training people

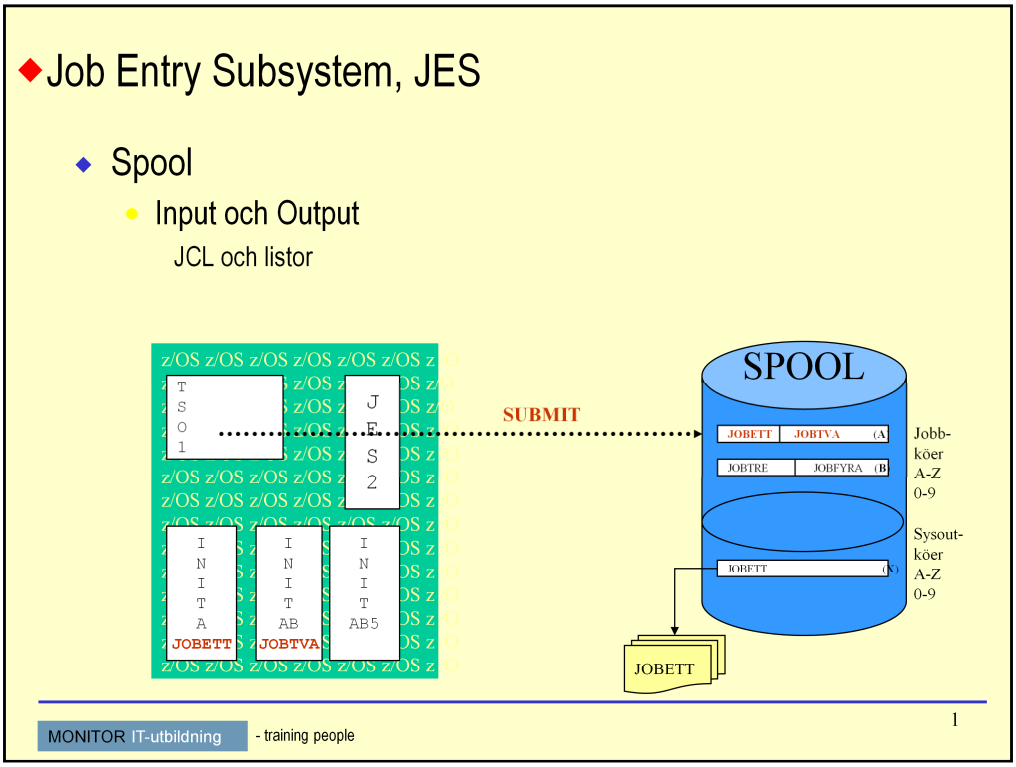
monitors

z/OS JCL och Utilities från grunden

Exempelsidor

Peter Sterwe

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



Spool

Simultaneous Peripheral Operation OnLine är ett dataset som JES använder för att lagra s.k JOB (JCL) och körningsresultat (*Job Log, Sysout*).

Input och output

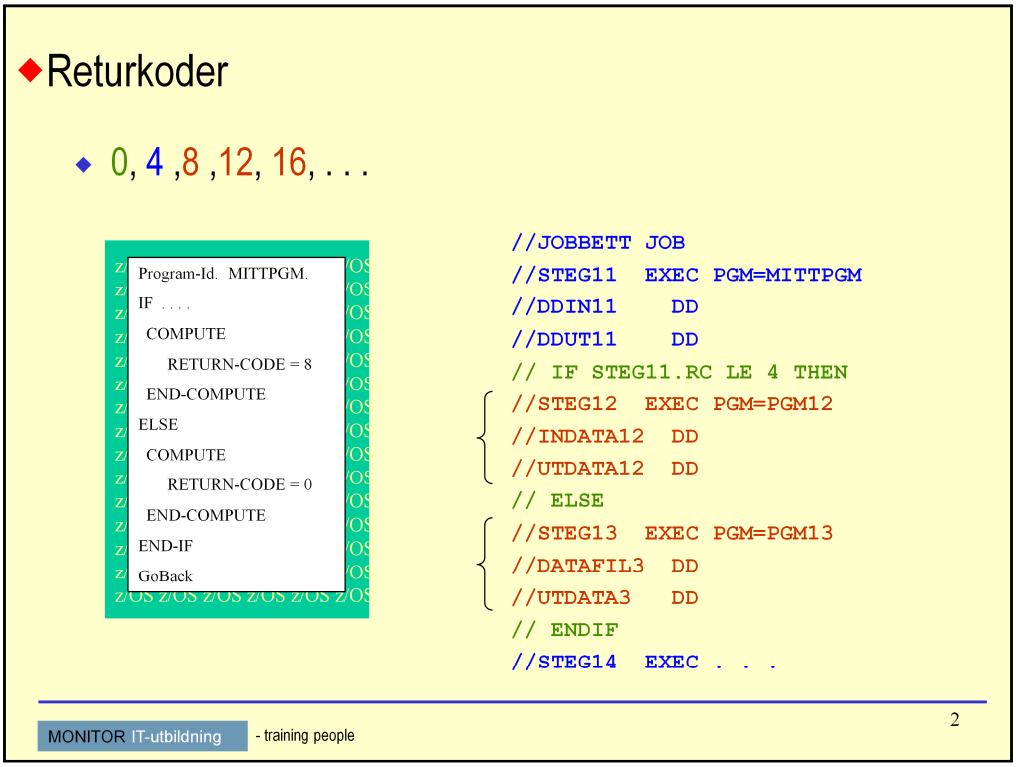
JCL beskriver vilket program som skall exekveras vid sett specifikt tillfälle och vilka dataset som skall göras tillgängliga för detta program. Detta är ett s.k JOB. Ett jobb kan bestå av flera s.k jobb-steg (*job step*). Dessa jobbsteg exekveras i sekvens d v s ett program i taget.

