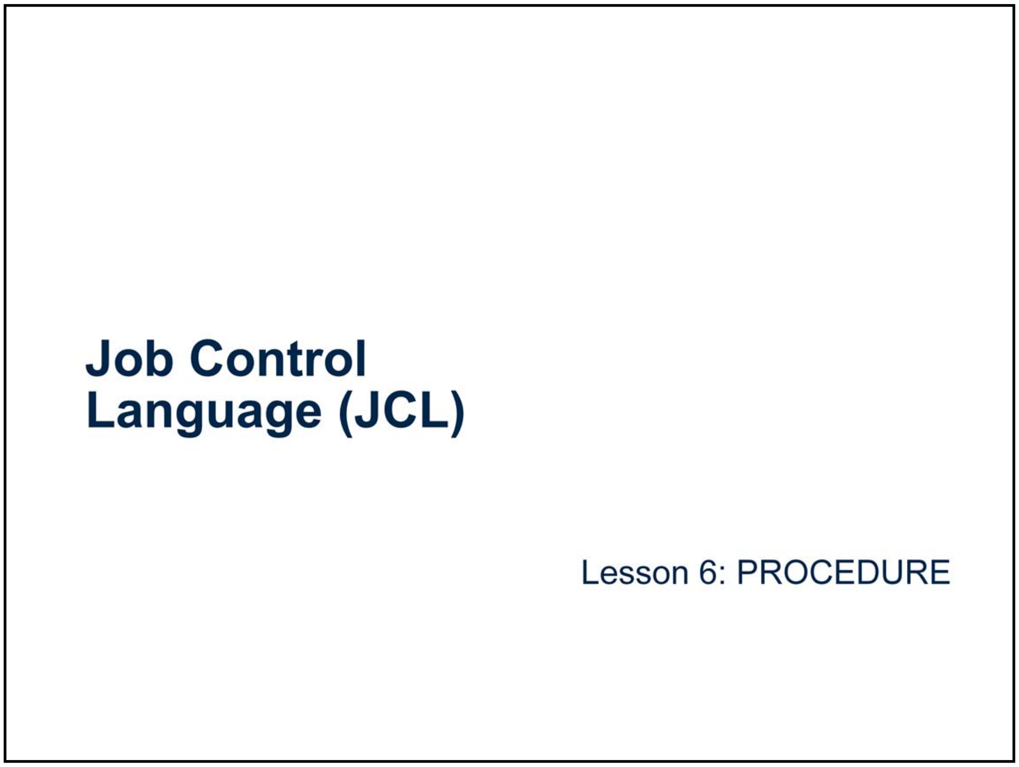
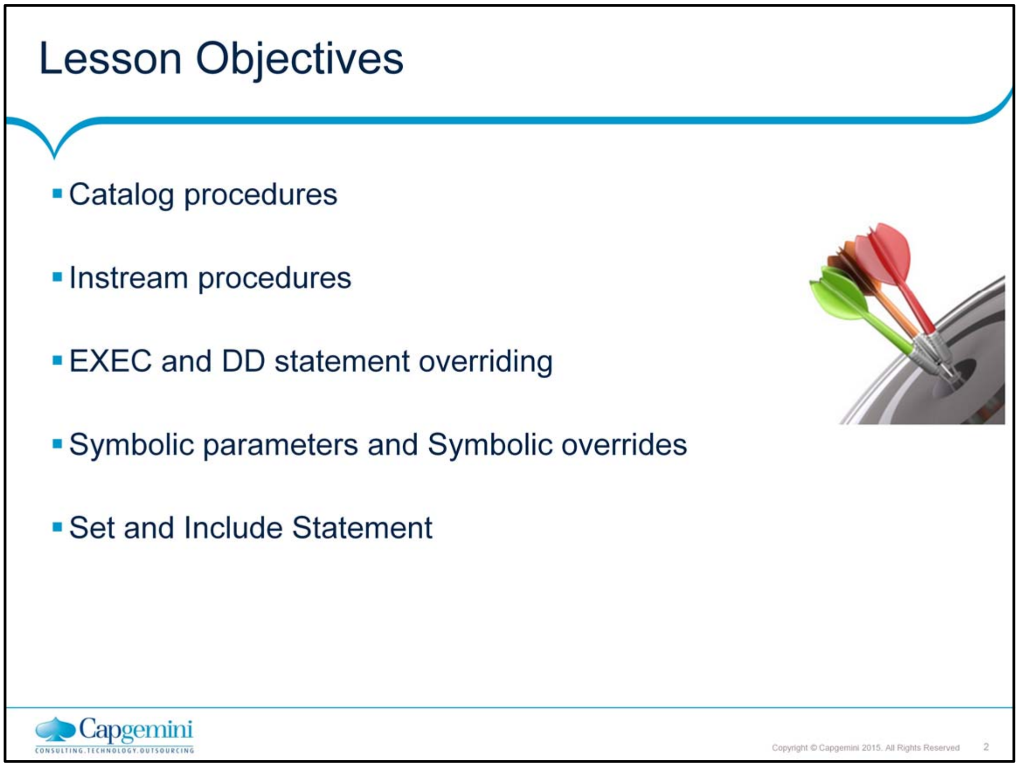


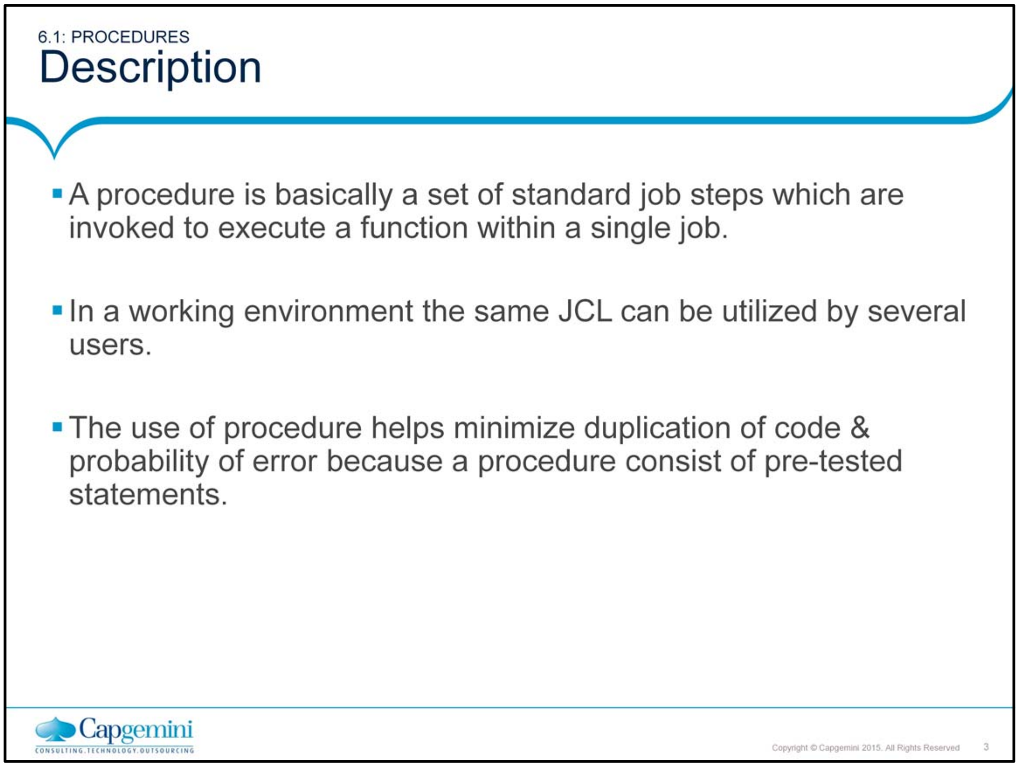
Job Control Language (JCL)

Lesson 6: PROCEDURE



Lesson Objectives

- Catalog procedures
- Instream procedures
- EXEC and DD statement overriding
- Symbolic parameters and Symbolic overrides
- Set and Include Statement



