I. INTRODUCTIE

I.1. INLEIDING

Database management systemen (DBMSen) zijn sedert het begin van de jaren zeventig bekende produkten. Deze systemen worden op grote schaal gebruikt voor het beheer van omvangrijke hoeveelheden gegevens in computersystemen. Onderling zijn grote verschillen in mogelijkheden, betrouwbaarheid, performance en gebruiksvriendelijkheid aan te geven.

De oude DBMSen verschaffen de gebruiker een netwerk van aan elkaar geknoopte gegevens. De gebruiker moet in computerprogramma's door al deze gegevens zich een weg weten te banen. De hiervoor benodigde informatica-kennis vormt een hindernis voor menig database gebruiker.

Sinds enige jaren verschijnen er DBMSen waarin deze bezwaren zijn weggenomen. Eenvoud wordt bereikt door de beschikbaarheid van slechts één gegevensrepresentatie: de tabel. Een hiermee samenhangend bezwaar is het gebruik van nieuwe structureringssymbolen. Iedere gebruiker moet hiervan op de hoogte zijn en precies weten hoe ze te hanteren. De betrouwbaarheid en performance komt hierdoor vaak in het gedrang.

Bij de meest recente ontwikkelingen in het vakgebied gaat men uit van een **semantisch model**. In zo'n model wordt betekenis toegekend aan de samenhang die er tussen gegevens bestaat. Met deze kennis is een DBMS in staat die samenhang te bewaken. Een voordeel hiervan is dat de gebruiker op de meest natuurlijke wijze zijn toepassingen kan formuleren.

XPLAIN is het eerste DBMS dat gebaseerd is op dit semantisch concept. Naast gebruiksvriendelijkheid hebben betrouwbaarheid, flexibiliteit en efficiency voorop gestaan bij het ontwerp. Het semantisch model heeft het mogelijk gemaakt zoveel mogelijk kennis over een toepassingsgebied door het systeem te laten beheren. Dit maakt het systeem aanzienlijk eenvoudiger te gebruiken voor zowel de geoefende als de beginnende gebruiker.

XPLAIN is een researchprodukt van de TU Delft. Theoretisch onderzoek op dit gebied is begonnen in het midden van de jaren zeventig op een moment dat de eerste tekortkomingen van de toenmalige DBMSen zichtbaar werden. Praktijkervaringen hebben daarna een grote invloed gehad op de theorievorming. Nadat de theorie was voltooid met de definitie van een volledige databasetaal, kon een aanvang worden gemaakt met de realisatie van het systeem. In maart 1984 en juli 1985 kwamen vervolgens de eerste experimentele Unix versies van het systeem gereed. Deze versies zijn van meet af aan getoetst aan praktijksituaties bij overheid en bedrijfsleven. In mei 1987 werd besloten tot commerciële beschikbaarstelling van het produkt. In februari 1996 verscheen versie 5.1 ook onder het Linux bedrijfssysteem. In januari 1999 is versie 5.7 het eerste DBMS met uitgebreide mogelijkheden voor recursieve queries.

Momenteel is XPLAIN meer dan een DBMS, het is een geavanceerd software ontwikkelings-gereedschap. Voornamelijk door de gebruiksvriendelijkheid, het interactieve karakter, het gebruik van de opgeslagen kennis en de geïntegreerde tools biedt het systeem een aanzienlijke besparing bij het ontwerpen en onderhouden van computertoepassingen.

I.2. KARAKTERISTIEKEN

XPLAIN is een flexibel en efficiënt database management systeem, gebaseerd op de nieuwste ontwikkelingen. Dit systeem maakt het mogelijk om snel informatie af te leiden uit een omvangrijke verzameling van geïntegreerde gegevens.

Het DBMS is te gebruiken zonder te beschikken over diepgaande specialistische kennis. Het systeem is gebruiksvriendelijk en biedt de mogelijkheid wensen op een zo natuurlijk mogelijke manier te formuleren. Afhankelijk van de ervaring van de gebruiker kan voor datadefinitie gekozen worden tussen de dialoogvorm en de commandovorm (DDL). XPLAIN bevat tevens on-line informatie (HELP) over het gebruik.

Het DBMS biedt de mogelijkheid om de werkelijkheid zo getrouw mogelijk weer te geven. Aanpassingen als gevolg van veranderde omstandigheden zijn met weinig moeite aan te brengen.

Het DBMS beschikt over een complete manipulatietaal waarmee de meest complexe gegevensmanipulatie door middel van enkele eenvoudige opdrachten kan worden aangegeven.

Het DBMS bevat een interactieve panelgenerator waarmee complexe gebruikersgerichte schermen voor on-line applicaties, visueel kunnen worden gedefinieerd. Het moeizame en kostbare programmeren van correct werkende applicaties inclusief alle integriteitscontroles is hiermee overbodig geworden. Eventueel kan het systeem zelf de schermindeling bepalen zodat de gebruiker bijna niets meer hoeft te doen om een complete applicatie te maken.

Het DBMS levert door toepassing van moderne opslag- en file management technieken een hogere prestatie dan vergelijkbare systemen, zonder dat aan de betrouwbaarheid afbreuk wordt gedaan. Resultaten van gegevensmanipulaties zijn altijd snel beschikbaar.

Het DBMS ondersteunt meerdere gebruikers die ieder toegang kunnen krijgen tot meerdere databases. Een database kan echter slechts door één gebruiker tegelijkertijd gebruikt worden.

Het DBMS vereist geen speciale hardware. Het DBMS vereist enkel het UNIX (System V R4) of LINUX bedrijfssysteem. De gebruikersinterface maakt gebruik van de gestandaar-diseerde VT-100 terminal aansturing. Dit maakt het DBMS geschikt voor een grote klasse van computersystemen. Momenteel is XPLAIN beschikbaar voor werkstations van HP (9000-serie), SUN (Solaris), DEC, en IBM. Daarnaast is XPLAIN middels Linux ook beschikbaar op diverse x86 PC’s.

