**Deel 5 - Performantie tweaks**

**Bestandsorganisatie**

* Gegevens bewaard op schijf

=> Voor een database gebeurd dat in logische eenheden (pages)

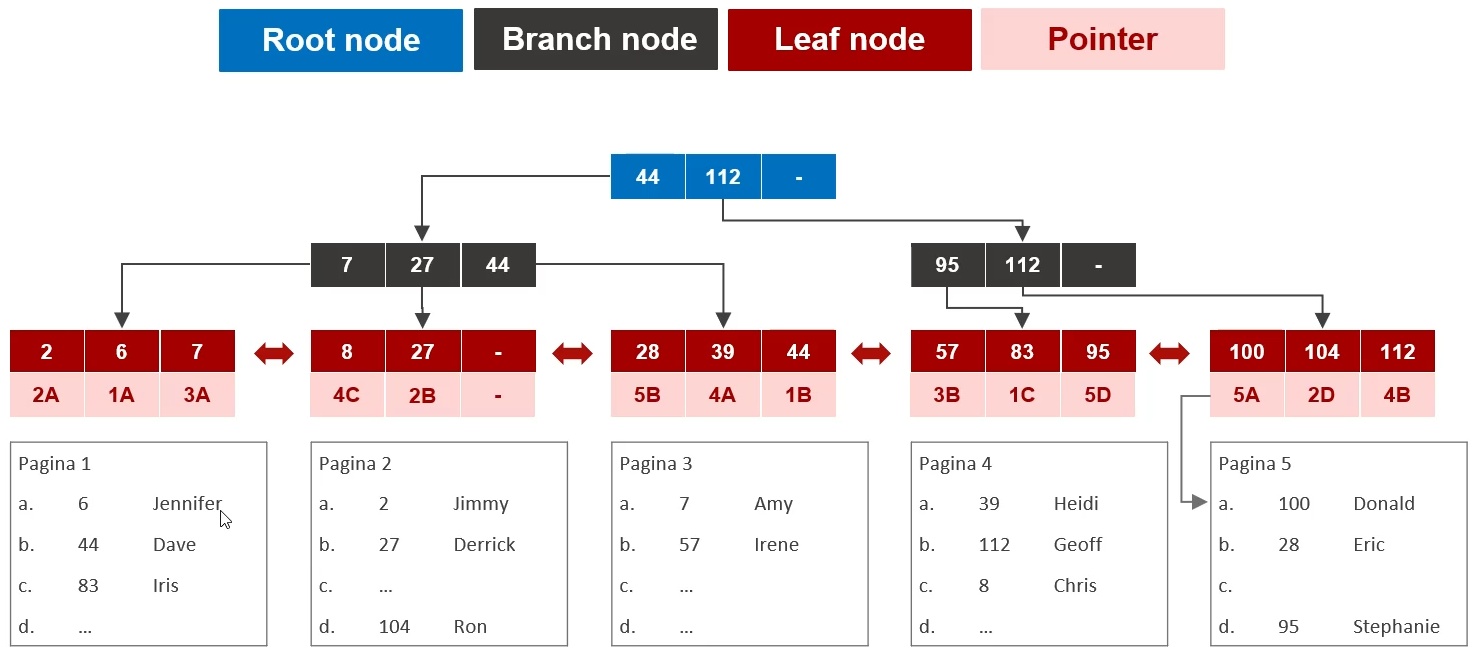
* Page
  + Vaste grootte ( 8 KB in SQL Server )

=> 2, 4 en 32 KB ook gangbaar

* + worden in SQL Server per 8 gebundeld in een extent

**Indexen - Opbouw**

* Index in termen van een boek
  + Terug te vinden achteraan in het boek
  + Gesorteerde lijst van keywords
  + paginanummers naast elk keyword
  + Doel => Snel terugvinden van woord in het boek
* Index in een database
  + Terug te vinden op schijf
  + Gesorteerde lijst van waarden voor een kolom
  + Voor elke waarde een of meerdere pointers naar de locatie op schijf
  + Doel => Snel terugvinden van waarden in de tabel
* Index wordt meestal opgebouwd als Boomindex (B-Tree)
* Opbouw met nodes (knooppunten)
  + Root node
  + Branch node
  + Leaf node



