

monitors

COBOL

från grunden

Exempelsidor

Peter Sterwe

Lär dig grunderna i COBOL-programmering på ett översiktligt och pedagogiskt sätt från företaget som har mer än trettio års erfarenhet av utbildning inom IBM z/OS Mainframe.

Nivåer

- ◆ Nivåindikatorer 01 - 49 används för att indikera inbördes förhållande
- ◆ **Gruppnamnet** avser samtliga underliggande element
- ◆ Grupper kan förekomma på flera nivåer

Exempelsidor

```
* Detta är en Kund
01 Kund.
  05 Namn.
    10 Fornamn          Pic X(20) .
    10 Efternamn        Pic X(30) .
  05 Adress.
    10 Postnr           Pic X(05) .
    10 Postadress        Pic X(25) .
```

Nivåindikatorerna 01-49 indikerar olika nivåer. Ett lägre nummer indikerar en överordnad nivå. I exemplet ovan så ser du att "stegen" är 5, men det behöver inte vara så.

Vid referens till ett gruppnamn, t.ex. Namn, så avses samtliga element som är underordnade denna grupp, i detta fall både Fornamn och Efternamn.

◆ Namnlösa variabler

- ◆ Dataelement som ej skall refereras kan vara namnlösa
- ◆ Dataelement som ej skall refereras kan ha det reserverade namnet `Filler`

```
01 Kundrac  
  05 Namn  
    10 Fornamn          Pic X(20).  
    10                  Pic X(05).  
    10 Efternamn        Pic X(30).  
    10 Filler           Pic X(10).
```

Ibland händer det att man behöver lite utfyllnad i en struktur, eller att man helt enkelt inte behöver referera vissa platser i en datapost. Dessa variabler kan ha det reserverade ordet *Filler*, eller så utelämnar du helt enkelt namnet.

Namn i samma struktur måste vara unika, och genom att inte namnge en variabel alls, så besparar du dig problemet med att hitta ett namn, men det förutsätter att du inte ha behov av att referera detta data.

Numeriska variabler

- ◆ Kan innehålla ett "tänkt" decimaltecken
 - **V** i deklarationen anger decimaltecknets placering

```
77 Summa          Pic S9(03)      Value Zero.  
77 Totsummala     Pic S999V99.  
77 Totsummalb     Pic S9(03)V9(02).
```

```
* Kan innehålla värdet -999,99  
Fill +999,99
```

- Decimaltecknet "syns" ej vid utskrift
- För utskrift finns redigeringskonstanter

När du behöver använda decimaltal, så anger du ett V på den plats där du vill att ditt decimaltecken finns. Detta V tar ingen plats i din variabel, utan är endast information till COBOL att du vill ha det antal decimaler i variabeln som antalet nior efter tecknet V beskriver.

Variablerna Totalsummala och Totalsummalb i exemplet ovan har samma noggrannhet i värdet men den senare är deklarerad med repetitionsfaktor. Deklarera alltid för maximal tydlighet.

Notera också att om du skulle göra en utskrift av en V-deklarerad variabel, så syns bara siffrorna. Decimaltecknet syns inte. För att decimaltecknet skall synas vid utskrift, så måste du flytta värdet till en variabel som innehåller en s.k redigeringskonstant, som beskriver decimaltecknet.

Add

- ◆ Addera innehållet i en operand till innehållet i en annan variabel, lagra svaret i en tredje variabel

```
Add
    Summa to Delsumma
    Giving Totalsumma
End-Add
```

- ◆ Med avrundning

```
Add
    Summa to Delsumma
    Giving Totalsumma Rounded
End-Add
```

Innehållet i variabeln `Summa` adderas till innehållet i variabeln `Delsumma` och resultatet sparas i variabeln med namnet `Totalsumma`.

Det undre exemplet har kompletterats med operanden `Rounded`, vilket innebär att slutresultatet kommer att avrundas matematiskt till det antalet decimaler som variabeln `Totalsumma` har.