I.3. THEORETISCHE ASPECTEN

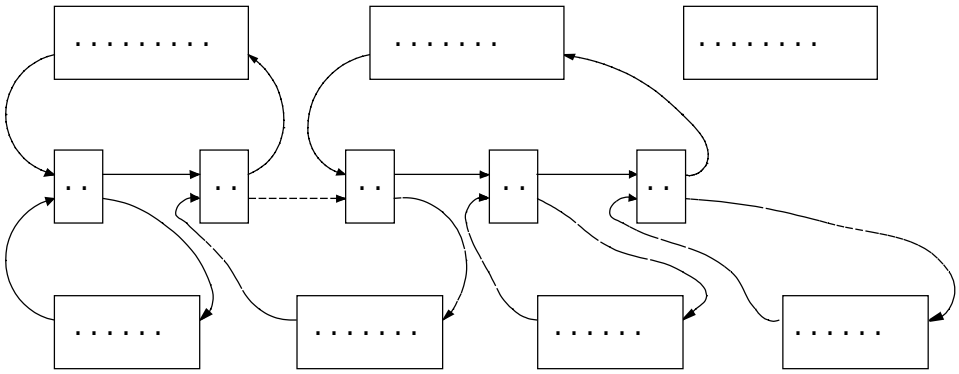
I.3.1. Inleiding

XPLAIN is het eerste DBMS dat gebaseerd is op het semantisch model. Omdat het systeem ontwikkeld is in de jaren tachtig, zijn de meest moderne technieken hierin toegepast. Hierdoor is het systeem gebruiksvriendelijker, flexibeler, efficiënter en betrouwbaarder dan systemen gebaseerd op oudere of verouderde database benaderingen.

Alvorens in te gaan op de architectuur van het DBMS zal een indruk gegeven worden van de kracht van de gegevensstructurering en -manipulatie in het semantisch model. Voor dit doel zullen alleen de belangrijkste verschilpunten tussen de diverse database benaderingen aan de orde worden gesteld.

I.3.2. Benaderingen

DBMSen worden voornamelijk gekarakteriseerd door de wijze waarop gegevens aan de gebruiker worden gerepresenteerd. Belangrijke benaderingen uit het verleden waren de **netwerk- en** de **relationele benadering**. Bij een netwerk DBMS worden gegevens aan elkaar geknoopt.



Een belangrijk nadeel van de netwerkbenadering is de expliciete specificatie van navigatiepaden. De hierdoor ontstane complexiteit maakt toepassing van dit model niet eenvoudig.

In de relationele benadering bestaat slechts één gegevensstructuur. Alle gegevens worden geplaatst in tweedimensionale tabellen.

deb#	adres
....	.....
....	.....
....	.....

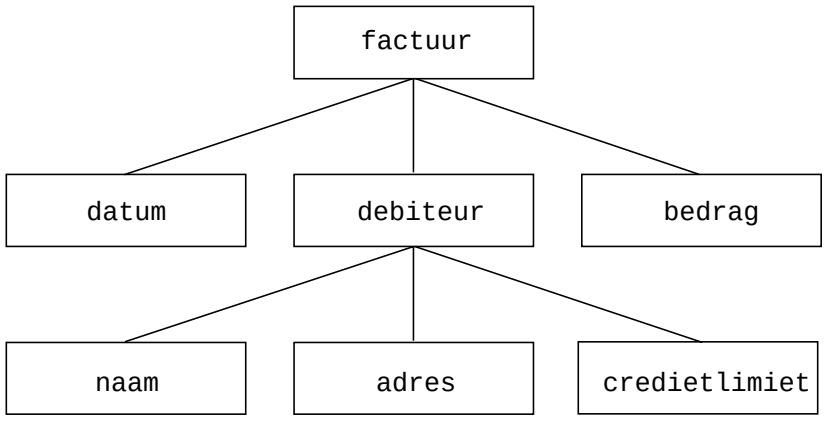
deb#	datum	bedrag
....	.....	....
....	.....	.....
....	.....	....
....	.....	...
....	.....	.....

Het relationele model werd verwelkomd vanwege de stap voorwaarts in eenvoud van gegevensbeheer. Er is echter een aantal belangrijke nadelen verbonden aan deze benadering. Ten eerste voorziet het relationele model niet in een structurele bewaking van de samenhang tussen gegevens (bijvoorbeeld facturen zijn geregistreerd maar de bijbehorende debiteuren zijn gewist). Ten tweede, omdat samenhang tussen tabellen ontbreekt, is iedere gebruiker of applicatie verplicht deze samenhang telkens opnieuw te specificeren. Ten derde zijn computers niet in staat manipulaties op onsamenhangende tabellen efficiënt uit te voeren.

Een semantisch DBMS gaat uit van het **semantisch model**, een moderne datamodelleringswijze. Het semantisch model kent evenals het relationele model eenvoudige gegevensstructuren maar bevat bovendien de samenhang tussen tabellen. Door de bewaking van de integriteit (samenhang) is het voor de gebruiker onmogelijk geworden opdrachten te geven die in strijd zijn met de in het model vastgelegde betekenis (bijvoorbeeld debiteuren kunnen pas worden gewist nadat alle facturen behorende bij deze debiteuren zijn verwijderd). Daarnaast kan iedere gebruiker, zonder nadere specificatie, direkt gebruik maken van de in het model eenmalig vastgelegde samenhang. Tenslotte ligt de performance van een semantisch DBMS ook aanzienlijk boven die van een relationeel DBMS.

