

7626  
98763  
CBM

R

SITY

UNIVERSITEIT  
BRABANT

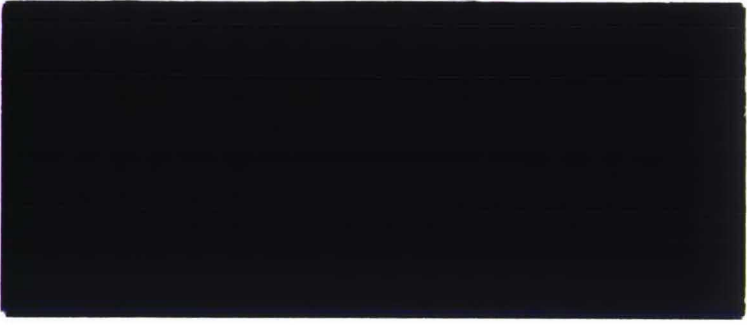
POSTBOX 90153

5000 LE TILBURG

THE NETHERLANDS



\* C I N O O 3 5 6 \*



DEPARTMENT OF ECONOMICS  
RESEARCH MEMORANDUM




 K.U.B.  
BIBLIOTHEEK  
TILBURG

**DATAAL - een hulpmiddel voor onderhoud  
van gegevensverzamelingen**

H.P.A. Mulders  
Drs. A.J. van Reeken

**FEW 253**



## Inhoudsopgave

1. Inleiding
2. Enkele notaties
3. Onderhoudsfunkties
4. Bewakingsfunkties
5. Beschrijving van het onderhoud en de bewaking
  - 5.1 Definities en structuur
  - 5.2 Expressie en procedures daarbij
6. Ondersteuning van de feitelijke gegevensmanipulatie
7. Commando's voor het onderhoud
8. Ervaring met DATAAL in het gebruik
  - 8.1 Van de eindgebruiker
  - 8.2 Van de maker van de beschrijving in DATAAL
  - 8.3 Van de maker van de interface

## DATAAL - een hulpmiddel voor onderhoud van gegevensverzamelingen

H.P.A. Mulders            en    Drs. A.J. van Reeken  
 dienst Rekencentrum        vakgroep ISA van de FEW  
                                   Katholieke Universiteit Brabant  
                                   Postbus 90153, 5000 LE Tilburg

### 1 Inleiding

In toepassingen met ( conventionele ) gegevensbestanden worden in de programma's, waarin de gegevens worden gemuteerd ( toevoegingen, verwijderingen, veranderingen ) omvangrijke controles opgenomen. Deze controles hebben tot doel te verhinderen dat onjuiste, onvolledige of ongeautoriseerde gegevens in de bestanden worden opgenomen of onterechte gegevens worden veranderd of verwijderd. De complexiteit van deze programma's is voornamelijk een gevolg van de op te nemen controles. De principes van Warnier en van Jackson, waarbij de structuur van het programma wordt afgeleid van de gegevensstructuur bieden daarbij weinig soelaas, omdat de algoritmische structuur van de controles domineert. Verder komt in elk programma, waarin het betreffende gegeven wordt gemuteerd de controle daarop voor. Wordt, zoals gebruikelijk, het werk over meerdere programmeurs verdeeld, dan vereist dit punt extra aandacht, maar "dubbel" werk en verschillen in de foutafhandeling zijn niet te voorkomen.

Ook bij DBMS toepassingen, waarbij slechts een "host-language" interface is, zijn deze problemen niet te vermijden. Is er een Query-language interface dan blijkt niet iedereen in staat hiervan op de juiste wijze gebruik te maken. Overigens hebben deze DBMS maar beperkte ( en vaak zeer beperkte ) mogelijkheden de ingevoerde gegevens aan voorwaarden te onderwerpen. Bij meerdere verschillende Databases en bij combinaties met conventionele gegevensbestanden zijn deze mogelijkheden er niet.

Bij de analyse van een concreet project in 1981, welke in deelprojecten gerealiseerd moest worden, rees de vraag of bovenstaande problemen niet op een andere wijze op te lossen waren. Het idee groeide een "taal" <sup>1</sup> voor het onderhoud te ontwerpen. Hierdoor wordt het mogelijk de verschillende onderhoudsaspecten op een hoger nivo te specificeren. Het wordt bovendien mogelijk alle onderhoudsaspecten met betrekking tot een gegeven in één blok met de specificatie van het gegeven op te nemen. Daardoor komen deze onderhoudsaspecten niet meer verstrooid voor in de toepassingsprogrammatuur. Dit laatste was "altijd al" een doorn in het oog en dat speciaal wanneer de betreffende statements om de een of andere reden moesten worden herzien.