◆ Reference Modification

- ◆ Position och längd kan anges vid referens till ett dataelement i i display-format

```
datanamn [startpos:[längd]]
```

- startpos anger teckennummer räknat från 1
- längd anger antalet tecken som avses
- startpos och längd kan båda vara konstanter, numeriska dataelement eller aritmetiska uttryck
- Utvärderingen av startpos måste vara större än noll(0) och ej utanför posten
- Standardvärdet för längd är "resten av dataelementet"

När du behöver referera en del av en variabel kan du använda det som kallas *Reference Modification*. Detta kallas oftast för *Substring* i andra programspråk.

När du använder denna teknik vid en Move, så ligger styrkan i att både sändande och mottagande variabel kan vara en delmängd. Du anger endast vilken position du avser samt eventuellt hur många tecken, separerat med ett kolon-tecken. Skulle den mottagande variabeln ha plats för fler tecken än de du angivit, så sker en utfyllnad, alternativt att du får en trunkering.

Längdangivelsen för den mottagande variabeln, om du anger en längd, får då ej vara så stor att du skulle nå utanför den mottagande variabeln. Detta ger ett kompileringsfel.

Om du inte anger en längd, avser både sändande och mottagande, så tolkas detta som att du avser hela variabeln från den angivna startpositionen.

Tabeller i tabeller

♦ Exempel (1)

```

01 Tabellerna.
05 Veckodagar.
   10 Pic X(10) Value 'Måndagen'.
   10 Pic X(48)
   10 Pic X(10) Value 'Tisdagen'.
   10 .....
05 Dagarna Redefines Veckodagar.
   10 Occurs 7.
      15 Dagen Pic X(10).
      15 Antal Pic 99 Occurs 24.

String 'Det var '
      Antal(Dnum,Tnum)
      ' stycken på ' Dagen(Dnum) 'kl ' Tnum
      Delimited By Size into Meddelande
End-String
Display Meddelande

```

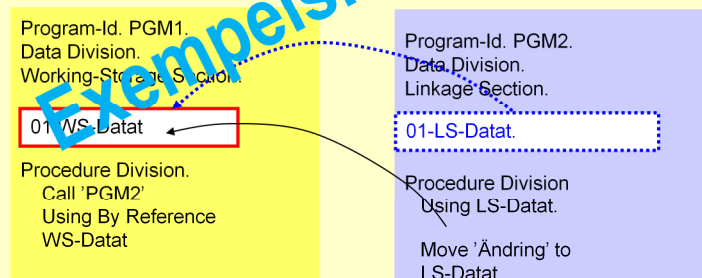
I denna tabell så finns det en yttre tabell med namnet Dagarna. Varje element i tabellen består av ett fält med namnet Dagen samt en inre tabell med namnet Antal, som förekommer 24 gånger. För att referera behövs ett dagnummer samt ett klockslag.

Om du behöver söka igenom hela tabellen så kan du göra på flera sätt.

Anropa subprogram

◆ Parameteröverföring - By Reference

- Mottagande program har en pekare i Linkage Section som kommer att peka ut överfört data
- Förändringar av det anropade programmet sker i anroparens data



Som du ser av denna bild så försöker jag visa att det mottagande programmet har en bild över parametern som sändande programmet överför.

Bilden av elementet eller gruppen skall ha samma utseende i de båda programmen, men det mottagande programmet har sin bild i Linkage Section. Namnet på strukturen eller elementen behöver dock ej vara samma.

Som du ser så har det mottagande programmet en Using-formulering i Procedure Division som refererar en 01-nivå i Linkage Section. När det mottagande programmet refererar en mottagen parameter i Linkage Section så sker referensen till det sändande programmens data i dess Working Storage.

Vi kommer att beskriva detta mer på senare bilder.

♦ Integer-Of-Date

- ♦ Konverterar från Gregorian-format till Integer-format
 - Argumentet måste vara ett giltigt datum i Gregorian-format (YYYYMMDD)
 - Returnerar ett 7 tecken långt numeriskt värde

```
9999999 = function Integer-Of-Date (gregdatum)
```

Om du har ett giltigt datum i Gregorian-format kan du med funktionen `Integer-Of-Date` konvertera det till ett numeriskt värde.

Mottagare av resultatet är en sju (7) tecken lång numerisk variabel.