(Coste)
        ).
```

```
factura_usuario(Nombre,Cost,M,A):-group_by(
        linea(Cod,L),usuario(Cod,Nombre),factura_linea(L,Coste,M,A),
        [Nombre,M,A],
        Cost=sum(Coste)
).
```

```
num_llamadas_linea(L,Num):-group_by(
        llamada(C,L,_,_,_,_,_),
        [L],
        Num=count(C)
).
max_llamadas_linea(Num):-group_by(
        num_llamadas_linea(_,N),
        [],
        Num=max(N)
).
linea_max(L):- num_llamadas_linea(L,Num),max_llamadas_linea(Num).
```

```
duracion_media(D):-group_by(
        llamada(_,_,S,_,_,_,_),
        [],
        D=avg(S)
).
usuarios_lineas_defectuosas(Nombre,L):-usuario(C,Nombre),linea(C,L),
llamada(Codigo,L,X,D,M,A),duracion_media(Q),X>3*Q.
```

Ejercicio 3. (2 puntos).

Una cadena hotelera necesita almacenar información sobre hoteles y habitaciones. Para identificar un hotel se utiliza un código. Para identificar una habitación se necesita el código del hotel en el que se encuentra y el número de habitación. Cada hotel es dirigido por un único administrador que tiene, al menos, un teléfono de contacto.

Chapuzas Informáticas S.A. ha ofrecido a la cadena hotelera el siguiente diseño de la base de datos:

- Tabla 1: hoteles(**cod-hotel**,nombre,dirección,ciudad,nif-administrador,telf-administrador)
- Tabla 2: estancias-habitaciones(**cod-hotel**,**num-habitación**,**fecha**,nif-cliente,días,tipo-habitacion)

Determina si esta base de datos está en 3FN. Si no es así, indica qué cambios harías para que esté en 3FN. Por supuesto, no es válido añadir nuevos campos.

Soluciones:

De la tabla 1 surgen las tablas:

- hoteles(**cod-hotel**,nombre,dirección,ciudad,nif-administrador)
- administradores(**nif-administrador**,**telf-administrador**)

Observación: la dirección no determina la ciudad. Por ejemplo, puede haber una calle que se llame Gran Vía en dos ciudades distintas.

De la tabla 2 surgen las tablas:

- estancias(**cod-hotel**,**num-habitación**,**fecha**,nif-cliente,días)
- habitaciones(**cod-hotel**,**num-habitación**,tipo-habitacion)

Duración del examen: 2:30 horas.