Universidad Carlos III

Arquitectura de Datos
Curso 2024-25

Práctica 1.2: Proceso de migración

Grupo 05

Ingeniería Informática, 4º Curso

Luis Otero Gómez (NIA: 100472104, 100472104@alumnos.uc3m.es)

Paula Morales García (NIA: 100472240, 100472240@alumnos.uc3m.es)

Alejandro Calderón Hernández (NIA:100472087, 100472087@alumnos.uc3m.es)

Profesora: Lourdes Moreno López

Grupo: 81

Índice

Introducción

El proceso de migración de información es una fase fundamental para poder manipular los datos exportados correctamente en las bases de datos. En esta práctica se utiliza la base de datos NoSQL orientada a documentos llamada MongoDB. Esta base permite almacenar y manipular datos no estructurados. En nuestro caso, utilizamos la información recogida en la carpeta Datasets Práctica 1.2.

Este directorio contiene varios archivos.csv con la información necesaria para completar el diagrama de clases en MongoDB. A continuación, se presenta una lista de cada archivo dentro de la carpeta, junto con una descripción de cómo se asignan con las clases.

- **AreasSucio.csv**:
  - Contiene información relacionada con las áreas dónde se agrupan un conjunto de juegos. Cabe mencionar que contiene información detallada sobre su ubicación, clasificación, estado operativo y un identificador para distinguir de forma unívoca las áreas.
  - Este csv es aplicado para crear la clase AreaRecreativa.

- **EncuestasSucio.csv**:
  - Contiene información relacionada con las encuestas de satisfacción realizadas por los usuarios registrados sobre las áreas recreativas y sus juegos.
  - Este csv es aplicado para crear la clase EncuestaSatisfacción.

- **estaciones_meteo_CodigoPostal.csv**:
  - Contiene información relacionada para poder asociar los datos almacenados en el archivo meteo24.csv con los datos en el archivo AreasSucio.csv a partir del código postal.

- **IncidenciasUsuariosSucio.csv**:
  - Contiene información relacionada con las incidencias reportadas por los usuarios sobre los juegos en las áreas recreativas, con datos como el tipo de incidencia, estado o fecha de reporte .
  - Este csv es aplicado para crear la clase Incidencia.

- **IncidentesSeguridadSucio.csv**:
  - Contiene información relacionada con los incidentes de seguridad que han ocurrido en las áreas recreativas.
  - Este csv es aplicado para crear la clase IncidenteSeguridad.

- **JuegoSucio.csv**:
  - Contiene información relacionada con los juegos disponibles entre todas las áreas, como por ejemplo, su modelo, estado operativo y un identificador para diferenciar de forma unívoca cada juego.
  - Este csv es aplicado para crear la clase Juego.

- **MantenimientoSucio.csv**:
    - Contiene información sobre las intervenciones realizadas sobre los juegos en las áreas recreativas.
    - Este csv es aplicado para crear la clase Mantenimiento.

- **meteo24.csv**:
    - Contiene información sobre los registros climáticos con información sobre las temperaturas, precipitaciones y otros eventos meteorológicos.
    - Este csv es aplicado para crear la clase RegistroClima.

- **UsuariosSucio.csv**:
    - Contiene información personal de los usuarios que han reportado incidencias o proporcionado comentarios sobre los juegos.
    - Este csv es aplicado para crear la clase Usuario.

Descripción de los agregados

En este apartado, se explica detalladamente las decisiones tomadas para conseguir las relaciones entre las clases según los agregados indicados en el enunciado. Las asociaciones se deben realizar utilizando atributos que sean identificativos y útiles para poder relacionar los elementos. A continuación, se justifica la elección de los atributos para relacionar las clases.

Agregado AreaRecreativa_Clima:

El objetivo de este agregado es mostrar un resumen sobre las áreas. Se deben incluir los juegos de cada área, sus datos de clima, una lista con sus encuestas de satisfacción y un resumen sobre sus incidentes de seguridad, incluyendo información sobre el tipo de incidente, su gravedad y la fecha del reporte. Además, se debe incluir una nueva variable que calcule su estado en función del número de juegos en mantenimiento y el número de incidentes de seguridad.

Para este agregado se necesita establecer relaciones entre las clases Juego, IncidenteSeguridad, RegistroClima, EncuestaSatisfacción con la clase AreaRecreativa, considerando esta última como la raíz. Seguidamente se detalla cómo se ha conseguido cada asociación:

- **AreaRecreativa - Juego**: se utiliza el atributo NDP en AreaRecreativa y NDP en Juego. Esta relación se hace por referencia, almacenando solo ID de cada juego.

