**Deel 1 - Installatie SQL Server**

DBA = DataBase Administrator

* beheert de database server en voert onderhoud uit
* Taken:
  + Setup + testen omgeving
  + Back-up & recovery plan
  + Tuning en optimalisatie
  + Security
  + Disaster recovery
  + ...

**Sequel Server overview**

Management tools



**Management Studio**

* en database configuratie

**Configuration Manager**

* Services (her)starten/stoppen
* Services configureren
* ...

**SQL Server Profiler**

* Tool om tracing te doen van “gebeurtenissen” op SQL Server
* Kan helpen bij het vinden van oorzaken bij problemen

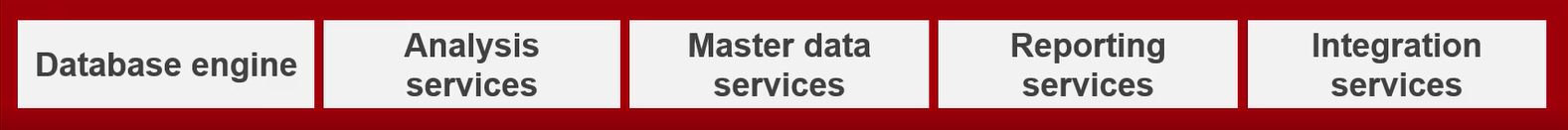
**Database Tuning Advisor**

* Analyseren van workloads
* Aanbevelingen voor verbetering genereren

**SQL Server Data Tools**

* Ontwerpen van packages voor data transformatie
* Maakt gebruik van Visual Studio en de Integration services component

Componenten



**Database engine**

core service voor:

* Het bewaren en verwerken van gegevens
* Het beveiligen van gegevens/database objecten

**Analysis services**

core service voor:

* Online Analytical Processing (OLAP)
* Data mining = gericht zoeken naar statistische verbanden in gegevensverzamelingen (bv aankoopgedrag klanten)

**Reporting services**

core service voor:

* Aanmaken,deployen en onderhouden van rapporten

**Integration services**

core service voor:

* Data integratie = gegevens van verschillende delen van het informatiesysteem migreren, combineren, …
* Data transformatie = converteren van gegevens van het ene formaat naar het andere (bv loongegevens)

**Licentiebeheer**

* Licentiebeheer of Software Asset Management (SAM)
* Twee opties:
  + Server/CAL (CAL = Client Access License)
  + Per-core

**Server/CAL optie**

* Server
  + Elk fysisch/virtueel Operating System Environment OSE, waarop een SQL server 2019 draait moet een aparte licentie hiervoor hebben
  + Per OSE meerdere instanties toegestaan
* CAL
  + Licentie voor elke gebruiker/device die toegang moet hebben tot de SQL server

**Per-core optie**

* Geen limiet op aantal gebruikers en/of devices
* Worden verkocht in pakketjes van 2

**=> In een fysisch OSE**

* Alle fysische cores op de server moeten een licentie hebben
* Minimum 4 cores voor elke fysische processor, ook al zijn er bv maar 2 aanwezig

**=> In een virtuele omgeving of container**

* Licentie per virtuele core (VC)
* Min. 4 cores per virtuele machine

**Algemene guidelines**

Software niet zomaar installeren zoals gewoon programma:

1. Juiste installatie en configuratie zeer belangrijk

* Beïnvloed security, schaalbaarheid, onderhoudbaarheid
* Voorkomt problemen in de toekomst op vlak van
  + Security
  + Locatie databasebestanden

1. Enkel componenten installeren die gebruikt zullen worden

* Vermindert resource gebruik
* Vermindert mogelijkheden tot aanval van buitenaf

1. Zowel OS als SQL Server moeten correct geconfigureerd worden

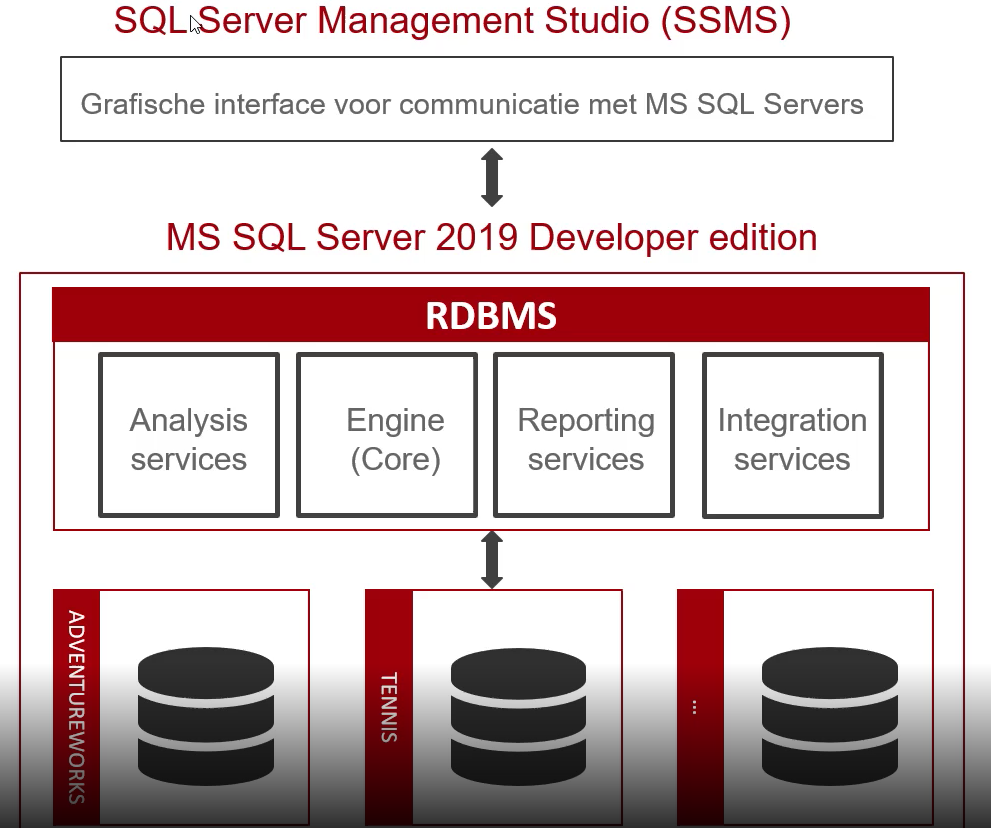
**Pre-installatie taken OS**

1. Windows Domain accounts aanvragen voor je SQL Server Service accounts
2. Windows Server installeren en patchen
3. BIOS en andere firmware updaten
4. Device drivers updaten
5. Power management settings checken
6. Bijkomende rechten geven voor SQL Server service accounts
7. Bereikbaarheid server instellen
8. Overige performantieverbeteringen doorvoeren
9. Laatste voorbereiding

