Universidad Pública de Navarra
Curso 2021/2022

Gestión de Sistemas de Información

Guión de la Práctica 02

Marilin Vega León, Dpto. de Automática y Computación

1. Contenidos del documento

Este documento recoge la Práctica 02 de la asignatura Gestión de Sistemas de Información correspondiente al curso 2021/2022.

2. Desarrollo de la práctica

Esta práctica ha de completarse en grupos, siguiendo la misma configuración que había en la Práctica 01. Si bien los ejercicios a realizar en esta práctica no son los necesarios para completar la siguiente, es conveniente tenerla terminada a tiempo.

No es estrictamente necesario tener conocimientos adicionales a los de la Práctica 01. Sin embargo, es conveniente que los alumnos dispongan de estos conocimientos antes de comenzar la práctica:

- Formatos de representación de información.
- Importación de librerías y clases ajenas en proyectos Java.
- Lectura, escritura y gestión de ficheros en Java.
- Técnicas de mapeo entre la vista lógica (entidades) y física (tablas) en el modelo relacional.

Los alumnos deben adquirir los siguientes conocimientos en el transcurso de la práctica:

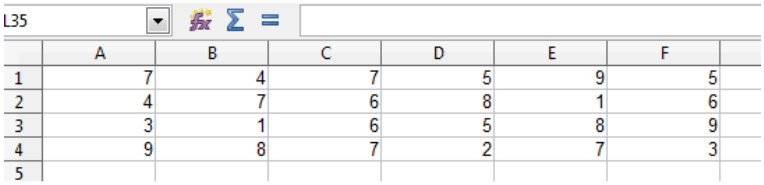
- Conocimientos teóricos:
  - Modelado de información bajo las restricciones de un formato físico (o lógico) determinado;
  - Representación de un modelo de objetos usando tablas.
- Conocimientos técnicos:
  - Aprendizaje y uso de librerías de *third parties*;
  - Almacenamiento automatizado de información en hojas de cálculo;
  - Integración de sistemas Java con herramientas ofimáticas. Generación y lectura de ficheros.

La asistencia al laboratorio es únicamente obligatoria el primero de los días asignados para esta práctica.

3. Fechas de interés

Las fechas de interés para el desarrollo de la práctica son las que siguen:

- 29 de Septiembre, 6 y 13 de octubre: Sesiones de laboratorio asignados para completar la prácticas;
- 19 de Octubre: Fecha límite de entrega (14h).



	A	B	C	D	E	F
1	7	4	7	5	9	5
2	4	7	6	8	1	6
3	3	1	6	5	8	9
4	9	8	7	2	7	3
5						

Figura 1: Detalle de la hoja de cálculo a generar en el Ejercicio 2.

4. Introducción a la práctica

El sistema de información creado en la Práctica 01 es realmente limitado, si bien cumple con las funciones más básicas que se esperan de él, como pueden ser almacenamiento o gestión de información. En cada una de las prácticas programadas para esta asignatura se va a desarrollar una ampliación del sistema. Esta práctica trata de la integración del sistema con herramientas ofimáticas, más concretamente con hojas de cálculo.

Muchos usuarios no están familiarizados con herramientas informáticas que, a ojos de estudiantes de ingeniería, pueden parecer básicas. Generalmente, los usuarios tienen problemas o inconveniencias en el uso de consolas de comandos, e incluso con ciertos tipos de interfaces gráficos. Por ello, muchos SIs se apoyan en herramientas con las que los usuarios finales tienen más afinidad. Las hojas de cálculo son, junto a las páginas/formularios web, la más común de todas. En esta práctica los alumnos deben aprender a manejar la librería jOpenDocument para la lectura/escritura de archivos ofimáticos Open Document Format.

5. Lista de ejercicios

Ejercicio 1. Descargue la librería jOpenDocument. Colóquela en una subcarpeta dentro del proyecto abierto en la Práctica 01, y proceda a incorporarlo dentro de las librerías en el proyecto de NetBeans. Cuando haga esto, asegúrese de que la ruta a la librería es local, no absoluta.

Ejercicio 2. Genere un paquete llamado `GSILabs.Misc`. Dentro de este paquete, cree una clase ejecutable llamada `SSTest01`. En el método `main` de dicha clase debe crear un *array* bidimensional de 4×6 números enteros. Esta matriz se almacenará en un fichero de nombre test01.ods, tal que la matriz ocupe las primeras 4 filas y 6 columnas de la primera de sus hojas. Use un programa ofimático para comprobar que el fichero generado tiene la información deseada.