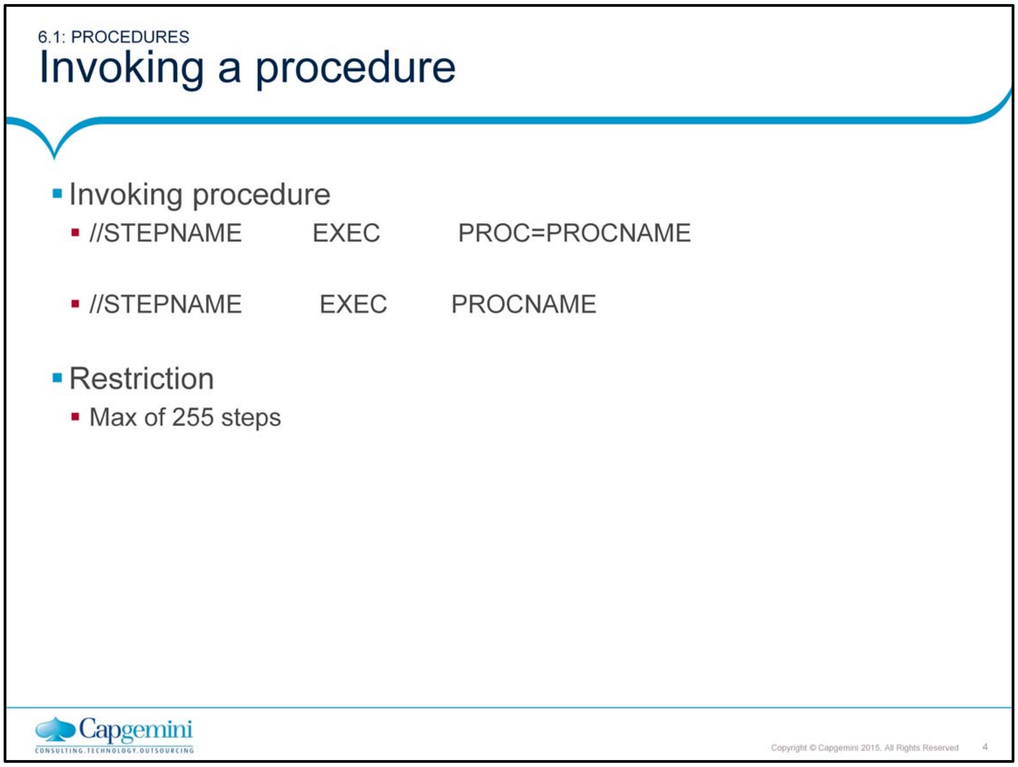
6.1: PROCEDURES

Description

- A procedure is basically a set of standard job steps which are invoked to execute a function within a single job.
- In a working environment the same JCL can be utilized by several users.
- The use of procedure helps minimize duplication of code & probability of error because a procedure consist of pre-tested statements.

The names of the procedures can be recorded and stored like any other members of PDS.

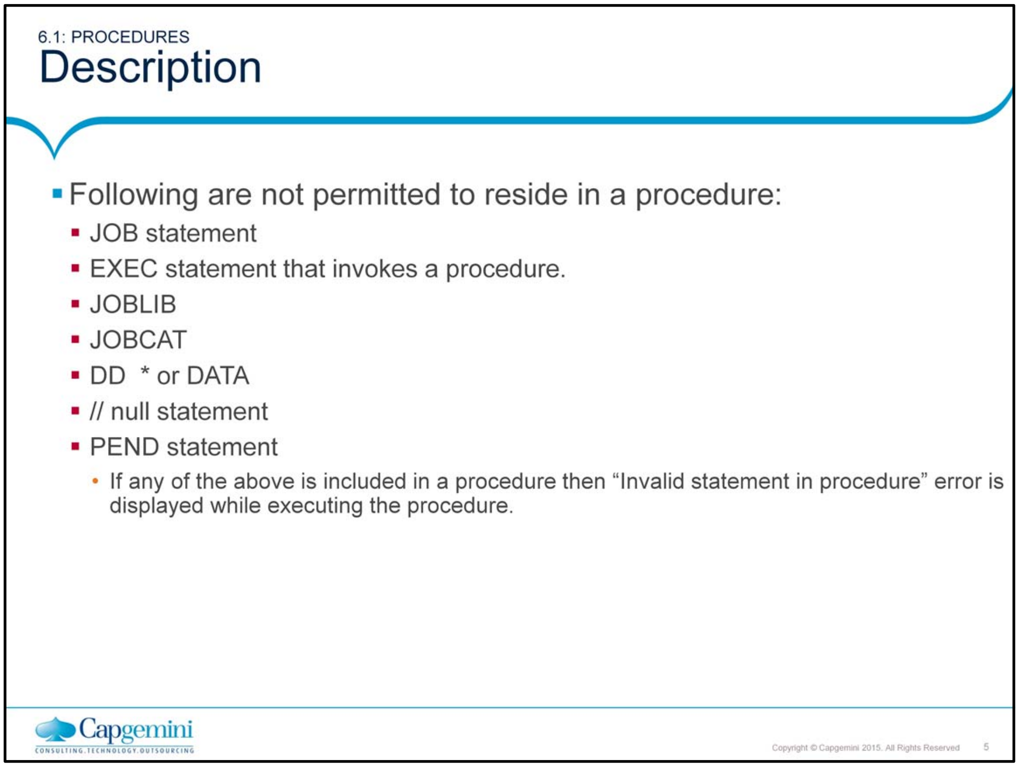
Sometimes the same JCL can be utilized by different users but with different parameters, in such cases JCL permits to override one parameter with another.



6.1: PROCEDURES

Invoking a procedure

- Invoking procedure
  - //STEPNAME EXEC PROC=PROCNAME
  - //STEPNAME EXEC PROCNAME
- Restriction
  - Max of 255 steps

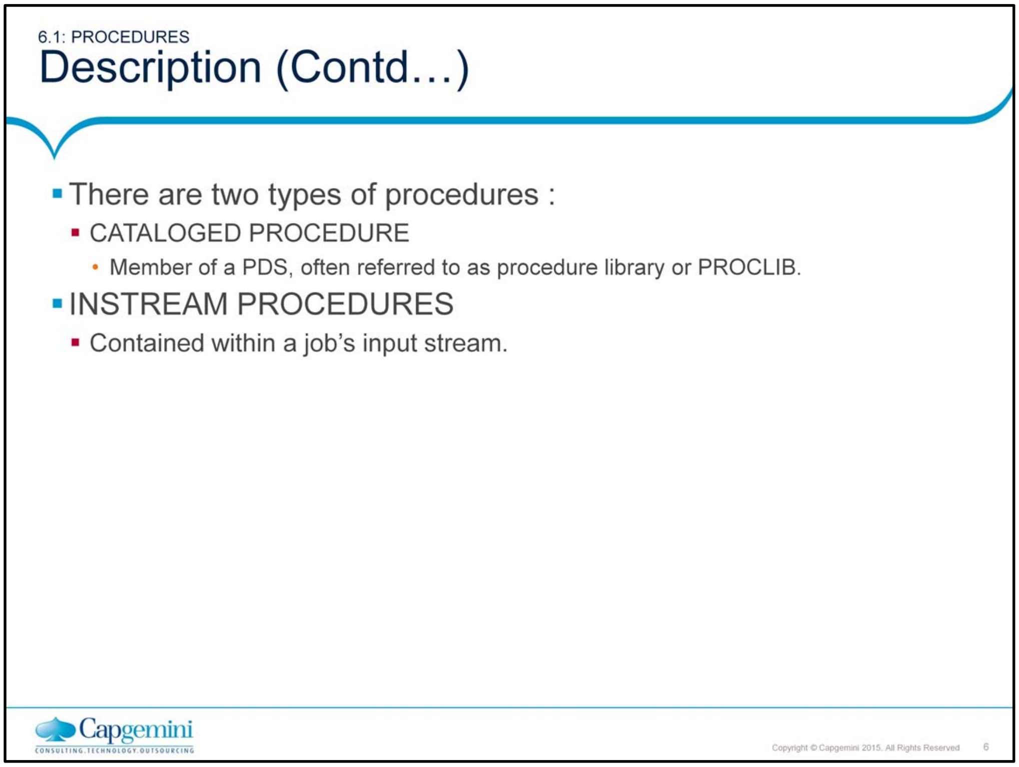


6.1: PROCEDURES

Description

- Following are not permitted to reside in a procedure:
  - JOB statement
  - EXEC statement that invokes a procedure.
  - JOBLIB
  - JOBCAT
  - DD * or DATA
  - // null statement
  - PEND statement
    - If any of the above is included in a procedure then “Invalid statement in procedure” error is displayed while executing the procedure.