- **AreaRecreativa - IncidenteSeguridad**: se utiliza el atributo ID en AreaRecreativa y AREA_RECREATIVA_ID en IncidenteSeguridad. Esta relación se hace por referencia con resumen, almacenando FECHA_REPORTE, TIPO_INCIDENTE y GRAVEDAD, además del ID.

- **AreaRecreativa - RegistroClima**: se utiliza el atributo COD_POSTAL en AreaRecreativa y en RegistroClima. Esta relación se hace por referencia dejando el ID de cada registro.

- **AreaRecreativa - EncuestaSatisfacción**: se utiliza el atributo ID en AreaRecreativa y AREA_RECREATIVA_ID en EncuestaSatisfacción. Esta relación se hace por referencia, almacenando el ID de cada encuesta.

Para el nuevo atributo ESTADO_GLOBAL_AREA, primero realiza un cálculo en función de los comentarios de los usuarios en las encuestas. Para ello se asignan los siguientes valores a cada tipo de comentario y se suma el valor total:

- Excelente ⇒ 5
- Muy bueno ⇒ 4
- Aceptable ⇒ 3
- Regular ⇒ 2
- Deficiente ⇒ 1

Posteriormente, se calcula el número de juegos cuyo estado es "EN REPARACIÓN" y el número total de incidentes. Finalmente, se realiza una suma de todos los datos calculados anteriormente para determinar el estado global de cada área.

Por último, se ha incluido el atributo CANTIDAD_JUEGOS_POR_TIPO como indica el diagrama de clases. Este atributo contiene información sobre la cantidad de juegos según el tipo en cada área recreativa.

Agregado Juego

El objetivo de este agregado es obtener una un resumen con la información esencial de las incidencias en los juegos, como por ejemplo, el tipo de incidencia, la fecha de reporte y su estado actual. Además, se incluyen dos nuevas variables. En particular, ULTIMA_FECHA_MANTENIMIENTO, que incluye la fecha del último mantenimiento realizado sobre cada juego, y TIEMPO_RESOLUCION, que representa un intervalo entre el reporte de una incidencia y su resolución para poder evaluar la calidad de mantenimiento de los juegos.

En este agregado es necesario establecer relaciones entre las clases Mantenimiento e Incidencia con las clase raíz Juego. A continuación, se indica cómo se han realizado las asociaciones:

- **Juego - Mantenimiento**: se utiliza el atributo ID en Juego y JUEGO_ID en Mantenimiento. Esta relación se hace por referencia, almacenando el ID de cada mantenimiento.
- **Juego - Incidencia**: se utiliza el atributo ID en Mantenimiento y MANTENIMIENTO_ID en Incidencia, porque no existe relación directa entre Juego e Incidencia. Esta relación se hace por referencia con resumen, guardando TIPO_INCIDENCIA, FECHA_REPORTE y ESTADO, además del ID de cada incidencia.

Respecto a ULTIMA_FECHA_MANTENIMIENTO, se ha realizado un bucle en el que se comprueban las fechas de todos los mantenimientos realizados a cada juego y se toma el valor mayor. Así, se consigue la fecha del último mantenimiento realizado en cada juego.

En cuanto a TIEMPO_RESOLUCION, tras incluir las incidencias relacionadas con el juego, se ordenan de menor a mayor en función del valor de su fecha de reporte. Esto permite extraer la primera incidencia y realizar una resta entre la fecha del último mantenimiento realizado y la primera fecha de reporte.

Por último, se han incluido los atributos DESGASTE_ACUMULADO e INDICADOR_EXPOSICION que indican el desgaste un juego en función de su tiempo de uso y el número de mantenimientos realizados y la exposición que tienen los juegos en las áreas recreativas (bajo, medio y alto).

Agregado Incidencia

El objetivo de este agregado es mostrar un resumen sobre información de los usuarios. En concreto, la información de contacto como el email o el teléfono. Además, se añade un nuevo atributo denominado NIVEL_ESCALAMIENTO para categorizar la urgencia de una incidencia.

En este agregado se asocian las clases Usuario e Incidencia, considerando esta última con la raíz. Para ello, se utiliza el atributo NIF de clase Usuario y USUARIOS_ID de la clase Incidencia. Esta única relación se hace embebida, almacenando todos los atributos de la tabla usuarios.

