## 2. Conclusiones tras ejecución de pruebas.

Con el fin de obtener la mejorar la percepción del comportamiento de la ejecución de la consulta preparada, tal y como se informa en el guion de esta práctica, se ha decidido montar un plan de pruebas con un total de 23 casos. Estos casos, vienen definidos en la Tabla 1.

|  |  |
| --- | --- |
| **Número de Caso** | **Número de hilos** |
| 1 | 1 |
| 2 | 2 |
| 3 | 4 |
| 4 | 8 |
| 5 | 10 |
| 6 | 16 |
| 7 | 20 |
| 8 | 30 |
| 9 | 32 |
| 10 | 40 |
| 11 | 50 |
| 12 | 60 |
| 13 | 64 |
| 14 | 70 |
| 15 | 80 |
| 16 | 90 |
| 17 | 100 |
| 18 | 120 |
| 19 | 128 |
| 20 | 130 |
| 21 | 140 |
| 22 | 144 |
| 23 | 255 |

Tabla 1. Plan de pruebas.

Se han escogido estos números debido a diferentes motivos. Primero de todo que, debido a la posibilidad de automatizar las ejecuciones, cuantas más mejor y así poder obtener una visión más perimétrica del comportamiento de estas conforme aumenta el número de hilos en los que se paraleliza una misma consulta ya preparada. Dicho lo cual, este plan de pruebas nace de unificar:

* Potencias de dos desde exponente uno hasta el exponente más cercano al límite de la propia máquina y, como es 144, potencia de dos con exponente 7.
* Ejecuciones de diez en diez hasta llegar al límite múltiple de 10 más cercano a 144.
* Forzar el doble de lo que, en teoría puede soportar la máquina. Como a la hora de la verdad una consulta con 288 provocaba una expulsión de la máquina, se decide lanzar la ejecución para 255, valor que aparentemente es el límite para el momento de las ejecuciones.

Cabe comentar que este plan de pruebas se ha ejecutado para los tres casos indicado, para las dos tesituras transaccionales y la única no transaccional.

### 2.1 Conclusiones de *No Transactional read-only*

Antes que nada, hacer hincapié en la razón de existencia de un apartado de ejecuciones no transaccionales en este caso de estudio.

Como se ha comentado previamente en esta memoria en lo referido en la definición de transacción, su única finalidad es la de mantener la consistencia de los datos a cambio de consumir tiempo de ejecución. Por tanto, una ejecución no transaccional se desmarca de la definición vista en el documento y en la teoría.

La razón de ser de este caso es para contrastar con las ejecuciones transacciones y ver todo lo que supone mantener la consistencia del conjunto de datos, especialmente cuando crece la cantidad de estos junto al deseo de paralelizar consultas.

Dicho esto, se procede a exponer los resultados obtenidos tras las ejecuciones. De los ficheros de *log* generados por las ejecuciones se han extraído dos parámetros: las transacciones por segundo y el tiempo total de ejecución. Después, se han calculado las transacciones totales, *Speed-up*, Coste y eficiencia. Los resultados se pueden apreciar en las gráficas de las siguientes páginas.

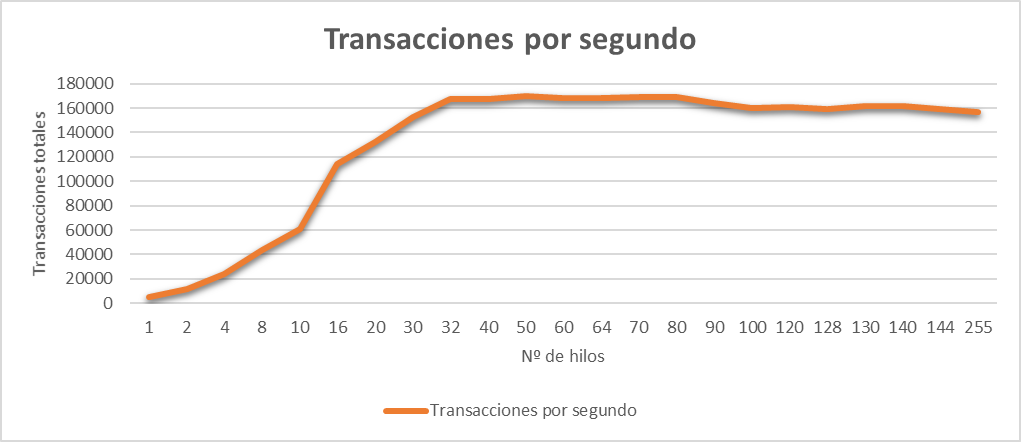


Tabla 2. Transacciones por segundo frente al número de hilos.