Jobbet måste placeras på en av de jobb-köer som finns i systemet. Jobbköerna identifieras med bokstäver A-Z samt siffror 0-9. Vilka jobbköer som används i systemet är installationsberoende. För att jobbet skall kunna köras av systemet, så måste det finnas en ledig s.k initierare (*initiator*) som är uppstartad för att ta hand om jobb med den jobbklass som beskrivs i JCL:en. Initieraren ’väntar på’ en eller flera jobbklasser och tar hand om jobben i sekvens.

På denna bild ser vi ett system med tre (3) initierade uppstartade. Detta innebär att maximalt tre batchprogram kan exekvera samtidigt.

När jobbet är klart placeras en jobblogg på en Sysout-kö. Vad som sedan sker med denna lista beror på om det finns någon systemskrivare som tar hand om listor i denna klass eller ej.

Vilken sysoutklass som listan hamnar i bestäms med JCL.



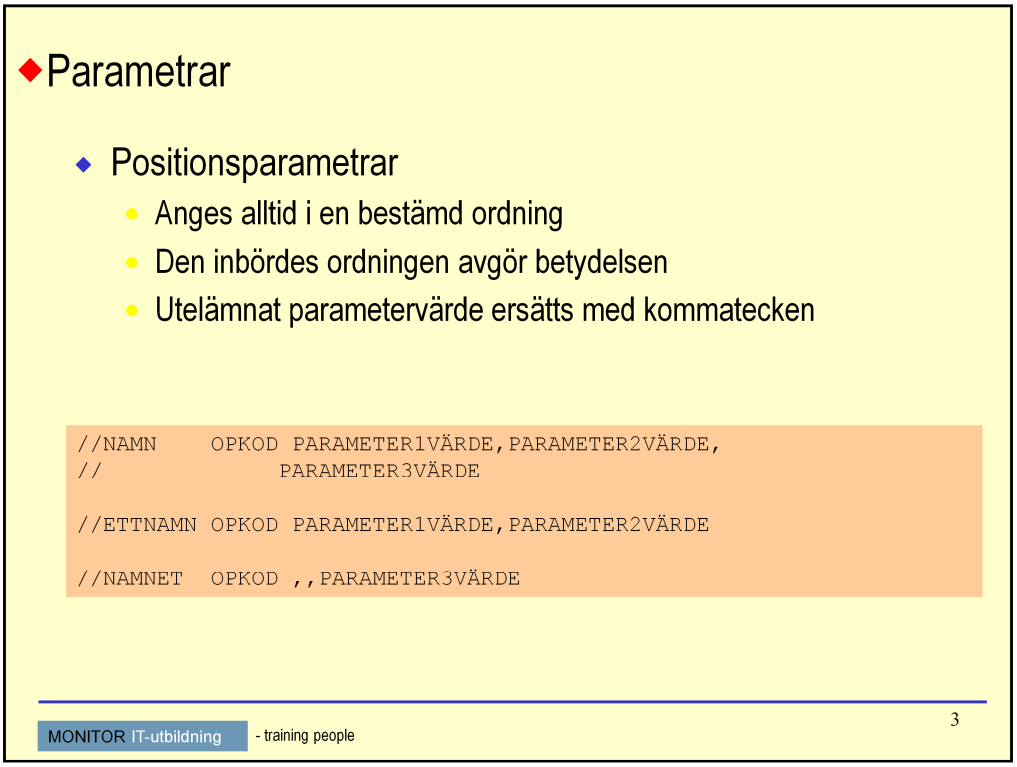
Returkoder

Innan ett program avslutar så kan det returnera en returkod. Denna blir tillgänglig vid ett ’stegbyte’ och kan testas med JCL.

Returkoder är ett sätt för ett program att kommunicera med omgivningen.

Returkod 0 innebär vanligen att programmet avslutats utan problem. Vanliga returkoder är multiplar av 4, men det behöver inte vara så. En programmerare kan signalera returkoder mellan 0 och 32767.