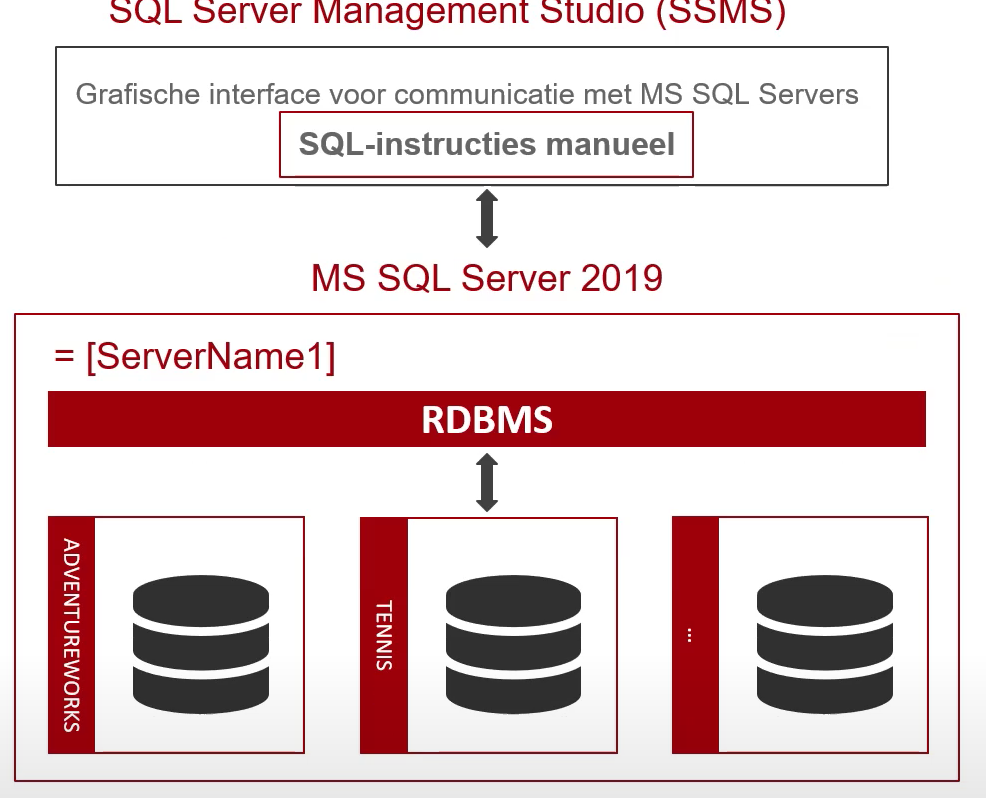
**Onderhoud**

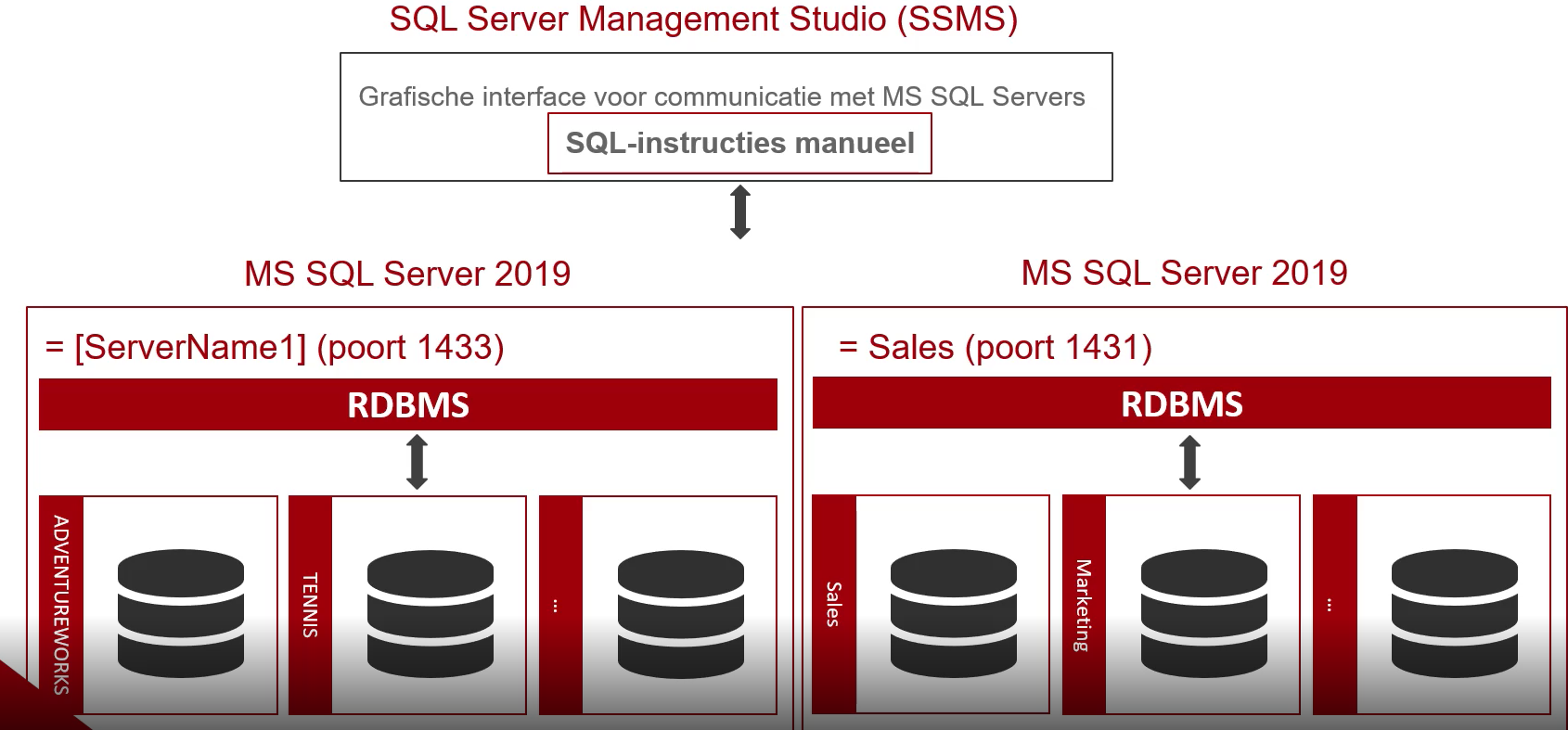
* Zowel OS als SQL Server moeten correct onderhouden worden
* Service packs en hotfixes op beide niveaus
* Proactief en preventief onderhoud heeft veel voordelen
  + Vermindert voorkomen van gekende problemen
  + Veelvuldig rapporteren van problemen kan leiden tot cumulatieve updates
* Betere ondersteuning bij verkopers (Hardware, storage, Microsoft)
  + Vermijden van wegvallen support door out-of-date zijn op gebied van firmware, drivers, patches
* 3 manieren om SQL Server up-to-date te brengen
  + Hotfixes
  + Cumulatieve updates
  + Service Packs
* **Hotfixes**: ontworpen om een specifiek defect op te lossen
* **Cumulatieve updates:** Pakketjes van hotfixes
* **Service Packs**: Ontworpen om een groot aantal defecten op te lossen

**RDBMS van SQL Server** (Relational Database Management System)



**Werkomgeving:**

* 1 instance 
* 2 instances

=> Wanneer meerdere sql server instances op eenzelfde (virtuele) server draaien moet elke instance een unieke naam krijgen en specifieke poort voor communicatie 

**Databases**

**Systeem databases**

* Wordt gebruikt door RDBMS (SQL server)
* Bevatten informatie over services, users en user databases

**User databases**

* Aangemaakt door users

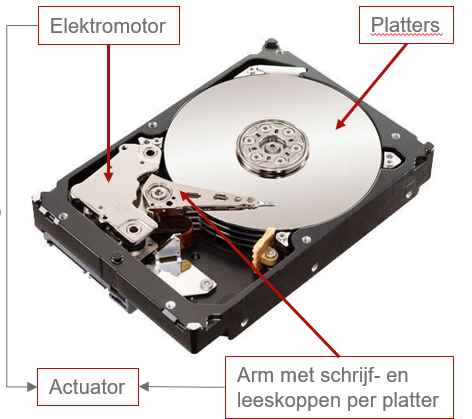
**Soorten**

* **master**
  + Container met systeem tabellen
  + Houdt info bij over services, users, …
* **msdb**
  + Houdt informatie bij over de Agent Service
  + Scheduled Jobs / Alert
* **model**
  + model database voor nieuwe user databases
  + Niet veel wijzigingen => Niet frequent backuppen
* **tempdb**
  + tijdelijk data in op slaan
  + Wordt leeggemaakt bij elke restart van SQL server

**Deel 2 - Opslag fundamentals**

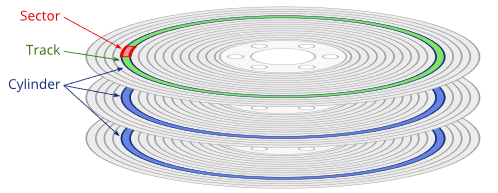
**I/O Patterns**

* Werken met een database houdt in
  + Lees- en/of schrijfopdrachten van gegevens
  + Van en naar database bestanden, transaction logfile op schijf
  + Van en naar data in het RAM geheugen
  + We spreken van I/O (Input/Output)
* Bedoeling is dit zo snel en efficiënt mogelijk te laten verlopen => Fundamentele kennis van opslag gerelateerde hardware = noodzakelijk

**Harde schijf**

* Actuator
  + Set van schrijf- en leeskoppen op een beweegbare arm
  + Elektromotor voor aandrijving arm
* Schijven draaien rond aan bepaalde snelheid
  + Eenheid = Revolutions Per Minute (RPM)

*(Desktop/server => van 7,2K tot 15K RPM)*



* Coating van platter ingedeeld in concentrische cirkels = **sporen (tracks)**
* Elk spoor onderverdeeld in stukken = **sectoren**
* Sporen die loodrecht op elkaar liggen = **cilinder**
* Sporen onderverdeeld in sectoren van 512 bytes

**I/O opdracht op 2 manieren**

* Sequentieel
* Random

1. **Sequentieel**

* Er wordt netjes van de ene sector naar de aangrenzende sector gesprongen
* Wanneer alle sectoren “verwerkt” zijn wordt naar het volgende spoor gesprongen
* Verwerking gebeurt **snel** omdat actuator weinig moet bewegen

1. **Random**

* Er wordt van de ene locatie naar de andere gesprongen
* Verwerking gebeurt **trager** omdat actuator regelmatig moet bewegen

**Database workloads**

* I/O opdrachten bij een database gebeuren volgens een bepaald patroon

=> **database workload**

* Belangrijkste database workloads
  + OLTP (Online Transaction Processing)
  + DW (Data Warehouse)
  + OLAP (OnLine Analytical Processing)

**OLTP (OnLine Transaction Processing)**

* Acties
  + Regelmatig schrijven naar database en transaction log bestanden
  + Leesoperaties om data in RAM te brengen indien nog niet aanwezig
  + Hoofddoel = Data processing
* Gevolg naar I/O Patterns toe
  + Leesoperaties indien van harde schijf
    - Random omwille van schrijfoperaties die zorgen voor verplaatsing actuator
  + Schrijfoperaties naar database
    - Random omwille van leesoperaties die zorgen voor verplaatsing actuator
  + Schrijfoperaties naar transaction log bestand
    - Sequentieel of random afhankelijk van aantal factoren (verderop meer)

**OLAP (OnLine Analytical Processing)**

* Acties
  + Voornamelijk leesoperaties van cube bestanden in data warehouse
  + Hoofddoel = Data analyse
* Gevolg naar I/O Patterns toe
  + Random leesoperaties van database omwille van vele gelijktijdige leesoperaties

**DW (Data Warehouse)**

* Acties
  + Grote sequentiële schrijfoperaties naar database bestanden (Dataload)
  + Schrijven naar transaction log bestand bij dataload
* Gevolg naar I/O Patterns toe
  + SchrijfOperaties naar database
    - Sequentieel want tijdens dataload meestal geen leesoperaties
  + SchrijfOperaties naar transaction log bestand
    - Sequentieel of random afhankelijk van factor opslag (verderop meer)