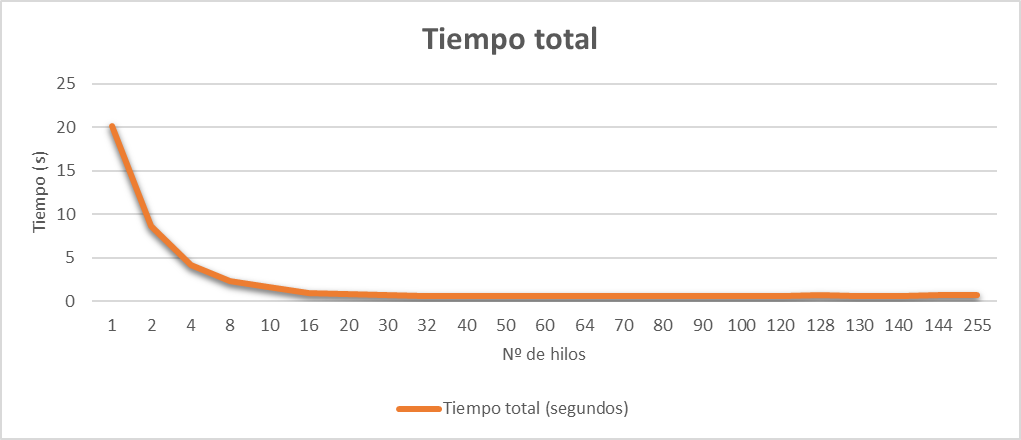


Tabla 3. Tiempo total requerido frente al número de hilos.

Tabla 4. Transacciones totales frente al número de hilos.

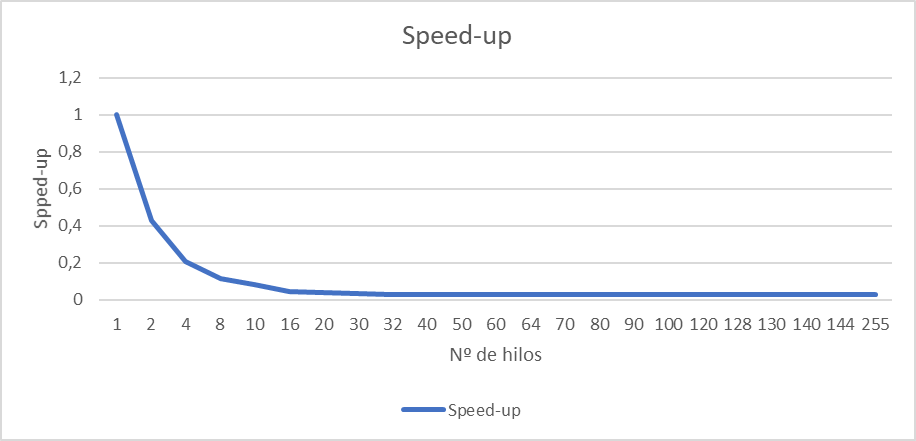


Tabla 5. Variación del incremento de la velocidad frente al número de hilos.

Tabla 6. Variación del Coste frente al número de hilos.

Tabla 7. Variación de la eficiencia respecto al número de hilos.

Tras haber realizado un análisis sobre el comportamiento reflejado en las gráficas se extraen las siguientes conclusiones.

Basándose en la gráfica de la Tabla 4, se puede apreciar como la cantidad de transacciones crece hasta llegar a un máximo absoluto y decae hasta, aparentemente, estabilizarse y sufrir grandes cambios. A esta conclusión se le suma la gráfica de la Tabla 6, la del coste, en la que puede apreciarse un comportamiento casi lineal del número de transacciones respecto al número de hilos. Con la mirada puesta en estas dos gráficas, se plantea la solución que se ha llegado a un punto de saturación ya que conforme se incrementa el número de hilos y el trabajo computacional de cada uno, no se aprecia un crecimiento notable del número de transacciones.

Análogamente se puede apreciar, centrando la mirada en las gráficas de las Tablas 5 y 7, en la que el incremento y eficiencia cae bruscamente y se mantiene prácticamente constante hasta el final. Esto, hace pensar que se alcanza el punto máximo con pocos hilos y conforme se van incrementando no se aprecia una mejora en el resultado de las ejecuciones.

Se puede concluir que el sistema no transaccional es asombrosamente rápido ya que, al no respetar la definición de transacción, la recuperación de datos es inmediata sin tiempos de esperas para respetar la consistencia de datos. Por tanto, el mismo sistema, pero transaccional, da pie a pensar que en el mismo periodo de tiempo se realizará mucho menos trabajo computacional.

### 2.2 Conclusiones de T*ransactional read-only.*

### 2.3 Conclusiones de T*ransactional read-Write.*

## 3. Conclusiones y comparativa de las tres tesituras.

Hablar un poco, muy breve, de lo guay que es este sistema. Estaría guay poner las gráficas to molonas que comparan los 3 tipos de ejecuciones.