UNIVERSIDAD DE INGENIERÍA Y TECNOLOGÍA

Curso:

Base de Datos II

Informe Laboratorio 2

Integrantes:

Melanie Alexia Cortez Rojas

Sofia Valentina Ku Paredes

Profesor:

Heider Sanchez Enrique

Lima, Perú

2025-1

1. Análisis comparativo entre las funciones de Sequential File y AVL File

a. Análisis Comparativo: Inserción

En la en la imágen 1 y 2, se puede observar una diferencia significativa en los tiempos de ejecución de la función de inserción del *sequential file* y *AVL file*. A primera vista, en ambos tipos de archivos, la inserción de datos es lineal, es decir que el tiempo aumenta conforme la cantidad de datos aumenta. Sin embargo, el *sequential file* con archivo auxiliar presenta tiempos significativamente menores (entre 0.01 y 0.2 segundos). En contraste, el *AVL file* presenta tiempos considerablemente más altos que el *sequential file* (entre 2.5 y 149.8 segundos), incluso en cantidades pequeñas de datos. Esto podría estar relacionado con las operaciones de balanceo necesarias en un AVL para preservar su estructura. Mientras que, en el sequential file, se accede, en primera instancia, al registro más cercano y luego solo se itera punteros hasta encontrar la posición indicada del nuevo registro. Incluso aunque el sequential file requiere de una reestructuración del archivo una vez que el archivo auxiliar llegue a su capacidad, parece evidencia que este tiene un costo mucho menor que el balanceo en el AVL.

Esta diferencia en el tiempo de ejecución es evidente en escenarios con un alto flujo de datos, en los cuales el *sequential file* resulta ser más eficiente. Por lo tanto, para el caso de inserción de datos, se recomienda utilizar un *sequential file* cuando se deba insertar una cantidad significativa de información en el archivo.

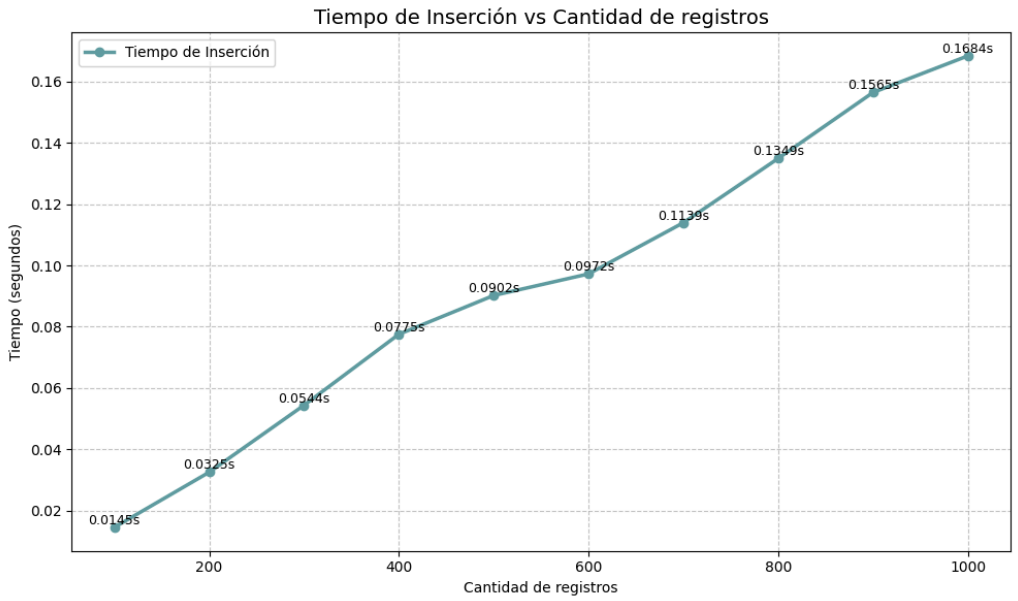

Imagen 1: Gráfica de tiempos de inserción por cantidad de registros en Sequential File

