Tablas y Conclusiones de los ejercicios 3 y 4

Ejercicio 3

**.3d**

**a.**

|  |  |  |
| --- | --- | --- |
| Acción/Implementación | HashSet  (tiempo en nanosegundos) | ArrayList  (tiempo en nanosegundos) |
| **.3a** Implementación con mapeo de una tabla por jerarquía | 231401604 | 241364932 |
| **.3a** Implementación con mapeo de una tabla por clase | 263875424 | 319187412 |
| **.3b** Implementación con mapeo de una tabla por jerarquía | 96368831 | 111051434 |
| **.3b** Implementación con mapeo de una tabla por clase | 99478674 | 120432089 |
| **.3c** Implementación con mapeo de una tabla por jerarquía | 43543362 | 15945266 |
| **.3c** Implementación con mapeo de una tabla por clase | 54411495 | 16503301 |

b. Conclusión

A simple vista observando los resultados, comparando los tipos de colecciones utilizadas en cada uno de los ejercicios notamos que:

- las operaciones de inserción(ejercicio a) utilizando ArrayList utilizarán más tiempo en ejecutarse ya que al ser un tipo de colección indexada necesita de manejo de índices, el cual adiciona más tiempo de ejecución , en cambio como la colección Set es un tipo de colección no indexada el tiempo utilizado para la inserción será menor.

-pero, como se describió anteriormente, las colecciones como el ArrayList son mucho más eficientes con respecto a la búsqueda de datos en dicha colección, esto es gracias al manejo de índices. Esto es muy evidente en el ejercicio c.

- la otra conclusión se basa en el tipo mapeo de clases y jerarquías, en esa práctica se distinguió entre mapeo de una tabla por jerarquía y mapeo de una tabla por clase. Tanto en el ejercicio a, b y c , la política de que toda clase es una tabla es la que más tiempo consume con respecto a consultas, ya sea alta, baja, búsqueda. Esto se debe a que al tener mayor cantidad de tablas ( comparado con el mapeo de una jerarquía en una tabla)el acceso es más lento ya que siempre se requiere del uso de “join”.

Ejercicio 5

**5a.**

|  |  |  |
| --- | --- | --- |
| Acción/Implementación | HashSet  (tiempo en nanosegundos) | ArrayList  (tiempo en nanosegundos) |
| **.4a** Implementación con mapeo de una tabla por jerarquía | 1347856000 | 1263530501 |
| **.4a** Implementación con mapeo de una tabla por clase | 1580191358 | 1437383050 |
| **.4b** Implementación con mapeo de una tabla por jerarquía | 7040727 | 8518965 |
| **.4b** Implementación con mapeo de una tabla por clase | 7994190 | 7255072 |
| **.4c** Implementación con mapeo de una tabla por jerarquía | 7819265 | 10005826 |
| **.4c** Implementación con mapeo de una tabla por clase | 10927878 | 24203075 |
| **.4d** Implementación con mapeo de una tabla por jerarquía | 344301452 | 305016649 |
| **.4d** Implementación con mapeo de una tabla por clase | 469502093 | 414199343 |
| **.4e** Implementación con mapeo de una tabla por jerarquía | 131434495 | 118659434 |
| **.4e** Implementación con mapeo de una tabla por clase | 179899160 | 112916478 |
| **.4f** Implementación con mapeo de una tabla por jerarquía | 48166553 | 49310340 |
| **.4f** Implementación con mapeo de una tabla por clase | 65096696 | 63906097 |
| **.4g** Implementación con mapeo de una tabla por jerarquía | 127331767 | 70897549 |
| **.4g** Implementación con mapeo de una tabla por clase | 85498234 | 95863765 |

5b. Conclusión

En este caso, se mide el tiempo de consultas (búsqueda) a la base de datos.

Lo que ocurrió es que al contrario del ejercicio 3 no todas las consultas realizadas sobre colecciones ArrayList obtuvieron un tiempo menor al de las consultas sobre HashSet. Solo 9 de 14 ejercicios (consultas) de las diferentes implementaciones sobre ArrayList obtuvieron un tiempo más bajo comparado con las implementadas en HashSet.

Otra de las conclusiones es que en la gran mayoría de las consultas se obtuvo un mejor tiempo mapeando toda una jerarquía a una sola tabla, gracias a la menor utilización de “joins” en las consultas, la cual es una de las operaciones más costosas (en tiempo) dentro del lenguajes de consultas de base de datos. Pero esto puede tener una desventaja ya que este tipo de mapeo requiere mucho más espacio de almacenamiento que el mapeo de “una tabla por clase”.