* Waarde in node wijst altijd naar hoogste waarde in onderliggende node
* Indexboom is altijd in overeenstemming met de inhoud van de tabel
* Index kan ook gedefinieerd worden op combinatie van kolommen
* Op een tabel mogen veel indexen gedefinieerd worden
* Index kost opslagruimte
* Mutaties in tabel => mutaties in de indextabel => extra write overhead
* Indexen kunnen op elk moment aangemaakt worden

**Indexen - Zoekalgoritme met unieke index**

Zoekopdracht **met filter voor een exacte waarde zonder sortering** van het resultaat

1. **Zoek de rootnode van de index**

=> deze node wordt een actieve node

1. **Is de node een leafnode?**
   1. Ja? => Ga naar stap 4
   2. Nee? => Ga naar stap 3
2. **Bevat de actieve node de gezochte waarde?**
   1. Ja?
      1. Maak van de node waarnaar verwezen wordt de actieve node
      2. Ga terug naar stap 2
   2. Nee?
      1. Bepaal in de actieve node de kleinste waarde die groter is dan de gezochte waarde
      2. Maak de node waarnaar deze waarde wijst de actieve node
      3. Ga terug naar stap 2
3. **Zoek in de actieve node de gezochte waarde op**

=> Deze waarde wijst nu naar alle pagina’s waar de gezochte waarde voorkomt

=> Laad al deze pagina’s voor verwerking

1. **Telkens voor elk van deze pagina’s**

=> Zoek de rijen op waar de kolomwaarde gelijk is aan de gezochte waarde

Zoekopdracht met **niet exacte filter en met sortering** van het resultaat

1. **Zoek de rootnode van de index, dit wordt de actieve node**
2. **Is de node een leafnode?**
   1. Ja?
      1. Ga naar stap 4
   2. Nee?
      1. Ga naar stap 3
3. **Bevat de actieve node de gezochte waarde?**
   1. Ja?
      1. Maak van de node waarnaar verwezen wordt de actieve node
      2. Ga terug naar stap 2
   2. Nee?
      1. Bepaal in de actieve node de kleinste waarde die groter is dan de gezochte waarde
      2. Maak de node waarnaar deze waarde wijst de actieve node
      3. Ga terug naar stap 2
4. **Zoek in de actieve node de gezochte waarde op**
5. **Zoek voor elk van deze pagina’s telkens de rijen op waar de kolomwaarde gelijk is aan de gezochte waarde**
6. **Laad alle pagina’s waarnaar deze leafnode verwijst in voor verwerking**
7. **Is er een volgende leafnode? (via double linked list)**
   1. Ja?
      1. Maak deze node de actieve node en ga terug naar stap 2
   2. Nee?
      1. Einde verwerking

**Indexen - Waar leggen**

* Automatisch
  + Bij definiëren primaire sleutel (Clustered index)
  + Bij definiëren alternatieve sleutels (Unique index)
* Zelf index leggen op
  + Refererende sleutels (Foreign keys)
  + Kolommen waarop geselecteerd en gesorteerd wordt

(WHERE, GROUP BY, ORDER, LIKE, DISTINCT)

* Nakijken van cardinaliteit voor effect van index
  + **Tabelniveau** (aantal rijen in tabel)

=> Hoe meer rijen, hoe meer effect

* + **Kolomniveau** (aantal verschillende waarden)

=> Hoe meer verschillende waarden, hoe meer effect index

* Nakijken van distributie van waarden in kolom
  + Hoe ongelijker verdeling, hoe meer effect index

**Indexen - Beheer**

* Data geraakt gefragmenteerd op schijf

=> Indexen ook

sys.dm\_db\_index\_physical\_stats

* Toont volgende informatie
  + Grootte en fragmentatie van data
  + Grootte en fragmentatie van index