I.3.3. Begrippen

Het typebegrip neemt een centrale plaats in bij de theorie van gegevens. Een **type** is gedefinieerd als zijnde een samenvoeging tot één geheel van een vast aantal **eigenschappen**. Een type kan ook opgevat worden als een eigenschap en als zodanig voorkomen in andere typen. Bijvoorbeeld: het type debiteur wordt gedefinieerd door middel van de eigenschappen: naam, adres en credietlimiet. Debiteur zelf kan een rol spelen binnen de typedefinitie van factuur; nu is debiteur een eigenschap van factuur.
Deze abstracties kunnen uitstekend worden weergegeven met behulp van een **diagram**.



Formeel weergegeven:

type factuur = datum, debiteur, bedrag.
type debiteur = naam, adres, credietlimiet.

In tabelvorm:

factuur	datum	debiteur	bedrag
.....	.....	.....	....
.....	.....	.....	.....
.....	.....	.....	.....
.....	.....	.....	....
.....	.....	.....	....
.....	.....	.....	.....
.....	.....	.....	...
.....	.....	.....	....
.....	.....	.....	.....
.....	.....	.....	...

debiteur	naam	adres	crediet limiet
.....	....	.....	.....
.....	..	.....	.....
.....	....	...	....
.....	...	.....	....
.....	....	.....	....
.....	...	....	....

In elke tabel staat links van de dubbele vertikale lijn de identificatie van een type en rechts daarvan de eigenschappen van een type.

Integriteitsregels:

1. **Relateerbaarheid**
   Volgens voorgaande formele typedefinities is iedere factuur gerelateerd aan precies één debiteur; een debiteur daarentegen kan corresponderen met diverse facturen. Het semantisch DBMS zal dan ook niet toestaan dat debiteurgegevens worden verwijderd zolang er nog een factuur voor die debiteur aanwezig is.
2. **Omkeerbaarheid**
   Iedere typedefinitie is uniek; er bestaat geen ander type met precies dezelfde eigenschappen. Zo zal naast de definitie van debiteur het volgende type niet toegestaan mogen worden:
   type klant = naam, adres, credietlimiet.

Bovenstaande samenhang wordt in de formele definities van een semantisch model éénmalig vastgelegd en vervolgens door het DBMS bewaakt. Hiermee zijn overtredingen van deze integriteitsregels uitgesloten.

Kenmerkend voor een semantisch DBMS is de ondersteuning van de begrippen generalisatie en specialisatie. Door deze begrippen wordt de mogelijkheid geboden om de werkelijkheid zo getrouw mogelijk weer te geven. **Specialisatie** wordt gedefinieerd door de eigenschappen die in één of meer andere deelnemende typen voorkomen. **Generalisatie** daarentegen wordt gedefinieerd door de eigenschappen die in alle deelnemende typen voorkomen.

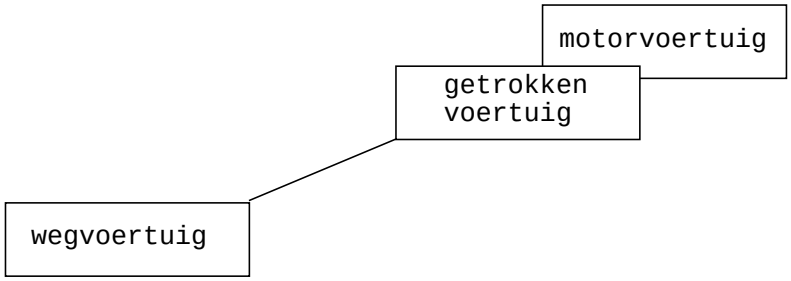
Bijvoorbeeld:

type motorvoertuig = merk, handelsaanduiding, cilinderinhoud, brandstof.
type getrokken voertuig = merk, handelsaanduiding, koppeling.

De generalisatie hiervan luidt:

type wegvoertuig = merk, handelsaanduiding.

Diagrammen worden ook gebruikt voor weergave van generalisatie / specialisatie:

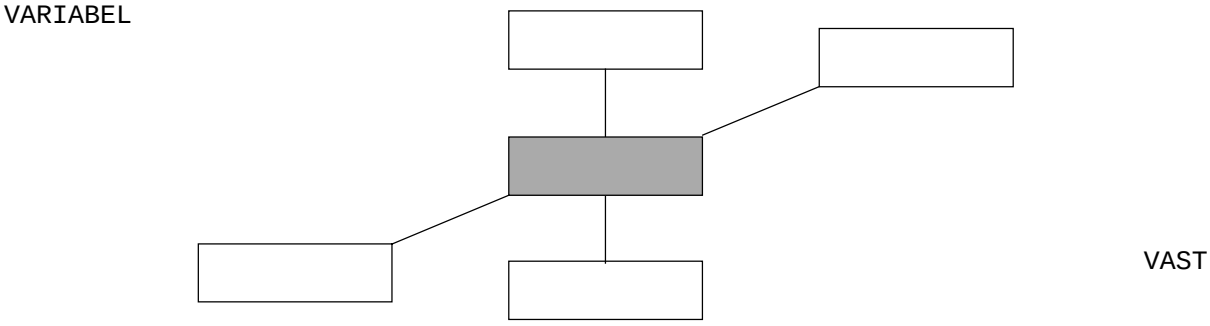


In de typedefinitie van een specialisatie treedt de generalisatie op als eigenschap en is omgeven door rechte haken:

type motorvoertuig = [wegvoertuig], cilinderinhoud, brandstof.
type getrokken voertuig = [wegvoertuig], koppeling.