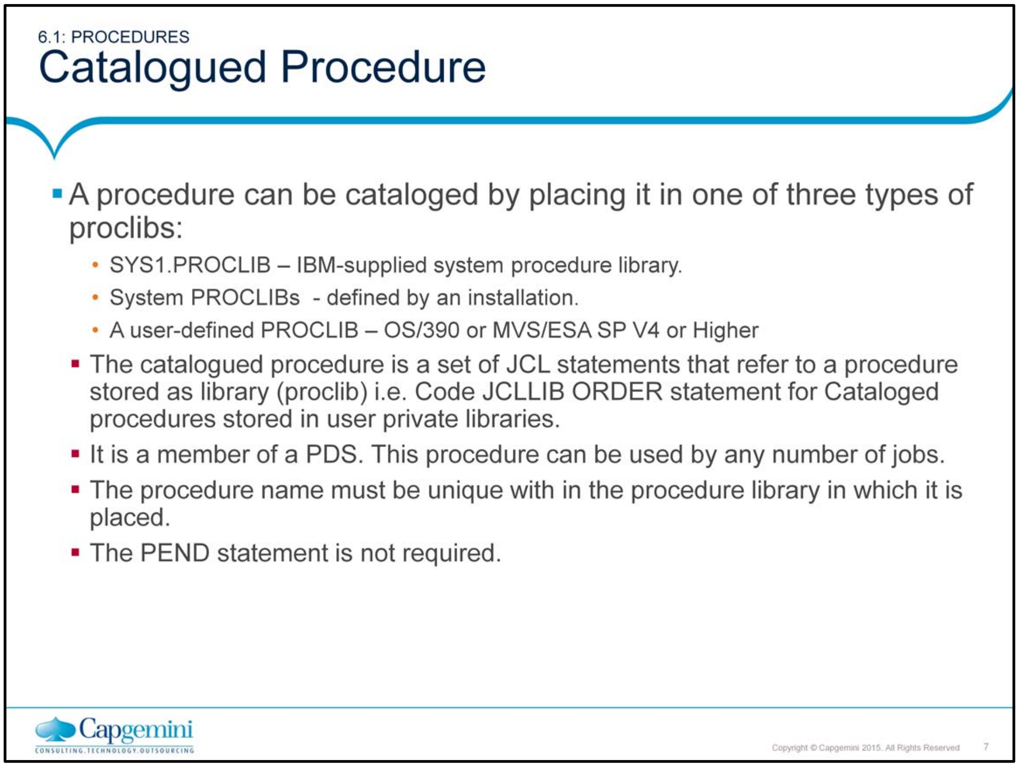
5



6.1: PROCEDURES

Description (Contd…)

- There are two types of procedures :
  - CATALOGED PROCEDURE
    - Member of a PDS, often referred to as procedure library or PROCLIB.
- INSTREAM PROCEDURES
  - Contained within a job’s input stream.

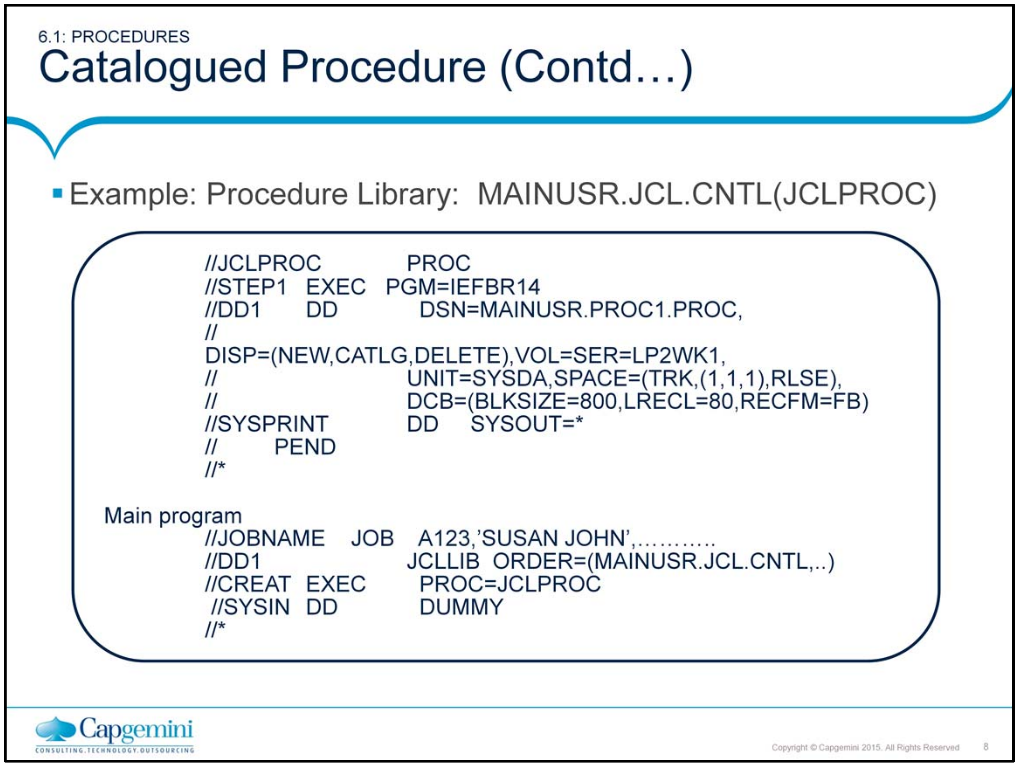


6.1: PROCEDURES

Catalogued Procedure

- A procedure can be cataloged by placing it in one of three types of proclibs:
  - SYS1.PROCLIB – IBM-supplied system procedure library.
  - System PROCLIBs - defined by an installation.
  - A user-defined PROCLIB – OS/390 or MVS/ESA SP V4 or Higher
  - The catalogued procedure is a set of JCL statements that refer to a procedure stored as library (proclib) i.e. Code JCLLIB ORDER statement for Cataloged procedures stored in user private libraries.
  - It is a member of a PDS. This procedure can be used by any number of jobs.
  - The procedure name must be unique with in the procedure library in which it is placed.
  - The PEND statement is not required.

7



6.1: PROCEDURES

Catalogued Procedure (Contd…)

- Example: Procedure Library: MAINUSR.JCL.CNTL(JCLPROC)

```
//JCLPROC            PROC
//STEP1   EXEC   PGM=IEFBR14
//DD1       DD            DSN=MAINUSR.PROC1.PROC,
//
DISP=(NEW,CATLG,DELETE),VOL=SER=LP2WK1,
//                           UNIT=SYSDA,SPACE=(TRK,(1,1,1),RLSE),
//                           DCB=(BLKSIZE=800,LRECL=80,RECFM=FB)
//SYSPRINT           DD     SYSOUT=*
//        PEND
//*
```

Main program

```
//JOBNAME    JOB    A123,'SUSAN JOHN',……….
//DD1                    JCLLIB  ORDER=(MAINUSR.JCL.CNTL,..)
//CREAT  EXEC         PROC=JCLPROC
 //SYSIN  DD            DUMMY
//*
```

Copyright © Capgemini 2015. All Rights Reserved 8

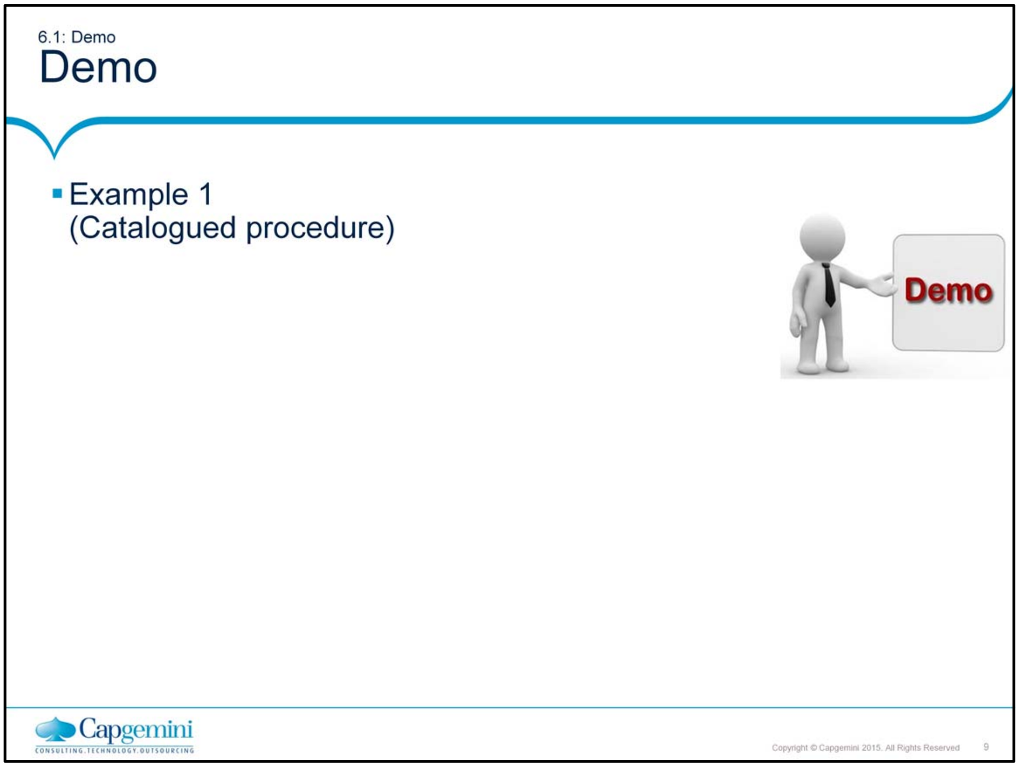
```
//PROC1  PROC
//STEP01   EXEC PGM=FIRST
//DD1  DD DSN=SPEC.A.INFILE1,DISP=SHR
//STEP02   EXEC PGM=SECOND
//DD1  DD DSN=SPEC.A.INFILE2,DISP=SHR
//                                          PEND
```