Para crear el atributo NIVEL_ESCALAMIENTO, se categorizan los tipos de incidencia según sus valores. A continuación, se muestra el valor y su equivalente para cada tipo de incidencia:

- Excelente ⇒ 5
- Desgaste ⇒ 4
- Vandalismo ⇒ 3
- Rotura ⇒ 2
- Mal funcionamiento ⇒ 1

Anomalías encontradas y resolución

Antes de la migración

Se han pasado todos los atributos de tipo texto a mayúsculas. Los atributos de texto se han estandarizado de forma que no tienen caracteres especiales como tildes o letras 'ñ' que pueden dar problemas a futuro. Se creó una función que permite cambiar el nombre de las cabeceras, lo cual permite poner todas las cabeceras también en mayúsculas.

Para las tablas de JuegosSucios y AreasSucio se ha establecido una relación, ya que comparten los campos NUM_VIA, TIPO_VIA y NOM_VIA. Además, se ha configurado de forma que los valores faltantes de estos atributos se obtengan del campo DIRECCION_AUX (filtrando el contenido mediante la segmentación de palabras separadas por espacios (split)), y en su defecto de los atributos obtenidos de la relación Juegos-Areas. También las columnas con códigos donde los valores deben ser enteros, se han modificado para que así sean (por ejemplo COD_DISTRITO, COD_POSTAL).

Todos los atributos de las distintas tablas cuyo contenido debe ser una fecha se han estandarizado al formato ISODATE 'YYYY-MM-DDThh:mm:ss', y en la tabla final de meteo se han tenido que eliminar fechas inexistentes (30 de febrero por ejemplo). Todos los valores ausentes en las tablas originales han sido reemplazados por una cadena con el formato '(ID)-(NOMBRE_DEL_ATRIBUTO)-ausente'. Sin embargo, el ID en la tabla de Mantenimientos se ha tenido que corregir porque no estaba en el orden correcto. Se eliminaron también todos los registros con ID repetido para deshacerse de los duplicados.

Por otra parte, el csv con los datos del clima tenía un separador diferente (punto y coma en lugar de coma) por lo que se ha cambiado por el carácter ",". Además, se adaptó este mismo csv para que tuviese una estructura más sencilla a la hora de trabajar con los datos, creando una nueva tabla con los atributos ID (nuevo atributo creado para futuros usos), FECHA, CODIGO, VIENTO, TEMPERATURA y PRECIPITACION.

Por último, en la tabla Usuarios también se estandarizó la columna con el número de teléfono, dejándolos todos de la forma '+34 XXX XXX XXX', y se ignoró una columna email porque estaba vacía y ya existía otra correcta con el mismo nombre.

Después de la migración

Tras importar los archivos .csv a MongoDB, los datos han tenido que pasar por otro proceso de preprocesado, debido a que no existen las variables de tipo 'array' ni 'date'. En aquellas colecciones que contenían campos con fechas o arrays, se ha realizado una conversión a sus tipos correspondientes. La conversión a tipo 'array' elimina los corchetes y comillas simples y divide los elementos usando la coma y el espacio. Esto se realiza sobre los atributos MANTENIMIENTO_ID y USUARIO_ID en la clase Incidencia.

Posteriormente, se ha añadido el campo COD_POSTAL para cada registro de la colección RegistroClima, y así poder establecer una relación entre las colecciones AreaRecreativa y Juego. Esto se ha realizado relacionando los códigos incluidos en la colección Estaciones y así se obtiene el código postal correspondiente.

Finalmente, se han creado las validaciones para cada campo para confirmar que los datos se han almacenado en el formato y en la forma correcta para poder realizar los agregados solicitados.

Entregables

En esta sección se documentan los archivos incluidos para la entrega de la práctica 1. A continuación, se indican los archivos con una breve descripción.

- **Datasets_Practica_1.2.zip**: archivo.zip con todos los archivos.csv sucios.
- **migrations.txt**: archivo.txt que contiene todos los comando necesarios para:
    - Preprocesar los datos en Mongodb una vez son insertados.
    - Crear la validación de esquemas.
    - Crear cada uno de los agregados con sus respectivas colecciones.
- **cleanData.py**: archivo con código en el lenguaje python que realiza la limpieza de los datos en los archivos.csv.
- **ejecucion.sh**: archivo bash que contiene los comandos necesarios para la ejecución completa:
    - Descomprimir Datasets_Practica_1.2.zip.
    - Ejecutar la limpieza de los csv con el archivo cleanData.py.
    - Realizar las importaciones en Mongodb.
    - Ejecutar migrations.txt.
- **Documentación_migración.pdf**: memoria dedicada a la parte de migración para la práctica 1.
- **Documentación_clúster.pdf**: memoria dedicada a la parte de fragmentación para la práctica 1.
- **Readme.txt**: archivo.txt con una explicación sobre qué pasos realizar para ejecutar los archivos.