INFORME TPE

Diseño y Procesamiento de Documentos XML - 72.34

Ahumada, Manuel, maahumada@itba.edu.ar
Hillar, Conrado, chillar@itba.edu.ar
Lategana, Juan Ignacio, jlategana@itba.edu.ar
Priotto, Nicolás, npriotto@itba.edu.ar

GRUPO 13 - 15 de noviembre de 2023

INTRODUCCIÓN

Se realizó un programa para la obtención y procesamiento de datos de una API de deportes “SportRadar”. A partir de dos parámetros, un nombre y un año, se selecciona una temporada de Rugby a procesar. El mismo permite de forma automática descargar los datos vía protocolo HTTP del servicio gratuito de SportRadar y mediante una serie de etapas de procesamiento generar un archivo para la visualización del usuario de los mismos. Se elaboraron documentos xQuery cuyos funcionamientos consisten en filtrar la información recibida a través de la API. Luego, con la información ya procesada, se diseñó un documento XSLT que se encargó de convertir los datos en un documento MarkDown. Este último documento presenta una página con el nombre de la temporada, información sobre ella, y diversas tablas que contienen los equipos y jugadores de forma ordenada.

DESARROLLO

El programa consiste de diversas tecnologías como XML, xQuery y XSLT. Para poder unificar las mismas en una sola ejecución, se creó un script de bash que une cada una de las etapas del proyecto. A continuación se desarrolla cada una de las etapas necesarias para el procesamiento de la información:

1. OBTENCIÓN DEL ID DE LA TEMPORADA

Utilizando el comando curl se descarga un documento seasons.xml en el cual se encuentran todas las temporadas disponibles en la API junto a datos básicos como su nombre o el año. Mediante un programa en xQuery se busca en este documento una temporada que coincida con el nombre y año recibidos por parámetros, y se da como respuesta su id. La misma es necesaria para pedir la información detallada de la temporada.

2. OBTENCIÓN DE DATOS DE LA TEMPORADA

Mediante dos nuevos pedidos GET a la API utilizando curl, se descargan los archivos season_info.xml y season_lineups.xml. Los mismos contienen información como los competidores de cada temporada y los jugadores presentes en cada partido.

3. COMPILADO DE DATOS Y GENERACIÓN DE XML AUXILIAR

En base a los dos archivos de datos específicos de la temporada obtenidos en la etapa anterior, se busca generar un nuevo archivo XML que contenga la información de forma ordenada para luego poder transformarla en el archivo de salida al usuario. Mediante un

programa xQuery, se procesan ambos archivos XML en paralelo para la creación de el archivo season_data.xml. El mismo posee una estructura acorde al xml-schema proporcionado por la cátedra. En caso de encontrarse errores en alguna etapa anterior, como puede ser que el ID no exista o que algún parámetro sea invalido, el xQuery los intercepta y devuelve una estructura específica de error como la encontrada en el sistema utilizado. De esta forma se puede informar al usuario ante cualquier problema en la ejecución del programa.

4. GENERACIÓN DE ARCHIVO MARKDOWN DE SALIDA

Utilizando el archivo XML auxiliar creado anteriormente, mediante un programa XSLT se crea un nuevo archivo MarkDown, respetando el formato pedido en la consigna.

ERRORES

Los errores son procesados en el programa xQuery anteriormente mencionado (etapa 3). Los mismos fueron encontrados en base a la utilización de la API con parámetros variantes que permitan cubrir el máximo de casos de uso posibles, y que el usuario obtenga la información más precisa posible sobre los problemas encontrados en ejecución. Los errores detectados por el programa xQuery son los siguientes:

- Parámetro “año” es un string vacío
- Parámetro “año” no es un número
- Parámetro “nombre” es un string vacío
- No existe una temporada con el ID elegido en la etapa 1
- No existe una temporada con los parámetros ingresados
- La API key no es válida
- No se encuentra un archivo de entrada (season_info.xml / season_lineups.xml)

DIFICULTADES

La mayor dificultad que se enfrentó al realizar este trabajo fue la manipulación de datos en el archivo “extract_season_data.xq”. Dentro de este documento, el primer problema que se encontró fue la manera en la que se debía extraer los datos de los jugadores, ya que estos aparecían reiteradas veces en el mismo documento. Para ello, mediante prueba y error, se llegó a la conclusión que la mejor manera de abordarlo era buscando los equipos de la temporada, y luego iterando sobre ellos buscando a todos los jugadores que pertenezcan a dicho equipo usando la función distinct-values, para que estos no se repitan. En segundo

lugar, el abordaje de los errores fue desafiante. Para poder verificar que el programa identifica los errores correctamente, primero se necesitó interpretarlos. Esto fue difícil porque era la primera vez que los integrantes del grupo utilizabamos una API con estos propósitos.

Del mismo modo, se encontraron limitaciones en el lenguaje xQuery. Una de ellas fue que la función distinct-values, mencionada previamente, devuelve un 'sequence of unique atomic values', el cual requería procesarlo nuevamente para poder obtener los resultados deseados. Este problema disminuye la eficiencia del procesamiento ya que hay muchas comparaciones de más. Esto podría haberse solucionado si xQuery incluyera alguna función que elimine repetidos o alguna estructura más compleja como un Set (Java).

Para poder ordenar los jugadores por la cantidad de partidos en los que aparecían, se tuvo que ordenar de manera descendente comparando por string, dado que la función 'sort' de xQuery funciona de dicha manera. Se consideró correcto ya que en el enunciado provisto por la cátedra la tabla estaba ordenada de la misma manera.

Finalmente, la API utilizada no permite realizar operaciones seguidas, requiere que estas estén espaciadas por un segundo. Para resolver este problema se utilizó una función sleep entre cada llamada a la API en el script de bash.

ROLES

Con respecto a los roles adoptados por cada integrante, se decidió un acercamiento distinto al propuesto por la cátedra. Para los documentos xQuery y XSLT, nos conectamos todos de forma sincrónica a través de la plataforma "Discord", utilizando la extensión de VSCode "LiveShare", la cual permite que varias personas puedan editar un mismo documento en simultáneo (como un Google Docs). A su vez se realizó un repositorio en GitHub para poder gestionar las versiones del código del proyecto.

CONCLUSIÓN

En resumen, creamos un programa que utiliza la API de SportRadar para obtener y procesar datos de temporadas de Rugby. Enfrentamos desafíos, especialmente al manipular datos en "extract_season_data.xq" y al manejar errores, pero implementamos un sistema efectivo en xQuery. Colaboramos usando Discord y LiveShare, gestionando versiones en GitHub.

Aunque hubo limitaciones en xQuery y restricciones en la API, superamos obstáculos y entregamos un programa funcional. Este proyecto no solo cumplió objetivos, sino que también proporcionó experiencia práctica y habilidades colaborativas al equipo.