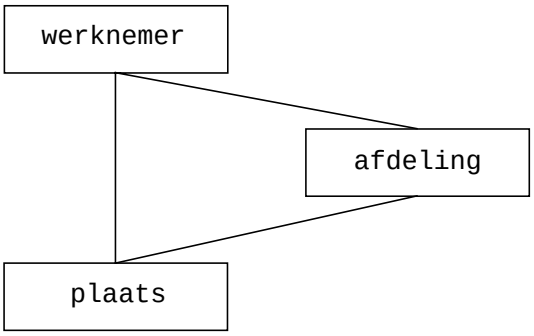
Een zogenaamd **Delfts Diagram** representeert een **abstractiehiërarchie** waarin twee soorten abstracties voorkomen:
- aggregatie (middens van zijden zijn met elkaar verbonden)
- generalisatie (hoekpunten zijn met elkaar verbonden)

Daarbij komen de vaste eigenschappen van een type altijd onder en de variabele eigenschappen van een type altijd boven het betreffende type.



Een Delfts Diagram is ook zeer bruikbaar bij manipulatie van gegevens. Voor vaste eigenschappen wordt altijd de **its**-constructie gebruikt, terwijl voor variabele eigenschappen altijd de **per**-constructie wordt gebruikt.

Een voorbeeld:



type werknemer = naam, woon_plaats, afdeling, ...
type afdeling = vestigings_plaats, ...

Vraagstelling: welke werknemers zijn forens?

Alle benodigde gegevens zijn onder het type werknemer te vinden, dus luidt de oplossing simpelweg:

get werknemer **its** naam
    where woon_plaats <> afdeling **its** vestigings_plaats.

Vraagstelling: geef van iedere afdeling het aantal werknemers?

Aangezien werknemer boven afdeling is geplaatst, moet de per-constructie worden gebruikt. De oplossing luidt nu:

extend afdeling **with** aantal =
    count werknemer **per** afdeling.

get afdeling.

Met "get afdeling" wordt verkregen de identificatie en alle eigenschappen van elke afdeling inclusief de afgeleide eigenschap ’aantal’.

Bij deze laatste vraagstelling is een belangrijke karakteristiek van de manipulatietaal zichtbaar. Complexe vraagstukken worden in gedeelten beschreven die elk een duidelijk effect hebben. Een bijzonder kenmerk van deze aanpak is dat deze opsplitsing voor ieder probleem maar op één manier kan worden gemaakt. Samen geven de delen een oplossing voor het gehele probleem. Het is niet nodig om complexe (niet unieke) geneste structuren te doorgronden, zoals dat bij formuleringen in SQL (de standaard manipulatietaal voor het relationele model) dikwijls voorkomt.

I.4. ARCHITECTUUR

I.4.1. Doelstellingen

De belangrijkste doelstellingen bij het ontwerp van het DBMS waren de volgende:

a. **Correctheid**

- Het DBMS mag geen tegenstrijdige modelspecificaties toelaten.
- Het DBMS moet alle integriteitsregels van een datamodel automatisch bewaken en niet toestaan dat integriteitsregels omzeild kunnen worden.
- Het DBMS moet functies voor database herstel bevatten.

b. **Flexibiliteit**

- Het DBMS moet opdrachten kennen om een model van een database eenvoudig te wijzigen. Dit zonder de integriteit van de database aan te tasten.

c. **Gebruiksvriendelijkheid**

- De bediening en dialogen moeten zodanig zijn ingericht dat ze zowel voor beginnende als voor geroutineerde gebruikers zinvol en eenvoudig zijn. Speciale voorzieningen moeten worden getroffen om interactief werken mogelijk te maken.
- Waar nodig moet de gebruiker extra ondersteuning kunnen krijgen van het systeem.
- Op ieder moment moet voor de gebruiker duidelijk zijn welke acties het DBMS heeft uitgevoerd.

d. **Performance**

- Het DBMS moet in staat zijn omvangrijke databases op een efficiënte manier te beheren en te manipuleren.
- Een goede performance mag niet gebaseerd zijn op de aanwezigheid van zeer speciale apparatuur. Gegevens moeten op standaard vaste schijven (hard disk) opgeslagen kunnen worden.

e. **Implementatie**

- Het DBMS moet de semantische concepten efficiënt implementeren rekening houdend met mogelijke latere uitbreidingen.
- Het DBMS moet modulair opgebouwd worden. De modules moeten een minimale koppeling bevatten met andere modules zodat aanpassingen in één module zo min mogelijke aanpassingen in andere modules veroorzaakt.
- Speciale aandacht moet besteed worden aan de portabiliteit van de software. Het DBMS mag hierdoor geen gebruik maken van specifieke hardware-faciliteiten.

I.4.2. Globale structuur

Het DBMS bestaat uit de volgende systeemcomponenten:

- **Kern**
  De kern van het DBMS bestaat uit programmatuur voor de fysieke opslag van gegevens. Hiervoor worden moderne opslag- en file management technieken toegepast die resulteren in een DBMS met hoge performance. Speciale functies zorgen voor onafhankelijkheid van bedrijfssysteem, terminal en printer.
- **B-tree management**
  Speciale programmatuur zorgt ervoor dat gegevens - ook in een omvangrijke database - zeer snel kunnen worden opgespoord. De implementatie met behulp van B^+-trees wordt ook ingeschakeld bij de handhaving van de integriteit van de gegevens in de database. Een B^+-tree is een speciale wijze van indexering.
- **Hoofdprogramma**
  Hierin zijn de menu's opgenomen. Speciale voorzieningen zijn getroffen om de dialoogteksten, menukeuzen, systeemmeldingen en hulpteksten in een andere taal (Nederlands, Engels, Duits, etc.) te kunnen presenteren.
- **Data Definition Language processor**
  Deze programmatuur verwerkt alle opdrachten die betrekking hebben op de definitie van een databasemodel. Gegevens die hierop betrekking hebben worden geplaatst in een semantische data dictionary. Hierbij spelen aantal metatabellen een belangrijke rol.
- **Data Manipulation Language processor**
  Deze programmatuur verwerkt alle opdrachten die betrekking hebben op het opvragen, wijzigen, toevoegen en verwijderen van gegevens in de database.
- **Autorisatiemanager**
  De autorisatie-module van het systeem zorgt ervoor dat ongeautoriseerde manipulatie of opvraging van gegevens niet mogelijk is. Iedere gebruiker heeft specifieke rechten. Hiertoe worden de gebruikers in groepen ingedeeld. Per groep kan aangegeven worden welke rechten de gebruikers van deze groep hebben.
- **Picklistgenerator**
  Deze programmatuur genereert dynamisch een volledige applicatie aan de hand van de typespecificaties van een te selecteren type. De interactieve applicatie is bruikbaar voor opvragingen, manipulaties en afdrukken van gegevens van het type. De schermindeling wordt geheel automatisch door het systeem bepaalt.
- **Panelgenerator**
  Deze programmatuur stelt de gebruiker in staat schermindelingen te definiëren naar eigen wens. Het gezichtspunt van de gebruiker hoeft niet beperkt te blijven tot één type, ook hiërarchieën van typen zijn eenvoudig te definiëren met behulp van enkele (functie)toetsen. Gedefinieerde schermen zijn te gebruiken voor alle soorten data-manipulatie, inclusief schermafdruk en complexe zoekopdrachten. Hierdoor kan iedere gebruiker, zonder kennis van de manipulatietaal, gebruik maken van de gegevens in de database.