Ejercicio 3. Dentro del paquete `GSILabs.Misc`, cree una clase ejecutable llamada `SSTest02`. En el método `main` de dicha clase debe crear un *array* bidimensional de 4×6 números enteros. Esta matriz se almacenará en un fichero de nombre test02.ods, dejando que las primeras 5 filas y 3 columnas estén vacías. Es decir, el elemento de la primera fila y columna (de la matriz) aparecerá en la cuarta fila, sexta columna (de la primera página de la hoja de cálculo). Además, el color de fondo de las celdas donde se almacena el *array* bidimensional estará determinado por el valor de los enteros. En caso de que los números sean mayores o iguales que 10, el fondo será azul, siendo rojo en caso contrario. Cualquier tono de azul y rojo será valido.

Ejercicio 4. Dentro del paquete `GSILabs.Misc`, cree una clase de nombre `SSTest03` cuyo método `main` sea capaz de leer los datos almacenados en el archivo `test02.ods` creado en el ejercicio anterior (el color

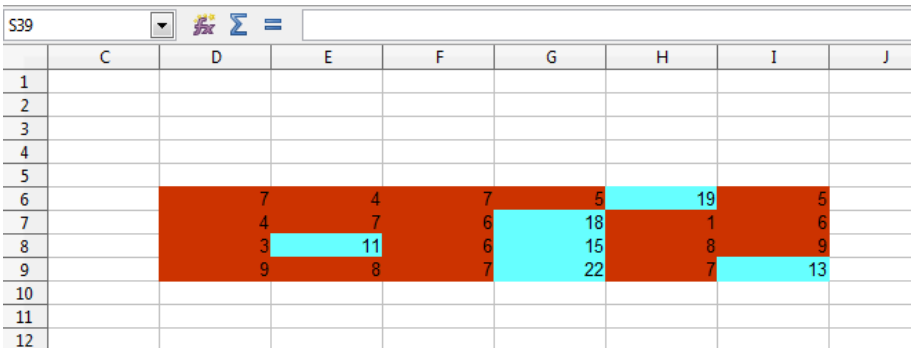


Figura 2: Detalle de la hoja de cálculo a generar en el Ejercicio 3.

es innecesario, pero los datos numéricos han de recuperarse). La posición donde se encuentra la matriz en la hoja de cálculo puede ser codificada de manera explícita (*hard-coded*) en la clase `SSTest03`.

Las hojas de cálculo se usan muy frecuentemente para almacenar información. Y, de hecho, soy muy prácticas cuando la información puede almacenarse en tablas individuales. De alguna manera, toda la información en una hoja de cálculo debe ser inventariable como tablas individuales, potencialmente generando un cubo tridimensional. Las aplicaciones informáticas gestionan información donde esta arquitectura no es muy adecuada, por razones que incluyen la cardinalidad de las relaciones entre instancias, la inclusión lógica de instancias dentro de otras, etc. En esos casos los sistemas de información deben complementar la información almacenada en la hoja de cálculo con elementos que no se soportan de manera nativa, como pueden ser las jerarquías o las referencias cruzadas, etc.

Ejercicio 5. Dentro del paquete `GSILabs.Misc`, cree una clase de nombre `SSTest04` cuyo método `main` almacene instancias de las clases que implementen `Local` en un fichero ODS. Para ello, la clase tiene que generar una instancia de la clase `BusinessSystem`, e introducir en ella una serie de eventos (incluyendo tanto bares como restaurantes y pubs), asociados a ciertas direcciones y propietarios.

Tras introducir datos en `BusinessSystem`, se recuperarán todas las noticias de ese artista para su almacenamiento en un fichero ODS. Para ello, se usarán tres hojas diferentes (una para cada tipo de evento), que se llamarán Bar, Restaurante y Pub. Las instancias se repartirán a razón de una fila por cada una, tendiendo en cuenta estas condiciones:

- En los casos en los que la información referente a a cada campo tiene cardinalidad variable (por ejemplo, varios tags), puede elegir entre considerar sólo uno de sus elementos o incluirlos todos, ocupando más de una celda.
- En los casos en que la información no sea almacenable en una celda (los propios tags, o quizá una Dirección), se evaluarán estrategias al resecto..

A la hora de comprobar los contenidos del fichero, una de las opciones es programar la lectura del propio fichero. Sin embargo, de cara a reducir la carga de trabajo, se recomienda la inspección ocular del mismo usando un programa ofimático. En las Figura 3 puede encontrar un detalle de la hoja Bar.