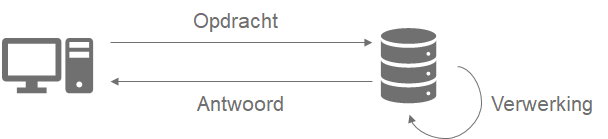
***Nadenken welke workload van toepassing zal zijn vooraleer SQL Server te installeren!***

**Opslag eigenschappen en performantie**

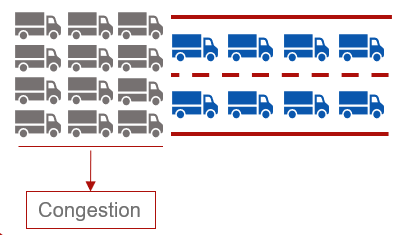
* Hardware eigenschappen hebben invloed op performantie
* **Belangrijkste eigenschappen in functie van opslag**
  + Latency (vertraging)
  + Throughput
  + I/O Block Size
  + IOPS

**Latency (vertraging)**

* Wachttijd nodig om I/O opdracht te volbrengen



* Hoe lager hoe beter
* Latency voor de harddisk
  + **Laag** voor Sequentiële I/O opdrachten
  + **Hoger** voor Random I/O opdrachten



**Throughput**

* Hoeveel data kan simultaan getransfereerd worden
* Hoe hoger de bandbreedte hoe meer data
* Opstopping (congestion) bij meer data dan bandbreedte

**I/O Block Size**

* **Blocking** = bundelen van gegevens (makkelijker en sneller met data werken)
* **Deblocking** = data uit een blok halen
* Data in SQL Server wordt gelezen/geschreven in blokken
* Block Size verwijst naar de hoeveelheid data vervat in 1 blok
* Kan variëren in functie van I/O operatie
* Hoge blocksize
  + Minder I/O operaties, kans op meer overtollige data
* Kleine blocksize
  + Meer I/O operaties maar minder overtollige data

**IOPS**

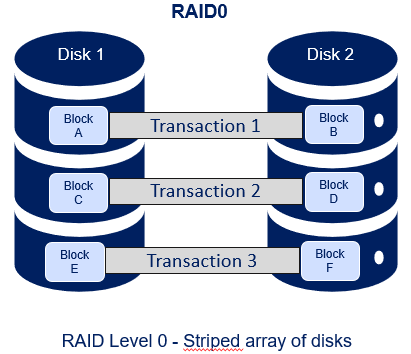
* **I/O operations per second**
* Alleenstaand is het een weinig zeggend gegeven
* Altijd in combinatie bekijken met de overige 3
* Wordt meestal bekeken op 4 meetpunten
  + Random read IOPS
  + Random write IOPS
  + Sequential read IOPS
  + Sequential write IOPS

**Belangrijkste karakteristieken m.b.t. database workload**

* **Online Transaction Processing (OLTP)**
  + Aantal IOPS
  + Latency
  + Throughput
* **Datawarehouse en OLAP (=Business Intelligence)**
  + Throughput
  + Latency
* Kiezen welk soort opslag te gebruiken
  + Hangt af van soort workload, performantie vereisten en beschikbaar budget
  + Bestaande infrastructuur
  + Capaciteiten/kennis van werknemers
  + Politiek

**Hardware - RAID configuraties**

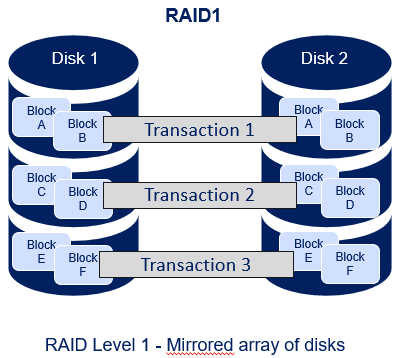
* Harde schijven kunnen af en toe crashen!
* Verhogen van snelheid/betrouwbaarheid door redundante disks toe te voegen
* RAID = Redundant Array of Inexpensive/Independent Disks
* Redundantie door data te spreiden over meerdere disks
* Grotere logische drives door combineren van meerdere fysische disks
* Doorstaan van faling van een disk
* Heeft minstens 2 disks nodig
* Betere I/O performantie door parallelle toegang
* Meerdere varianten
  + RAID0
  + RAID1
  + RAID5
  + (Ook RAID2,3,4,6)
* Combinaties
  + RAID01 (0+1)
  + RAID10 (1+0)
  + RAID50 (5+0)

**RAID0**

* **Striping**
* Striped array zonder fout tolerantie
* Bestaat uit n disks
* **Data wordt gelijkmatig verspreid over het aantal beschikbare disks**
* **Capaciteit = n x smin** waarbij
  + **n** = aantal disks
  + **smin** = capaciteit kleinste disk
* **Voordelen**
  + Hoge performantie
  + Simpel ontwerp
* **Nadelen**
  + Geen fout tolerantie
  + Alles weg bij crash van 1 disk
  + Betrouwbaarheid lager dan bij 1 disk
* **Gebruik**
  + Tijdelijk bewaren van data met  
    nood aan hoge bandbreedte
  + Compilatie van grote software projecten

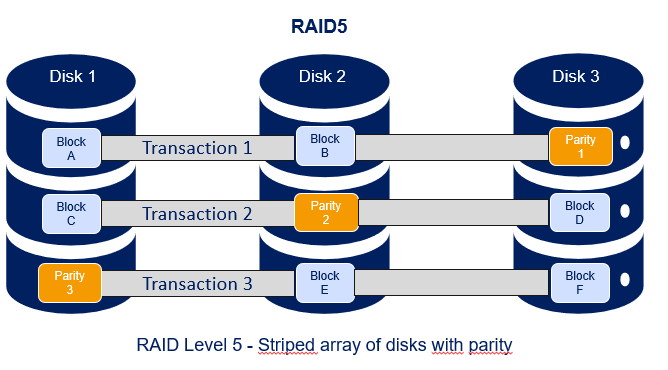
*Aantal IOPS en writes stijgt wanneer er meer disks worden toegevoegd*

**RAID1**

* **Mirroring without parity or striping**
* Bestaat uit minstens 2 disks
* **Data wordt gedupliceerd op alle disks in de array**  
  
* **Capaciteit**
  + 1x kleinste disk van de RAID array
* **Voordelen**
  + Kan overweg met falen van een disk
  + Hogere lees performantie door simultaan lezen van meerdere disks
  + Simpel ontwerp
  + Geen data reconstructie nodig
* **Nadelen**
  + Grote overhead
* **Gebruik**
  + Applicaties die hoge  
    betrouwbaarheid nodig hebben
  + Meestal gebruikt voor kleinere  
    data hoeveelheden

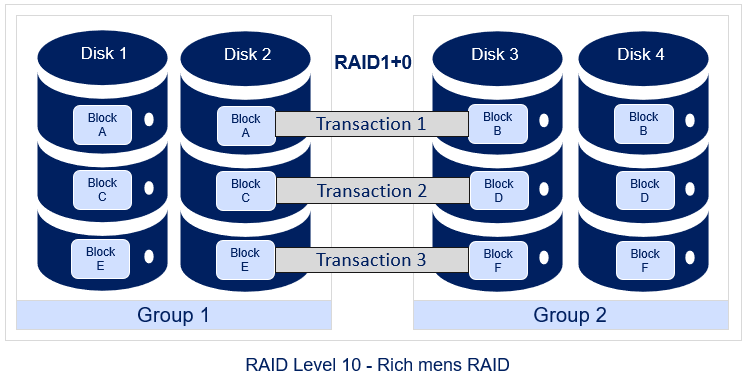
**RAID5**

* **Striping with distributed/interleaving parity**
* Bestaat uit n+1 (minimum 3) disks



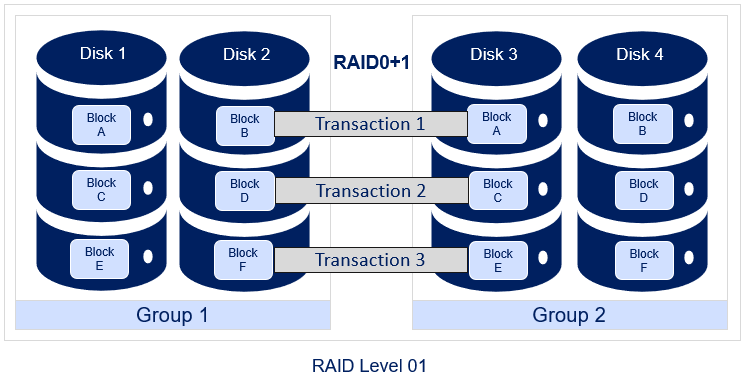
* **Data wordt verspreid over n disks (varieert van transactie tot transactie)**
* **Pariteitsinformatie wordt berekend en naar overige disk geschreven**
* **Capaciteit = (n -1) x smin**
* **Voordelen**
  + Hoge data rates
  + Weinig overhead (enkel pariteit bepalen)
  + Gelijkmatig gespreide toegang
* **Nadelen**
  + Schrijf operaties trager dan bij andere RAID configuraties
  + Disk falen heeft middelmatige impact op performantie
  + Complex ontwerp
  + Complexer om terug op te bouwen
* **Gebruik**
  + Applicaties die hoge betrouwbaarheid nodig hebben
  + Meest veelzijdige RAID niveau
  + Meest gebruikte RAID niveau