De oplossing gaat uit van een relationele benaderingswijze, maar de onderliggende datastructuur behoeft daar niet aan te voldoen ( zie sub. 6 ).

Alle bestanden worden gezien als een tabel met rijgewijs de tupels ( ook wel records genoemd ) en kolomgewijs de attributen ( ook wel velden genoemd ).

Het verwijderen, veranderen of toevoegen van tupels, is gelet op de daarbij betrokken attributen, aan voorwaarden gebonden. Bij veranderen of toevoegen kan overigens gebruik gemaakt worden van verstek-waarden ( "bij default" ), naast de door de gebruiker in te voeren waarden. Niet alleen de waarden van een attribuut kunnen aan voorwaarden zijn gebonden, maar ook tussen waarden van attributen in één tupel kunnen voorwaarden bestaan. Verder kunnen er voorwaarden zijn voor relaties tussen tabellen.

Veel later hebben wij begrepen dat er ook elders naar oplossingen voor deze problemen is gezocht en dat ontwerpvoorschriften ( in de literatuur ) zijn gegeven die van zo'n principe zijn uitgegaan. Zo is een verwant principe dat van "information hiding" en de "abstract data

<sup>1</sup>Voorlopig wordt dese "taal" ( onderstaand globaal beschreven ) met de naam DATAAL aangeduid

types" ( vgl. D.L. Parnas : A Technique for Software Module Specification with Examples, Communication of the ACM, May 1972 vol. 15 number 5, p.330-336 ). Wij menen dat in dit licht onze aanpak nog interessante aspecten heeft die zich ook nog wel eens voor Dataflow-machines zouden kunnen bewijzen.

## 2 Enkele notaties

Een logisch bestand kunnen we voorstellen als een tabel met rijgewijs de tupels en kolomgewijs de attributen. We gaan ervan uit dat de tabellen in de zogenaamde 3<sup>e</sup> normaalvorm zijn.

We noteren als volgt met

tab : een tabel

$d = \text{tab}(.,d)$  : een attribuut ("domain" genoemd) uit tab

$d_k$  : een sleutelattribuut

$t = \text{tab}(t,.)$  : een tupel ("tuple" genoemd) uit tab

Meestal zullen er meerdere tabellen in het geding zijn waartussen relaties bestaan.

We gebruiken de notatie

$\text{tab}_i.d_i \rightarrow \text{tab}_j.d_k$  : de waarde van  $\text{tab}_i(., d_i)$  moet als sleutelwaarde in  $\text{tab}_j$  voorkomen, waarbij  $\text{tab}_j(., d_k)$  sleutelattribuut is.

$d_i \rightarrow \text{tab}_j.d_k$  : is een verkorte schrijfwijze hiervan, waarbij  $\text{tab}_i$  de actuele tabel is.

$\text{tab}_i.d_i \rightarrow d_k$  : is eveneens een verkorte schrijfwijze, waarbij  $\text{tab}_j$  de actuele tabel is.

De bij de beschrijving gebruikte ( gereserveerde ) DATAAL-termen zullen onderstreept worden.

## 3 Onderhoudsfuncties

Voor onderhoud op de tabel zijn de volgende acties te onderscheiden :

- nieuw tupel toevoegen ( new )
- bestaand tupel veranderen ( alter of update )
- bestaand tupel verwijderen ( delete )

Om bovenstaande acties te kunnen uitvoeren, is het nodig te weten wat voor elk attribuut gedaan moet worden. Bij delete wordt op de attributen verder geen actie uitgevoerd. Bij new moeten alle attributen en bij alter of update moeten de te veranderen attributen een waarde krijgen. Deze waarde kan aan de gebruiker worden gevraagd maar kan ook gegenereerd worden. Het betreft dan bijvoorbeeld een "defaultwaarde" of aktuele datum.

## 4 Bewakingsfuncties

Voor de bewaking van de integriteit tijdens het onderhoud moeten worden beschreven :

- a. de voorwaarde(n) waaraan een attribuut moet voldoen ( pic , doc )
- b. acties voor het leggen c.q. behouden van een relatie tussen twee tabellen ( src , ssr )
- c. de voorwaarde(n) tussen waarden van attributen in één tupel ( tuc )

De DATAAL-termen worden hieronder toegelicht.

pic staat voor "picture" en daarmee worden de karakteristieke eisen van een attribuut vastgelegd.

doc staat voor "domain constraint" en daarmee wordt aangegeven welke waarden het desbetreffende attribuut mag aannemen.