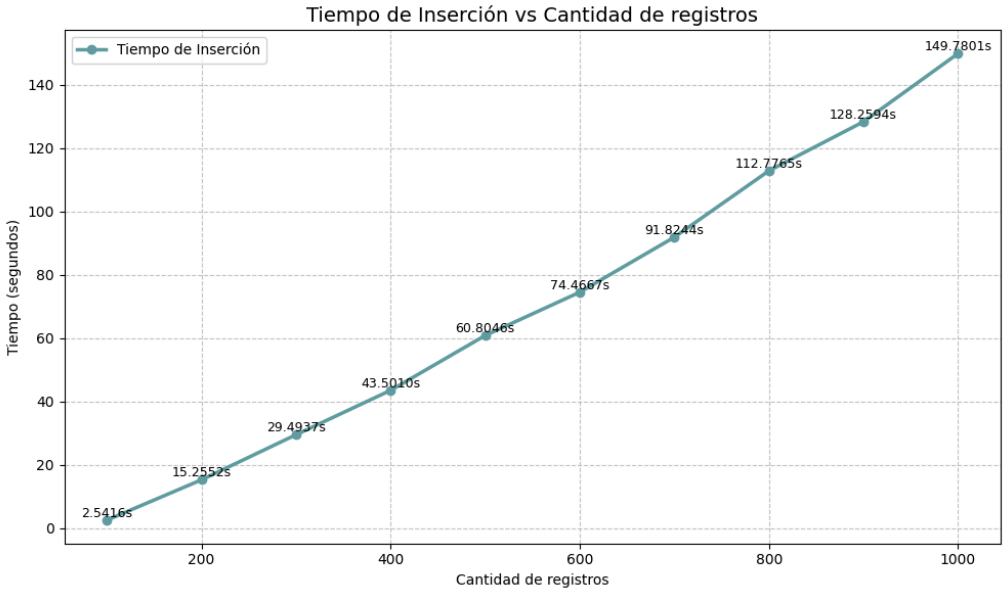

Imagen 2: Gráfica de tiempos de inserción por cantidad de registros en AVL File

b. Análisis Comparativo: Búsqueda

En la en la imágen 3 y 4, se observa una clara diferencia en el rendimiento de las operaciones de búsqueda entre ambos tipos de archivos. El *sequential file* muestra tiempos notablemente más bajos (0.0009 a 0.0021 segundos) que el *AVL file* (0.0050 a 0.0140 segundos), a pesar de que ambos tienen complejidad teórica O(log n). Este comportamiento se puede deber a que se utiliza búsqueda binaria por el cual se comienza la búsqueda desde la posición más cercana al ID buscado, lo cual podría llegar a optimizar el tiempo de búsqueda en este tipo de archivos. En cambio, en el *AVL file*, se puede llegar a recorrer cada nivel desde la raíz del archivo para poder encontrar un registro.

Tal como se observó en la inserción, la diferencia en el tiempo de ejecución en ambos tipos de archivos es evidente en escenarios con alta cantidad de datos, en los cuales el *sequential file* resultaría ser más eficiente. Por lo tanto, para el caso de búsqueda de datos específicos, se recomienda utilizar un *sequential file* cuando se deba insertar una cantidad significativa de información en el archivo.

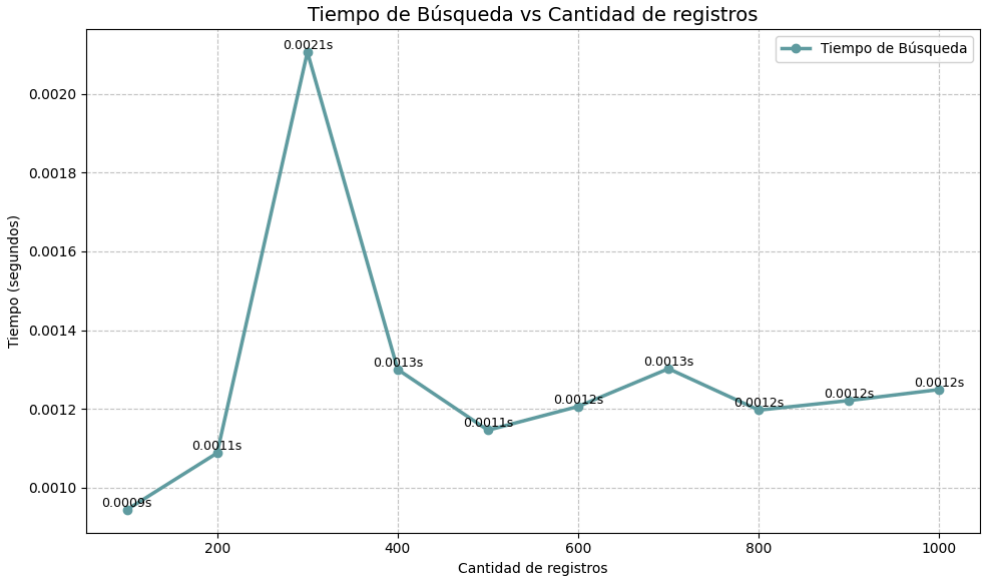

Imagen 3: Gráfica de tiempos de búsqueda por cantidad de registros en Sequential File