Returkoder ’sparas undan’ av systemet så att de kan testas i efterföljande steg, inte nödvändigtvis det närmast efterföljande.



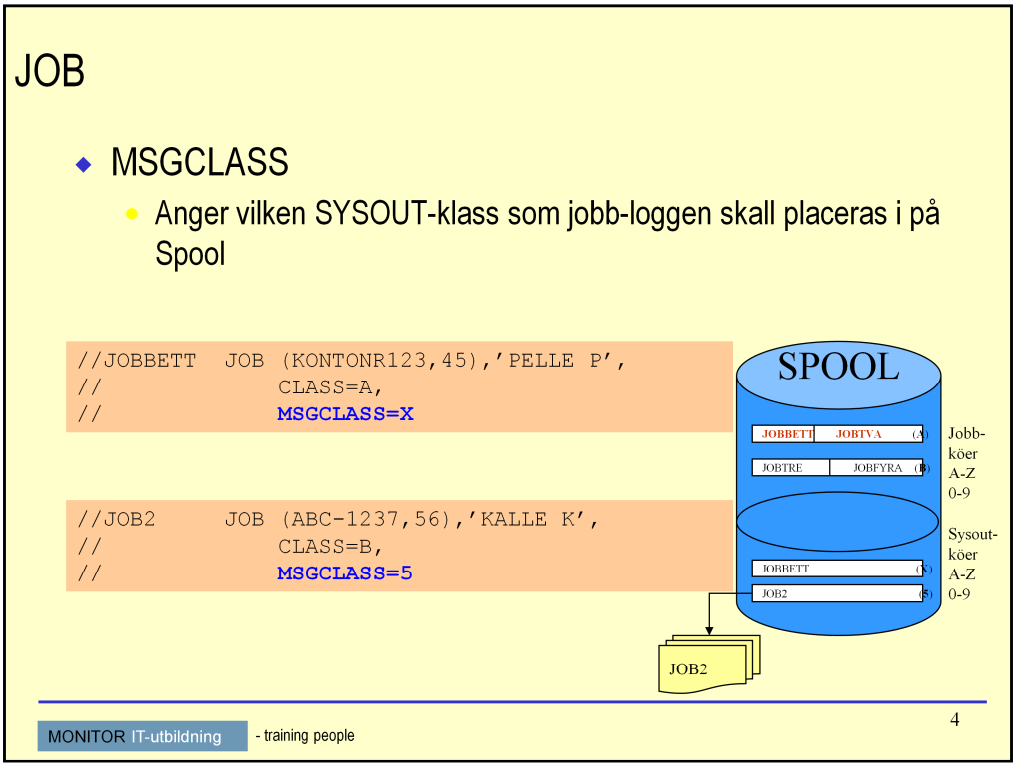
Positionsparametrar

Parametrar i ett JCL-uttryck är alltid valfria. Om en parameter är en positionsberoende parameter så är det den inbördes ordningen som avgör betydelsen. Detta innebär att man inte kan utelämna ett parametervärde utan vidare. Det utelämnade parametervärdet ersätts med ett kommatecken (,).

Om man vill utelämna det sista parametervärdet i en lista behöver det ej ersättas med ett kommatecken.

Alla parametervärden åtskiljs med kommatecken.

Det får inte förekomma blanktecken i uttrycken.