- **Rapportgenerator**
  Deze generator kan gegevens verwerken die afkomstig zijn van verschillende XPLAIN databases, maar is ook in staat gegevens te verwerken die niet door XPLAIN zijn aangemaakt.
- **Data Editor**
  Deze programmatuur maakt het mogelijk om op interactieve wijze gegevens, behorende tot één type, te manipuleren.

Via menu's kan de gebruiker verschillende componenten benaderen. Het menu staat onderaan het scherm. Afhankelijk van het niveau bevat het menu steeds andere keuzemogelijkheden. De huidige keuze wordt als een blokje aangegeven. Met de pijltjestoetsen kan een andere keuze geselecteerd worden. Het indrukken van de <Return>-toets activeert de keuze.

Het hoofdmenu van XPLAIN bevat de volgende keuzemogelijkheden:

EXIT
Hiermee verlaat men het systeem. Overigens moet daarna achter "login:" ook nog **end** (of **exit** of **logout**) gevolgd door <Return> ingetoetst worden.

DEFINE
Hiermee betreedt men het databasedefinitie-menu. Het is dan mogelijk het toevoegen, verwijderen of veranderen van databases te realiseren. Ook definitie van een databasemodel is mogelijk, zowel door middel van Data Definition Language (DDL) als via dialogen tussen DBMS en gebruiker. In deze fase moet aan een database een naam worden toegekend. Alleen de Database Administrator (**dba-groep**) is geautoriseerd tot het vastleggen van database definities en/of wijziging hiervan.

USE
Deze optie maakt het mogelijk manipulaties te doen op de inhoud van reeds eerder gedefinieerde databases. Hierbij kan men kiezen tussen het gebruik van de data manipulation language (DML) via Query, de picklistgenerator (applicatieschermen voor manipulaties op gegevens van één type), de panelgenerator (voor complexere applicatieschermen) of de data-editor (voor manipulaties op gegevens van één type).

REPORT
Met deze optie kunnen rapporten worden verwerkt. Rapporten kunnen van diverse databases afkomstig zijn, van extern worden geimporteerd en naar extern worden geexporteerd. Relationele operatoren en operaties voor de opmaak van rapporten zijn aanwezig.
Deze optie biedt ook de mogelijkheid om grote hoeveelheden data te importeren in bestaande XPLAIN databases. Daarnaast kunnen queries en modellen worden geimporteerd in XPLAIN databases.

De rapportgenerator kan bij batchverwerking worden opgevat als de enige sluis waarmee XPLAIN kan communiceren met de buitenwereld.

ACCESS
Met deze optie wordt de autorisatie-manager betreden. Hiermee is het mogelijk rechten te geven aan of af te nemen van groepen van gebruikers. Een gebruiker met voldoende rechten mag weer gebruikers en/of groepen creëren, wijzigen of verwijderen. Indien een gebruiker z'n eigen password wil wijzigen dan moet dit via de optie ACCESS. Overigens is het doorgeven van rechten hiërarchisch geregeld, d.w.z. een gebruiker mag hoogstens de rechten van zichzelf doorgeven.

BACKUP
Hiermee is het mogelijk om een copie van de gehele database, inclusief modeldefinities, queries en panels te maken. Een backup wordt alleen gemaakt indien er geen onregelmatigheden zijn geconstateerd tijdens het gebruik van de database.

RESTORE
Deze optie maakt het mogelijk de gemaakte backup terug te zetten.

HELP
Het help-menu geeft toelichting op de hierboven genoemde opties.

I.5. GEBRUIK VAN HET DBMS

I.5.1. Gegevensdefinitie

Het conceptuele databasemodel kan in het DBMS worden opgebouwd met behulp van commando's en/of dialogen. Het commandogedeelte bevat hiervoor een aantal eenvoudige opdrachten. Dialogen stellen de gebruiker in staat om een bestaand databasemodel op een interactieve manier te definiëren of aan te passen.

Commando's

De belangrijkste opdrachten zijn:

1. **Definitie basistypen**
   Basistypen zijn typen die alleen voorkomen als elementair type (d.w.z. niet samengesteld zijn). Deze typen worden geïntroduceerd met behulp van de base-opdracht, bijvoorbeeld:

base naam (A20).	(20 karakters)
base salaris (R6,2).	(6 posities links en 2 rechts van punt)
base gehuwd (B).	(logische waarde)
base aantal (I4).	(4 decimale cijfers)
base datum (D).	(datum bestaande uit 8 cijfers: jjjjmmdd)
	Opmerking: operatie op een **datum vereist** gebruik van datumfunctie.

