Database snelheidsrapport

Systeembeschrijving:

* CPU: AMD Ryzen 3700x
* GPU: AMD Radeon 5800xt
* RAM: 16gb 3200mhz DDR4 Ram
* Opslag: Adata SX8200 Pro 1TB
* OS: Windows 10

# Betrouwbaarheid:

Om de betrouwbaarheid omhoog te krijgen is er voor gekozen om alle databases lokaal op de zelfde computer te draaien hierdoor kan er geen delay zitten tussen de server en de client. Ook is er hierdoor geen verschil in hardware waardoor de CRUD operaties even snel zouden kunnen draaien en er geen grote verschillen in kunnen zitten qua tijden.

Ook zijn alle CRUD operaties op de elke database 3 keer uitgevoerd en hier het gemiddeld uit gehaald zodat snellere en slomere runs gecompenseerd worden.

# Bevindingen

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ADO.NET | Create | Read | Update | Delete |
| 1 | 0.007 | 0.35 | 0.001 | 0.0003 |
| 1000 | 0.203 | 0.074 | 0.136 | 0.197 |
| 100000 | 19.425 | 6.330 | 12.715 | 18.412 |
| 1000000 | 181.254 | 59.10 | 119.305 | 172.633 |

Uit de tabel hierboven is te zien dat ADO.NET het snelst read operaties kan uitvoeren in grootte hoeveelheden en create operaties het sloomst in grootte hoeveelheden. In kleine hoeveelheden CRUD operaties is de read operatie het sloomst en de delete in kleine hoeveelheden het snelst. Terwijl de delete in kleine hoeveelheden het snelst is wordt het minder snel als er meerdere uitgevoerd worden. Bij de read operatie is dit andersom hierbij word het sneller als er meer operaties worden uitgevoerd.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Entity Framework | Create | Read | Update | Delete |
| 1 | 0.173 | 0.110 | 0.057 | 0.002 |
| 1000 | 0.062 | 0.178 | 0.584 | 0.019 |
| 100000 | 47.587 | 45.034 | 77.753 | 0.338 |
| 1000000 | 465.928 | 455.292 | 780.037 | 3.54 |

Bij de Entity Framework database is de delete operatie het snelst in kleine hoeveelheden operaties en de create het sloomst. In een grote hoeveelheid CRUD operataties is de delete operatie het snelst en de update het sloomst. Wat erg opvalt is dat de update operatie in kleine hoeveelheden operatie één van de snelste is en bij een grote heoveelheid operaties er bijna twee keer zo lang over doet als de op een na sloomste operatie. Ook valt het erg op dat de delete operatie extreem snel is. De delete operatie is meer dan 200 keer zo snel als de update operatie in grote hoeveelheden.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| MongoDB | Create | Read | Update | Delete |
| 1 | 0.666 | 0.696 | 0.733 | 0.737 |
| 1000 | 0.220 | 0.200 | 0.285 | 0.251 |
| 100000 | 19.109 | 17.356 | 24.667 | 23.158 |
| 1000000 | 191.894 | 172.536 | 246.505 | 230.739 |
|  |  |  |  |  |

Bij MongoDB is in kleine hoeveelheden operaties de delete de sloomste en de create het snelst. Echter zit er niet veel verschil tussen de CRUD operaties en zijn ze haast even snel in kleine hoeveelheden operaties. In grote hoeveelheden operaties is de read het snelst en de update het sloomst. Wat erg opvalt is dat de CRUD operaties van MongoDB qua snelheid erg dicht bij elkaar in de buurt komen in tegenstelling tot de Entity framework database waar de tijden erg verschillen.

# Conclusie

Elke database kan heeft zijn eigen voor en nadelen. Hierdoor is voor elke database wel een use case. Uit de tabelen in het kopje bevindingen is gebleken dat voor create, read en update operaties ado.net het snelst is in kleine en grote hoeveelheden operaties. Dit maakt ADO.NET een systeem waar veel data in word opgeslagen en uitgelezen maar niet vaak word verwijderd. Verder is Entity framework het snelst met delete operaties maar het sloomst met alle andere operaties. Dit kan handig zijn voor databases waar vaak data uit word gedelete. Echter zijn alle andere operaties het sloomst van alle databases wat de gemiddelde CRUD operatie tijd omlaag haalt. MongoDB heeft geen operatie die hij het snelst doet maar bij MongoDB komen de snelheden van alle operatie dicht bij elkaar in de buurt. Dit kan handig zijn voor een systeem waarbij alle CRUD operatie haast even snel moeten zijn.