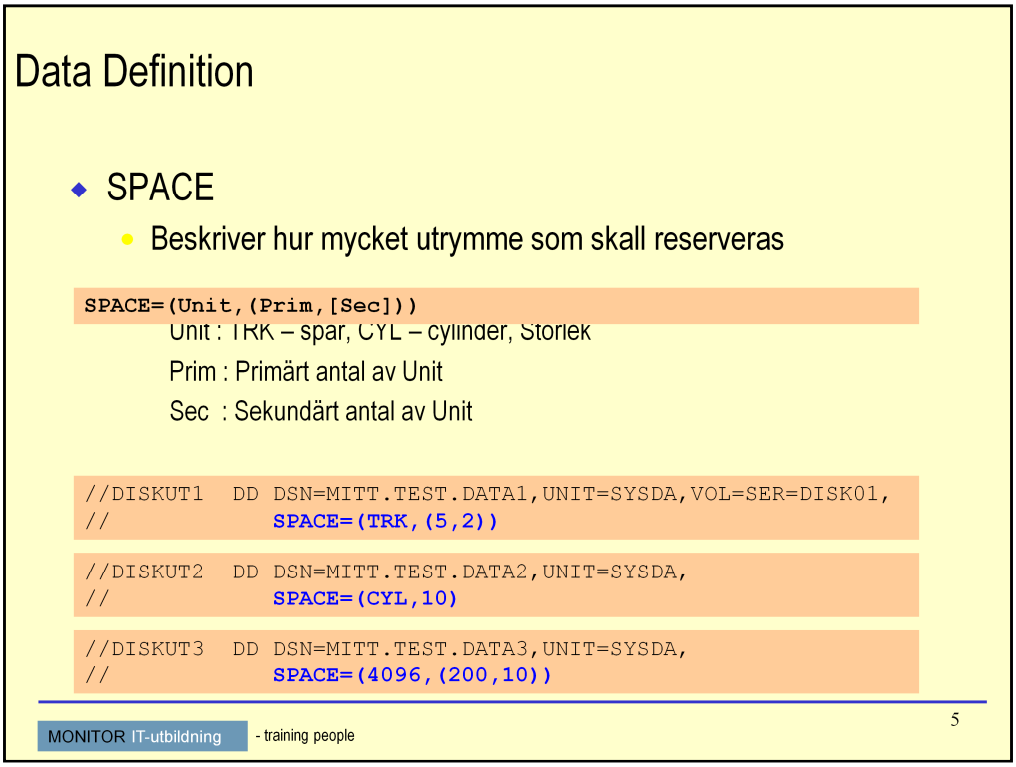
MSGCLASS

Varje körning av ett jobb resulterar i en jobblogg (*job log*). Denna innehåller information om körningen som t.ex starttid, sluttid och exekveringstid. Man ser även de returkoder som de olika jobbstegen resulterat i.

Det finns ingen förutbestämd betydelse av de olika Sysout-klasserna.

Vilka Sysout-klasser som kan användas är helt och hållet installationsberoende. För vissa klasser kan det finnas olika skrivare som skriver ut informationen på papper.

Andra klasser har inga skrivare utan är s.k Hold-klasser. Sådana klasser används för att man skall kunna 'titta på' körningsresultatet med verktyg som t.ex SDSF eller SysView.



Unit

Beskriver vilken ”mått” som skall användas när ett nytt dataset skall skapas.

TRK – anger att utrymme skall reserveras i enheten spår.

CYL – anger att utrymme skall reserveras i enheten cylinder.

Storlek – anger blockstorlek som skall reserveras.

Prim

Här anges hur många enheter (*unit*) som skall reserveras när datasettet skapas. Detta utrymme måste finnas tillgängligt annars signaleras JCL-fel.

När ett dataset fylls med data så kan det primära utrymmet bli helt fyllt och då kan ytterligare utrymme reserveras enligt specifikationen i Sec.

Sec

Anger hur många enheter som skall reserveras när datasettet är fullt. Den sekundära allokeringen/tilldelningen kan ske högst 15 gånger, sedan betraktas datasettet som fullt. Denna prameter är inte obligatorisk.

Abend B37 - Det finns inte mer utrymme på volymen eller samtliga 16 extents har använts.

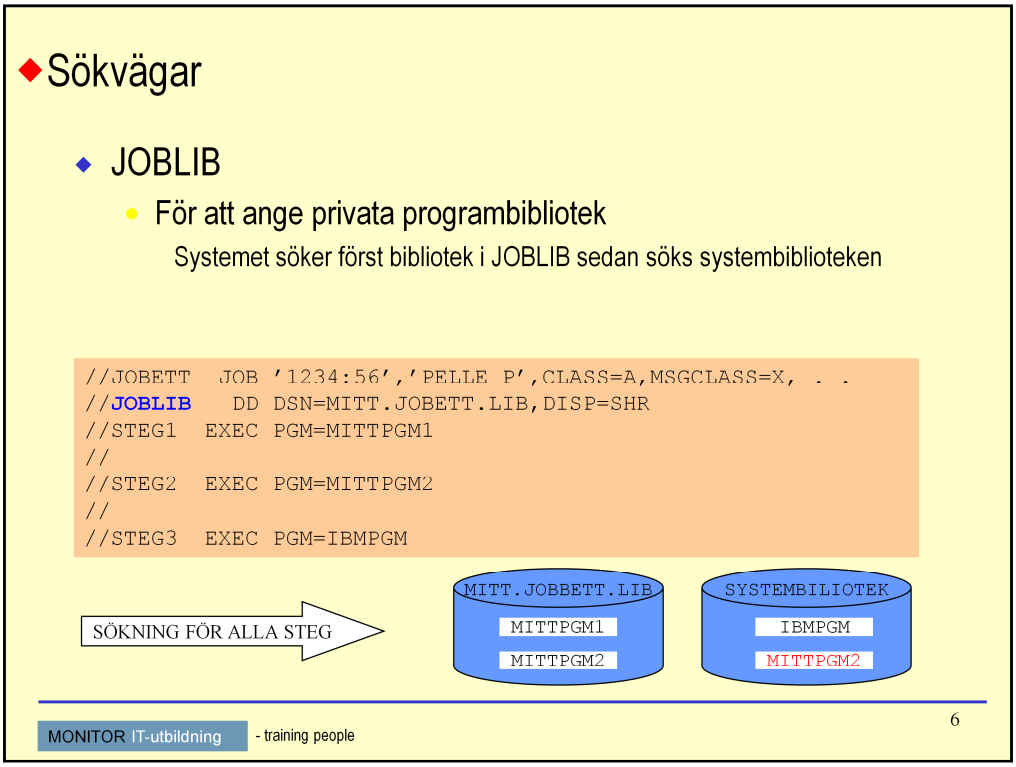
Mer information finns i meddelande IEC030I B37-rc.....