**RAID10 (1+0)**

* **Stripeset of mirrors**
* Bestaat uit 2n (minimum 4) disks
* Voorkeursconfiguratie voor I/O intensieve applicaties zoals databases
* **Capaciteit = p x smin** (waarbij **p** aantal RAID1 pairs is)
* **Voordelen**
  + Combinatie van voordelen RAID0 (performantie) en RAID1 (redundantie)
  + Simpel ontwerp
  + Hoge data rates
  + Betere throughput (Behalve t.o.v. RAID0) en latency dan alle andere RAID levels
  + Zeer geschikt voor write-intensieve workloads
* **Nadelen**
  + Grote overhead
  + Duurder dan andere RAID configuraties
* **Gebruik**
  + Applicaties die hoge betrouwbaarheid en performantie nodig hebben

**RAID01 (0+1)**

* **Mirrored stripesets**
* Bestaat uit 2n (minimum 4) disks



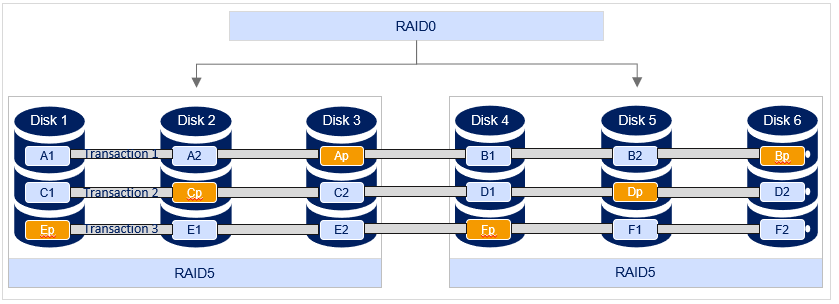
* **Voordelen**
  + Hoge performantie
  + Kan overweg met single disk failure (wordt dan RAID0)
  + Simpel ontwerp
* **Nadelen**
  + Grote overhead
* **Gebruik**
  + Applicaties die matige betrouwbaarheid en hoge performantie nodig hebben

**RAID10 vs RAID01**

* **Gelijk(w)aardig** op vlak van
  + Performantie
  + Opslagcapaciteit
* **Verschil** zit in fouttolerantie bij gebruik van >= 6 disks
  + Hoger in RAID10 dan in RAID01

**RAID50 (5+0)**

* **Parity and disk striping across multiple drives**
* Bestaat uit minstens twee, 3-disk RAID5 arrays
* **Splitst data op in kleinere blokken die gestriped worden over de RAID5 arrays**



* **Voordelen**
  + Betere redundantie dan RAID5
  + Betere performantie dan RAID5, vooral bij writes
  + Kan overweg met single disk failure in elke RAID5 array  
      
     Indien nog een disk uitvalt in die array valt de ganse RAID
  + Betere performantie bij verlies van 1 disk dan RAID5
  + Goed compromis tussen RAID5 en RAID10
  + Minder duur dan RAID10
* **Nadelen**
  + Duurder dan RAID5
* **Gebruik**
  + Data warehousing
  + Applicaties met hoge vereisten naar capaciteit en fout-tolerantie toe

**Aandachtspunten**

* Hou rekening met volgende punten bij bepalen RAID level
  + Het aantal spindles in een array is heel belangrijk
    - Groot aantal kleinere disks   
      => betere performantie dan klein aantal grote disks
  + RAID5 heeft een write performantie handicap
    - Kan geen verlies overleven van meer dan 1 disk in de array
    - Hoe meer disks in de array, hoe meer kans op verliezen van een disk
    - In geval budget een rol speelt, gebruik RAID5 dan om weinig gebruikte data op te zetten  
      (bv weinig veranderende data bestanden, back-ups)
  + RAID10 en 1 hebben zeer goede write performantie
    - RAID10 meer redundantie dan RAID5
    - Probeer altijd RAID1 of RAID10 te gebruiken voor log-bestanden
  + Onderhandel niet met jezelf omtrent opslag
    - Vraag altijd RAID10 aan en onderhandel dan naar beneden indien het moet
    - Gebruik RAID5 voor weinig gebruikte data bestanden en back-ups indien je geen RAID10 kan krijgen

**Overige opslag gerelateerde hardware**

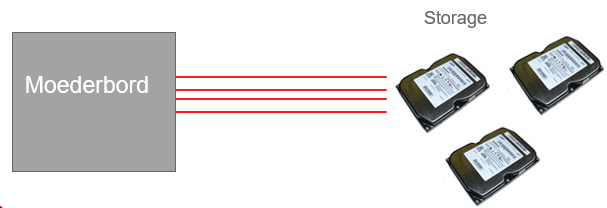
**SSD (Solid State Drive)**

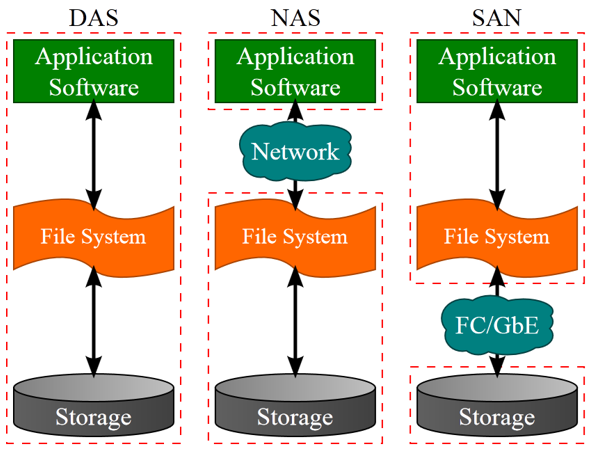
* Te gebruiken als een gewone standaard disk
* Volledig elektronisch (geen bewegende delen)
* Energiezuiniger
* Sneller dan een gewone standaard disk, vooral voor random I/O operaties
* Minder gevoelig voor schade
* Geen fragmentatie zoals bij een gewone standaard disk
* Nadelen
  + Limiet op aantal schrijf-operaties bij niet-vluchtige uitvoering
  + Niet-vluchtige versie kan alleen maar schrijven naar lege geheugencellen
  + Geheugencellen waarvan “vermoed” wordt dat ze data bevatten moeten eerst leeggemaakt worden
  + Schrijven naar lege geheugenblokken zeer snel, maar pak trager indien een overwrite moet gebeuren

**Tips aanvragen opslagruimte**

* Vraag niet enkel opslagruimte aan op basis van opslagcapaciteit vereisten
  + Kans op minder goede performantie omdat men niet weet waarvoor het gebruikt zal worden
* Specificeer je performantie vereisten
  + Sequentiële performantie in MB of GB per seconde
  + Random performantie in IOPS
* Overweeg gebruik van “short-stroking” om I/O performantie te verbeteren
  + Bewust kleiner percentage van beschikbare opslagruimte gebruiken
  + Actuator moet minder ver bewegen om data te vinden
* Meer opslagruimte aanvragen dan wat je nodig zal hebben
  + Handig bij onderhandelingen

**Host Bus Adapter (HBA)**

* Component die voorziet in fysische verbinding tussen een computer en opslag media of een netwerk
* Voorziet ook in het verwerken van I/O i.p.v. dat CPU dit doet
* Stuurt de beweging van data aan tussen het moederbord en de disks volgens een bepaalde interface
* Belangrijk in opslag en I/O discussies
* IDE (Integrated Drive Electronics => Verouderd!)
* SATA (Serial Advanced Technology Attachments => Thuisgebruik)
* SCSI (Small Computer System Interface => Servers)
* Firewire
* Over het netwerk via netwerk protocols
  + AoE (ATA over Ethernet)
  + iSCSI (internet Small Computer System Interface)
  + Fibre Channel (over ethernet, over IP, internet Fibre Channel)

**DAS - NAS - SAN**

**DAS (Direct Attached Storage)**

* Rechtstreeks gekoppeld aan de server

**NAS (Network Attached Storage)**

* Verbonden via het netwerk
* Technologie voor file-and-print servers
* Maakt gebruik van bestandsgeoriënteerde protocollen waarbij een deel van een bestand wordt opgevraagd i.p.v. een blok op een opslagmedium
* Niet geschikt voor SQL Server data-, transaction log en back-up bestanden

**SAN (Storage Area Network)**

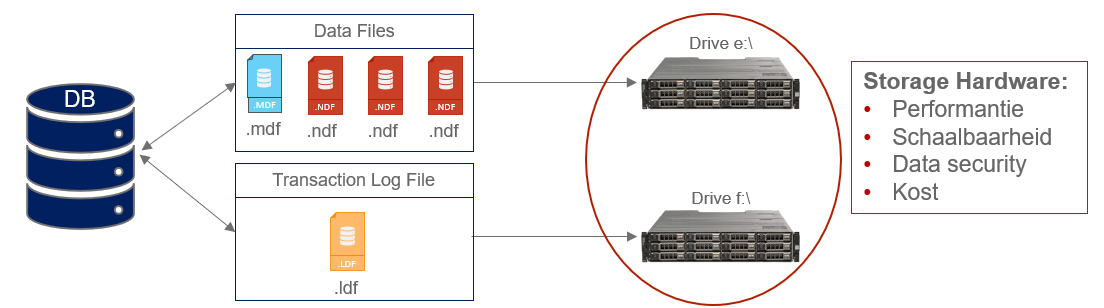
* Architectuur die dient als koppeling tussen servers en opslagapparaten zodanig dat het OS denkt dat het opslagapparaat rechtstreeks is aangekoppeld
* Meestal iets trager dan DAS (intensiever gebruik door meerdere toepassingen)
* Duurder, in aanschaf en onderhoud maar zeer flexibel en onderhoudbaar  
   Beheer opslagsysteem mogelijk zonder server uit bedrijf te nemen
* Maakt meestal gebruik van het iSCSI protocol (niet de fysieke aansluiting) of Fibre Channel