El ejercicio anterior supone un reto de transformación de información entre formatos. Se trata, en definitiva, de adecuar la información en la capa (vista) lógica (en términos de objetos) a su representación en la capa (vista) física (en términos de tablas). Si algún alumno tuvo problemas para realizar esta conversión, debería reflexionar acerca de cómo lo habría hecho en caso de utilizar una base de datos

	A	B	C	D	E	F	G	H
1	Rincon las palmeras	Travesia Cervantes 174	Mataro	Barcelona	Carlos Perez	Calamares		
2	Tasca andalucia	Calle Cervantes 161	Tarrasa	Barcelona	Carlos Gutierrez	Quesadillas	Tapas	Tapas
3	Rincon santi	Plaza Libertad 95	Algeciras	Cadiz	Juan Martin	Desayuno	Cerveza	Albondigas
4	Taberna la perla	Paseo Zaragoza 160	Albacete	Albacete	Marcos Perez	Camarones	Embutidos	Catalana
5	Tasca acuario	Plaza Espana 78	Sevilla	Sevilla	Pedro Garcia	Torreznos	Cerveza	
6	Rincon la pena	Plaza Quijote 9	Vitoria	alava	Josefina Garcia	Croquetas	Chorizo	Tapas
7	Taberna el puerto	Paseo Industria 151	Tarrasa	Barcelona	Carlos Rodriguez	Crudivoro	Olivas	Fritos

Figura 3: Ejemplo de hoja Bar para el Ejercicio 5.

relacional. Es decir, como habría hecho la estructura de clases a entidades, y finalmente a tablas relacionales. Dado que las tablas relacionales son (aquí no cabe la sorpresa) tablas, pueden almacenarse en la hoja de cálculo a razón de una tabla por hoja. Esto quiere decir que el almacenamiento en una hoja de cálculo tiene la misma potencia que el almacenamiento en una base de datos relacional[1]. Aquí, quizá, sí cabe la sorpresa.

Ejercicio 6. Incorpore a la clase `GSILabs.BSystem.BusinessSystem` un método `importaPubs(File f):int` que cargue un listado de bares desde un fichero ods. Para ello, `f` debe apuntar a un fichero existente. Dicho fichero debe tener una única página, tal que el nombre del bar al que se refiere se almacene en la primera columna. El resto de información se almacenará a partir de la segunda columna, incluyendo su dirección. Los bares deben comenzar en la primera fila de la hoja, y se considerará que no hay bares más allá de la primera fila cuya primera columna esté vacía. Puede descargar un ejemplo de fichero desde MiAulario, con el nombre P02Ej05.ods. El valor que devuelve el método es el número de bares incorporados con éxito al sistema (nótese que el fichero podría no cumplir con las normas de la política de negocio).

Nota: En caso de que la lógica de negocio implementada en la Práctica 01 no sea acorde a la información del fichero P02Ej05.ods, consúltelo con el docente.

6. Metodología de entrega

Cada grupo debe realizar una única entrega a través de la Sección Tareas de MiAulario. Cada grupo de alumnos debe entregar en una carpeta comprimida el proyecto, bien sea de Eclipse o NetBeans, así como cualquier documento auxiliar o de información relevante que los alumnos crean necesario incluir. Dicha carpeta se llamará `GrXX.zip` o `GrXX.7z`, donde `XX` debe reemplazarse por el número del grupo.

El proyecto debe ser totalmente autocontenido, no haciendo uso de rutas absolutas o dependencias de archivos no incluidos. Cualquier violación de estas normas tendrá una penalización de $-20\,\%$ en la nota del grupo.

La nota de cada grupo se asignará, sobre un máximo de 12 puntos, en función de los siguientes criterios:

e1-2 La clase `SSTest01.java` está completa y bien documentada (4 punto).

e3 La clase `SSTest02.java` está completa y bien documentada (3 punto).

e4 La clase `SSTest03.java` está completa y bien documentada (3 puntos).

e5 La clase `SSTest04.java` está completa y bien documentada (1 puntos).

e6 El método `importaPubs` de la clase `BusinessSystem` es funcional y está bien documentado (1 puntos).

1. No hablamos de las propiedades de la gestión de información.

Cada uno de los criterios podrá evaluarse como *fallido* (0 %), *incompleto* (25 %, 50 %, 75 %) o *completo* (100 %).

Adicionalmente, los alumnos podrán subir la nota final si alcanzan estos objetivos:

e2.1 Añada un método a la clase `BusinessSystem` para que, al igual que `importaPubs` hace con los tickets, permita importar la información de bares (`Bar`). Para ello, debe encontrar algún tipo de convenio para gestionar los tipos multivaluados (p.e. tags) y las posibles referencias. Este convenio debe explicarse en la documentación del método.(+1 puntos)

e2.2 Descargue el de MiAulario el *interface* `ODSPersistente` y haga que la clase `BusinessSystem` lo implemente (+1 puntos).

Cada día (o fracción de día) de retraso en la entrega, tomando como referencia la fecha límite, supondrá una penalización del $-25\,\%$ en la nota del grupo.

7. Control de versiones

Este documento no ha sufrido modificaciones desde su publicación (XX/XX/2021).