Hoofdstuk 1

Databases en databasesystemen

Doelen

Na het doornemen van dit hoofdstuk:

- ben je vertrouwd met een aantal basisconcepten rond databases;
- heb je inzicht in de hoofdcomponenten van een databasesysteem;
- kun je de belangrijkste gebruikersprofielen van elkaar onderscheiden;
- ken je de belangrijkste functies van een databasemanagementsysteem;
- weet je hoe een databasemanagementsysteem is opgebouwd;
- weet je wanneer een databasesysteem gebruikt kan worden en wanneer niet.

Inleiding

Dit hoofdstuk gaat dieper in op databases en databasesystemen. Deze spelen een centrale en vaak cruciale rol bij het geïnformatiseerd gegevensbeheer van veel ondernemingen. Inzicht in de gebruikte terminologie, de ontstaansgeschiedenis en de belangrijkste componenten en ontwikkelingen rondom databasesystemen is onontbeerlijk als basis voor het begrijpen, het opzetten van, het leren omgaan en het werken met databases. Iedereen die te maken heeft met databases en deze beter wil doorgronden – van eindgebruiker tot applicatieontwikkelaar, van databaseontwerper tot ict-manager – is daarom gebaat bij een degelijke basiskennis.

Het hoofdstuk is als volgt opgebouwd: in paragraaf 1.1 worden enkele basisconcepten geïntroduceerd. Daarna geven we in paragraaf 1.2 een kort historisch overzicht van de evolutie van de menselijke behoefte om informatie te conserveren met het oog op later hergebruik. Daarbij geven we aan hoe de hedendaagse databasesystemen zijn ontstaan. In paragraaf 1.3 bespreken we de belangrijkste componenten van een databasesysteem. De belangrijkste component, namelijk het databasemanagementsysteem behandelen we uitvoeriger in paragraaf 1.4. Daarbij besteden we in het bijzonder aandacht aan de functionaliteit en architectuur. Tot slot geven we in paragraaf 1.5 enkele opmerkingen over de keuze voor het al dan niet gebruiken van een databasemanagementsysteem.

1.1 Enkele basisconcepten

Computersystemen worden tegenwoordig gebruikt voor de automatisering van allerlei aspecten van menselijke activiteit. Een belangrijke activiteit, die niet meer weg te denken is uit de huidige informatiemaatschappij, is het automatisch beheren van informatie met het oog op latere verwerking en hergebruik. In eenvoudige gevallen komt dit louter neer op het bijhouden van gegevens voor later onderzoek. In meer geavanceerde toepassingen mondt dit uit in complexe taken voor de ondersteuning van diverse activiteiten binnen een onderneming. Naast het doorzoeken, zijn het toevoegen, verwijderen, aanpassen en consistent houden van de gegevens typische basistaken die voorkomen bij gegevensbeheer. In dit werk maken we een duidelijk onderscheid tussen de termen 'data' (of gegevens) en 'informatie'.

Met **data** bedoelen we 'gegeven feiten'. Hiermee wordt verwezen naar de Latijnse betekenis van het woord 'dare' (=geven) waarvan het woord 'data' is afgeleid. Zo zijn de cijfers, getallen, symbolen, karakters en woorden die je dagelijks tegenkomt verschillende vormen van data.

Voorbeeld 1.1
Voorbeelden van data zijn de woorden 'Rotterdam', 'Vissershuis', 'Monet' en het getal 1882.

Informatie verwijst daarnaast ook naar de betekenis van gegevens: naast de data beschikt men ook over de betekenis die een bepaalde gebruiker of groep van gebruikers aan de data toekent.

Voorbeeld 1.2
Een voorbeeld van informatie is vervat in de zin: 'In het museum Boijmans Van Beuningen in Rotterdam bevindt zich het doek 'Vissershuis' dat in 1882 door Monet werd geschilderd'.

In de context van informatiebeheer is het verschil tussen de concepten 'data' en 'informatie' belangrijk. De huidige computersystemen zijn vrijwel alleen in staat om op een efficiënte manier data te beheren. Dit heeft rechtstreeks geleid tot de term 'database' die we als volgt kunnen omschrijven:

Een **database** is een collectie van persistente data.