**src** staat voor "subset requirement control" en daarmee wordt aangegeven welke acties nodig zijn voor het leggen c.q. behouden van een relatie van deze tabel (via betreffend attribuut) met een andere tabel ( via zijn sleutelattribuut ).

**ssr** staat voor "subset requirement" en daarmee wordt de vereiste  $tab_i.d_i \rightarrow tab_j.d_k$  gedefinieerd.

**tuc** staat voor "tuple constraint" en daarmee wordt aangegeven waaraan de waarden van attributen, ten opzichte van elkaar, in één tupel moeten voldoen.

De hierboven gegeven volgorde is ook de volgorde waarin de controle wordt uitgevoerd.

## 5 Beschrijving van het onderhoud en de bewaking

### 5.1 Definities en structuur

Het onderhoud van de data kan nu beschreven worden door per tabel de volgende gegevens vast te leggen :

- de tabelnaam en een indicatie welke interface naar de gegevens zelf gebruikt wordt ( table )  
Het is mogelijk door als interface 'VI' te specificeren een virtuele tabel te maken.
- de attributen ( domains ) met per attribuut de naam gevolgd door een of meerdere van de volgende opties :
  - de attribuutdefinitie, met een van de volgende mogelijkheden :
    - a. type aanduiding ( mode ), indien een virtueel attribuut
    - b. herdefinitie van een rij bestaande attributen ( redefine )
    - c. een bestaand attribuut een andere naam geven of een deel van een bestaand attribuut als een attribuut definiëren ( define )
    - d. een expressie ( exp ). Indien deze een niet bestaand attribuut betreft dan wordt deze als virtueel attribuut beschouwd
    - e. een bestaand tabel ( subtuple ). Het attribuut staat voor een tabel, welke beschouwd wordt als deel van de onderhavige tabel. Er kunnen voor dit attribuut geen andere opties gespecificeerd worden. De tabel kan slechts eenmaal als subtuple voorkomen.
  - de actie bij nieuw tupel ( new )
  - de actie bij veranderen van tupel ( alter en/of update )
  - het beeld voor de waarde van het attribuut ( pic )
  - de range van de waarde voor het attribuut ( doc )
  - actie nodig voor het leggen c.q. het behouden van relaties ( src )
  - toelichtende ("help") tekst voor het desbetreffende attribuut ( help )
- het sleutelattribuut ( key )
- de vereiste relaties naar andere tabellen ( ssr )
- voor een tupel ( tuple ) de volgende gegevens
  - actie bij een nieuw tupel ( new )  
In sommige gevallen moeten namelijk bij het toevoegen van een nieuw tupel ook veranderingen in andere tabellen worden uitgevoerd.

- actie bij veranderen van een tuple ( alter en/of update )
  - actie bij verwijderen van een tuple ( delete )
  - voorwaarden waaraan attributen onderling, in een tuple moeten voldoen ( tuc )
- voor een tabel kan nog een window ( window ) gedefinieerd worden. Met behulp van een window kan bepaald worden dat alleen tupels welke aan de gestelde voorwaarden (logische expressie) voldoen tot de actuele tabel behoren.

Een beschrijving voor het onderhoud van data en de bewaking daarvan heeft per tabel de volgende structuur

```





```

Het geheel wordt afgesloten door end ;

Van mode , exp , define en redefine mag er maar één per attribuut voorkomen.



## 5.2 Expressies en procedures daarbij

Voor het beschrijven van bovenstaande acties en voorwaarden zijn eenvoudige expressies nodig. Een expressie kan samengesteld worden uit :

- procedures, zonder of met een of twee parameters
- constante
- waarde van een attribuut
- operatoren
- := ("wordt"-teken)
- ( logische waarde ! expressie ! expressie )  
betekenis : *if* logische waarde *then* expressie *else* expressie *fi*
- ( integer waarde ! expressie, ..., expressie ! expressie )  
betekenis : *case* integer waarde  
          *in* expressie<sub>1</sub>, ..., expressie<sub>n</sub>  
          *out* expressie<sub>n+1</sub>  
          *esac*

Als de integer waarde  $k$  is dan wordt indien  $1 \leq k \leq n$  expressie <sub>$k$</sub>  uitgevoerd, anders expressie<sub>n+1</sub>.