Abend D37 - När det primära utrymmet är fyllt och det inte finns något värde för sekundär allokering.

Mer information i meddelande IEC031I D37-rc....

Abend E37 - Kan jämföras med B37 men avser dataset av type bibliotek (*Partitioned Data Set, PDS*).

Mer information i meddelande IEC032I E37-rc....

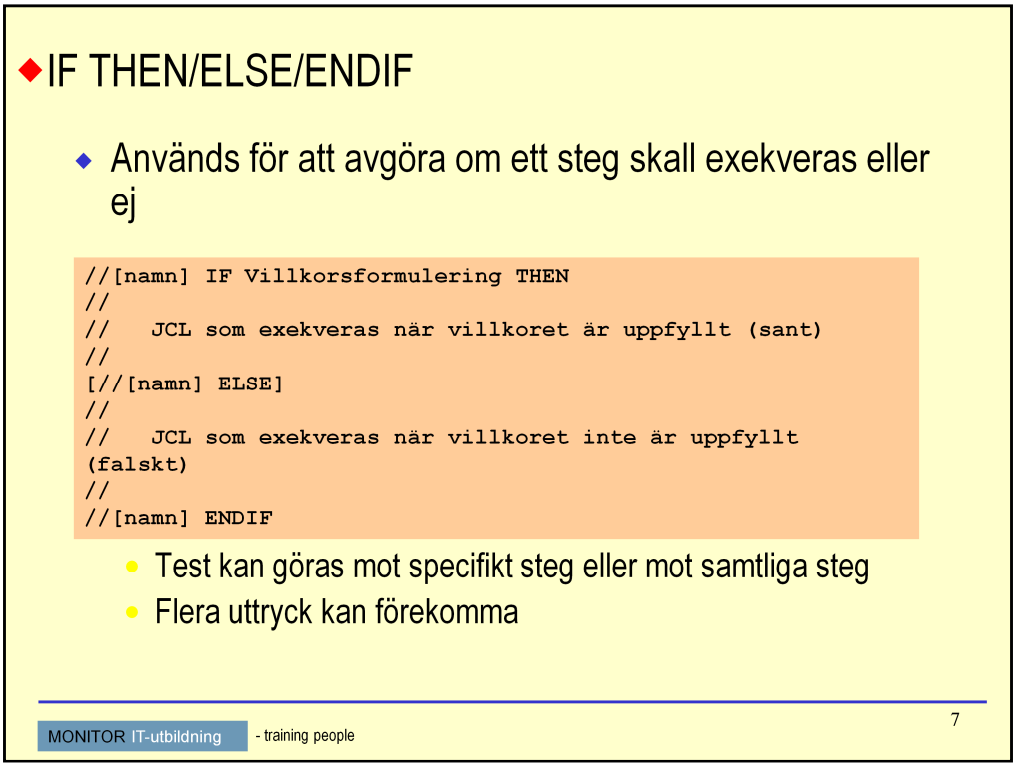


JOBLIB används för att informera systemet om vilket eller vilka programbibliotek som skall sökas för att finna de program som refereras i samtliga steg.

Det eller de bibliotek som finns med i JOBLIB kommer alltid att sökas FÖRE systembiblioteken. Skulle samma programnamn finnas i systembiblioteken så är det ändå det program som påträffas i JOBLIB som exekveras.

Detta kan utnyttjas för att testa nya versioner av program. De förändrade/uppdaterade programversionerna placeras på ett bibliotek som beskrivs med JOBLIB-uttrycket.

Skulle man upptäcka något programfel och behöver göra en omkörning och använda den gamla versionen så kan man bara plocka bort JOBLIB-kortet och systemet hämtar då den föregående versionen från systembiblioteken.



IF THE, ELSE samt ENDIF är enskilda JCL-uttryck som används vid villkorsformulering för att ”omsluta” den JCL som berörs av villkoret eller villkoren.

