Laboratorio #6
Interacción con bases de datos parte 2 - *PAREJAS*

I. Modalidad y fecha de entrega

a) El laboratorio debe hacerse en parejas
b) Debe ser enviado antes de la fecha límite de entrega: martes 6 de Marzo a las 23:59
c) Luego de la fecha límite se restarán 10 puntos por cada hora de atraso en la entrega

II. Descripción de la actividad

En base a si diseño de base de datos trabajado anteriormente y a la información contenida en **bulk-loading.pdf** desarrolle un *script* en Python que permita transformar la información de sus subastas en archivos **.dat** que poder cargar en bloque (*bulk load*) hacia una base de datos SQLite.

Actividad 1

Asegúrese de familiarizarse con el documento adjunto que describe el proceso de carga en bloque a SQLite.

Su tarea consiste en desarrollar un programa que transforme la data XML en archivos de carga en bloque SQLite consistentes con el esquema relacional desarrollado durante el Laboratorio #5.

Para iniciar se le provee un *skeleton_parser.py* que contiene una implementación incompleta del parser, pero con funciones útiles. Su tarea consiste en implementar la función **parseXml**, que debe *parsear* y extraer cada archivo XML y generar los archivos para carga en bloque apropiados de acuerdo a su esquema relacional.

Por favor copie *skeleton_parser.py* hacia *parser.py* y desarrolle su programa en ese archivo.

La forma de uso del parser debe ser mediante el comando: **python parser.py data/ítems-0.xml**, en donde el archivo **ítems-0.xml** puede ser sustituído por un conjunto de archivos: **python parser.py data/ítems-0.xml data/ítems-1.xml data/ítems-2.xml**.

Se sugiere encarecidamente *testear* su script con el archivo **ítems-0.xml** únicamente antes de utilizarlo con el conjunto completo de archivos.

Se recomienda utilizar los caracteres **<>** como delimitadores.

Asegúrese de utilizar la extensión **.dat** para los archivos generados.

Valores de moneda y fechas-horas

El parser inicial provisto contiene las funciones **transformDollar(string)** y **transformDttm(string)** que le permitirán hacer la conversión entre los textos de valores monetarios y fechas-horas de tal forma que pueda generarlos correctamente hacia los archivos **.dat**.

Eliminación de duplicados

Si nota que durante el *parsing* algunas tuplas se generan múltiples veces pero a usted le interesa conservar solo una copia puede desarrollar la funcionalidad de eliminación de duplicación como parte de su parser.

Actividad 2

Modifique el archivo **schema.sql** desarrollado en el Laboratorio #5 de tal forma que incluya los comandos necesarios para reconstruir las tablas definidas incluso si están ya creadas, por ejemplo:

```
DROP TABLE IF EXISTS Item;

CREATE TABLE Item ( … );
```

Adicionalmente trabaje un archive **carga.sql** que contenga las instrucciones necesarias para cargar la información a sus tablas. Este archivo debe verse similar a:

```
.separator <>
.import ítems.dat Items
UPDATE Items SET …
.import users.dat User
….
```

Si no sabe a qué se refieren estos comandos refiérase a **bulk-loading.pdf**.

Tanto **schema.sql** como **carga.sql** deben ejecutarse correctamente cuando se corren contra el comando **sqlite**, por ejemplo:

```
sqlite <db_name> < create.sql
```

Actividad 3

La última actividad consiste en *testear* su nueva base de datos.

Inicie realizando *queries* sencillos que le permitan verificar que la data fue cargada correctamente. Cuando esté seguro acerca de que la data cargada es correcta prepare consultas SQL para cumplir con las siguientes tareas:

1. Encuentre el número de usuarios en la base de datos (R: 13422)
2. Encuentre el número de usuarios de *New York* (es decir, usuarios cuya *location* sea el string “New York”) (R: 80)
3. Encuentre el número de subastas que pertenecen exactamente a cuatro categorías (R: 8365)
4. Encuentre el ID de la subasta con el valor más alto de *current price* (pueden ser múltiples) (R: 1046871451)
5. Encuentre el número de vendedores cuyo *rating* es mayor a 1000 (R: 3130)
6. Encuentre el número de usuarios que son *sellers* y también *bidders* (R: 6717)
7. Encuentre el número de categorías que incluyen al menos un elemento con una subasta de más de $100.00 (R: 150)

Ponga cada query en su propio archivo: *query1.sql*, *query2.sql*, *query3.sql*, etc.

Hints: si el resultado del query2 es incorrecto esto puede deberse a que la ubicación del seller haya sido especificada incorrectamente. La localidad del seller es la localidad de su ítem. Si el resultado del query 3 es incorrecto esto puede deberse a que su base de datos contiene duplicados incorrectos.

III. Temas a reforzar

- Interacción con bases de datos
- Consultas SQL

IV. Documentos a entregar

Debe publicar los siguientes documentos:

Archivo de diseño con ER y descripción textual de BD (en PDF)

parser.py
schema.sql
carga.sql
query1.sql
query2.sql
query3.sql
query4.sql
query5.sql
query6.sql
query7.sql

*No debe incluir los archivos **.dat** ni **.xml** en su entrega.*

V. Evaluación

- Entregables de la actividad 1: 40 puntos
- Entregables de la actividad 2: 20 puntos
- Entregables de la actividad 3: 40 puntos
- **Total: 100 puntos**