The segment of JCL that executes this above procedure

```
//MYJOB JOB (),CLASS=A
//PROCLIB JCLLIB ORDER=SPECTRUM.A.CBL
//BATCH1  EXEC PROC1
```

At Runtime:

```
//MYJOB    JOB
//PROCLIB JCLLIB ORDER=SPECTRUM.A.CBL
//* PROC1  PROC=PROC1
//*  This is what the system visualizes
// STEP01   EXEC PGM=FIRST
//DD1  DD DSN=SPEC.A.INFILE1,DISP=SHR
//STEP02   EXEC PGM=SECOND
//DD1  DD DSN=SPEC.A.INFILE2,DISP=SHR
```

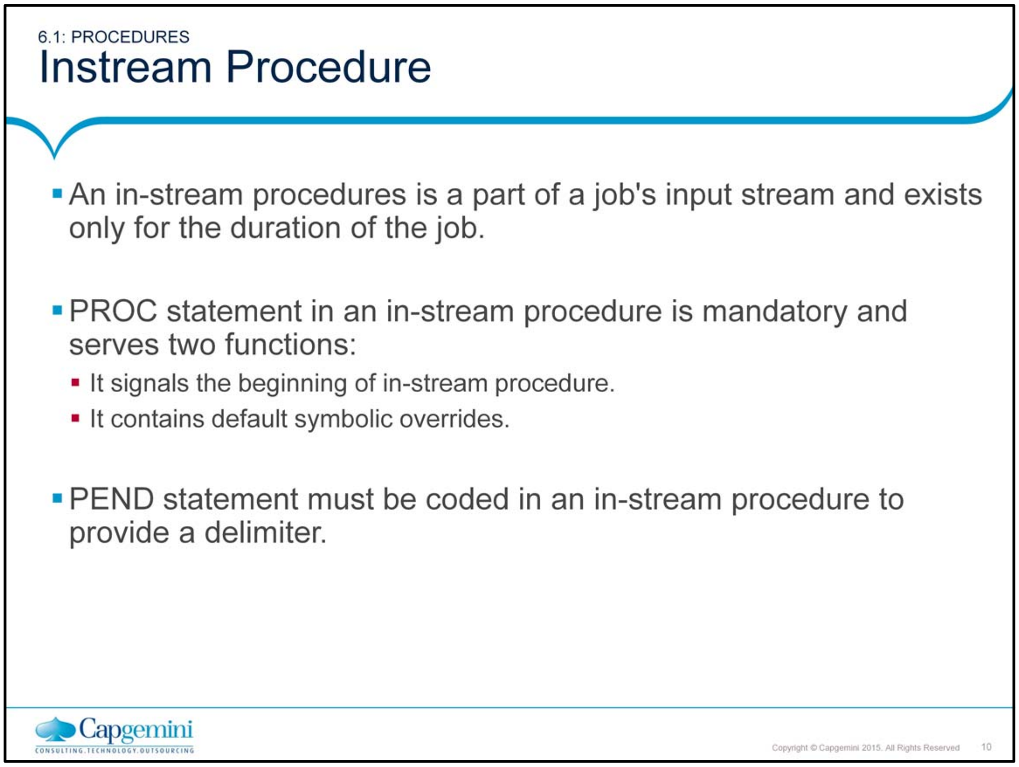


6.1: Demo

Demo

- Example 1
  (Catalogued procedure)

9



6.1: PROCEDURES

Instream Procedure

- An in-stream procedures is a part of a job's input stream and exists only for the duration of the job.
- PROC statement in an in-stream procedure is mandatory and serves two functions:
  - It signals the beginning of in-stream procedure.
  - It contains default symbolic overrides.
- PEND statement must be coded in an in-stream procedure to provide a delimiter.

As the name says it is a set of JCL statements which is contained within an input stream of a job.

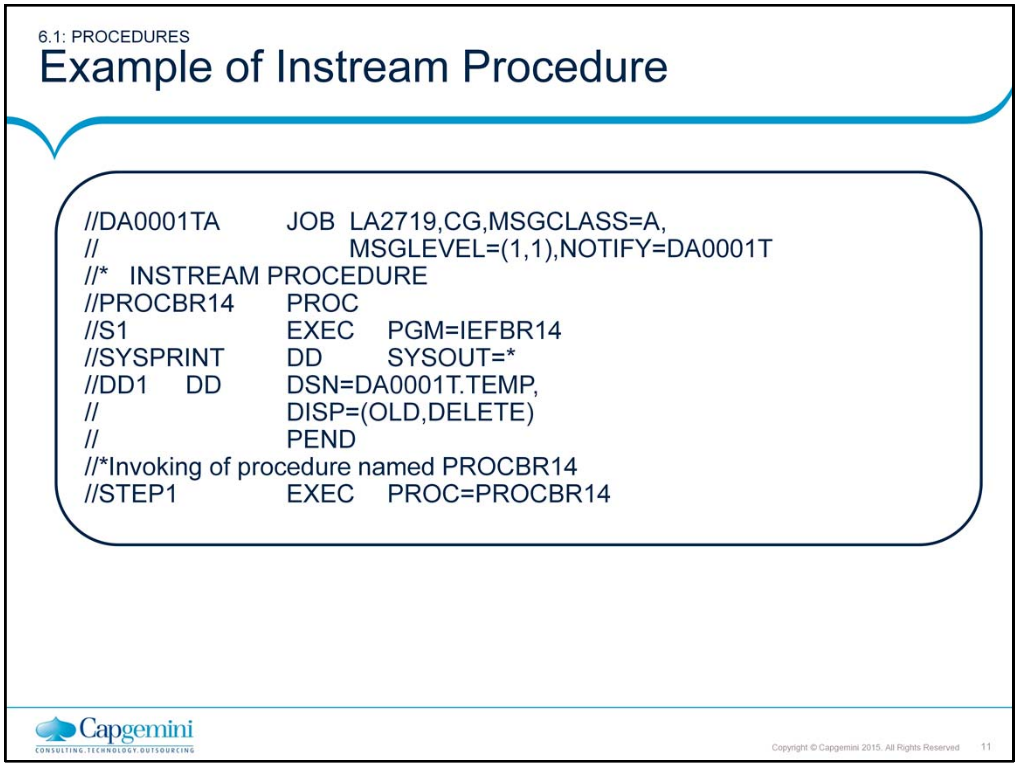
This procedure can be used by only one job but can be executed any number of times within a job.

Must begin with PROC & end with PEND statement

Must be coded immediately after the JOB statement and before the first EXEC statement.

No more than 15 in-stream procedures can be coded in one JOB.

```
//MFCVT01 JOB (),CLASS=A
//PROC1  PROC
//STEP01   EXEC PGM=FIRST
//DD1  DD DSN=SPEC.A.INFILE1,DISP=SHR
//STEP02   EXEC PGM=SECOND
//DD1  DD DSN=SPEC.A.INFILE2,DISP=SHR
//           PEND
//STEP03 EXEC PROC1
```