”OM villkoret är uppfyllt,

exekvera dessa rader

ANNARS

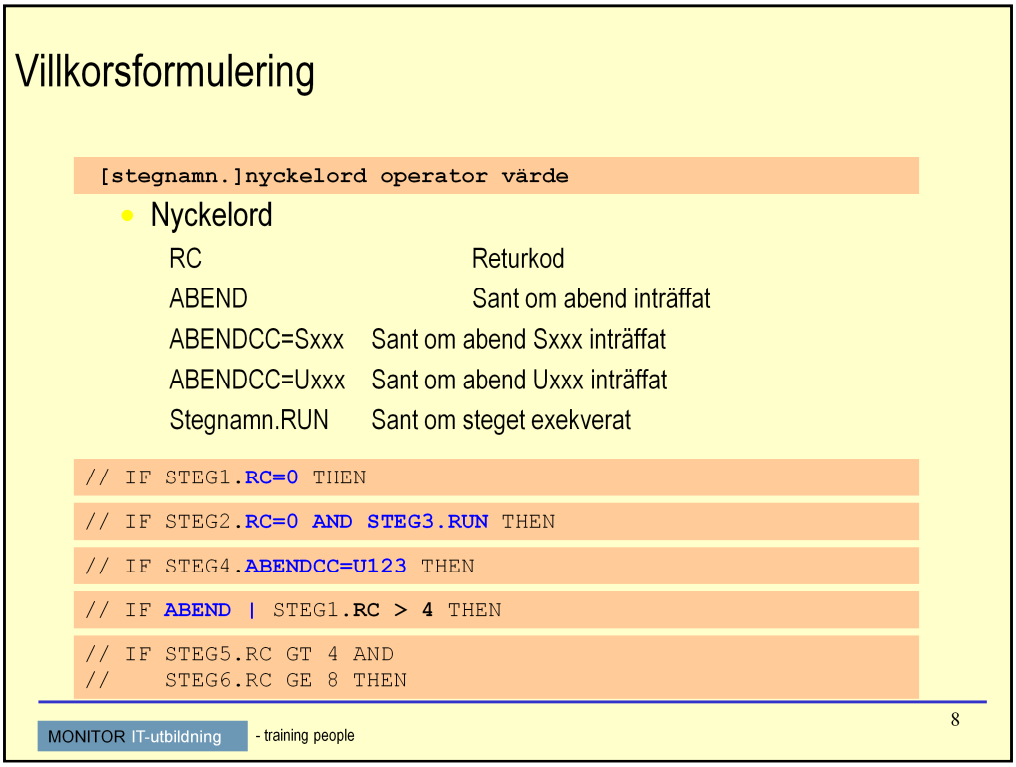
exekvera dessa rader

SLUT-OM”

Notera att villkoret ELSE är valfritt, vilket innebär att det inte behöver finnas ett alternativ, om villkoret inte skulle vara uppfyllt.

Man kan även ha s.k. ”nästlade” formuleringar, vilket innebär att man kan skriva på detta sätt:

```
//A    IF villkorsformulering1 THEN
//. . . . .
//     ELSE
//B        IF villkorsformulering2 THEN
//. . . . .
//         ELSE
//. . . . .
//B        ENDIF
//A    ENDIF
```



Operatorer förenar nyckelord med värden, för att tillsammans utgöra en villkorsformulering.

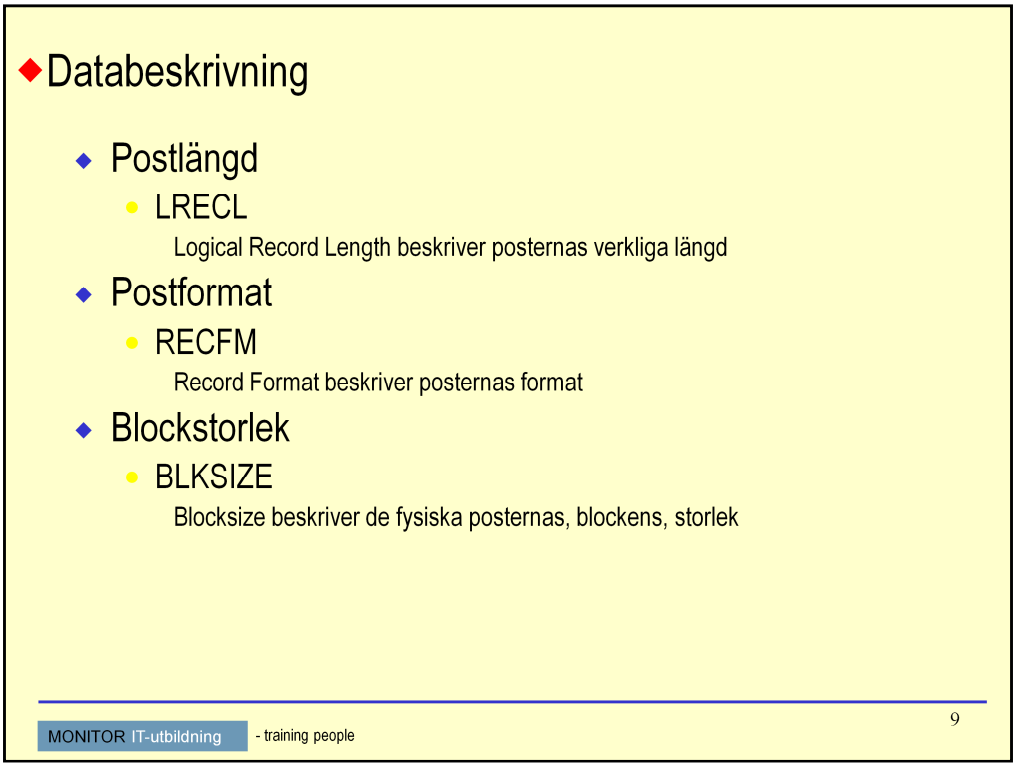
Exempel 1 är en enkel formulering där steget kommer att exekveras om STEG1 fick returkod 0.