Het woord 'persistent' geeft hierbij aan dat de data gedurende een zekere tijd worden opgeslagen in het permanent geheugen van een computersysteem. Daarmee bedoelen we het geheugen waarvan de inhoud niet verloren gaat bij het uitschakelen van het systeem. Voorbeelden van permanent geheugen zijn magneetbandgeheugen, mag-

neetschijfgeheugen en cd-romgeheugen. In normale omstandigheden blijven de data in het permanent geheugen opgeslagen tot ze expliciet, door tussenkomst van een gebruiker of toepassing, worden gewist.

De data in een database kunnen alle gegevens zijn die significant zijn voor het individu of de onderneming die de database wenst op te zetten.

Voorbeeld 1.3
Voorbeelden van databases zijn een patiëntendatabase in een huisartsenpraktijk, een productiedatabase in een bedrijf, een collectiedatabase in een museum, een geografische database in een geografisch informatiesysteem, een bibliotheekdatabase, een ledendatabase van een vereniging en een adressendatabase voor thuisgebruik.

Wat betreft het beheer van de betekenis van de data, met andere woorden het beheer van informatie, is de beschikbare databasetechnologie momenteel nog ontoereikend. Een substantieel deel van het huidig onderzoek rond databases richt zich erop om te komen tot semantisch rijkere datamodellen, met als ultieme doelstelling het ontwikkelen van meer geavanceerde systemen die volwaardige 'informatiebases' kunnen beheren.

Een computersysteem voor het beheer van databases wordt een **database-systeem** genoemd.

Verder beschrijven we in dit hoofdstuk verschillende componenten van een databasesysteem. Bijzondere aandacht gaat naar de architectuur en de werking van een databasemanagementsysteem. Om dit te plaatsen in een context bespreken we de historische ontwikkeling van gegevensbeheer.

1.2 Gegevensbeheer door de eeuwen heen

Gegevensbeheer is een zeer oude activiteit. Vrijwel alle moderne beschavingen danken hun bestaan grotendeels aan hun vermogen om gegevens en kennis fysiek te *registreren* met het oog op conservatie en kennisoverdracht. Sinds de Codex van Hammoerabi (een van de oudste schriftelijke wetten, ongeveer 4000 jaar geleden) waren steen, hout, perkament en papier achtereenvolgens de dominante informatiedragers. Aanvankelijk deed men weinig moeite om de geregistreerde informatie te *structureren*. Het schema van in elkaar sluitende classificaties voor informatie uit het werk 'Fysice akroasis' (Physica) van Aristoteles (384-322 v.Chr.) wordt beschouwd als één van de oudste bekende inspanningen om informatie te ordenen. Stilaan begon de mens het nut te erkennen van het groeperen van data volgens een vaste structuur: Men kan gestructureerde data veel efficiënter doorzoeken en bewaren.

Al bij vroegste pogingen om computers te ontwerpen, zoals bij de 'analytical engine' van Charles Babbage rond 1830, maakte men plannen om niet alleen de

machine-acties te programmeren, maar ook de data volgens een voorgedefinieerde structuur te rangschikken. Deze ideeën werden vrijwel onmiddellijk met succes in de praktijk toegepast: in het begin van de negentiende eeuw werden de besturingsinstructies van het automatisch Jacquard-weefgetouw voorgesteld door gestructureerde perforaties in papieren kaarten. Herman Hollerith gebruikte ook soortgelijke kaarten bij de machine die hij ontwierp voor de statistische verwerking van de Amerikaanse volkstellingsgegevens van 1890. Een nieuwe gegevensdrager, de geperforeerde kaart of ponskaart, was 'geboren'.

Het duurde tot de jaren 1960-1970 voordat ponskaarten geleidelijk aan werden verdrongen door *sneller toegankelijke* en gemakkelijker te hanteren magnetische gegevensdragers met veel *grotere opslagcapaciteit*: de magneetbandgeheugens en de magneetschijfgeheugens. Men moet de magneetbanden door hun fysieke beperkingen steeds sequentieel (één voor één) doorzoeken. Hierdoor zijn ze minder efficiënt bij toepassingen die elk willekeurig gedeelte van de data in willekeurige volgorde moeten kunnen verwerken. Magneetschijven kun je efficiënter adresseren, maar waren aanvankelijk veel duurder. In dezelfde periode kwam ook een einde aan het 'predatabasetijdperk'. Met de komst van de magnetische gegevensdragers begon men de voordelen te zien van het gebruik van een *'tweelagen'-systeemarchitectuur* met aparte datastructuren voor de opslag (interne laag) en het verwerken (logische laag) van de data. De fysieke eigenschappen van de gebruikte gegevensdrager bepaalt immers in belangrijke mate de structuur waarin de data fysiek worden weggeschreven en is daarom doorspekt met technische details die de verdere verwerking van de data bemoeilijken. Het gebruik van een extra, abstractere (logische) datastructuur helpt om dit euvel te voorkomen.