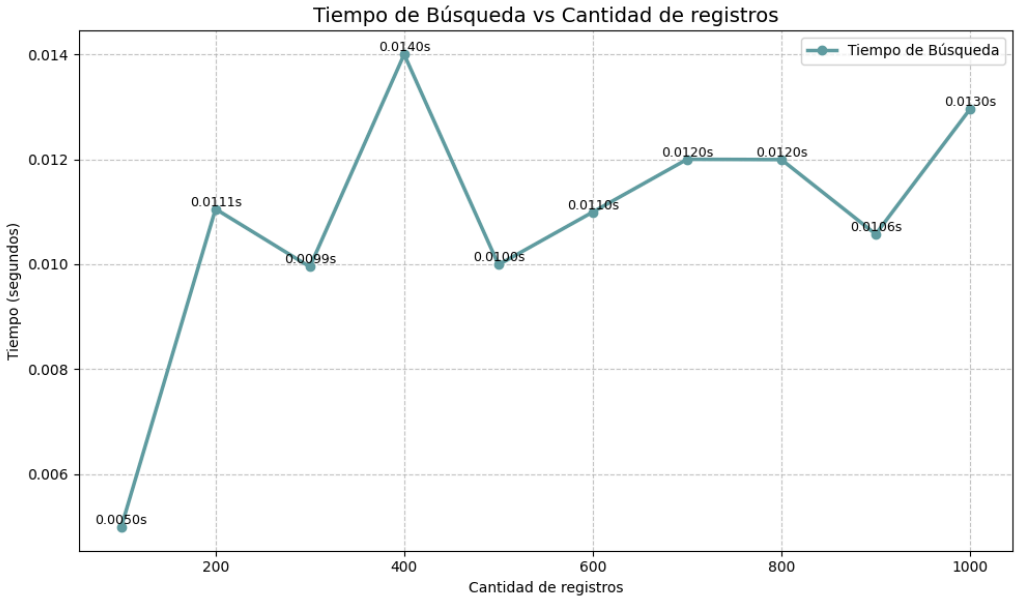

Imagen 4: Gráfica de tiempos de búsqueda por cantidad de registros en AVL File

c. Análisis Comparativo: Búsqueda por rango

Los resultados obtenidos para operaciones de búsqueda por rango muestran una tendencia lineal en el tiempo de ejecución conforme a la cantidad de datos en ambos tipos de archivos. El sequential file mantiene tiempos de ejecución consistentemente bajos (de 0.0012 a 0.0111 segundos), mientras que el AVL file presenta valores significativamente mayores (0.0050 a 0.0540). Esto puede ser debido a que el *sequential file* tiene los datos casi ordenados en el archivo principal, por lo que las búsquedas por rango pueden aprovechar el acceso básicamente lineal. En cambio, los AVL file, incluso

aunque su estructura balanceada garantiza búsquedas O(log n) para elementos individuales, las consultas por rango requieren recorrer a profundidad, además de recorrer múltiples nodos en diferentes ramas del árbol y realizar comparaciones adicionales para identificar todos los elementos dentro del rango.

Por ende, así como se pudo observar en las funciones anteriores, el sequential file es la opción óptima debido a su acceso predecible y bajo costo computacional a comparación de *AVL file*.

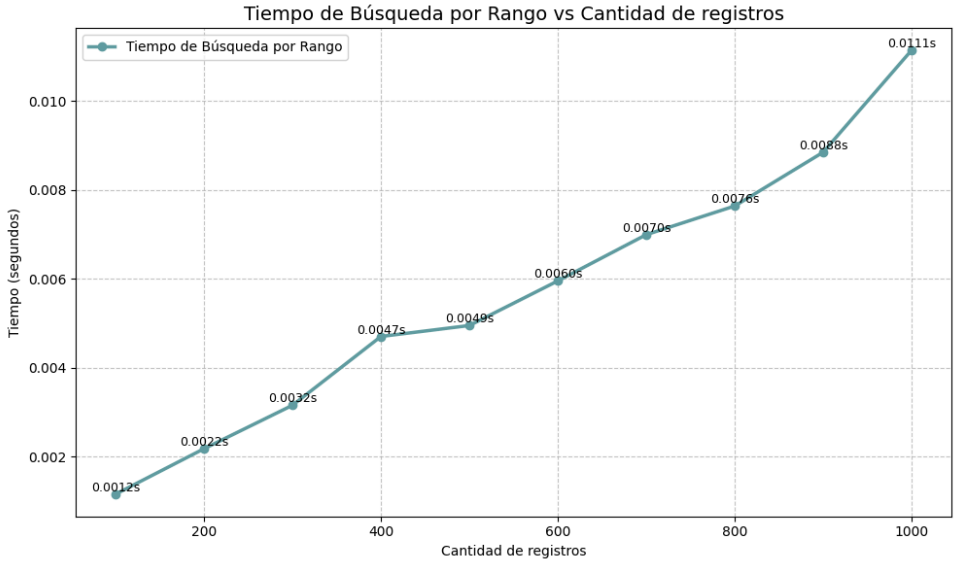

Imagen 5: Gráfica de tiempos de búsqueda por rango por cantidad de registros en Sequential File