- attribuutnaam gevolgd door [m:n], met m en n constanten. Het attribuut moet van het type karakter zijn. Er wordt een deelrij, zijnde het m<sup>de</sup> tot en met het n<sup>de</sup> karakter van het attribuut, genomen. Vooral nodig bij define .

Er zijn speciale procedures voor gebruik in bovenbedoelde expressies. De procedures leveren allemaal een waarde op, zodat ze in expressies met operatoren gebruikt kunnen worden. Bij de specificatie van de procedures komen als parameters tab(.,d) en d<sub>k</sub> voor. Bij deze parameters wordt steeds verondersteld dat het sleutelattribuut, d<sub>k</sub> behorende bij de betreffende tabel, een waarde heeft en de actie wordt voor het tupel t<sub>j</sub> met die sleutelwaarde uitgevoerd. Verder wordt er bij de procedures gesproken over het actuele attribuut, waarmee het attribuut bedoeld wordt waarbij de uit te voeren expressie staat.

De procedures zijn :

<u>ask</u>	:	drukt de naam van het actuele attribuut af en vraagt de waarde
<u>show</u>	:	drukt de naam en de waarde van het actuele attribuut af
<u>today</u>	:	levert de datum als rij karakters af
<u>testdate</u>	:	kontroleert of de waarde van het actuele attribuut een correcte datum is
<u>message</u> (s)	:	drukt de string s welke een mededeling bevat af
<u>errmsg</u> (s)	:	drukt de string s welke een foutmelding bevat af. Er wordt verder gegaan alsof er een foutsituatie was
<u>escmessage</u> (s)	:	drukt de string s welke een foutmelding bevat af. Er wordt verder gegaan alsof er een ontsnappingsituatie was
<u>question</u> (s)	:	drukt de string s welke een vraag bevat af. Als het antwoord y of yes is, levert de procedure 'true' op, anders 'false'
<u>give</u> (expressie)	:	geeft, eventueel na conversie, de waarde van de expressie als geheel getal

- give (expressie) : geeft, eventueel na conversie, de waarde van de expressie als een rij karakters
- delspaces (expressie) : hetzelfde als give, maar waarbij alle spaties vóór het eerste en ná het laatste niet-spatie-karakter worden weggelaten
- concat (exp<sub>1</sub>, exp<sub>2</sub>) : concatenatie van delspaces (exp<sub>1</sub>) en delspaces (exp<sub>2</sub>). Indien het laatste karakter van give (exp<sub>1</sub>) een spatie is, dan staat tussen exp<sub>1</sub> en exp<sub>2</sub> ook een spatie
- count (tab<sub>j</sub>(., d<sub>1</sub>), s) : de procedure levert 'true' indien het aantal tupels in tab<sub>j</sub> met de waarde van d<sub>1</sub> aan de in s gestelde voorwaarde voldoet. Hierbij is s een rij karakters in de vorm van "(on)gelijkteken getal", bijvoorbeeld : ≥ 3
- ssr (d<sub>1</sub>, tab<sub>j</sub>(., d<sub>k</sub>)) : controleert of de relatie d<sub>1</sub> → tab<sub>j</sub>.d<sub>k</sub> bestaat. Zo ja, dan levert de procedure 'true' op, anders 'false'
- alterkey (d<sub>k</sub>) : veranderen van de sleutelwaarde. Normaal mag een sleutelwaarde niet veranderd worden, omdat relaties verbroken kunnen worden. Deze procedure zorgt ervoor dat alle tab<sub>j</sub>.d<sub>1</sub> waarvoor geldt tab<sub>j</sub>.d<sub>1</sub> → d<sub>k</sub>, mee veranderd worden
- deletekey (d<sub>k</sub>) : verwijderen van alle tupels waarvoor geldt tab<sub>j</sub>.d<sub>1</sub> → d<sub>k</sub>, zodanig dat aan alle ssr's voldaan blijft
- newtuple (tab<sub>j</sub>(., d<sub>k</sub>)) : voert voor alle attributen van het tuple met sleutelwaarde d<sub>k</sub> van tabel tab<sub>j</sub>, de expressie bij new uit, controleert de pic en doc en voert de src uit. Vervolgens wordt voor het tuple de new en tuc uitgevoerd en wordt (indien geen fouten) het tuple in de tabel toegevoegd
- newtuple (tab<sub>j</sub>) : hetzelfde als bij newtuple (tab<sub>j</sub>(., d<sub>k</sub>)) maar nu wordt eerst de waarde van d<sub>k</sub> gevraagd
- altertuple (tab<sub>j</sub>(., d<sub>1</sub>)) : voert de expressie bij alter van attribuut d<sub>1</sub> van het actuele tuple van tab<sub>j</sub> uit, controleert de pic en doc en voert de src uit
- altertuple (tab<sub>j</sub>(., all)) : voert voor alle attributen van het actuele tuple van tabel tab<sub>j</sub> de expressie bij alter uit, controleert de pic en doc en voert de src uit. Vervolgens wordt voor het tuple de alter en tuc uitgevoerd en worden (indien geen fouten) de veranderingen aangebracht
- altertuple (tab<sub>j</sub>) : hetzelfde als bij altertuple (tab<sub>j</sub>(., all)) maar nu wordt eerst de waarde van d<sub>k</sub> gevraagd
- updatetuple (tab<sub>j</sub>(., d<sub>1</sub>)) : hetzelfde als bij altertuple (tab<sub>j</sub>(., d<sub>1</sub>)) maar met update in plaats van alter
- updatetuple (tab<sub>j</sub>(., all)) : hetzelfde als bij altertuple (tab<sub>j</sub>(., all)) maar met update in plaats van alter
- updatetuple (tab<sub>j</sub>) : hetzelfde als bij altertuple (tab<sub>j</sub>) maar met update in plaats van alter
- deletetuple (tab<sub>j</sub>(., d<sub>k</sub>)) : verwijdert het tuple van tab<sub>j</sub> met als sleutelwaarde de actuele waarde van d<sub>k</sub>
- deletetuple (tab<sub>j</sub>) : vraagt de waarde van het sleutelattribuut d<sub>k</sub> en verwijdert het tuple van tab<sub>j</sub> met als sleutelwaarde de waarde van d<sub>k</sub>