De 'tweelagen'-systeemarchitectuur effende het pad voor de eerste volwaardige databasesystemen en databasetoepassingen: omdat data op een uniforme, logische manier werden weergegeven, konden databasesystemen groeien naar systemen die data uit verschillende toepassingen op een uniforme wijze beheren. Tot de eerste grote databasetoepassingen uit die periode behoren het vluchtreserveringssysteem 'Sabre' van American Airlines (1964) en de bibliografische databases van de NASA.

Een volgende technische mijlpaal kwam er in de jaren 1970–1980 met de *'drielagen'-systeemarchitectuur* [Ans75, Tsi78]. In deze architectuur wordt er, naast de interne en logische laag, nog een derde, zogenaamde externe laag beschouwd. De externe laag kent diverse presentatievormen van de logische data en laat het toe om deze data 'op maat' weer te geven volgens datastructuren die het best passen bij de individuele behoeften van eindgebruikers en toepassingsprogramma's. De 'drielagen'-systeemarchitectuur beschrijven we gedetailleerder in paragraaf 1.4.3 .

Geïnspireerd door de programmeerconcepten van Edsger Dijkstra – software en data moeten worden gecreëerd volgens gestructureerde standaardvormen, zodat ontwikkelaars kunnen voortbouwen op het werk van anderen – werden databasesystemen in het begin van de jaren tachtig voor het eerst uitgerust met een eigen *datadefinitie-* en *datamanipulatietaal.* In juni 1983 kondigde IBM het eerste commerciële relationele databasesysteem aan, DB2 [Dat84]. Dit systeem was gebaseerd op het relationeel databasemodel dat in 1970 werd voorgesteld door Edgar (Ted) Codd, een toenmalige onderzoeker aan het 'IBM San Jose Research Laboratory'

[Cod70, Cod71]. Sindsdien zijn relationele systemen gegroeid tot de meest gebruikte en succesvolste databasesystemen. Volgens het relationeel databasemodel worden databases logisch gezien als verzamelingen van samenhangende tabellen. Alle relationele databasesystemen werken met een datadefinitie- en datamanipulatietaal die een 'dialect' is van de gestandaardiseerde *SQL-taal*.

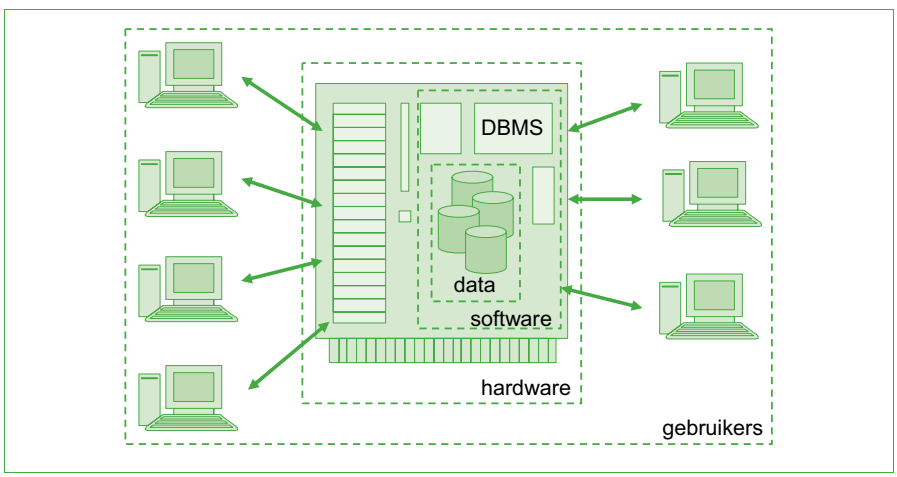
Tegenwoordig zijn databases niet meer weg te denken. Het is niet verwonderlijk dat er voortdurend verder wordt gezocht naar nieuwe technieken voor het efficiënter beheren van nog grotere databases, met nog betere faciliteiten om de informatie die erin besloten ligt te exploiteren. De technologische vooruitgang heeft zo geleid tot tal van nieuwe technieken, waaronder:

- *Gedistribueerde databases* die gebruikmaken van communicatienetwerken om data te verspreiden en te repliceren over verschillende locaties.
- *Multimediadatabases* die diverse multimedia, zoals audio, beeld en video, kunnen beheren.
- *Geografische databases* voor het beheer van cartografische data.
- *Datawarehouses* die erop gericht zijn om zeer grote datavolumes, vaak met een historisch karakter te beheren en te analyseren.
- *Dataminingtoepassingen* die bedoeld zijn om voorheen onbekende informatie te ontginnen uit een database of datawarehouse.

1.3 Databasesysteem

We hebben in paragraaf 1.1 al kort aangegeven dat een databasesysteem een computersysteem is dat tot doel heeft data te bewaren en dat gebruikers of toepassingssoftware toelaat om deze data te manipuleren en te doorzoeken.

In een **databasesysteem** onderscheiden we globaal genomen drie *hoofdcomponenten*: de *hardware*, de *data* (inhoud) en de *software*. Daarnaast is het belangrijk om ook de gebruikers van het systeem te beschouwen. In figuur 1.1 zijn deze schematisch voorgesteld.



Figuur 1.1 Hoofdcomponenten en gebruikers van een databasesysteem

1.3.1 Hardware

Hoewel het zeker niet de bedoeling is om in dit boek veel aandacht te besteden aan hardwareaspecten, bespreken we toch in het kort de belangrijkste hardwarecomponenten binnen een databasesysteem.

Essentieel zijn de *central processing unit* (CPU) en het *computergeheugen* waarin de database wordt opgeslagen. De CPU kan fysiek bestaan uit één of meer *processoren* en zorgt voor het uitvoeren van de software-instructies. Om redenen van onder meer efficiëntie en implementeerbaarheid is het geheugen van een computer hiërarchisch georganiseerd. Daarbij onderscheiden we twee hoofdcategorieën:

Primair geheugen. Geheugens uit deze categorie zijn rechtstreeks aanspreekbaar door de CPU. Het primaire geheugen staat fysiek het dichtst bij de CPU en is daarom het snelst toegankelijk. De geheugencapaciteit is echter beperkt.

Tot het primaire geheugen behoren:

- *Statisch RAM-geheugen (Random Access Memory)*. Statisch RAM-geheugen wordt ook wel het cachegeheugen genoemd en wordt door de CPU in hoofdzaak gebruikt om de uitvoering van programma's te versnellen. Statisch RAM-geheugen is het snelste, doch tegelijk het duurste geheugen binnen een databasesysteem.
- *Dynamisch RAM-geheugen.* Dit geheugen is het belangrijkste werkgeheugen voor de CPU en noemen we daarom ook wel het hoofdgeheugen. We gebruiken het voor de tijdelijke opslag van data en programmacode. Het voordeel van dynamisch RAM-geheugen is de lagere kostprijs. Nadeel is echter de tragere werking vergeleken met statisch RAM-geheugen.

Vaak worden er in het primaire geheugen minstens twee zogenaamde **databasebuffers** opgezet. Terwijl de inhoud van de ene buffer wordt ingelezen of weggeschreven, kan de CPU tegelijkertijd (parallel) de data uit de andere buffer verwerken. Een direct gevolg hiervan is dat de hoeveelheid data die per lees- of schrijfoperatie wordt overgedragen van en naar het primaire geheugen gelijk is aan de grootte van een databasebuffer.

Het primaire geheugen heeft een tijdelijk karakter. Hierdoor is het niet bedoeld als hoofdopslagmedium voor databases. Daar een dynamisch RAM-geheugen van 8 tot 16 gigabytes op één enkel computersysteem geen uitzondering is, bestaan er databases die integraal in het RAM-geheugen zijn opgeslagen. Vaak gaat het daarbij om toepassingen waarbij extreem hoge overdrachtsnelheden zijn vereist, wat bijvoorbeeld het geval is bij databases voor telefooncentrales. Uiteraard dienen bij dergelijke databases zeker veiligheidsvoorzieningen tegen dataverlies te worden getroffen, zoals het bijhouden van een kopie van de database in een permanent geheugen.