Na de basistypenaam volgt het domein (representatie) van het gegeven. Hiervoor zijn vijf mogelijkheden:

A voor tekstuele gegevens (karakterstring),
R voor gebroken getallen (Real),
I voor gehele getallen (Integer),
B voor logische waarden waar/onwaar (Boolean).
D voor datumwaarde d.w.z. geldige kalenderdatum (Date).

Daarnaast zijn waardebeperkingen op basistypen mogelijk:

- een waardebeperking in de vorm van een traject. Deze beperking is mogelijk bij Integer-, Real en Date typen. De beperking moet worden opgegeven door het opgeven van een onder- en bovengrens. Bij een datum moeten geldige datum-grenzen gekozen worden.
  Voorbeeld: **base** korting (I2) (0..25).
- een waardeperking in de vorm van een patroon. Deze beperking kan bij karakterstrings worden opgegeven.
  Voorbeeld: **base** postcode (A6) "9999xx".
- tenslotte is er nog de mogelijkheid om de toegestane waarden die het type aan kan nemen, één voor één op te sommen. Deze mogelijkheid geldt alleen voor Integer en Alfanumerieke basistypen.
  Voorbeeld: **base** werkdag (A2) = "ma", "di", "wo", "do", "vr".

2. Definitie samengestelde typen

Samengestelde typen worden geïntroduceerd met de type-opdracht, zoals:

type werknemer (I6) = naam, salaris, gehuwd.

In deze definitie is alleen het domein van het nieuwe samengestelde type nodig, daar alle eigenschappen in het rechterlid eerder zijn gedefinieerd als basis- of als samengesteld type. Aan het domein van een samengesteld type kan **alleen** een patroon (bij alfanumeriek) of traject (bij integer) worden toegevoegd, dus geen opsomming!

3. Restricties

Met dynamische restricties kunnen bij toevoeging (insert) beperkingen worden opgelegd aan één of meerdere attribuutwaarden van een samengestelde type. Deze beperkingen kunnen al dan niet afhankelijk zijn van de databasetoestand.

Voorbeelden:
init factuurregel **its** bruto_prijs = artikel **its** prijs.
init factuurregel **its** korting = **if** aantal < 10 **then** 0 **else** 10.

4. Verwijderingen

Tenslotte is er de purge-opdracht waarmee (basis)typen en inits kunnen worden verwijderd, bijvoorbeeld:
purge afdeling.

Deze opdracht zal alleen worden uitgevoerd wanneer de integriteit van het conceptuele databasemodel niet in gevaar komt, met andere woorden alleen als er geen enkele referentie naar het te verwijderen type is.

De gebruiker van het DBMS kan met de voorgaande opdrachten conceptuele database-modellen definiëren. Het is niet nodig extra index-tabellen te definiëren. Het DBMS zorgt zelf voor introduktie en onderhoud van deze zoek-structuren.

Dialogen

Dialogen laten een zeer flexibele manier van gegevensdefinitie toe. De volgende opdrachten zijn beschikbaar:
- voeg type (basis of samengesteld) toe
- verwijder type
- wijzig type
  - . wijzig typenaam
  - . wijzig domein
  - . wijzig rolnaam
  - . wijzig aggregatie/specialisatie
  - . voeg eigenschap toe
  - . verwijder eigenschap

Bij iedere keuze wordt in dialoogvorm het conceptuele databasemodel opgebouwd of aangepast. In dialogen worden alleen relevante gegevens aan de gebruiker gevraagd. Dit houdt in dat de gebruiker zeer weinig ervaren kan zijn in het gebruik van DBMSen. Het DBMS zelf zorgt voortdurend voor handhaving van de integriteit en stelt de gebruiker voortdurend op de hoogte van de acties die zijn uitgevoerd.

Zowel met commando's als via dialogen kan de gebruiker relevante delen van de data dictionary (gegevensmodel) zichtbaar maken op het scherm. De inhoud van de data dictionary wordt gepresenteerd in termen die voor de gebruiker begrijpelijk zijn. Implementatie aspecten komen hierin niet voor.

I.5.2. Gegevensmanipulatie

Voor gegevensmanipulatie zijn in het DBMS zes krachtige opdrachten aanwezig, te weten:
- **get** opdracht: voor het ophalen van bepaalde gegevens uit de database eventueel met zoekcriteria.

- **insert** opdracht: voor toevoeging van gegevens (een instance) aan de database.
- **delete** opdracht: voor verwijdering van gegevens (instances) uit de database.
- **update** opdracht: voor wijziging van gegevenswaarden in de database.
- **extend** opdracht: voor tijdelijke toevoeging van een eigenschap aan een type, met gedefinieerde waarde.
- **value** opdracht: voor het creëren of opvragen van een tijdelijke variabele.

De volgende standaard **set-functies** zijn mogelijk bij de get, extend en value-opdracht:

max: bepaling van de maximale waarde van een collectie van gegevens.
min: bepaling van de minimale waarde van een collectie van gegevens.
total: bepaling van de som van een collectie van gegevens.
count: bepaling van het aantal waarden in een collectie van gegevens.
any: vaststelling of de collectie van gegevens waarden bevat (waar/ onwaar)
nil: vaststelling of de collectie van gegevens leeg is (waar/onwaar).
some: bepaling van een willekeurige waarde uit een collectie van gegevens.

Rekenkundige bewerkingen zijn optellen (+), aftrekken (−), vermenigvuldigen (*), delen (/) en modulus (%). Bovendien zijn de operatoren =, <>, <, <=, >, >=, **not**, **and**, **or** aanwezig. Met haakjes (en) kunnen verwerkingsprioriteiten aangegeven worden.

Ter illustratie van de kracht van de manipulatieopdrachten volgen nu enkele voorbeelden. Deze vraagstellingen hebben allen betrekking op een database met het volgende conceptuele model:
type werknemer = naam, adres, salaris, commissie, afdeling.
type afdeling = afdnaam, plaats.