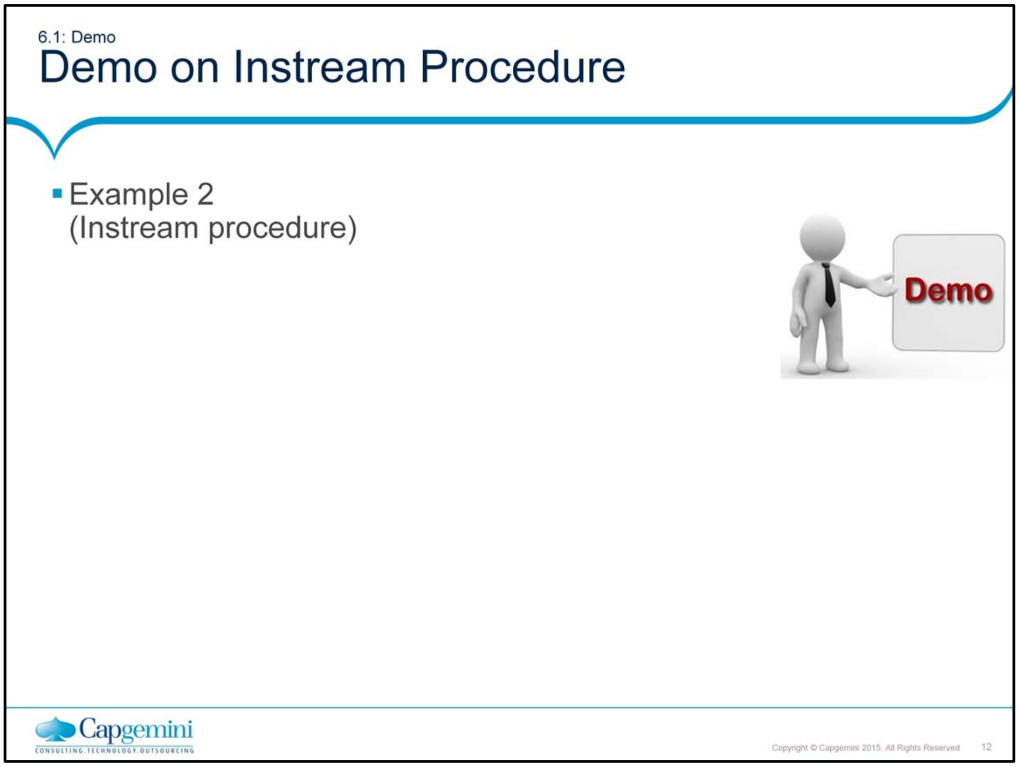


6.1: PROCEDURES

Example of Instream Procedure

```
//DA0001TA        JOB  LA2719,CG,MSGCLASS=A,
//                     MSGLEVEL=(1,1),NOTIFY=DA0001T
//*   INSTREAM PROCEDURE
//PROCBR14        PROC
//S1              EXEC    PGM=IEFBR14
//SYSPRINT        DD      SYSOUT=*
//DD1     DD      DSN=DA0001T.TEMP,
//                DISP=(OLD,DELETE)
//                PEND
//*Invoking of procedure named PROCBR14
//STEP1           EXEC    PROC=PROCBR14
```

11

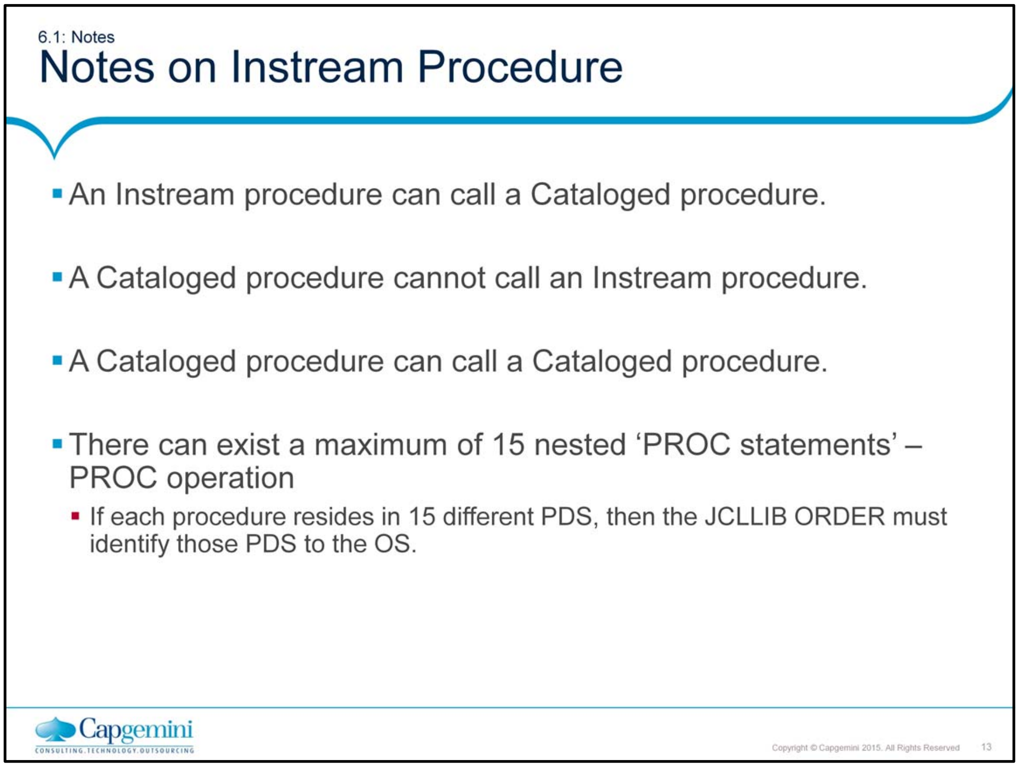


6.1: Demo

Demo on Instream Procedure

- Example 2
  (Instream procedure)

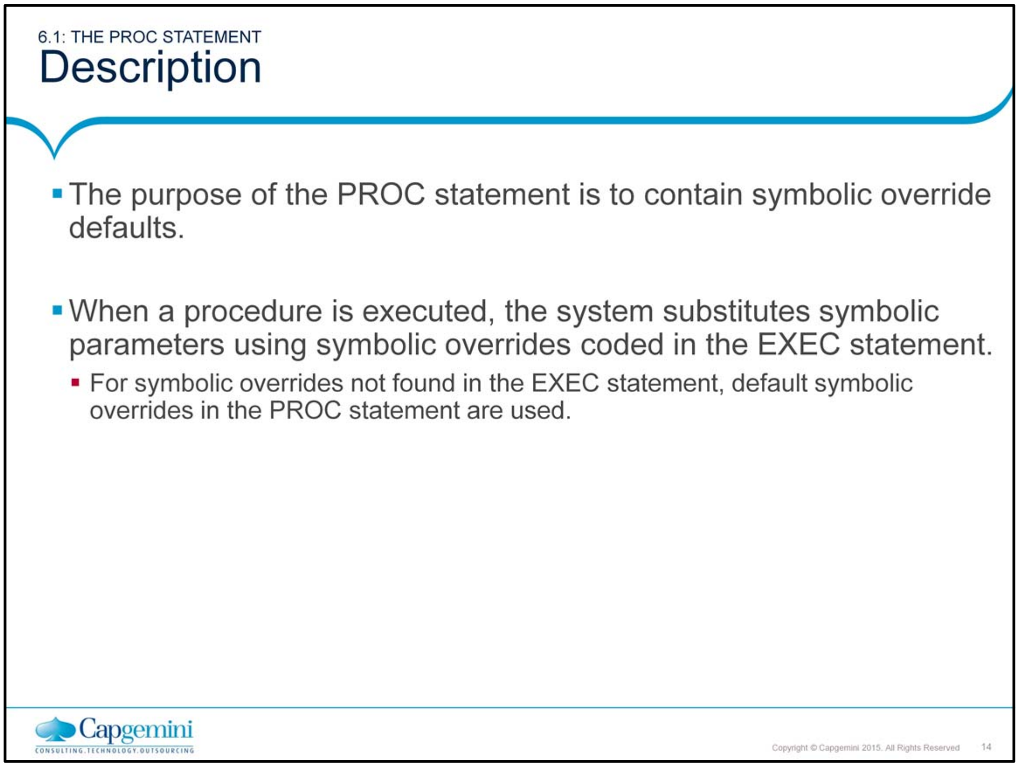
Copyright © Capgemini 2015. All Rights Reserved 12



6.1: Notes

Notes on Instream Procedure

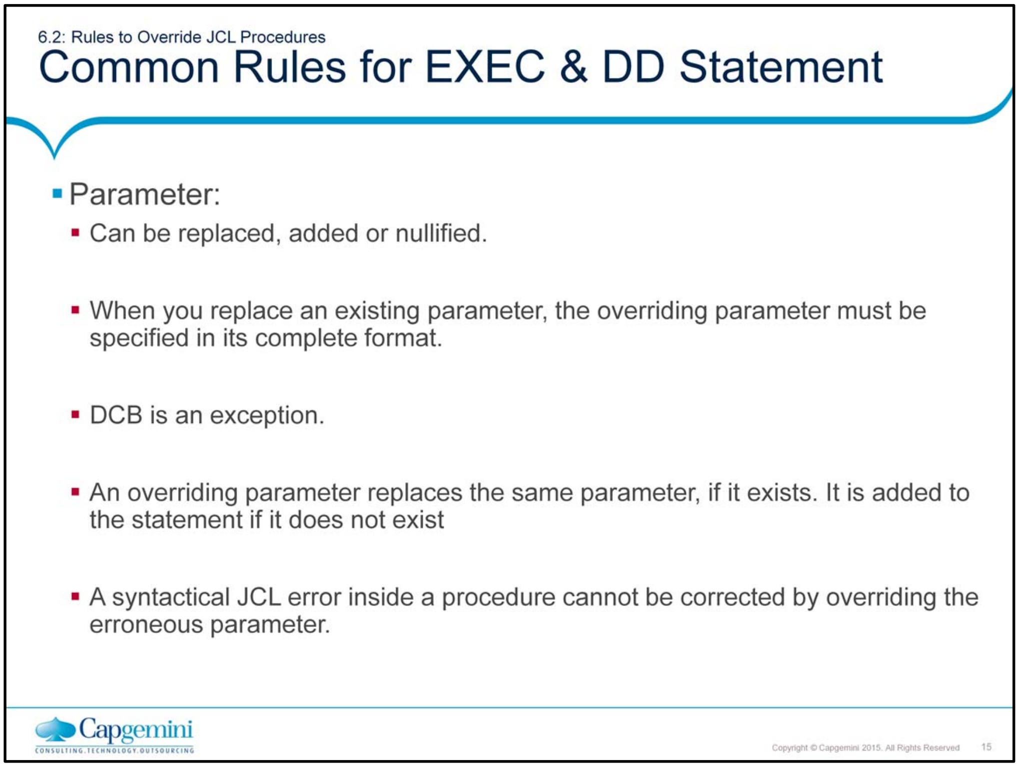
- An Instream procedure can call a Cataloged procedure.
- A Cataloged procedure cannot call an Instream procedure.
- A Cataloged procedure can call a Cataloged procedure.
- There can exist a maximum of 15 nested ‘PROC statements’ – PROC operation
  - If each procedure resides in 15 different PDS, then the JCLLIB ORDER must identify those PDS to the OS.