Indien een attribuut niet mag worden veranderd dan zullen alter of update natuurlijk ontbreken. Indien new, alter, update en delete bij tuple ontbreken, dan wordt alleen de standaard actie uitgevoerd. Dit zijn respectievelijk de procedures newtuple, altertuple,

updatetuple en deletetuple .

Wanneer er een ssr  $d_1 \rightarrow \text{tab}_j.d_k$  gedefinieerd is dan wordt voor het attribuut  $d_1$  de volgende actie uitgevoerd : ( ssr ( $d_1, \text{tab}_j(., d_k)$ ) ! 'true' ! newtuple ( $\text{tab}_j(., d_k)$ ) )

Bij aanroep van ssr ( $d_1, \text{tab}_j(., d_k)$ ) krijgt het attribuut  $d_k$  van de tabel  $\text{tab}_j$  de waarde van  $d_1$ .

Er is een logische variabele before waarmee aangegeven kan worden of een actie vóór ( before = 'true' ) of ná ( before = 'false' ) het toevoegen respectievelijk veranderen van het actuele tuple moet worden uitgevoerd.

Wanneer bijvoorbeeld bij tuple  $t_j$  staat new : ( before ! expressie<sub>1</sub> ! expressie<sub>2</sub> ), dan wordt eerst expressie<sub>1</sub> uitgevoerd, vervolgens wordt  $t_j$  in de tabel toegevoegd en daarna wordt expressie<sub>2</sub> uitgevoerd.

## 6 Ondersteuning van de feitelijke gegevensmanipulatie

Het voorgaande betrof de bewaking van het onderhoud van de gegevens, niet het onderhoud van de gegevens zelf, d.w.z. de feitelijke toevoeging, verandering of verwijdering van gegevens.

Natuurlijk moet voor de eigenlijke gegevensmanipulatie nog een interface naar de gegevens zelf aanwezig zijn.

Daarvoor is minimaal het volgende nodig :

- De manipulatiefuncties
  - geef een tuple uit de tabel ( get )
  - voeg een tuple in de tabel toe ( insert )
  - verander een tuple in de tabel ( modify )
  - verwijder een tuple uit de tabel ( delete )
- de layout van de attributen
  - de naam
  - type ( karakters, integer, real of boolean )
  - opslag ( karakter, binair )
  - verwijzing; een "pointer" of de plaats binnen het record
- controle-informatie ( b.v of een tuple verwijderd mag worden )  
 Indien de gebruikte data-base hiervoor geen goede mogelijkheden heeft kan er een extra attribuut, \$count genaamd, aan elke tabel worden toegevoegd. DATAAL houdt met dit veld rekening indien het aanwezig is.