Algemeen geldt dat hoe meer dynamisch RAM-geheugen je voorziet en hoe meer werkruimte (voor databasebuffers) je hierin voor het databasesysteem reserveert, hoe beter de prestaties van het databasesysteem zullen zijn. Hierop komen we nog kort terug aan het einde van paragraaf 6.5.2.

Een speciale vorm van geheugen dat sommigen eveneens tot het primaire geheugen rekenen, is het zogenaamde flashgeheugen, waarbij EEPROM-technologie (Electrically Erasable Programmable Read-Only Memory) wordt gebruikt. Voor-

beelden van (kleine) databases op flashgeheugen zijn de datacollecties in mp3-spelers, digitale fototoestellen, pda's en camera's.

Secundair geheugen. Gegevensdragers uit deze categorie hebben gewoonlijk een veel grotere opslagcapaciteit, zijn goedkoper, maar hebben een veel tragere gegevenstoegang. Data in het secundaire geheugen kunnen niet rechtstreeks worden verwerkt door de CPU, maar moeten hiertoe eerst naar een databasebuffer worden gekopieerd. Deze datastromen zijn gevisualiseerd in figuur 1.2.

Tot het secundaire geheugen behoren:

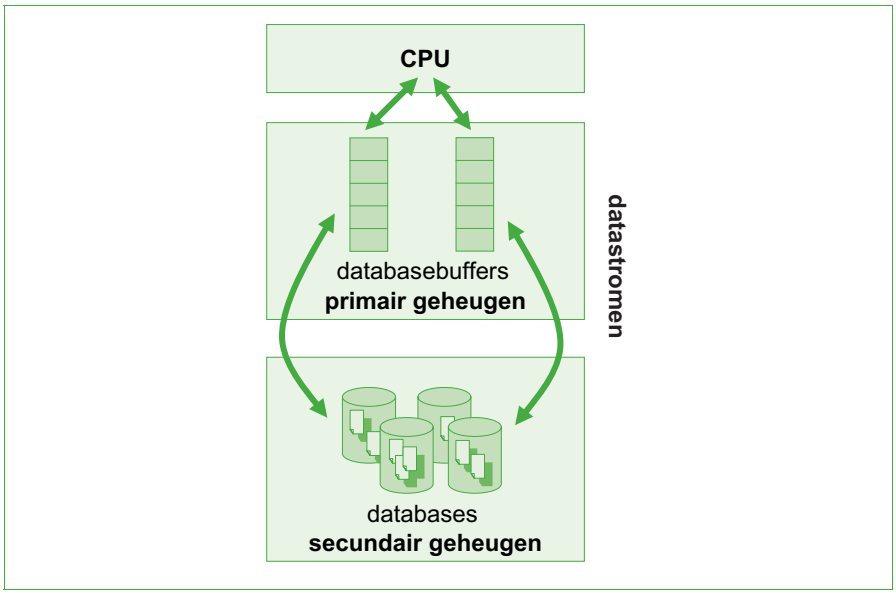
- *De magneetschijfgeheugens*. Deze geheugens zijn opgebouwd uit één of meer roterende schijven die langs één of twee zijden beschrijfbaar zijn. Elke schijf is door de fabrikant onderverdeeld in cirkels die we *tracks* noemen. Bij verschillende schijven vormen de tracks die zich op een gelijke afstand van het middelpunt van de schijf bevinden een *cilinder*. De tracks zijn op hun beurt onderverdeeld in sectoren.

  Bij het formatteren, verdeelt het besturingssysteem, bijvoorbeeld Windows, Linux of Unix, de sectoren onder in **pagina's** van gelijke grootte. Elke pagina bestaat uit een vast aantal sectoren. De databasebuffers uit het primaire geheugen worden even groot gekozen als de grootte van een pagina. Daardoor wordt per lees- en schrijfoperatie één pagina gekopieerd.

  Om een pagina te kunnen lezen of beschrijven moet deze vooraf worden gelokaliseerd. Hierbij moet rekening worden gehouden met drie tijdspannes: de tijd die nodig is om de lees/schrijfkoppen van het opslagmedium te verplaatsen naar de juiste *track* of cilinder, de tijd die nodig is om de schijf te roteren tot aan het begin van de eerste sector van de betreffende pagina en de tijd die nodig is om de data over te brengen. De eerste twee tijdspannes duren vaak veel langer dan de derde en vormen een prestatie-'bottleneck'. Om prestatieredenen is het belangrijk dat we de data in de schijfbestanden zodanig organiseren dat het aantal paginatransfers minimaal wordt gehouden.