6.1: THE PROC STATEMENT

Description

- The purpose of the PROC statement is to contain symbolic override defaults.
- When a procedure is executed, the system substitutes symbolic parameters using symbolic overrides coded in the EXEC statement.
  - For symbolic overrides not found in the EXEC statement, default symbolic overrides in the PROC statement are used.

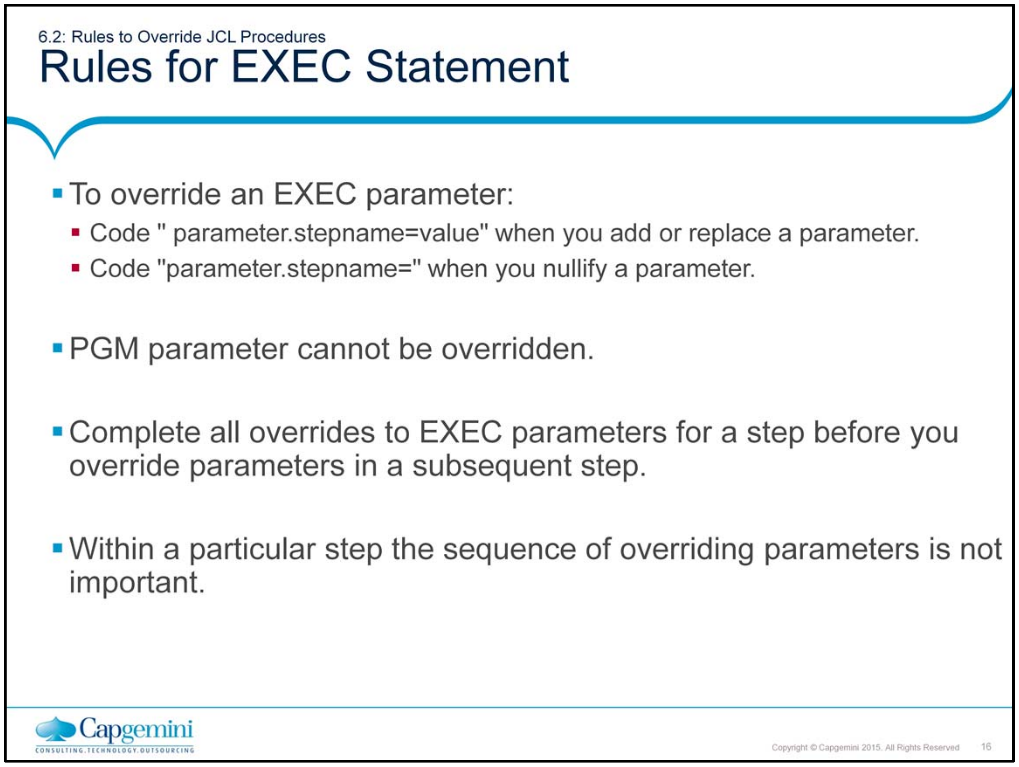


6.2: Rules to Override JCL Procedures

Common Rules for EXEC & DD Statement

- Parameter:
  - Can be replaced, added or nullified.
  - When you replace an existing parameter, the overriding parameter must be specified in its complete format.
  - DCB is an exception.
  - An overriding parameter replaces the same parameter, if it exists. It is added to the statement if it does not exist
  - A syntactical JCL error inside a procedure cannot be corrected by overriding the erroneous parameter.

15

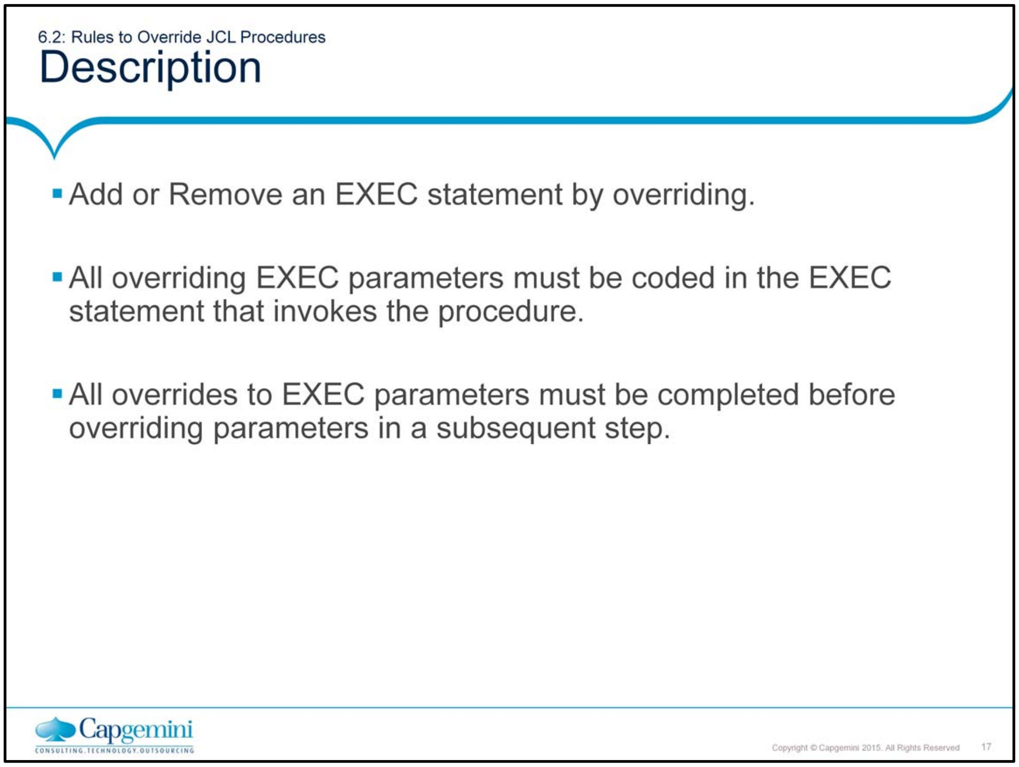


6.2: Rules to Override JCL Procedures

Rules for EXEC Statement

- To override an EXEC parameter:
  - Code " parameter.stepname=value" when you add or replace a parameter.
  - Code "parameter.stepname=" when you nullify a parameter.
- PGM parameter cannot be overridden.
- Complete all overrides to EXEC parameters for a step before you override parameters in a subsequent step.
- Within a particular step the sequence of overriding parameters is not important.

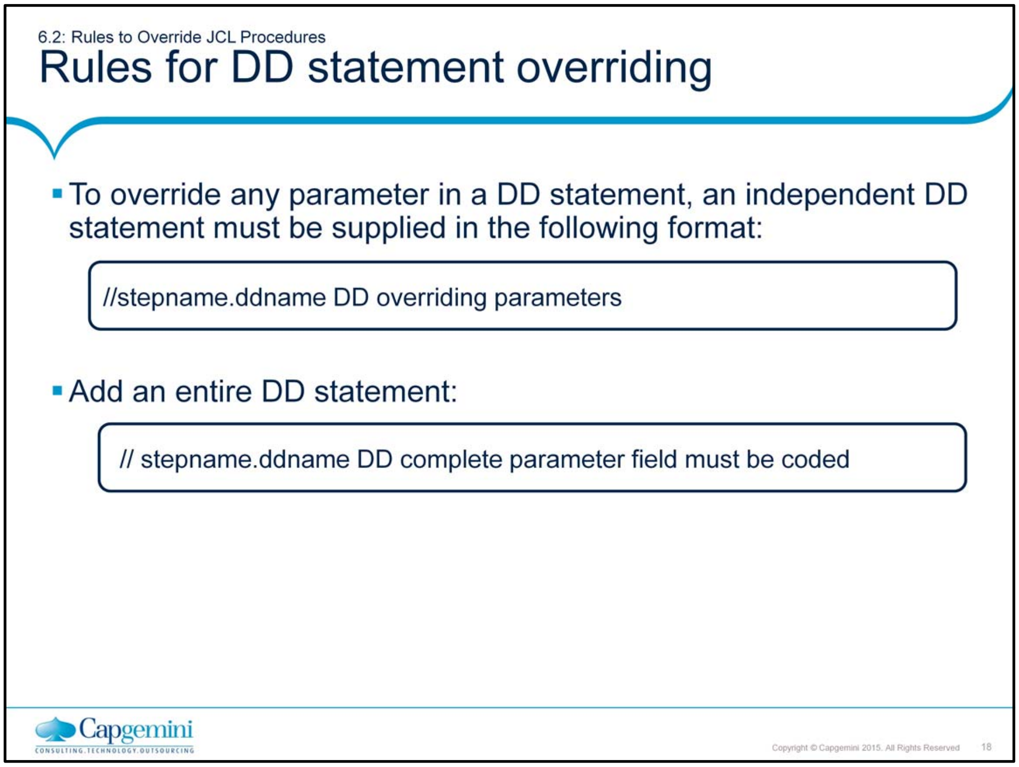
16



6.2: Rules to Override JCL Procedures

Description

- Add or Remove an EXEC statement by overriding.
- All overriding EXEC parameters must be coded in the EXEC statement that invokes the procedure.
- All overrides to EXEC parameters must be completed before overriding parameters in a subsequent step.