**Logische drives**

* Drive arrays moeten aangemaakt en aangeboden worden aan de host (server)
* Gebruik Windows Disk Manager om drives te initialiseren en formatteren
  + Wijs drive letters en naam toe aan elke LUN
  + Gebruik hierbij je naamgeving schema
* Nodige directories op deze drives aanmaken voor de installatie
* Testen van de performantie van je logische drives met CrystalDiskMark

**Database bestanden**

* Database in SQL Server bestaat uit >= 2 bestanden
  + MDF = Primary Data File (1x) Bevat database metadata en eigenlijke data
  + NDF = Secondary Data File Bevat eigenlijke data, meerdere bestanden kunnen bestaan
  + LDF = Transaction Log File Bevat historiek van alle data gerelateerde bewerkingen
* **Belangrijke instellingen van bestanden:**
* File type
  + ROWS Databestand (.mdf standaard aanwezig, extra bestanden .ndf)
  + LOG Transaction log bestand (Best slechts 1 per database!)
* Size
  + Initiële grootte van bestand op schijf
  + Deze ruimte wordt gealloceerd en kan dan gevuld worden
* Autogrowth
  + Geeft aan met hoeveel ruimte het bestand mag vergroot worden
  + Kan in MB of percentages ingesteld worden, inclusief limiet
* Path en filename



**Performance tweaks**

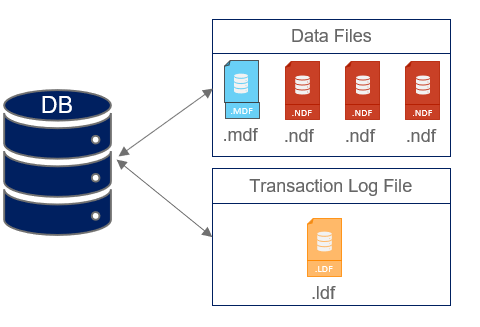
**I/O opsplitsen**

* I/O opsplitsen voor betere performantie en betrouwbaarheid
* Databases, transaction log, OS op aparte storage hardware plaatsen

**Database bestanden**

* Hoeveel Data Files gebruiken in een database?

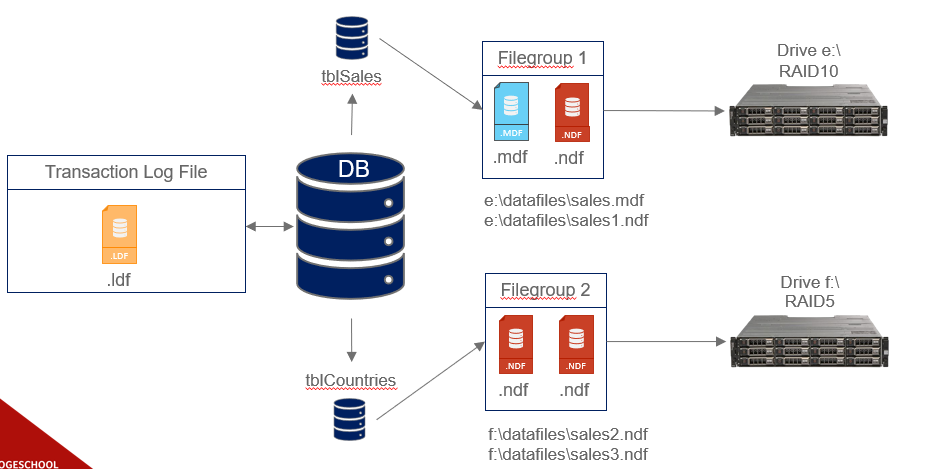
=> Hanteer verhouding 1 datafile per 2-4 processorkernen

* SQL Server Storage Engine regelt wegschrijven van data bij gebruik meerdere files
* Schrijft data proportioneel weg over files (Proportional Fill Algorithm)
  + Elk bestand krijgt data (groter bestand = meer data)
* Datafile groter maken zorgt voor vertraging van database operaties
* Groei zorgt ook voor meer fragmentatie

=> Settings voor filegrowth voldoende hoog instellen om frequente automatische groei te vermijden

**=> Shrinking** toepassen

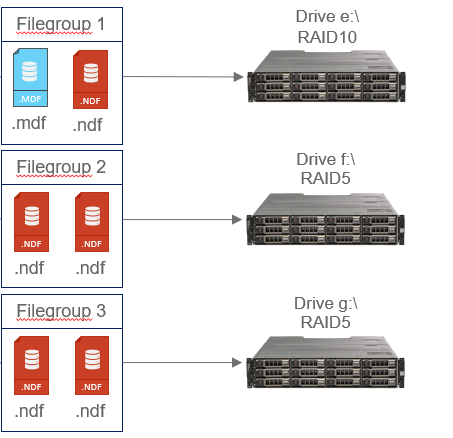
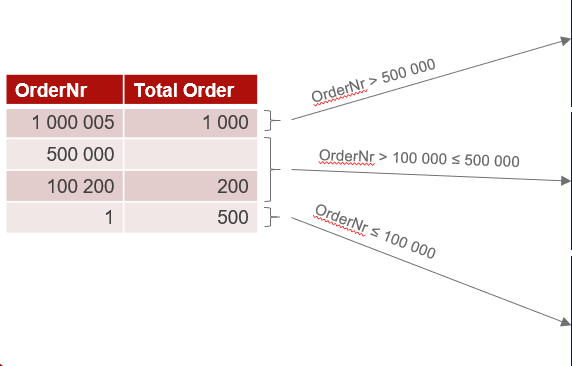
* + Verplaatst volledige pagina’s aan het einde van data file(s) naar de eerste vrije ruimte.
* Beter niet toepassen indien niet noodzakelijk
  + Intensief proces naar I/O toe
  + Kan resulteren in ernstige fragmentatie van indexen waarna deze best opnieuw opgebouwd worden

**Filegroups**

* Elk databasebestand heeft locatie op schijf
* Databasebestand(en) worden toegewezen aan een filegroup
* Filegroup is logische container
* Databasebestanden met eenzelfde locatie kunnen gebundeld worden in 1 filegroup
* Tabel kan niet worden toegewezen aan databasebestand => Wel aan filegroup
* Tabel toewijzen aan filegroup, bij creatie van tabel

**Data partitionering**

* Inhoud van een tabel kan opgedeeld worden in stukken
* Elk stuk op zijn beurt toewijsbaar aan een filegroup
* Op basis van in te stellen regel(s), bv:
  + Datum > 01-01-2021
  + …
* SQL Server past regels toe op data om partitionering door te voeren
* Kan bv. gebruikt worden om meest recente data op performante hardware te plaatsen

**Best practises**

* Configuratie van filegroups en databasebestanden
  + Default configuratie is ok voor kleine databases
  + Tuning en aanpassing voor high-performance databases
* Databasebestanden
  + Pre-configureer de grootte van files
  + Stel waarde van autogrowth voldoende groot in
  + Niet automatisch verkleinen
  + Databasebestanden, transaction log, OS bestanden op fysisch gescheiden volumes
  + Databasebestanden en transaction log op (gescheiden) RAID volumes
  + Creëer bijkomende databasebestanden met identieke grootte

**Deel 3 - Snapshots en backups**

**Snapshots**

* Een (read-only) moment in tijd
* Afhankelijk van de bron database
* Bewaar de toestand van je database op het punt in tijd waarop de snapshot wordt uitgevoerd
* Snel en niet resource intensief
* Meerdere snapshots mogelijk voor dezelfde bron database

**Transacties**

* Belangrijk bij het werken met database is:
  + Performantie
  + Consistent zijn van de database
* Hoe kunnen we hiervoor zorgen?

**Performantie**

* SQL Server leest data van schijf naar werkgeheugen (buffer) met pages van 8 KB
* Verwerking in werkgeheugen sneller dan op disk
  + Lees Operaties en wijzigingen op gegevens in die page in het werkgeheugen gedaan => Bij wijziging wordt page dan als “Dirty” gemarkeerd
  + Aantal dirty pages mogelijk kleiner dan aantal wijzigingen

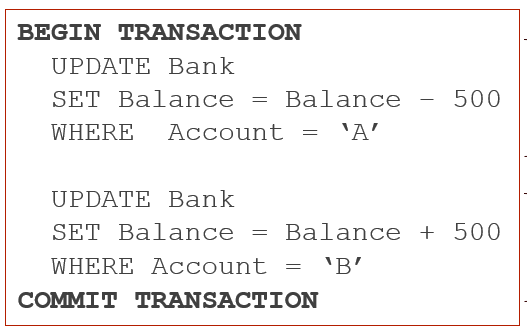
=> Minder I/O operaties (1 page wegschrijven i.p.v. x-aantal wijzigingen)  
=> Betere performantie

* Werken met gegevens in werkgeheugen zorgt voor
  + **Betere performantie**
  + **Groot risico op dataverlies**  
     => Evenwicht vinden tussen performantie en betrouwbaarheid data!