=> Indexen opnieuw opbouwen om performantie te vergroten

**Performantie meting tellers**

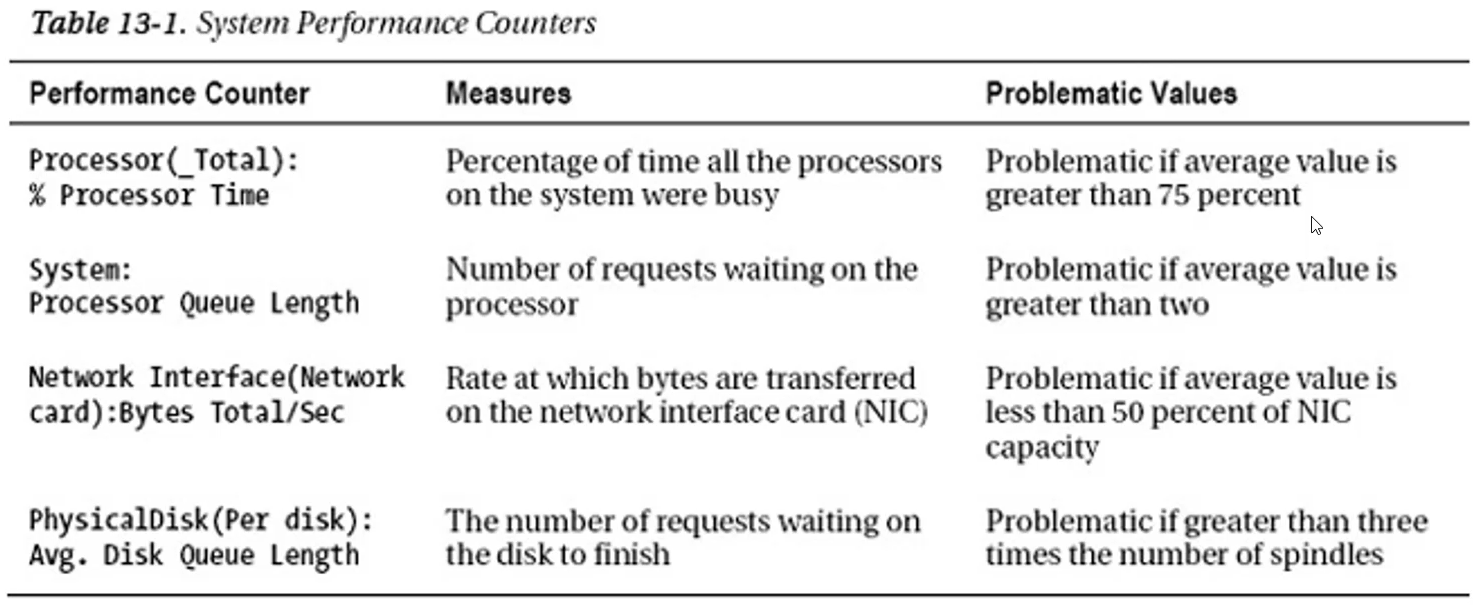
**Waarom meten?**

* Kijken dat al de aanwezige resources goed samenwerken (geheugen, processor kracht, schijven, netwerk,..)
* Kijken dat de jobs/taken in SQL server goed uitgevoerd worden
* Integriteit in de gaten houden
* Errors bekijken
* De opslagruimte in de gaten houden
* Potentiële problemen en dingen die geoptimaliseerd kunnen worden identificeren
* Verhogen van efficiëntie
* Kosten besparen
* Frustraties verminderen

**Performantie - Tellers**

* Tellers om performantie te meten
* Bestaan op 2 niveau’s
  + Systeem
  + SQL Server
* Treshold bepalen per teller
  + Kan overschreden of onderschreden worden
  + Teller bepaalt in welk geval het duidt op een problematische situatie

**Performantie - Systeem tellers**



**Performantie - Strategie**

* Creëer een performantie basislijn
  + Specifieer wat te controleren
  + Specifieer tellers en metingen bv.:
    - % Processor Time
    - Memory Pages/Sec
    - Memory Available Bytes
    - % Disk Time
    - SQL Server Buffer Cache Hit Ratio
    - SQL User Connections
    - SQL Locks
    - …
* Voer periodieke performantie controles uit
  + Plan periodiek performantie controles om te controleren dat de performantie niet gezakt is
* Voer veranderingen uit in SQL Server of Windows en evalueer de impact ervan
  + Na een audit kan je misschien dingen vinden die een aanpassing vereisen
  + Wees nauwgezet!