  Bij de bestandsorganisatie maken we een onderscheid tussen primaire en secundaire bestandsorganisatie. **Primaire bestandsorganisatie** heeft betrekking op de manier waarop een database fysiek wordt opgeslagen. De data kunnen daarbij bijvoorbeeld al dan niet geordend worden. **Secundaire bestandsorganisatie** heeft tot doel de toegang tot de fysiek opgeslagen database verder te optimaliseren door extra data over bijvoorbeeld de locatie van gegevens en verbanden tussen gegevens bij te houden. Indexen zijn voorbeelden hiervan, we bespreken ze verder in hoofdstuk 4.

  Magneetschijfgeheugens worden momenteel het meest gebruikt als secundair geheugenmedium voor databases.
- *De cd-rom en dvd-geheugens.* Dit zijn recentere opslagmedia. In configuraties waarbij verschillende van dergelijke geheugens worden samengebracht in een soort 'jukebox'-systeem kunnen grote opslagcapaciteiten worden bereikt. Door hun minder goede schrijffaciliteiten zijn cd's en dvd's eerder geschikt voor de opslag van databases die enkel moeten kunnen worden geraadpleegd. Voorbeel-

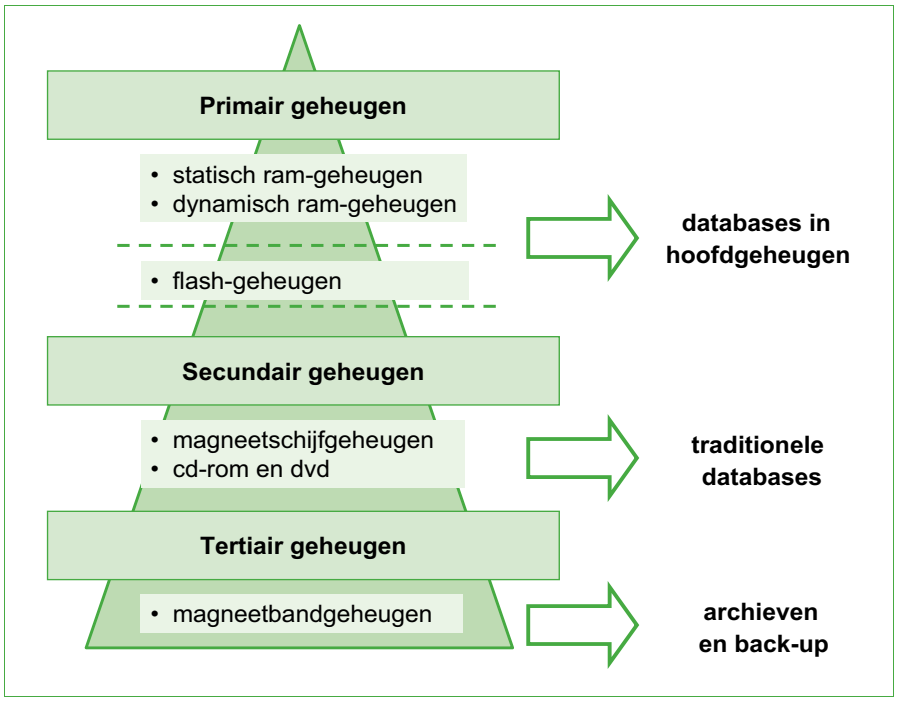


Figuur 1.2 Datastromen van het secundaire en primaire geheugen naar de CPU en vice versa

den zijn digitale telefoongidsen, digitale archieven, geografische databases en digitale woordenboeken en encyclopedieën. Een belangrijk minpunt is de beperkte duurzaamheid van zelfbeschreven cd's en dvd's, waardoor we hun bruikbaarheid voor kritische toepassingen op zijn minst in twijfel moeten trekken.