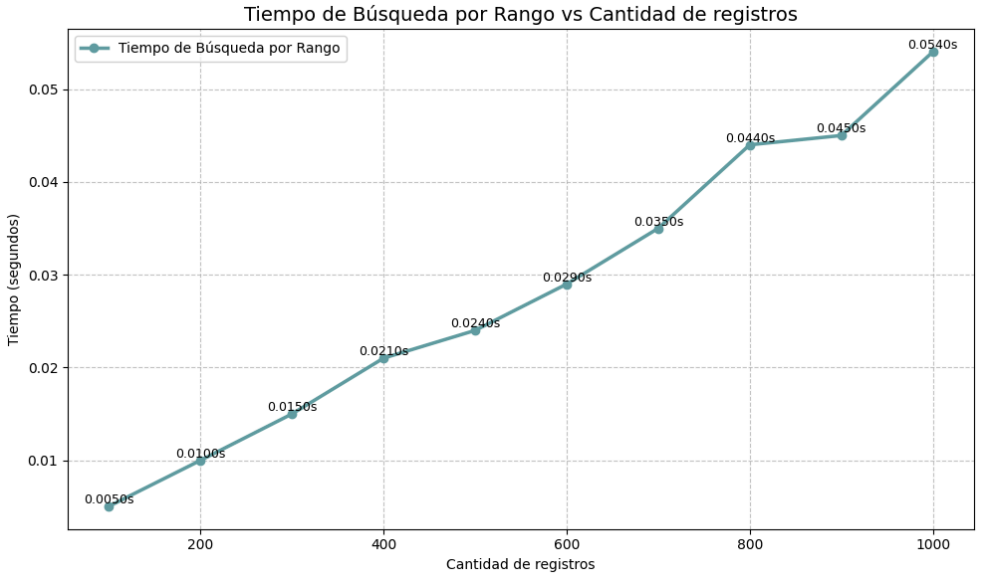

Imagen 6: Gráfica de tiempos de búsqueda por rango por cantidad de registros en AVL File

d. Análisis Comparativo: Eliminación

Los resultados obtenidos para las operaciones de eliminación muestran una tendencia casi lineal en el rendimiento en cuestión a la cantidad de datos entre ambos tipos de archivos. El sequential file muestra tiempos de ejecución notablemente menores (de 0.0049 a 0.0436 segundos) en comparación al AVL file (de 0.0163 a 1.7740 segundos), el cual muestra además una alta variabilidad en sus tiempos de ejecución.

Este comportamiento es debido al manejo de eliminación. En el caso de sequential file, la eliminación de de un registro requiere solo encontrar el registro a eliminar y el previo para poder actualizar el puntero del previo al siguiente registro. En cambio, para el AVL file, también se debe buscar el registro, el cual hemos observado previamente que es más costoso que la búsqueda del sequential file. Así mismo, al eliminar un nodo o registro del árbol AVL se debe hacer múltiples operaciones de balanceo para mantener su estructura, lo cual hemos concluido previamente que aumenta el tiempo de ejecución de la función.

Tal como se observó en las funciones anteriores, la diferencia en el tiempo de ejecución entre los dos tipos de archivos, el *sequential file*, demuestra ser claramente superior que el *AVL file*. El comportamiento predecible y estable del *sequential file* lo hace ideal para cargas de trabajo con operaciones de eliminación frecuentes.