Bij de tot nu toe aanwezige toepassingen van DATAAL zijn er drie interfaces gemaakt, te weten :

- a. een voor index-sequentiele bestanden.  
 Voor de layout van de attributen wordt aan elk bestand een beschrijving toegevoegd. Voor de controle-informatie moet er een extra attribuut, \$count toegevoegd worden.
- b. een voor het query-systeem DATATRIEVE van Digital
- c. een voor de database RDB van Digital.

## 7 Commando's voor het onderhoud

De gebruiker krijgt een aantal "commando's" - in ons geval in het Nederlands - ter beschikking, waarmee hij aangeeft wat hij wil doen. Deze commando's zijn o.a. :

- nieuw        xyz : in de tabel met naam xyz moet een nieuw tupel worden toegevoegd. Dit commando impliceert dat de procedure newtuple (xyz) wordt opgeroepen en uitgevoerd.
- wijzig        xyz : in de tabel met naam xyz moet een tupel veranderd worden. Dit commando impliceert dat de procedure altertuple (xyz) wordt opgeroepen en uitgevoerd.
- bijwerken    xyz : in de tabel met naam xyz moet een tupel veranderd worden. Dit commando impliceert dat de procedure updatetuple (xyz) wordt opgeroepen en uitgevoerd.
- verwijderen xyz : in de tabel met naam xyz moet een tupel verwijderd worden. Dit commando impliceert dat de procedure deletetuple (xyz) wordt opgeroepen en uitgevoerd.
- zien          xyz : een tupel uit de tabel met naam xyz wordt op het scherm zichtbaar. De gebruiker wordt de waarde gevraagd van het sleutelattribuut van het tupel dat hij wil zien.
- continue    xyz : herhaling voorgaand commando.

De gebruiker mag voor commando's en namen van tabellen en attributen willekeurige unieke afkortingen gebruiken, zo is con maar ook c toegestaan voor continue.

Er zijn enkele karakters welke tijdens het invoeren van gegevens een speciale betekenis hebben en niet als eerste karakter van een gegeven gebruikt mogen worden.

Deze karakters zijn :

? het help karakter

Indien help bij het attribuut is gedefinieerd wordt de erbij staande tekst afgedrukt. Anders wordt indien doc is gedefinieerd de erbijstaande expressie afgedrukt. In de overige gevallen wordt afgedrukt het maximaal aantal karakters dat voor het attribuut mag worden ingevoerd.

§ het edit karakter

Voor het actuele tupel kunnen alle reeds ingevoerde gegevens veranderd worden.

@ het escape karakter

Het invoeren van gegevens voor het actuele tupel wordt gestopt. Het laatste actuele attribuut van het voorgaande actuele tupel wordt actueel.

@@ het all-escape karakter

Alle akties worden gestopt en er wordt terug gegaan naar commando niveau.

## 8 Ervaringen met DATAAL in het gebruik

De ervaringen in het gebruik van DATAAL kan naar drie gezichtspunten geordend worden :

- a. naar dat van de eindgebruiker, dus degene die de gegevens muteert en raadpleegt;
- b. naar dat van degene die de beschrijvingen in DATAAL maakt, en
- c. naar dat van de maker van de interface naar de eigenlijke gegevens.

### 8.1 Van de eindgebruiker (a)

Bij het muteren en raadplegen blijkt dat

- de eindgebruiker nauwelijks instructie ( cursus ) nodig heeft; het een à twee keer voordoen van de werking is meestal voldoende;
- de eindgebruiker zijn eigen, in zijn dagelijkse werk gebruikelijke, namen van velden en tabellen kan kiezen, omdat, ook als dezelfde data gebruikt worden, per toepassing een beschrijving in DATAAL gemaakt kan worden;
- de eindgebruiker direkt die akties uit kan voeren die hij wenst, want indien hierdoor andere tabellen moeten worden aangepast dan kan dit door een goede specificatie in de DATAAL beschrijving, automatisch worden verzorgd;
- de eindgebruiker van een verandering van de feitelijke gegevensopslag als gevolg van bijvoorbeeld een nieuw database systeem, *niets* merkt.
- de eindgebruiker geen kennis nodig heeft van database-commando's of (kriptische) menu's.