**Performantie - Mythes**

* Mythes:
  + Wanneer processor gebruik hoog is, is een snellere processor nodig

=> Eén ding is zelden de dader

* + 80 % van de performantie word tbepaald door de applicatie code

=> Betere code is beter, maar een beter ontwerp is het best

* + Een geoptimaliseerde server is de enige sleutel tot database performantie

=> Alles moet ook nog steeds over het netwerk gaan

**Performantie - Tools**

* Performance Monitor (Windows)
* Dynamic Management Views
* Activity Monitor
* System stored Procedures
* Profiler + Tuning Advisor

**Performantie - Performance Monitor**

* (Prestatiemeter)
* Windows tool
* Monitored een object van het systeem (geheugen, processor)
* Toont statistische informatie (teller) over een object
* Monitored live en/of monitored over een bepaalde tijdspanne
* Vuurt alerts af wanneer performantie tellers buiten gespecificeerde tresholds komen

**Performantie - Dynamic Management Views**

* Views in MS SQL server die dezelfde informatie aanbieden als performance counters
* Bv. sys.dm\_os\_performance\_counters

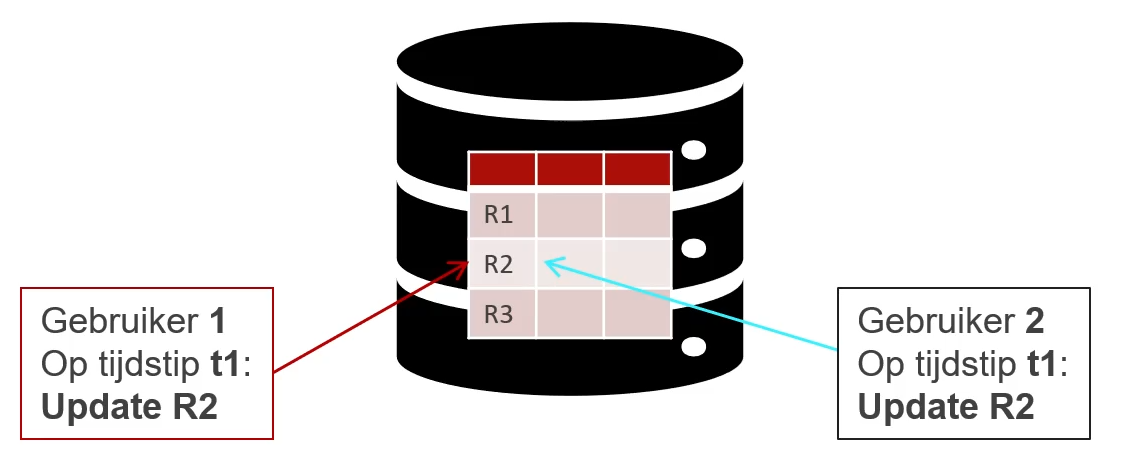
(operating system informatie in relationeel formaat)

**Performantie - Activity monitor**

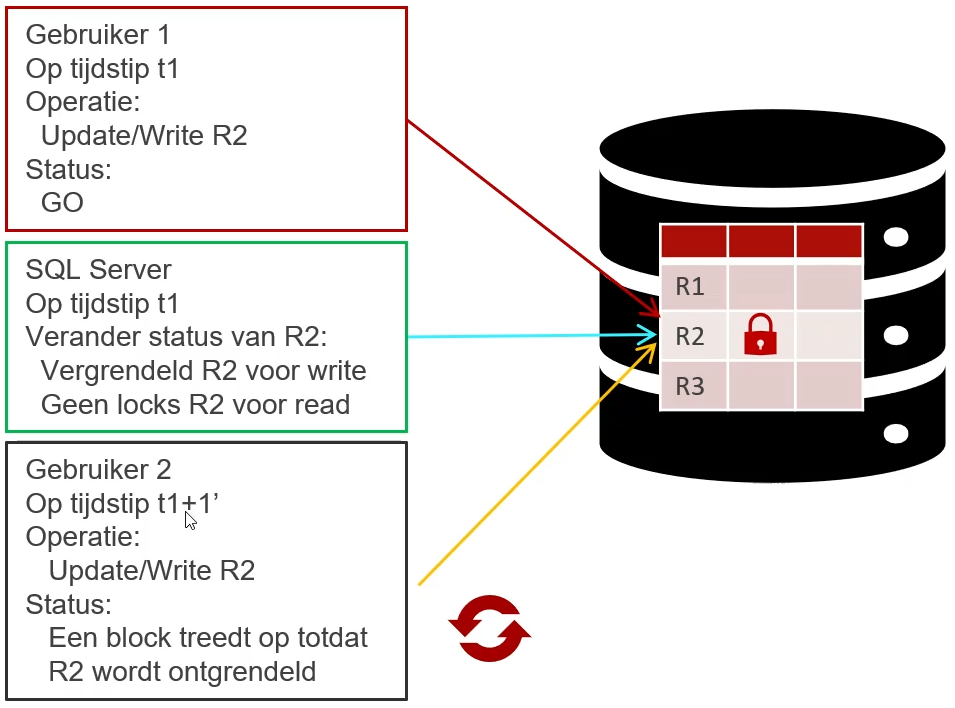
* Bijna real-time, klein grafisch performantie dashboard in MS SQL Server
* 5 belangrijkste voorgedefinieerde secties:
  + Bijna real-time grafieken
  + Lijst van actieve processen met gedetailleerde informatie
  + Resource Waits
  + Database bestand I/O
  + Recente “dure” queries
* Bruikbaar om bv. Deadlocks te detecteren en op te lossen