6.2: Rules to Override JCL Procedures

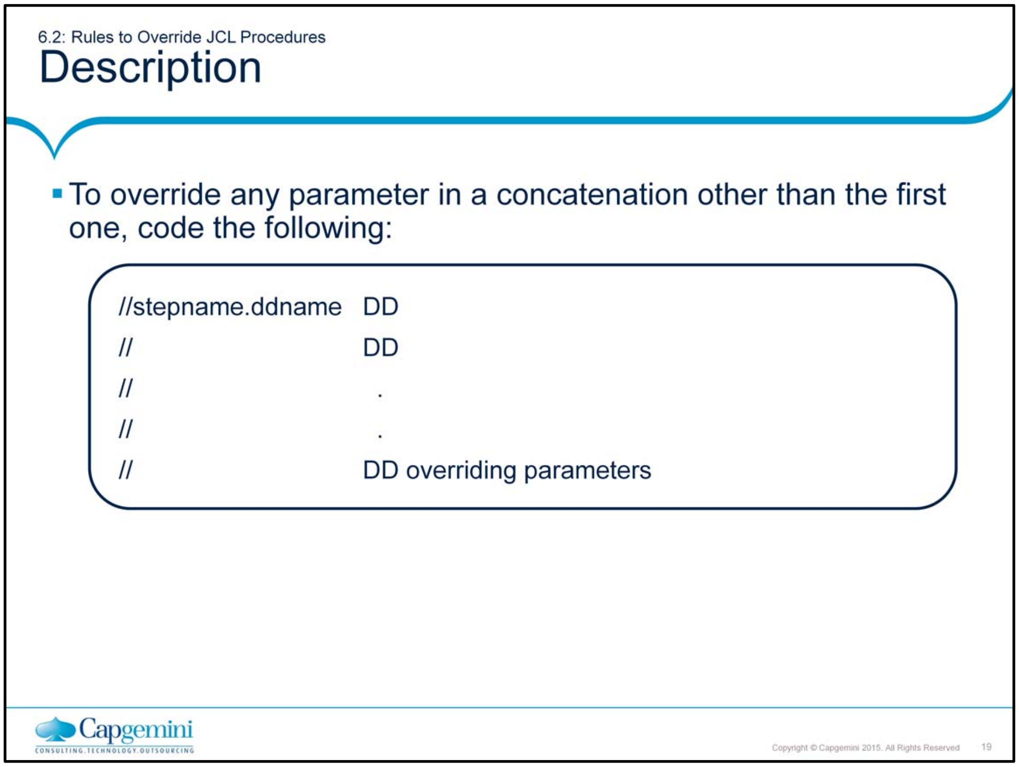
Rules for DD statement overriding

- To override any parameter in a DD statement, an independent DD statement must be supplied in the following format:

```
//stepname.ddname DD overriding parameters
```

- Add an entire DD statement:

```
// stepname.ddname DD complete parameter field must be coded
```

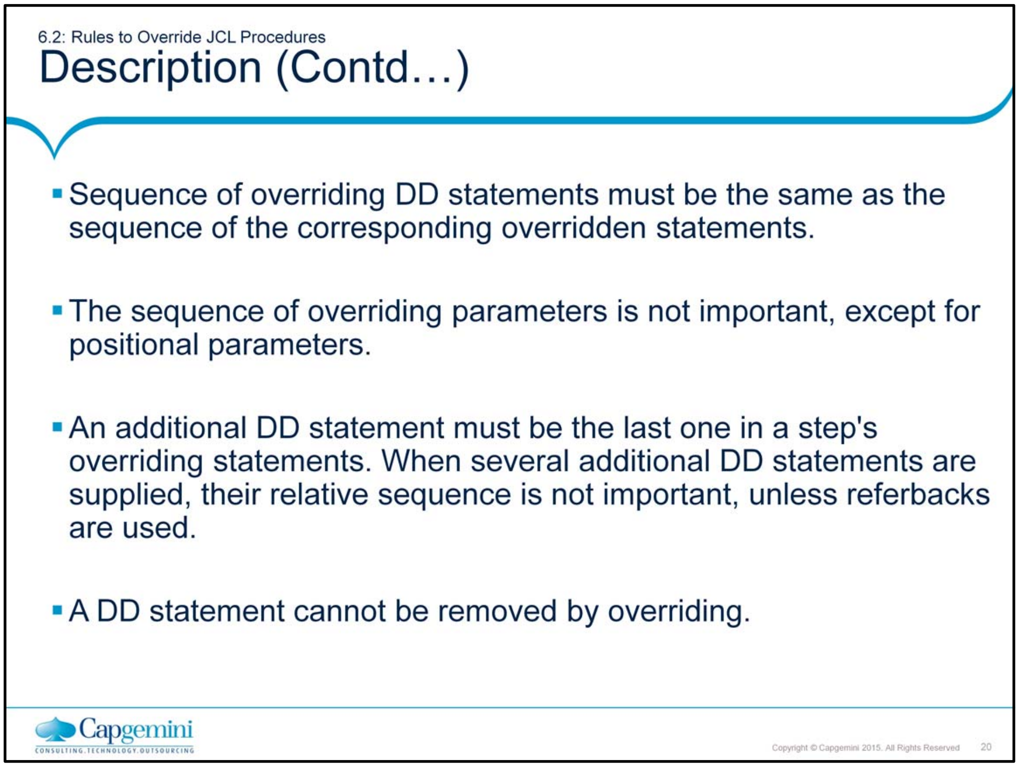


6.2: Rules to Override JCL Procedures

Description

- To override any parameter in a concatenation other than the first one, code the following:

```
//stepname.ddname   DD
//                  DD
//                    .
//                    .
//                  DD overriding parameters
```



6.2: Rules to Override JCL Procedures

Description (Contd…)

- Sequence of overriding DD statements must be the same as the sequence of the corresponding overridden statements.
- The sequence of overriding parameters is not important, except for positional parameters.
- An additional DD statement must be the last one in a step's overriding statements. When several additional DD statements are supplied, their relative sequence is not important, unless referbacks are used.
- A DD statement cannot be removed by overriding.

20

Overriding DD statements of a Procedure

```
//MYJOB  JOB
//MYPROC PROC
//STEP1  EXEC PGM=IEFBR14
//DD1    DD DSN=MAINUSR.JCL.PDS,DISP=SHR,
//                                          VOL=SER=LP1WK1,LRECL=80
//       PEND
//STEP2 EXEC MYPROC
//STEP1.DD1 DD VOL=SER=LP2WK1,LRECL=
//STEP1.DD2 DD DSN=MAINUSR.COPYLIB,DISP=SHR
```

AT RUNTIME

```
//MYJOB JOB
//STEP1   EXEC PGM=IEFBR14
//DD1 DD   DSN=MAINUSR.JCL,PDS,DISP=SHR,VOL=SER=LP2WK1
//DD2     DD   DSN=MAINUSR.COPYLIB,DISP=SHR
```

In the above example

In DD1, LP1WK1 was changed to LP2WK1

The LRECL parameter was nullified (discarded)

A new DD statement DD2 was added

Overriding EXEC statements of a Procedure

```
//MYJOB  JOB  NOTIFY=USERID
//MYPROC PROC
//STEP1  EXEC PGM=IEBGENER,TIME=NOLIMIT,REGION=4M
//SYSUT1 DD DUMMY
//SYSUT2 DD DUMMY
//       PEND
//STEP2 EXEC MYPROC,TIME.STEP1=10,REGION.STEP1=
```