**Consistente database mbv transacties**

* Wijzigingen wegschrijven naar permanent opslagmedium
* Gebruik maken van transacties
  + Record(s) die geaffecteerd worden door INSERT/UPDATE/DELETE instructies
  + Worden in hun totaliteit bewaard (=commit)
  + Ofwel niet waarbij eventuele wijzigingen ongedaan gemaakt worden (=rollback)
* Transactie
  + Kan impliciet (insert/update/delete) of expliciet (begin transaction) gestart worden
  + Garandeert ook relationele integriteit

Vb: Overschrijven $500 van rekening A naar B



* Tijdens de transactie kan de server crashen om diverse redenen
  + Power outage
  + Disk failure
* Dit kan ook gebeuren tijdens het wegschrijven van gegevens
* Hoe oplossen?

=> “Transaction log” file gebruiken

**Transaction log**

* Extra bestand dat bij een database hoort (extensie .ldf)
* DBMS schrijft “gebeurtenissen” naar dit bestand
* Gebeurtenissen worden sequentieel (in volgorde van gebeuren) weggeschreven
* **Gebeurtenissen** zijn:
  + Insert van record(s)
  + Update van record(s)
  + Delete van record(s)
* Transaction log file is nodig voor transactie en database recovery
* **Write-ahead logging:**
  + Eerst wijzigingen wegschrijven naar transaction log
  + Daarna wegschrijven naar database
* Wegschrijven naar transaction log is sequentiële write
  + Sneller dan random writes
* Er wordt een entry in de transaction log geschreven voor:
  + Start en einde van een transactie
  + Rollback operaties
  + Wijziging van data, ongeacht hoe deze tot stand kwam
    - Insert
    - Update
    - Delete
  + Wijziging (create, drop, alter) van tabellen en/of indexen

=> Transacties lopen door elkaar

* Elke entry in de transaction log file krijgt een unieke nummer  
   => Dit noemen we de **Log Sequence Number (LSN)** (numerieke waarde die telkens ophoogt)
* Als transacties door elkaar lopen en door elkaar worden weggeschreven
* Hoe weten we tot welke transactie een entry behoort?

=> Start transactie is beginpunt  
=> Elke transactie krijgt een ID

=> Elke (data-gerelateerde) entry krijgt een verwijzing naar die ID

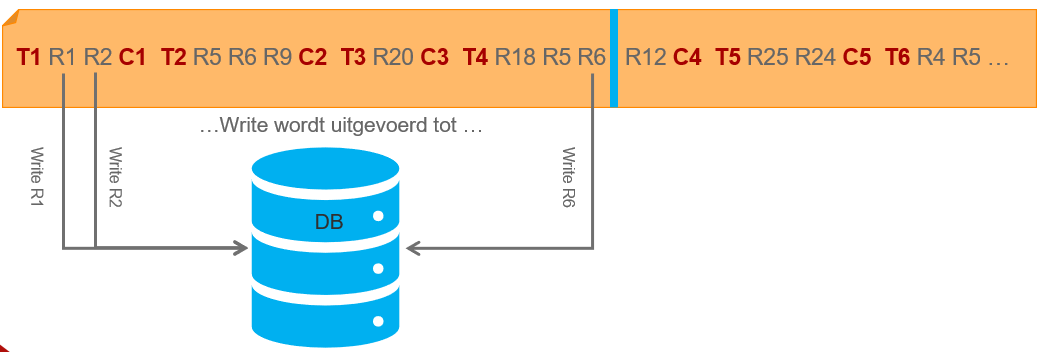
**Durability**

* Ook transaction log file wordt in geheugen geladen
* Writes naar TL worden initieel in geheugen geschreven

=> Risico op dataverlies bij crash

* 2 manieren om hier mee om te gaan
  + **Full transaction durability** (synchroon)
    - Gebruiken indien dataverlies niet tolereerbaar is
  + **Delayed transaction durability** (=asynchroon)
    - Potentieel dataverlies groter dan bij Full transaction durability
    - Gebruiken indien dataverlies minder erg is

**Recovery mechanisme**



* Blauwe lijn markeert een bepaald tijdstip
* Hier beslist SQL Server om alle rijen (R) vanaf vanaf T1 tot blauwe lijn weg te schrijven naar database op disk
* Na de laatste Write, wordt een ijkingspunt toegevoegd aan de transaction log file
  + We noemen dit een Checkpoint
* Het geeft aan vanaf waar we bij de volgende “Write to disk” moeten starten
* Transaction log file is circulair bestand
  + Vullen start aan het begin van de file
* Op regelmatige tijdstippen gebeurt een truncate van de transaction log file
* **Truncate** is verwijderen van niet gebruikte data
* Truncate <> File shrink
* Truncate gebeurt automatisch, tijdstip afhankelijk van recovery model

**Back-up’s**

**Recovery models**

* Recovery Model van een database
  + Heeft invloed op hoe transaction logging kan gebeuren
  + Bepaalt of een back-up van de de transaction log moet/kan gemaakt worden
  + Bepaalt welke restore mogelijkheden er zijn

Drie modellen

* **Simple recovery model**
  + Meest eenvoudige model
  + Beter niet gebruiken indien data-verlies niet tolereerbaar is
  + Laat toe om database te herstellen tot meest recente back-up
  + Voert bij elk checkpoint automatisch een truncate uit van transaction log tot aan oudste niet gecommitte transactie



* **Bulk-logged recovery model**
  + Te gebruiken bij veelvuldig toepassen van bulk-operaties
  + Geen point-in-time recovery (herstellen tot op een bepaald tijdstip)
* **Full recovery model**
  + Point-in-time recovery
  + Transacties met commit worden hersteld
  + Transacties zonder commit worden teruggedraaid

**Types en werking**

* Verschillende back-up types
  + **Full back-up**
  + **Differential back-up**
  + **Transaction log back-up**
* Alle back-up types hangen af van bestaan van full back-up
* Back-ups worden bewaard in een back-up toestel (back-up device)

**Full back-up**

* Traag
* Resource intensief

**Differential back-up**

* De differentiële back-up bewaart alle wijzigingen aan de database sinds de laatste volledige back-up
* Hangt af van het bestaan van de volledige back-up
* Is sneller en kleiner dan een full back-up

**Transaction log**

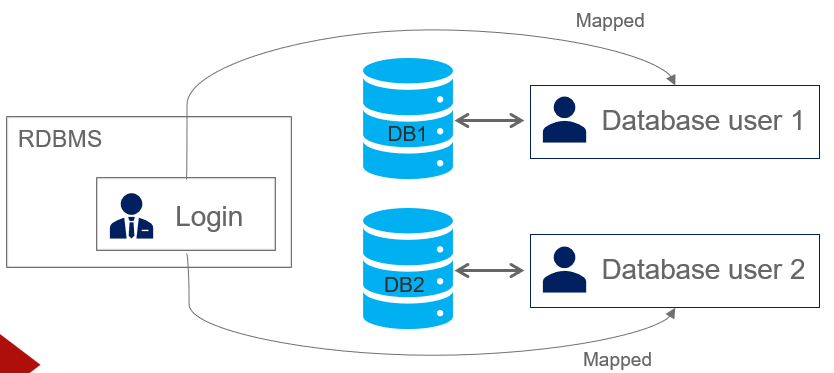
* De Transaction log back-up bewaart secties van de transaction log file
* Laat toe om terug te zetten tot aan het punt van falen of eender welk punt in de tijd (na volledige- of differentiële back-up).
* Hangt af van het bestaan van de volledige back-up.
* Snel
* De Transaction log back-up verwijdert oude transacties van de transaction log file

**Welk type van back-up wanneer te gebruiken:**

* Alleen volledige back-ups
  + Ok voor kleine databases (backup time < 4 uren)
  + Snelle terugzetting
  + Geen leegmaken van Transaction log file
* Volledige back-ups met differentiële back-ups
  + Ok voor grote databases
  + Snelle differentiële back-up
  + Trage terugzetting (Volledig + Diff)
  + Geen leegmaken van Transaction log file
* Volledige, differentiële en Transaction log back-ups
  + Punt-in-tijd terugzetten
  + Leegmaken van Transaction log file
  + Snelle differentiële- en trans- back-up

**Deel 4 - Security**

**Logins**

* Twee belangrijke concepten in security
  + Authenticatie => Wie ben je?
  + Autorisatie => Wat mag/kan je doen?
* SQL Server werkt op 2 niveaus voor authenticatie
  + Niveau van de database server Login
  + Niveau van de database Database user
* SQL Server voorziet 2 soorten logins
  + Windows login
  + SQL login
* Keuze uit 2 authentication modes in SQL Server
  + Windows Authentication mode alleen Windows logins
  + Mixed mode Windows logins en SQL logins

**Windows login**

* Bestaande accounts bewaard in Windows
* Windows zal verantwoordelijk zijn om de authenticiteit van de login te valideren