**Performantie - Locks, blocks & deadlocks**

* Concurrency
  + Meerdere gebruikers of processen benaderen dezelfde database op hetzelfde moment
  + Dit kan resulteren in conflicten over resource toegang (Bv. Deadlocks)
  + Een of ander mechanisme moet bestaan dat deze conflicten voorkomt

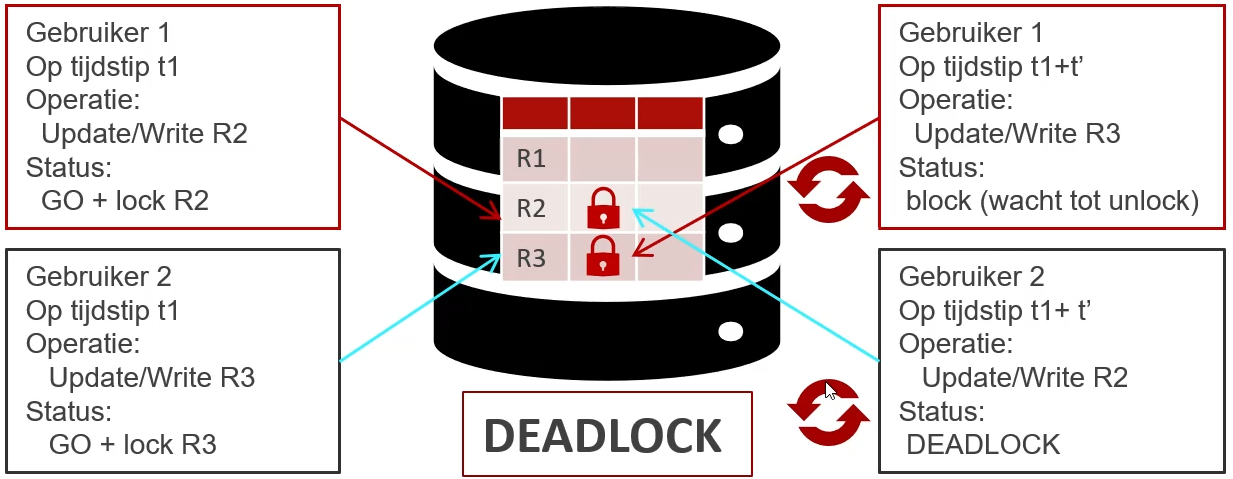


**Block**



=> Een **block** die de 2de update verhinderd tot de 1ste gebruiker zijn operatie heeft afgerond

**Deadlock**



=> Beide gebruikers zitten te wachten op de andere gebruiker om een transactie te committen

Oplossing => 1 van de 2 gebruikers zal moeten stoppen met de operatie

**Performantie - Systeem stored procedures**

* Activity Monitor is een GUI tool die vaak de output van stored procedures gebruikt
* Bijvoorbeeld:
  + EXEC sp\_who
  + EXEC sp\_who2
  + EXEC sp\_lock

**Performantie - SQL Server Profiler**

* SQL Server Profiler
  + Mogelijkheid om zo goed als elke vent in SQL Server te monitoren (=Trace)
  + En dit op te nemen
* Trace resultaten kunnen
  + Bekeken worden via de GUI
  + Bewaard worden als een workload file
* Workload file kan geanalyseerd worden
  + Gebeurt met de **Database Engine Tuning Advisor**
  + Resultaat is aanbevelingen ter verbetering van performantie op basis van analyse