Curso Bases de datos y Programación Orientada a Objetos
Master Universitario en Modelización Matemática, Estadística y Computación

Problema JAVA
(I/O.- Adaptación de datos a requerimientos de base de datos)

Supongamos que tenemos una base de datos con referencias bibliográficas, y que cada cierto tiempo recibimos un fichero con datos extraídos de scholar.google.es. Será preciso adaptar estos datos a nuestras necesidades concretas para incorporarlos. En estos ficheros tendremos una línea por cada dato bibliográfico, que estará precedido por un número de referencia asignado previamente, y un carácter que indica el tipo de dato (t=título, y=año, p=publicación, r=referencias, a=autor, l=link).

Ejemplo:

```
101t Polymerase chain reaction can detect bacterial DNA in aseptically loose total hip arthroplasties
101y 2004
101p journals.lww.com
101r 8 455 30 532 31 234 21 333 13 19
101a MT. Clarke "University of Toronto, utoronto.ca, Canada"
101a CP. Roberts "Rheinische Friedrich Wilhelms Universitat Bonn, uni-bonn.de, Germany"
101a PTH. Lee "University of Massachusetts Amherst, umass.edu, USA"
101a J. Gray "University of California Irvine, uci.edu, USA"
101l http://journals.lww.com/corr/2004/10000/Polymerase_Chain_Reaction_Can_Detect_Bacterial_DNA.23.aspx
```

La información correspondiente a autores(a) puede aparecer varias veces, de modo que en cada ocasión se referirá a un autor y llevará siempre asociada la filiación del mismo entre comillas (en la imagen vemos 4). Una referencia bibliográfica puede estar completa o no; cuando no lo está deberá consignarse algo predeterminado (p.ej., si no hay autores se considerará uno con el texto "indeterminado"). El número de referencia y el carácter que indica el tipo de información van juntos y seguidos por un espacio en blanco. El número de referencia no tiene una longitud determinada (en la imagen es de tres cifras, pero no siempre lo será). No hay un orden predeterminado para las líneas, de modo que la información referente a los distintos artículos podrá intercalarse sin restricciones. Tampoco se asegura que todos los campos de un mismo artículo aparezcan seguidos, sino que puede haber cierto desorden en este sentido.

La tarea consiste en realizar un programa capaz de recoger toda la información que se nos aporta y generar los elementos necesarios para cumplimentar en una base de datos cada una de las tablas necesarias. Como comprobación del funcionamiento se aplicará a un caso concreto (fichero "DB.txt")

Las tablas serán las generadas en la base de datos como:

```
CREATE TABLE papersreferences (
 ref_paper INTEGER,
 cited_ref_paper INTEGER,
 PRIMARY KEY(ref_paper,cited_ref_paper)
);
```

```
CREATE TABLE publication (
  ref_publication INTEGER PRIMARY KEY,
  title VARCHAR(200)
);
```

```
CREATE TABLE papersauthors (
  ref_paper INTEGER,
  ref_author INTEGER,
  PRIMARY KEY(ref_paper,ref_author)
);
```

```
CREATE TABLE authorafiliation (
  ref_author INTEGER,
  ref_afiliation INTEGER,
  PRIMARY KEY(ref_author,ref_afiliation)
);
```

```
CREATE TABLE afiliations (
  ref_afiliation INTEGER PRIMARY KEY,
  university VARCHAR(65),
  webpage VARCHAR(30),
  country VARCHAR(15)
);
```

```
CREATE TABLE papersinfo (
  ref_paper INTEGER PRIMARY KEY,
  title VARCHAR(200),
  publiyear INTEGER,
  ref_publication INTEGER,
  url VARCHAR(200) );
```

```
CREATE TABLE authors (
  ref_author INTEGER PRIMARY KEY,
  name VARCHAR(25) );
```

A tener en cuenta:

1.- las referencias de los artículos (ref_paper y cited_ref_paper en las tablas) vienen dadas en el fichero de entrada y suponemos que son consistentes con el estado previo de la base de datos (es decir, como en la imagen se aportan datos del artículo 101, damos por hecho que en la base datos no existe ya un artículo con esa referencia. Por otro lado, como vemos que referencia al 532, suponemos que eso es consistente: o está más adelante en el fichero o se encuentra ya en la base de datos). No hacerlo así complicaría sustancialmente el ejercicio.

2.- El fichero ***DB.txt*** se ha generado extrayendo automáticamente la información de Google Scholar y puede presentar las faltas de homogeneidad propias de dicha aplicación web (p. ej. Un mismo autor con variantes en su denominación que le harán aparecer como más de uno). Al examinar los resultados esto puede inducir en algún caso a pensar que el programa no ha funcionado correctamente, por lo que conviene "sospechar" también de los datos de origen. En todo caso lo que se pretende es que la salida sea consistente con la entrada[1].

[1] Tomamos el camino sencillo por tratarse de un ejercicio, pero podría ganarse en "robustez" haciendo cosas como considerar que los separadores pueden no ser únicos o seguir un patrón determinado (p.ej. es igual <J. Gray> que <J. Gray>), no permitir que la capitalización de caracteres influya (p.ej. es igual <J. Gray> que <j. gray>), admitir líneas vacías, etc.