**SQL Login**

* Accounts bestaan niet in windows

**Server rollen**

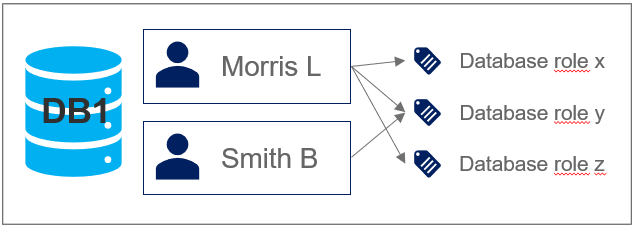
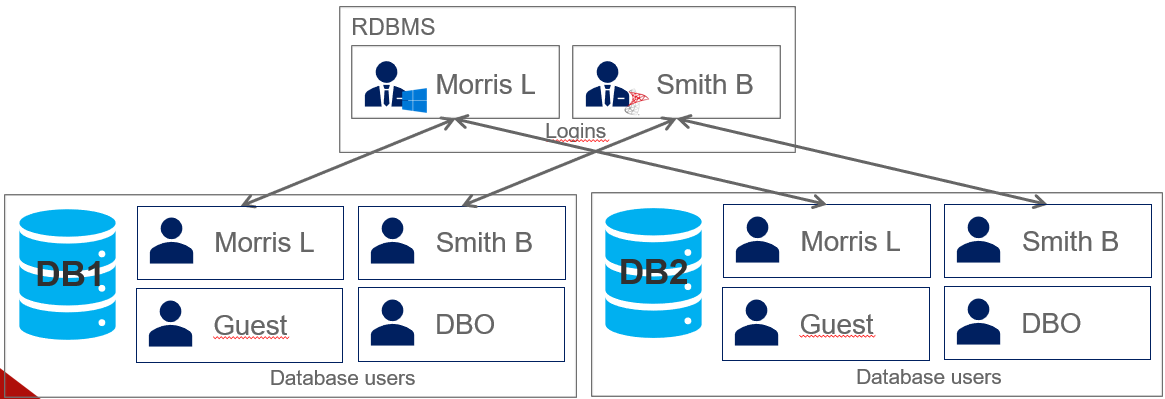
* Een rol is een set van vooraf gedefinieerde permissies
* Server rol bepaalt wat je mag op niveau van de SQL Server instance
* Login kan een of meerdere rollen toegewezen krijgen
* Login erft de permissies van de toegewezen rol(len)
* Een server rol kan fixed of user-defined zijn
  + **Fixed** => Bijhorende permissies kunnen niet gewijzigd worden
  + **User-defined** => Bijhorende permissies kunnen wel geconfigureerd worden

**Vb:**

* **Dbcreator** => Kan eender welke database creëren, droppen, wijzigen en terugzetten
* **Public** => De “Public” rol kan elke database zien (lezen)
* **Securityadmin** => Kan logins en permissies beheren
* **Sysadmin** => Kan eender welke taak uitvoeren in SQL Server

**Database users**

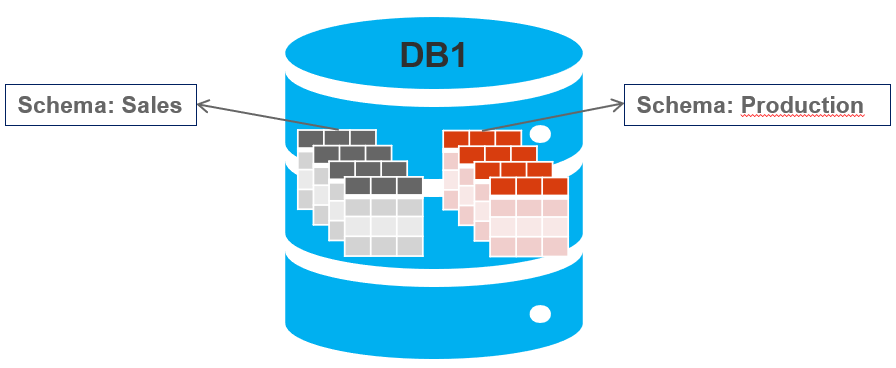
* Worden gedefinieerd binnen een database en bestaan enkel daar
* Worden gebruikt om toegang tot database objecten in te stellen
* Database user is geassocieerd (gemapped) met een login
* Aan 1 login kan voor elke database 1 database user gekoppeld worden
* Authenticatie
  + Is niet van toepassing voor een database user
  + Gebeurt overkoepelend via de Login die geassocieerd wordt met de database user
* By default 2 database users in een database
  + Guest
  + DBO (DataBase Owner)



**Database rollen**

* Bepaalt wat je mag op niveau van de database
* Database user kan een of meerdere rollen toegewezen krijgen
* Een database rol is “Fixed” of “User-defined”
* Ook op dit niveau is er een onderscheid tussen fixed en user-defined database rollen
* **Fixed database rollen** 
  + Zijn minder flexibel om in te stellen wat iemand (niet) mag
* **User-defined database rollen** 
  + Bieden grotere controle over het beheren van permissies en toegang tot resources in de database

**Database schema**

* Container die een aantal database objecten bevat
* Schema heeft een naam (bv. Production)
* Schema heeft slechts één eigenaar (Database user of database Role)
* De volledige naam van een object is = schemaname.objectname  
  Bv. Production.persons

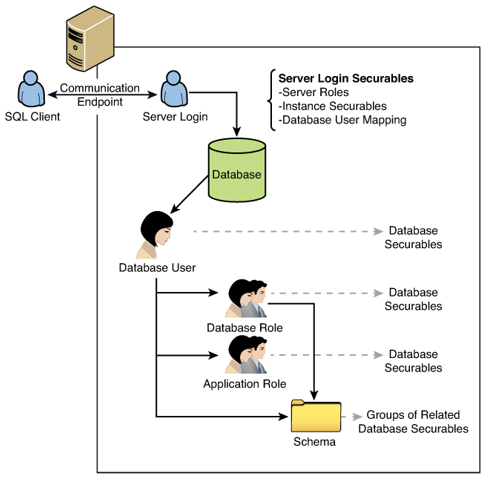
**Expliciet instellen permissies**

**DENY:** Het verbieden van een bepaalde permissie. Hierdoor wordt expliciet aangegeven dat een bepaalde permissie niet toegekend wordt aan een principal.  
Als de permissie impliciet zou worden toegekend zal Deny ervoor zorgen dat dit alsnog overruled wordt.

**GRANT:** Wil zeggen dat je de permissie die je zelf hebt ook kan doorgeven aan anderen.

**REVOKE:** Het intrekken van een reeds toegekende permissie.

**Schematisch overzicht**



**Geef de login/user gepaste permissies om:**

**=>** Zijn/haar werk te kunnen doen

**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

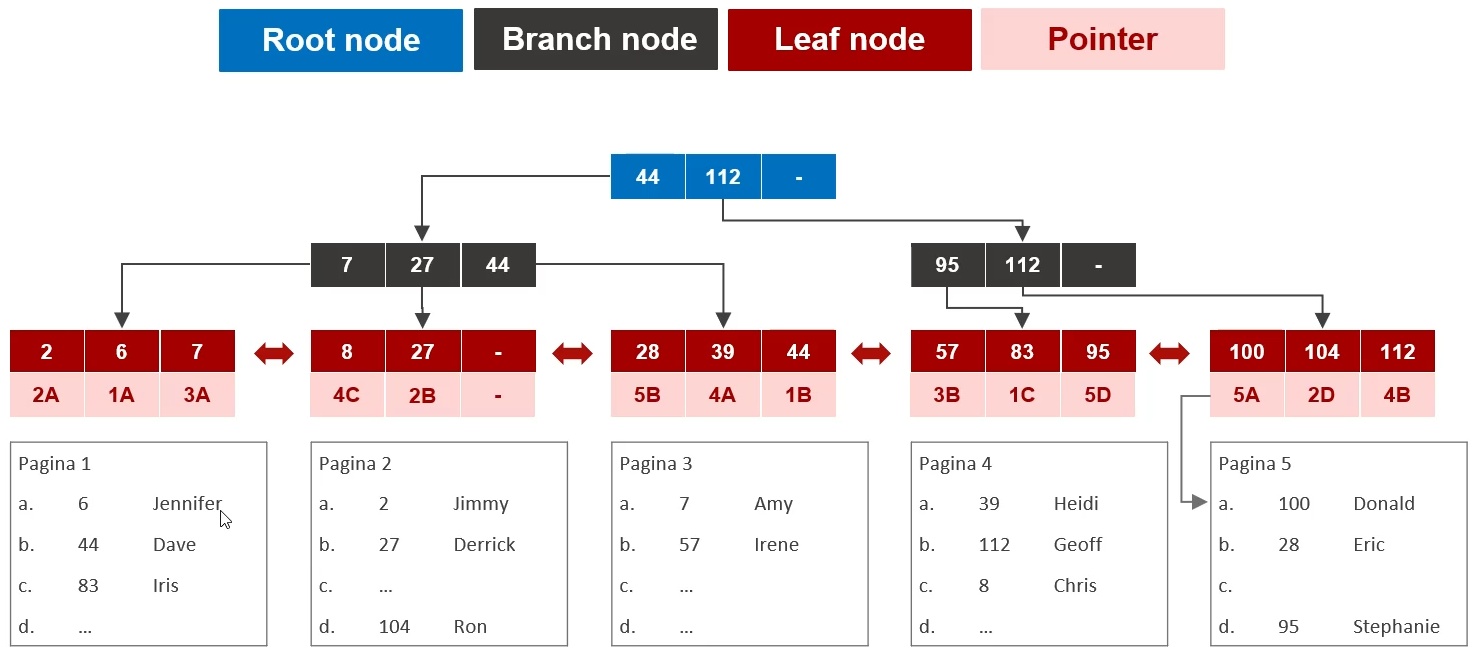
* + worden in SQL Server per 8 gebundeld in een extent