1. **Eenvoudige vraagstelling**
   Geef de gegevens van de werknemer met identificatie 132.
   get werknemer "132".

2. **Eenvoudige selectie**
   Geef de namen van de werknemers die meer commissie dan salaris ontvangen.
   get werknemer **its** naam
   where commissie > salaris.

3. **Gerelateerde gegevens**
   Geef de gegevens van de werknemers die in Delft werken.
   get werknemer **where** afdeling **its** plaats = "Delft".

4. **Standaard functie**
   Geef de werknemer(s) met het hoogste inkomen (d.w.z. salaris + commissie).
   value maxink = **max** werknemer **its** salaris + commissie.
   get werknemer **where** maxink = salaris + commissie.

5. **Extensie**
   Geef de afdeling met een salarisbudget boven 100.000.
   extend afdeling **with** budget =
   total werknemer **its** salaris **per** afdeling.
   get afdeling **where** budget > 100000.

6. **Toevoeging**
   Voeg afdeling A22 toe met afdelingsnaam Onderzoek en vestigingsplaats Delft.
   insert afdeling "A22" **its** afdnaam = "Onderzoek", plaats = "Delft".

7. **Verwijdering**
   Verwijder alle gegevens van werknemer 192.
   delete werknemer "192".

8. **Wijziging**
   Verhoog de salarissen van de werknemers die in Delft werken met 10%.
   update werknemer **its** salaris = 1.1 * salaris
   where afdeling **its** plaats = "Delft".

De mogelijkheden van de manipulatietaal worden uitvoerig behandeld in het hoofdstuk (MANIPULATIETAAL). In het hoofdstuk (APPENDICES) wordt o.a. de syntaxis van de semantische datadefinitie en datamanipulatie taal gegeven.

I.5.3. Panelgenerator

De interactieve panelgenerator wordt gebruikt voor het genereren van nieuwe database toepassingen (applicatieschermen) zonder gebruik te maken van conventionele programma's. Bij schermdefinitie is de lay-out van het uiteindelijke scherm zichtbaar en direct manipuleerbaar. Schermdefinitie bestaat uit vastlegging van de applicatiestructuur en schermindeling.

Met een gegenereerde applicatie kunnen gegevens in een database worden toegevoegd, opgezocht, gewijzigd en verwijderd. Gegevens worden opgeslagen in één of meer al dan niet samenhangende tabellen. Zowel bij definitie als bij gebruik is het overbodig eventueel bestaande samenhang aan te geven met behulp van complexe data manipulatie opdrachten. Bewaking van de integriteit gebeurt automatisch door het DBMS.

Een scherm bestaat in het algemeen uit vrije tekst waarin opgenomen enkele velden uit de database. De onderste twee schermregels worden gebruikt voor aanduiding van relevante toetsen, foutboodschappen en een geheugensteuntje (dat betrekking heeft op het relevante gedeelte van het conceptuele databasemodel). Tijdens schermdefinitie en -gebruik vindt direkte terminalbesturing plaats. Hierdoor kan terstond gereageerd worden op eventuele ongeldige invoer van de gebruiker.

De panelgenerator is bedoeld voor gebruikers die enige kennis hebben van het semantisch

model. Het DBMS ondersteunt de gebruiker tijdens de definitie van de applicatiestructuur door alleen de zinvolle en correcte typen of attributen te tonen, waaruit de gebruiker dan simpel kan kiezen. De definitie kan hierdoor zeer snel en altijd correct gedaan worden. Tekst kan gemakkelijk toegevoegd worden.

Om een indruk te geven van de schermindeling tijdens definitie, een eenvoudig voorbeeld dat betrekking heeft op het volgende model:

type klant = naam, voorl, adres, postc, woonplaats.
type artikel = beschrijving, kleur, voorraad, prijs, soort.
type verkoop = artikel, hoeveelheid, bedrag, klant, datum, aanbet.

In dit voorbeeld wordt gebruik gemaakt van een view met verkoopgegevens per klant. Geheel bovenaan staan identificatie en eigenschappen van het type klant. Van de verkopen aan de klant worden aangegeven: artikel (ARTNR.), hoeveelheid (HOEV.) en bedrag (BEDRAG). Verder staan er nog een aantal eigenschappen van artikel op het scherm: beschrijving (ART.BESCHRIJVING) en prijs (PRIJS).

```
KLANT 1.ID
NAAM  1.voorl      1.naam
ADRES       1.adres           1.postc            1.woonplaats

ARTNR.   ART.BESCHRIJVING      HOEV       PRIJS         BEDRAG
 2.ID      2.beschrijving    2.hoev       2.prijs     2.bedrag
 2.ID      2.beschrijving    2.hoev       2.prijs     2.bedrag
 2.ID      2.beschrijving    2.hoev       2.prijs     2.bedrag
 2.ID      2.beschrijving    2.hoev       2.prijs     2.bedrag
 2.ID      2.beschrijving    2.hoev       2.prijs     2.bedrag
 2.ID      2.beschrijving    2.hoev       2.prijs     2.bedrag
PF1=EXIT  PF2=DEFINE  PF3=INS/DEL LINE  PF4=HELP  Enter=Insert/Typeover
2. verkoop = artikel, hoeveelheid, bedrag, klant, datum, aanbet.
```

Hetzelfde scherm kan tijdens gebruik de volgende gegevens bevatten:

```
KLANT  232
NAAM   E T   Peters
ADRES  Brederodelaan 14       7312 NJ            Apeldoorn

ARTNR.   ART.BESCHRIJVING     HOEV        PRIJS         BEDRAG
9832416  gereedsch.kist          1        29.50          29.50
5639810  pullover                2        49.75          99.50
3420186  zoomlens+kap            1       452.75         452.75
8392950  comtoise                1       399.00         399.00
2834084  afzuigkap               1       189.50         189.50
0482910  zwembroek               1        37.25          37.25
 EXIT      <      FILTER      >        UP      INSERT    UPDATE     HELP
CASCADE   <<      TOP >>      >>      DOWN     CLEAR     DELETE    OUTPUT
```

De relateerbaarheid door middel van de identificaties maakt zulke views over meerdere objecttypen - hier: verkoop, artikel en klant - mogelijk.