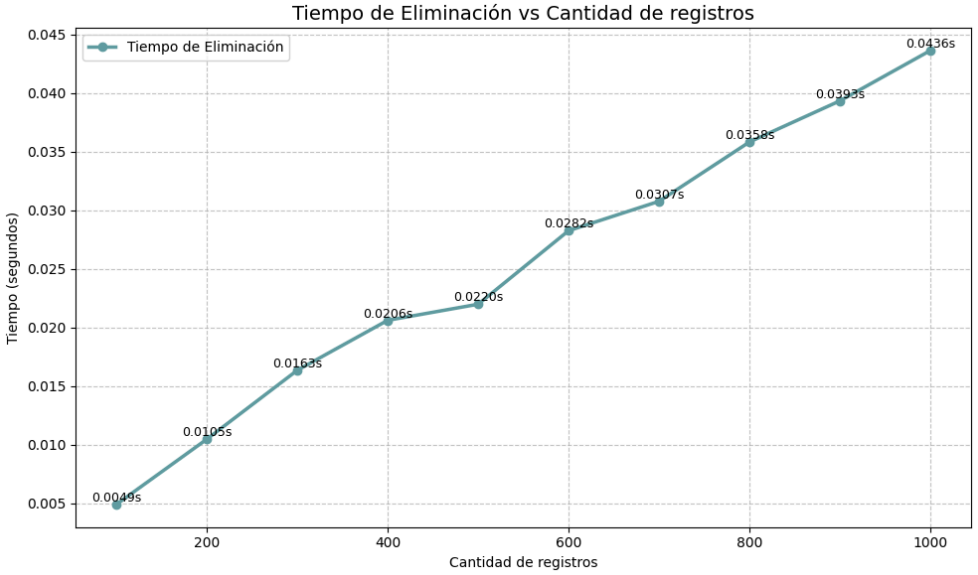

Imagen 7: Gráfica de tiempos de eliminación por cantidad de registros en Sequential File

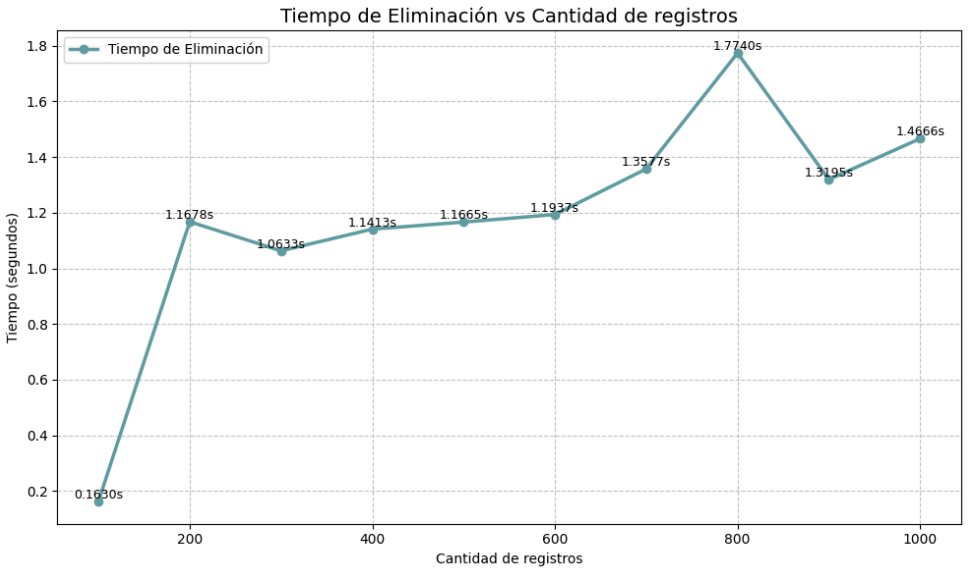

Imagen 8: Gráfica de tiempos de eliminación por cantidad de registros en AVL File

2. Conclusión

En base a las discusiones y observaciones que se hicieron en base a las gráficas de las operaciones básicas del *sequential file* y *AVL file*, podemos concluir que el utilizar un *sequential file* daría un mejor rendimiento computacional que un *AVL file*.

El *sequential file* demuestra ser más eficiente tanto en escenarios con altos como bajos volúmenes de operaciones, destacando en inserción, búsqueda, búsqueda por rango y eliminación. Esto se debe principalmente a su acceso secuencial a los registros, debido a que los datos se almacenan de forma ordenada en el archivo principal, y al bajo costo computacional de su mantenimiento. Incluso considerando la reestructuración cuando el archivo auxiliar alcanza su capacidad, el costo de dicha operación es considerablemente menor en comparación con las múltiples rotaciones y rebalanceos que exige la estructura de un AVL file.

El análisis comparativo entre ambos tipos de archivo demuestra que la elección de una estructura de organización adecuada influye directamente en el rendimiento de las operaciones básicas. Esta elección afecta la escalabilidad del almacenamiento. En ese sentido, elegir la estructura adecuada es una parte fundamental para una buena organización de archivos y tener un sistema ágil a medida que crece el volumen de datos.

3. Repositorio

https://github.com/sofkp/lab2_bd2