- *De magneetbandgeheugens.* Magneetbandgeheugens worden in hoofdzaak gebruikt voor archivering en back-up. Omwille van hun fysieke karakteristieken werken magneetbandgeheugens alleen efficiënt bij sequentiële toegang (in volgorde van fysieke opslag) tot de data. Ook hier kan een soort 'jukebox'-systeem worden gebruikt om de opslagcapaciteit te vergroten. Een voorbeeld van een database op magneetbandgeheugen is het EOS-archief (Earth Observation Satellite) van de NASA. Sommige auteurs categoriseren magneetbandgeheugens als tertiair geheugen.

In figuur 1.3 stellen we de hiërarchie van het computergeheugen voor. De magneetbandgeheugens zijn hierin apart vermeld als tertiair geheugen. Aan de top van de hiërarchie staat het snellere, duurdere geheugen met doorgaans kleinere opslagcapaciteit. De basis wordt gevormd door het tragere, minder snel toegankelijk geheugen met een grotere opslagcapaciteit.



FIGUUR 1.3 De geheugenhiërarchie

1.3.2 Data

Centraal in een databasesysteem staan de databases. Een database wordt meestal in het geheugen gestructureerd opgeslagen als een collectie van records. Elk **record** is opgebouwd uit één of meer velden die data kunnen bevatten. Deze velden zijn vastgelegd in het **recordtype** van het record en worden elk gekarakteriseerd door een naam en een datatype. Analoog als bij programmeertalen legt het datatype de toegelaten waarden en operatoren voor het veld vast. Naast velddefinities heeft een recordtype ook een naam. Doorgaans bevat een database verschillende records die gegroepeerd zijn volgens diverse recordtypes. In het secundaire geheugen worden de records opgeslagen in **bestanden**. Meestal bevat een bestand een aantal records. Hoewel dit geen vereiste is, kunnen er in hetzelfde bestand records van verschillende recordtypes zijn opgeslagen.

In de praktijk kan het aantal records in een database variëren van enkele tot miljoenen. Daarnaast kan een databasesysteem tegelijkertijd instaan voor het beheer van *meerdere* databases.

Voorbeeld 1.4

Figuur 1.4 bevat een voorstelling van de recordtypes en records uit een voorbeeld-database voor schilderkunst. De database bestaat uit een collectie van records van drie recordtypes: 'Schilderij', 'Artiest' en 'Eigenaar'. Omwille van de leesbaarheid zijn de records geordend per recordtype en worden de veldnamen voorafgaand aan de records herhaald. De gebruikte datatypes zijn:

- CHAR(n) voor het modelleren van karakterstrings van lengte n.
- INTEGER voor het modelleren van gehele getallen (die geïnterpreteerd worden als jaartallen).
- REAL voor het modelleren van reële getallen.

RECORDTYPE Schilderij (ID:**CHAR**(3); Naam:**CHAR**(30); Artiest:**CHAR**(30); Periode:**INTEGER**; Waarde:**REAL**; Eigenaar:**CHAR**(30))

ID	Naam	Artiest	Periode	Waarde	Eigenaar
S01	Vissershuis	Monet	1882	16.000.000	Boijmans
S02	De balletles	Degas	1872	8.500.000	Louvre
S03	Mona Lisa	Da Vinci	1499	75.000.000	Louvre
S04	Namiddag te Oostende	Ensor	1881	200.000	KMSK

RECORDTYPE Artiest (Naam:**CHAR**(30); Voornaam:**CHAR**(20); Geboren:**INTEGER**; Gestorven:**INTEGER**)

Naam	Voornaam	Geboren	Gestorven
Da Vinci	Leonardo	1452	1519
Degas	Edgar	1834	1917
Ensor	James	1860	1949
Monet	Claude	1840	1926

RECORDTYPE Eigenaar (Naam:**CHAR**(30); Plaats:**CHAR**(20); Land:**CHAR**(20))

Naam	Plaats	Land
Boijmans	Rotterdam	Nederland
Louvre	Parijs	Frankrijk
KMSK	Antwerpen	België

FIGUUR 1.4 Records uit de voorbeelddatabase 'Schilderkunst'

Besturingssystemen zoals MVS, Windows, Linux en Unix, bieden geen oplossing voor het vastleggen en het behandelen van de soms complexe verbanden die kunnen bestaan tussen de data in een bestand. Als ze voorkomen, zijn dergelijke verwantschappen echter zeer belangrijk voor de gebruikers en toepassingsprogramma's: ze verduidelijken de semantiek van de verschillende entiteiten uit het toepassingsdomein.