Exempel 2 är ett uttryck med sammansatta villkor. Steget exekveras BARA om STEG2 fick returkod 0 OCH STEG3 har exekverats. Båda villkoren måste vara uppfyllda.

I nästa exempel kontrolleras om STEG4 har fått en abendkod U123. När detta är sant så exekveras jobbsteget.

Exempel 4 är också ett exempel på sammansatta villkor, men operatorerna är ej alfabetiska. Villkoret är uppfyllt om ett av delvillkoren uppfylls, d v s OM det har varit någon abend i tidigare steg ELLER om STEG1 fick returkod 5 eller högre.

Det sista exemplet visa att man skriva uttrycken på flera rader. Detta kan öka tydligheten vid sammansatta villkor. Inga speciella fortsättningstecken behövs. Inled fortsättningsraden med //.



Posternas verkliga längd beskrivs med parametern LRECL (*Logical Record Length*) när ett dataset skapas. Denna information sparas sedan i diskvolymens VTOC i ett DSCB1 (se sid 4-12)eller i etiketter på magnetband.

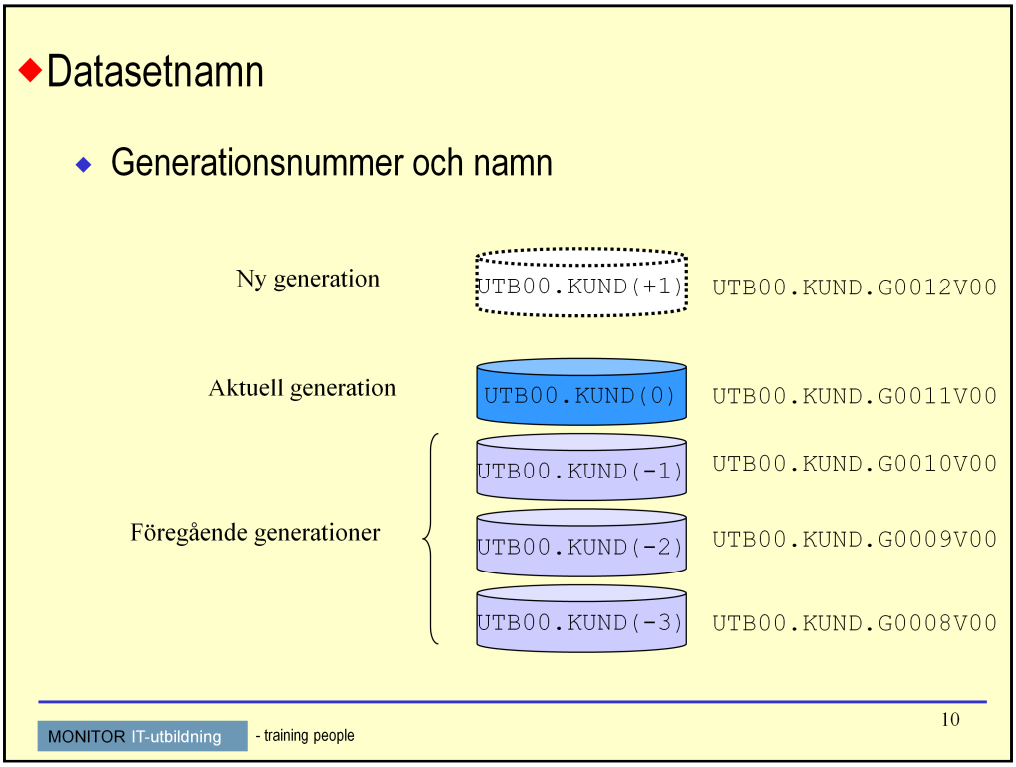
Programmets postbeskrivning måste stämma överens med filens verkliga fysiska utseende. Detta kontrolleras av systemet när programmet gör ”Open” av filen.

Skulle det vara några olikheter så kommer programmet att avslutas med abend av typen 013-rc, där rc varierar beroende på vilken typ av fel som systemet upptäck.Meddelandet IEC141I beskriver ett femtiotal olika värden för rc i detalj.

Vid läsning och/eller skrivning så transporteras posterna i fysiska poster eller mer vanligt kallat block. Blockstorleken (*blocksize*) kan man beskriva själv när datasettet skapas. Storleken måste alltid vara en multipel av postlängden.