Merk op dat alleen relevante verkoopgegevens gedefinieerd zijn en dat ter verduidelijking het scherm andere tekst kan bevatten dan alleen de termen waarmee het model beschreven is.

Tijdens het gebruik van het panel zal XPLAIN zorgen voor controle van de waarde-beperkingen en handhaving van de samenhang tussen gegevens. Dit houdt in dat afleidbare gegevens niet hoeven (sterker nog: niet kunnen) worden ingebracht. Het DBMS zorgt zelf voor de opslag van de benodigde gegevenswaarden in de database.

Een voorbeeld ter illustratie van deze faciliteiten van het DBMS. Er is sprake van een bedrijf met meerdere vestigingen, waarbij afdelingsnamen meerdere keren kunnen bestaan. De afdelingsnaam kan dus niet dienen als identificatie. Stel dat de gegevens over werknemers per afdeling moeten worden opgenomen in de database.

Het conceptuele databasemodel luidt:

type afdeling = afdnaam, plaats.
type werknemer = naam, plaats, afdeling.

In het volgende overzicht van werknemers per afdeling moet gebruik worden gemaakt van de samenhang tussen afdeling en werknemer.

```
AFDELING ==> A21   Inkoop      Delft

  WERKNEMER          NAAM        PLAATS
    1024             Jansen      Delft
    1025             Peters      Gouda
    1026             de Goede    Leiden
```

Bij een dergelijke schermdefinitie zou opname van de afdeling bij werknemer kunnen leiden tot inconsistenties (de gebruiker zou immers telkens de waarde A21 moeten intoetsen). Vandaar dat het DBMS deze laatste opname verbiedt en de waarde A21 bij toevoeging van een medewerker automatisch vult bij het attribuut afdeling.

Hierboven is een eenvoudig voorbeeld gegeven van de integriteitsbewaking door het DBMS. Een uitputtende behandeling hiervan moet echter in dit globale overzicht achterwege blijven.

I.5.4. Picklistgenerator

Dit is een hulpmiddel dat op basis van een te selecteren typenaam geheel automatisch een applicatiedefinitie samenstelt en daarna het panel genereert. Alle velden worden overzichtelijk gepresenteerd op het scherm in een formulier-achtige vorm voorzien van begeleidende tekst zoals bij een zelf gedefinieerd panel.

De picklist bevat, zoals ieder panel die met de panelgenerator gegenereert is, alle standaard manipulatiemogelijkheden zoals zoeken op waarden, een volgend of vorig scherm met gegevens laten zien, nieuwe gegevens invoeren, gegevens wijzigen en gegevens verwijderen. Dit allemaal met controles op integriteit en waardebereiken. Dit hulpmiddel is een zeer handige, snelle en gebruiksvriendelijke manier om met een database te werken.

I.5.5. Rapportgenerator

De interactieve rapportgenerator kan onder meer de volgende soorten gegevens in rapporten verwerken: database modellen, query resultaten, query formuleringen, applicatieschermen, externe files, database imports en database imports. Op het hoofdniveau bestaan de volgende operaties:

- NEW: toevoegen van een nieuw rapport;
- INSERT: bestaande file opnemen als rapport;
- FIND: opsporen van files voor opname als rapport;
- SELECT: rapportselectie t.b.v. rapportopmaak;
- RENAME: herbenoemen van rapport;
- REMOVE: verwijderen van rapport;
- PRINT: afdrukken van rapportinhoud;
- EXPORT: rapport copiëren naar andere directory;
- IMPORT: rapportgegevens opnemen in XPLAIN database;
- SUBTRACT: rapportverschil bepalen (relationele operator);
- JOIN: rapporten samenvoegen (relationele operator);
- DOCUMENT: rapport inkijken voor afdruk.

De rapporteditor stelt de gebruiker in staat rapporten op diverse manieren op te maken:

- EDITING: tekstopmaak faciliteiten;
- SORT: mogelijkheden tot sorteren van rapportgegevens;
- HEADER: opmaak van kopregel van rapport;
- FOOTER: opmaak van voetregel van rapport;
- TABSET: herstel van tabsetting in rapport;
- APPEND: toevoegen van rapport aan bestaand rapport;
- UNIQUE: verwijderen van duplicaatregels (relationele operator);
- STRIP: verwijdering van kolom in rapport (t.b.v. de relationele project operator).

I.5.6. Data editor

Het DBMS bevat een eenvoudige data editor waarmee op gestandaardiseerde wijze gegevens kunnen worden ingebracht, gewijzigd, gezocht en verwijderd. Deze editor is beschikbaar voor ieder samengesteld type uit het databasemodel.

De editor geeft een standaard representatie van de gegevens. Eventueel maakt het DBMS automatisch gebruik van een 132-karakter scherm in plaats van het gebruikelijke 80-karakter scherm.

Tijdens editing zijn de volgende commando's toegestaan:
- specificatie van relevante eigenschappen;
- toevoeging van gegevens (horizontaal/vertikaal);
- verwijdering van gegevens;
- wijziging van gegevens;
- zoeken van gegevens;
- zichtbaar maken van alle gegevens van een type;
- volgend blad met database gegevens;
- vorig blad met database gegevens;
- volgend exemplaar in de database;
- maak alle eigenschappen van een type zichtbaar.

Wellicht ten overvloede wordt opgemerkt dat de editor automatisch zorg draagt voor de integriteit van de database.