**Indexen - Opbouw**

* Index in 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

=> vergelijken met de index van een boek

* 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

**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 raakt 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 en opslagruimte in de gaten houden
* Errors bekijken
* Potentiële problemen en dingen die geoptimaliseerd kunnen worden identificeren
* Verhogen van efficiëntie, kosten besparen

**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**

* **Processor Time in %** = Percentage van tijd dat alle processors bezig zijn
* **Processor Queue length** = Aantal requests wachtende op de processor
* **Network Interface card(Bytes Total/Sec)** = Snelheid van bytes die worden getransporteerd op de network interface card
* **Physicaldisk: Avg. Disk Queue Length** = Aantal requests wachtende op de disk

**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
    - ...
* 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

**Performantie - Mythes**

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

=> Eén ding is zelden de dader

* + 80 % van de performantie wordt bepaald 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

**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

**Dynamic Management Views**

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

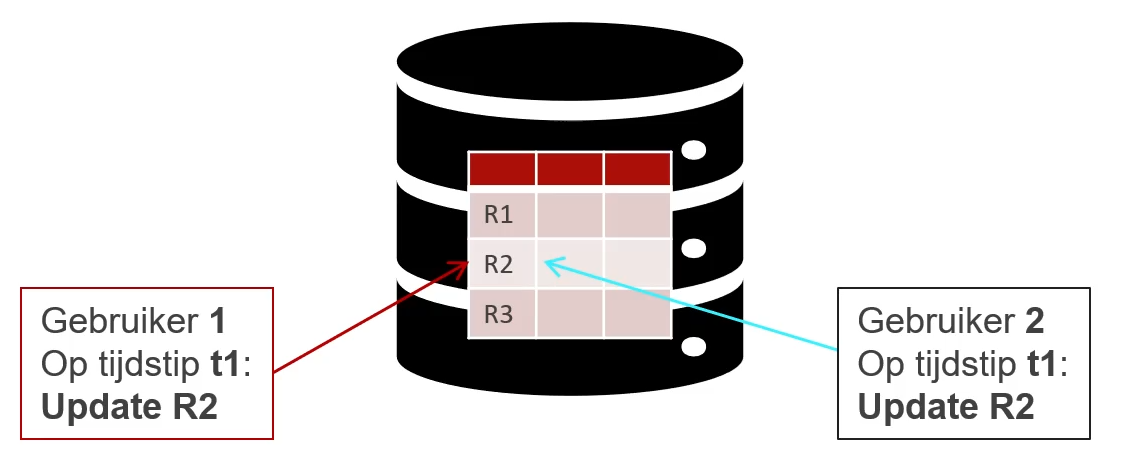
(operating system informatie in relationeel formaat)

**Activity monitor**

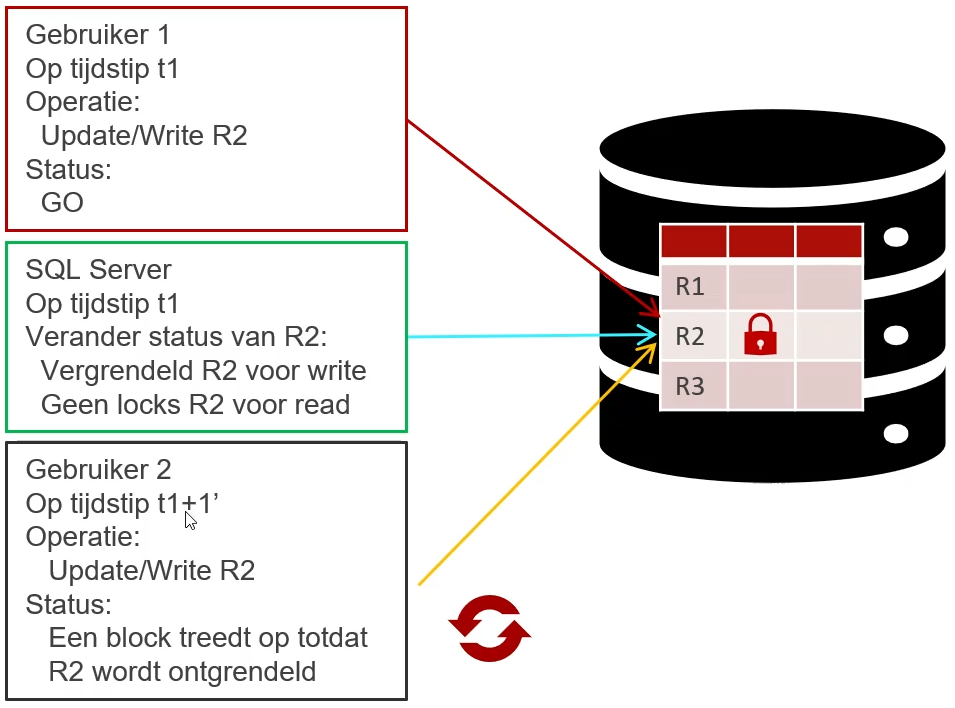
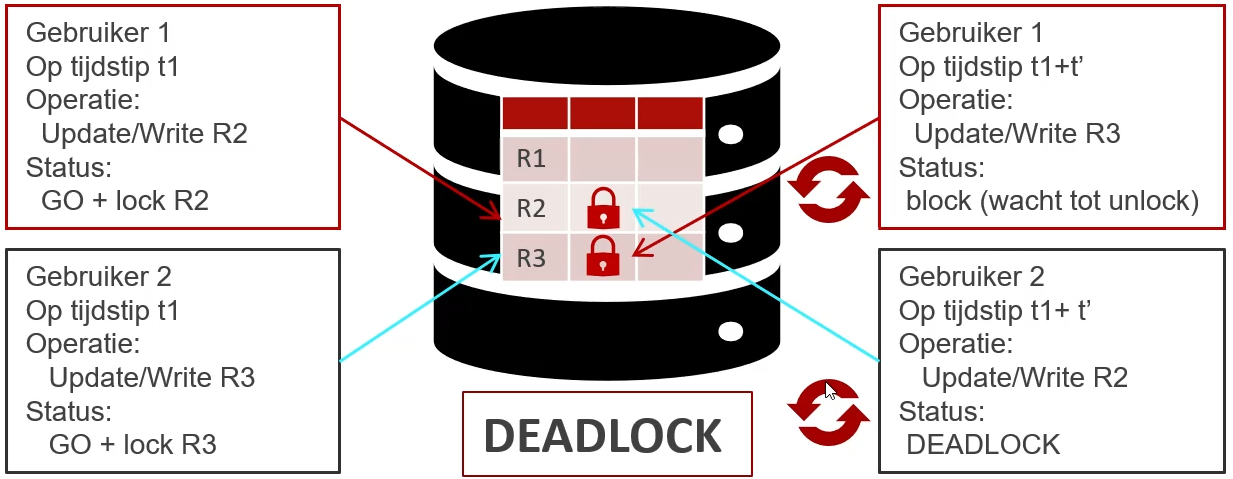
* Bijna real-time, klein grafisch performantie dashboard in MS SQL Server
* Bruikbaar om bv. Deadlocks te detecteren en op te lossen

**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 Deadlock**



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

=> 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

**Systeem stored procedures**

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

**SQL Server Profiler**

* SQL Server Profiler
  + Mogelijkheid om zo goed als elk event 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

**Deel 6 - Jobs en alerts**

**SQL Server Agent**

**Overview**

* Taak scheduler
  + Laat toe om administratieve taken uit te voeren en/of te plannen
* Biedt mogelijkheden toe om je server op te volgen
  + Via alerts
  + Via logbestanden
* Maakt gebruik van 4 elementaire componenten
  + Operator => Degene die op de hoogte gebracht wordt van het resultaat
  + Job => De taken die uitgevoerd moeten worden
  + Alert => Geautomatiseerd antwoord op een gebeurtenis (event)
  + Schedule => Planning die aangeeft wanneer een job uitgevoerd moet worden



**Operator**

* Een gebruiker/groep die berichten (notificaties) kan ontvangen
* Kan berichten enkel ontvangen via mail
* Systeem hangt af van database mail
  + Systeem dat mailservers kan gebruiken om mails door te geven
  + Dient geconfigureerd te worden
  + Dient toegang te hebben tot een smtp server en bijhorende mail account

**Alert**

* instelbaar alarm
* Kunnen aan en uitgezet worden
* Te gebruiken wanneer je op de hoogte wil/moet gehouden worden van bepaalde gebeurtenissen

=> Systeem monitoring via alerts is light-weight en low-overhead

* Drie types alerts
  + SQL Server events
  + SQL Server Performance conditions
  + WMI (Windows Management Instrumentation) events
* Response
  + Wat moet er gebeuren bij optreden gebeurtenis
  + Job uitvoeren en/of operator op de hoogte brengen

**Job**

* Taken moeten regelmatig uitgevoerd worden
* Telkens manueel uitvoeren => tijdrovend, resulteren in vergeten van de taak

**=>** Job maken voor dergelijke taken

**Schedule**

* Voorwaarde(n) instellen om een job te starten zonder interactie gebruiker
* Schedule kan gebaseerd worden op tijdstip of event
  + Tijdstip
    - Eenmalig
    - Terugkerend
    - Automatisch als SQL Server Agent start
  + Event
    - Job uitvoeren op basis van een alert die afgaat bij een event
  + Eén schedule kan gekoppeld worden aan meerdere jobs
* Uitvoering van een geplande job verhinderen:
  + De job disablen
  + De schedule disablen of verwijderen
  + De job loskoppelen van de schedule
  + SQL Server Agent stoppen