Vid lagring på disk eller tape så separeras blocken fysiskt. Det innebär att det reserveras utrymme mellan två block som inte kan användas till något vettigt. Om blocken är förhållandevis små, så blir det fler sådana utrymmen som inte kan utnyttjas och ett dataset rymmer då inte så mycket verklig data som det skulle göra om blocken hade varit större.

Det optimala är att låta systemet avgöra hur stora blocken skall vara. Olika enheter har olika optimala storlekar, men systemet väljer alltid den mest optimala. Detta uppnås genom att utelämna parametern BLKSIZE.



Dataset i en generationsdatasetgrupp innehåller ett generationsnummer samt ett versionsnummer. Generationsnumret förändras av systemet för varje ny generation som skapas. Den första generationen har generationsnummer G0001, den andra G0002 o.s.v. Dessa generationsnummer är absoluta, till skillnad från de generationsnummer som anges i JCL, som är relativa.

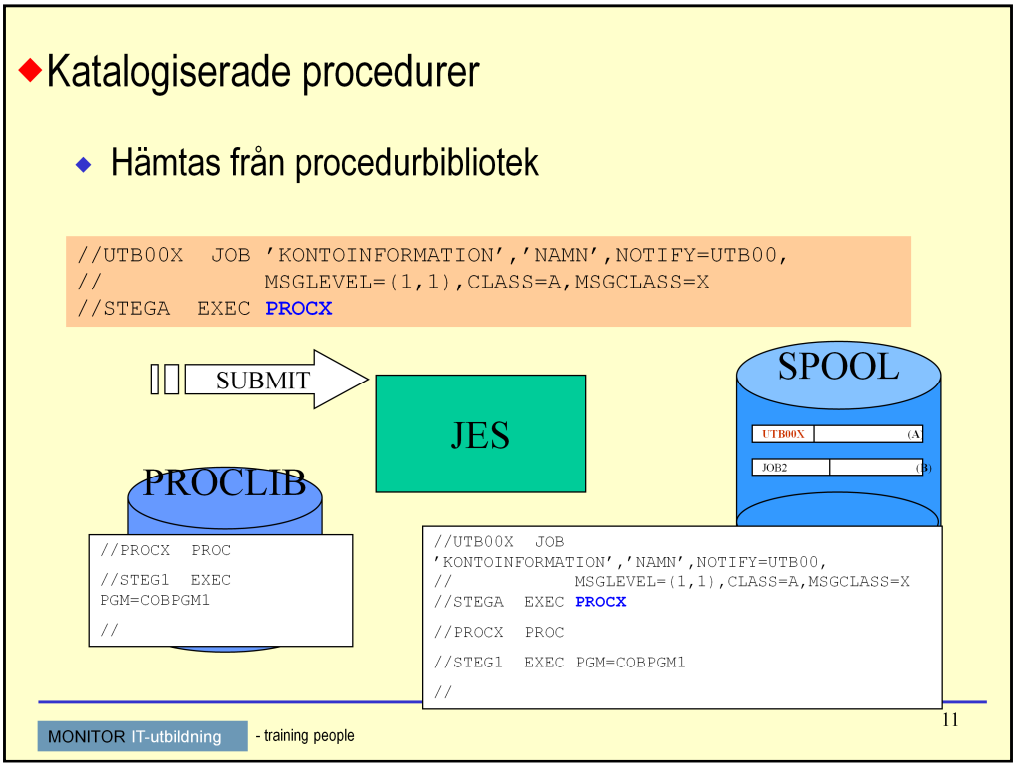
De absoluta generationsnumren förändras kontinuerligt, men de relativa generationsnumren är alltid de samma och bara beroende av storleken på generationsgruppen.

Versionsnumret används ej.

Ett generationsdataset med det högsta absoluta generationsnumret är alltid den aktuella generationen.

En generationsdatasetgrupp innehåller ett bestämt antal generationer som bestäms när generationsdatasetgruppen skapas.

När en ny generation skapas, så måste den äldsta generationen tas bort ur gruppen. Vad som skall hända med ”föråldrade” generationer anges när gruppen skapas.



Vid ”SUBMIT” av jobbet kommer JES att ta hand om den JCL vi skrivit. I JES finns en funktion som kallas ”*Reader/Interpreter*” som kontrollerar JCLen med avseende på korrekt syntax. När JES ser att vi refererar en procedur, så kontrolleras i första hand om procedurbeskrivningen finns i jobbet.

Om den inte finns där så söker JES efter en medlem, med samma namn som procedurnamnet, på de bibliotek som är systemets procedurbibliotek. Vilka bibliotek som är procedurbibliotek beskrivs i uppstartsparametrar för JES.

Man kan även ha privata procedurbibliotek. Detta behandlas senare.

Den översatta versionen av jobbet, som nu även innehåller den JCL som finns i proceduren, placeras på den jobbkö som finns angiven i jobbuttrycket.