Bij het muteren en raadplegen blijken ook enkele nadelen van het huidige systeem, zoals

- de onmogelijkheid om de schermopmaak te veranderen.  
(Hieraan is tot nu toe geen aandacht besteed.);
- het soms traag of erg traag werken van de gegevens-manipulatie.  
(Dit is een gevolg van de noodzakelijke algemene interface naar de database toe.);
- het ontbreken van een eenvoudige query-taal, voor bijvoorbeeld het geven of afdrukken van alle tupels uit een tabel welke aan een bepaalde voorwaarde voldoen. (Voor sequentiële gegevensbestanden is hierin voorzien middels het door onszelf ontwikkelde QS-systeem.)

Samenvattend blijkt met name de eenvoud van het DATAAL-systeem het grootste voordeel voor de eindgebruikers op te leveren.

### 8.2 Van de maker van de beschrijvingen in DATAAL (b)

Het maken van de beschrijvingen in DATAAL blijkt meestal erg eenvoudig te zijn. Ook veranderingen als gevolg van wensen van de gebruiker of veranderingen van de onderliggende gegevensopslag zijn snel en eenvoudig aan te brengen.

Er zijn ook tekortkomingen. Een hiervan is het ontbreken van een herhalingsopdracht. (Denk hierbij aan een klant die een of meer bestellingen doet.) Verder zijn geen tabel- en database-voorwaarden geïmplementeerd. Deze zijn echter zelden nodig en bovendien is dit probleem bijna altijd, zoals ook bij vele database's gebruikelijk, op te lossen door een of meer extra velden in relevante tabellen of extra tabellen in de database op te nemen.

### 8.3 Van de maker van de interface (c)

Het maken van de interface naar de feitelijke data blijkt vaak geen eenvoudige zaak te zijn. De database-pakketten bieden meestal geen elegante *algemene* interface naar andere programatuur. Enkele obstakels zijn :

- hoe is de layout van de attributen( zie § 6 )?
- hoe moeten de feitelijke gegevens naar de ( eenvoudige ) DATAAL-types worden geconverteerd?
- hoe *algemeen* moeten de manipulatiefuncties ( zie § 6 ) gemaakt worden?
- "echte" variabele lengte van velden is in de database-wereld zo goed als onbekend.

Het lijkt erop dat binnen niet al te lange tijd door ontwikkelingen in de database-wereld de onder punt c genoemde obstakels geheel of gedeeltelijk zullen verdwijnen.

Overigens begint een belangrijk aspect van DATAAL, namelijk het via één interface ( "chapeau" ) kunnen muteren en raadplegen van verschillende andere database-systemen aandacht te krijgen. We denken hierbij onder andere aan INGRES <sup>1</sup>.

---

<sup>1</sup>zie bijvoorbeeld : Elektronica Aktueel, 7 november 1986, p.7-8

## IN 1986 REEDS VERSCHENEN

- 202 J.H.F. Schilderinck  
Interregional Structure of the European Community. Part III
- 203 Antoon van den Elzen and Dolf Talman  
A new strategy-adjustment process for computing a Nash equilibrium in a noncooperative more-person game
- 204 Jan Vingerhoets  
Fabrication of copper and copper semis in developing countries. A review of evidence and opportunities
- 205 R. Heuts, J. van Lieshout, K. Baken  
An inventory model: what is the influence of the shape of the lead time demand distribution?
- 206 A. van Soest, P. Kooreman  
A Microeconomic Analysis of Vacation Behavior
- 207 F. Boekema, A. Nagelkerke  
Labour Relations, Networks, Job-creation and Regional Development. A view to the consequences of technological change
- 208 R. Alessie, A. Kapteyn  
Habit Formation and Interdependent Preferences in the Almost Ideal Demand System
- 209 T. Wansbeek, A. Kapteyn  
Estimation of the error components model with incomplete panels
- 210 A.L. Hempenius  
The relation between dividends and profits
- 211 J. Kriens, J.Th. van Lieshout  
A generalisation and some properties of Markowitz' portfolio selection method
- 212 Jack P.C. Kleijnen and Charles R. Standridge  
Experimental design and regression analysis in simulation: an FMS case study
- 213 T.M. Doup, A.H. van den Elzen and A.J.J. Talman  
Simplicial algorithms for solving the non-linear complementarity problem on the simplotope
- 214 A.J.W. van de Gevel  
The theory of wage differentials: a correction
- 215 J.P.C. Kleijnen, W. van Groenendaal  
Regression analysis of factorial designs with sequential replication
- 216 T.E. Nijman and F.C. Palm  
Consistent estimation of rational expectations models

- 217 P.M. Kort  
The firm's investment policy under a concave adjustment cost function
- 218 J.P.C. Kleijnen  
Decision Support Systems (DSS), en de kleren van de keizer ...
- 219 T.M. Doup and A.J.J. Talman  
A continuous deformation algorithm on the product space of unit simplices
- 220 T.M. Doup and A.J.J. Talman  
The 2-ray algorithm for solving equilibrium problems on the unit simplex
- 221 Th. van de Klundert, P. Peters  
Price Inertia in a Macroeconomic Model of Monopolistic Competition
- 222 Christian Mulder  
Testing Korteweg's rational expectations model for a small open economy
- 223 A.C. Meijdam, J.E.J. Plasmans  
Maximum Likelihood Estimation of Econometric Models with Rational Expectations of Current Endogenous Variables
- 224 Arie Kapteyn, Peter Kooreman, Arthur van Soest  
Non-convex budget sets, institutional constraints and imposition of concavity in a flexible household labor supply model
- 225 R.J. de Groof  
Internationale coördinatie van economische politiek in een twee-regio-twee-sectoren model
- 226 Arthur van Soest, Peter Kooreman  
Comment on 'Microeconomic Demand Systems with Binding Non-Negativity Constraints: The Dual Approach'
- 227 A.J.J. Talman and Y. Yamamoto  
A globally convergent simplicial algorithm for stationary point problems on polytopes
- 228 Jack P.C. Kleijnen, Peter C.A. Karremans, Wim K. Oortwijn, Willem J.H. van Groenendaal  
Jackknifing estimated weighted least squares
- 229 A.H. van den Elzen and G. van der Laan  
A price adjustment for an economy with a block-diagonal pattern
- 230 M.H.C. Paardekooper  
Jacobi-type algorithms for eigenvalues on vector- and parallel computer
- 231 J.P.C. Kleijnen  
Analyzing simulation experiments with common random numbers



- 232 A.B.T.M. van Schaik, R.J. Mulder  
On Superimposed Recurrent Cycles
- 233 M.H.C. Paardekooper  
Sameh's parallel eigenvalue algorithm revisited
- 234 Pieter H.M. Ruys and Ton J.A. Storcken  
Preferences revealed by the choice of friends
- 235 C.J.J. Huys en E.N. Kertzman  
Effectieve belastingtarieven en kapitaalkosten
- 236 A.M.H. Gerards  
An extension of König's theorem to graphs with no odd- $K_4$
- 237 A.M.H. Gerards and A. Schrijver  
Signed Graphs - Regular Matroids - Grafts
- 238 Rob J.M. Alessie and Arie Kapteyn  
Consumption, Savings and Demography
- 239 A.J. van Reeken  
Begrippen rondom "kwaliteit"
- 240 Th.E. Nijman and F.C. Palmer  
Efficiency gains due to using missing data. Procedures in regression models
- 241 Dr. S.C.W. Eijffinger  
The determinants of the currencies within the European Monetary System

## IN 1987 REEDS VERSCHENEN

- 242 Gerard van den Berg  
Nonstationarity in job search theory
- 243 Annie Cuyt, Brigitte Verdonk  
Block-tridiagonal linear systems and branched continued fractions
- 244 J.C. de Vos, W. Vervaat  
Local Times of Bernoulli Walk
- 245 Arie Kapteyn, Peter Kooreman, Rob Willemse  
Some methodological issues in the implementation  
of subjective poverty definitions
- 246 J.P.C. Kleijnen, J. Kriens, M.C.H.M. Lafleur, J.H.F. Pardoel  
Sampling for Quality Inspection and Correction: AOQL Performance  
Criteria
- 247 D.B.J. Schouten  
Algemene theorie van de internationale conjuncturele en structurele  
afhankelijkheden
- 248 F.C. Bussemaker, W.H. Haemers, J.J. Seidel, E. Spence  
On  $(v,k,\lambda)$  graphs and designs with trivial automorphism group
- 249 Peter M. Kort  
The Influence of a Stochastic Environment on the Firm's Optimal Dynamic  
Investment Policy
- 250 R.H.J.M. Gradus  
Preliminary version  
The reaction of the firm on governmental policy: a game-theoretical  
approach
- 251 J.G. de Gooijer, R.M.J. Heuts  
Higher order moments of bilinear time series processes with symmetrically  
distributed errors
- 252 P.H. Stevers, P.A.M. Versteijne  
Evaluatie van marketing-activiteiten

**Bibliotheek K. U. Brabant**



**17 000 01059968 7**