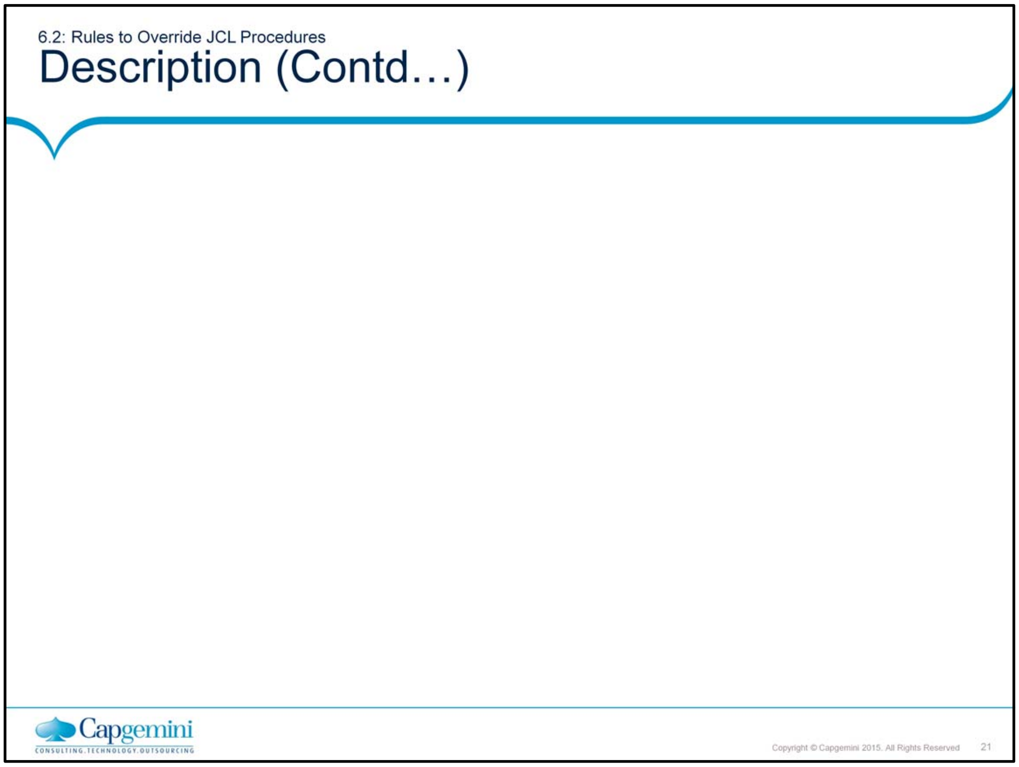
AT RUNTIME

```
//MYJOB  JOB NOTIFY=USERID
//STEP1  EXEC PGM=IEBGENER,TIME=10
//SYSUT1 DD DUMMY
//SYSUT2 DD DUMMY
```

In the example above:

The value of the TIME parameter has been changed from NOLIMIT to 10

The REGION parameter has been nullified



6.2: Rules to Override JCL Procedures

Description (Contd…)

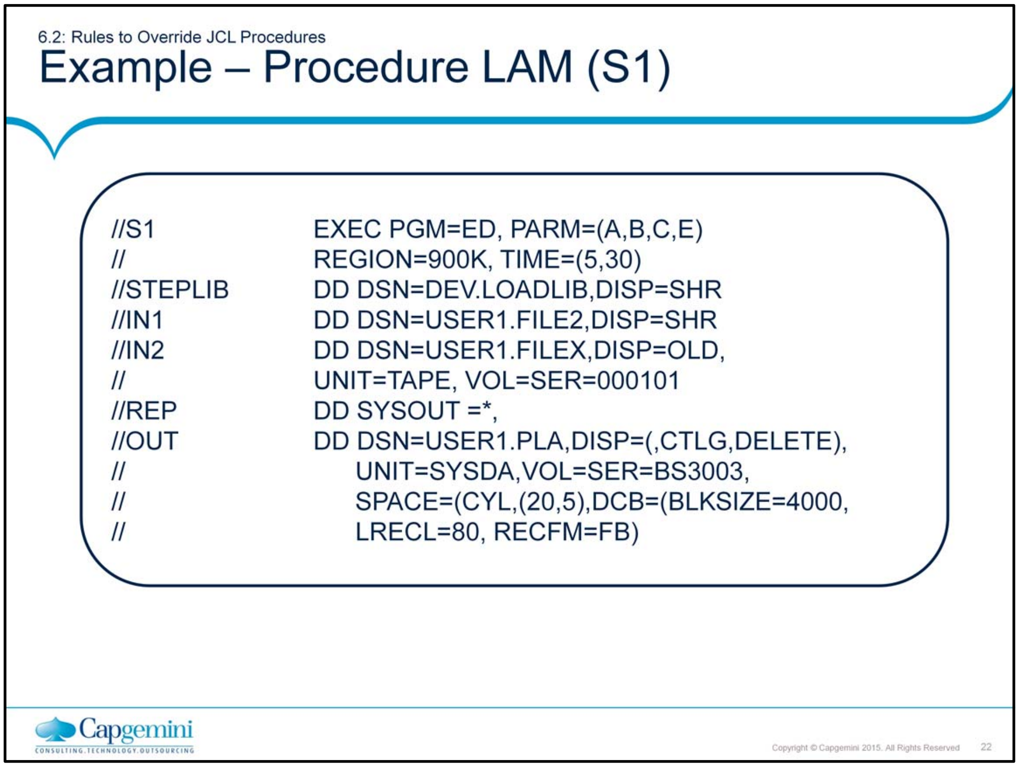
Overriding EXEC statements of a Procedure

```
//MYJOB  JOB  NOTIFY=USERID
//MYPROC PROC
//STEP1  EXEC PGM=IEBGENER,TIME=NOLIMIT,REGION=4M
//SYSUT1 DD DUMMY
//SYSUT2 DD DUMMY
//       PEND
//STEP2 EXEC MYPROC,TIME.STEP1=10,REGION.STEP1=
```

AT RUNTIME

```
//MYJOB  JOB NOTIFY=USERID
//STEP1  EXEC PGM=IEBGENER,TIME=10
//SYSUT1 DD DUMMY
//SYSUT2 DD DUMMY
```

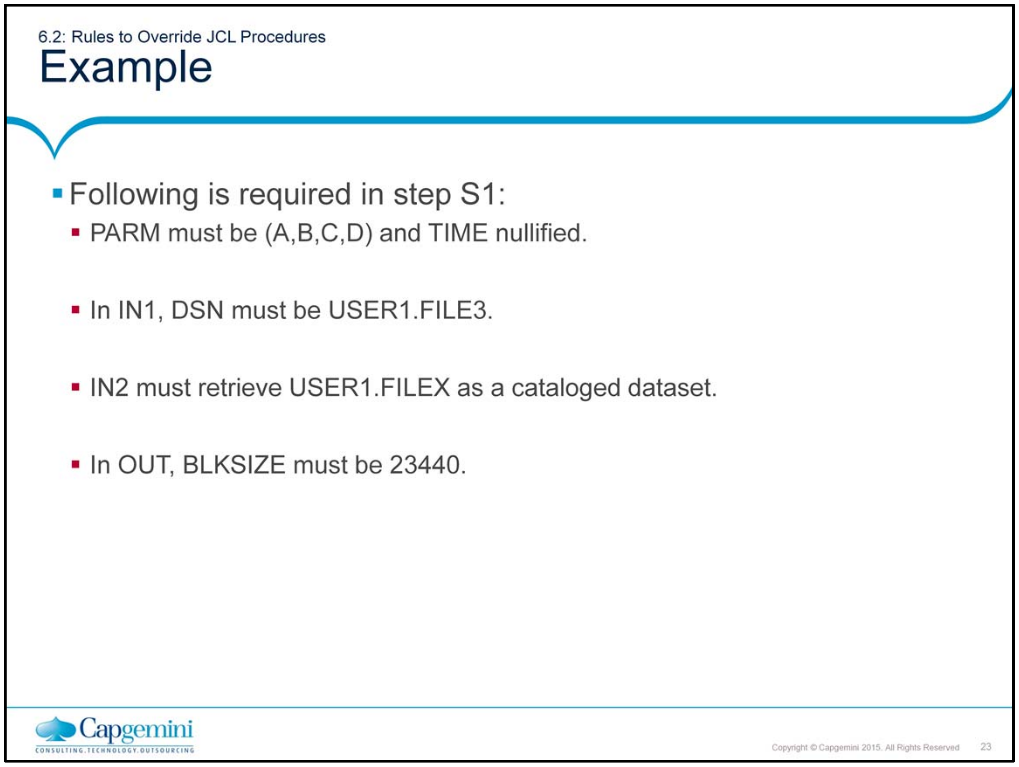
In the example above:
The value of the TIME parameter has been changed from NOLIMIT to 10
The REGION parameter has been nullified



6.2: Rules to Override JCL Procedures

Example – Procedure LAM (S1)

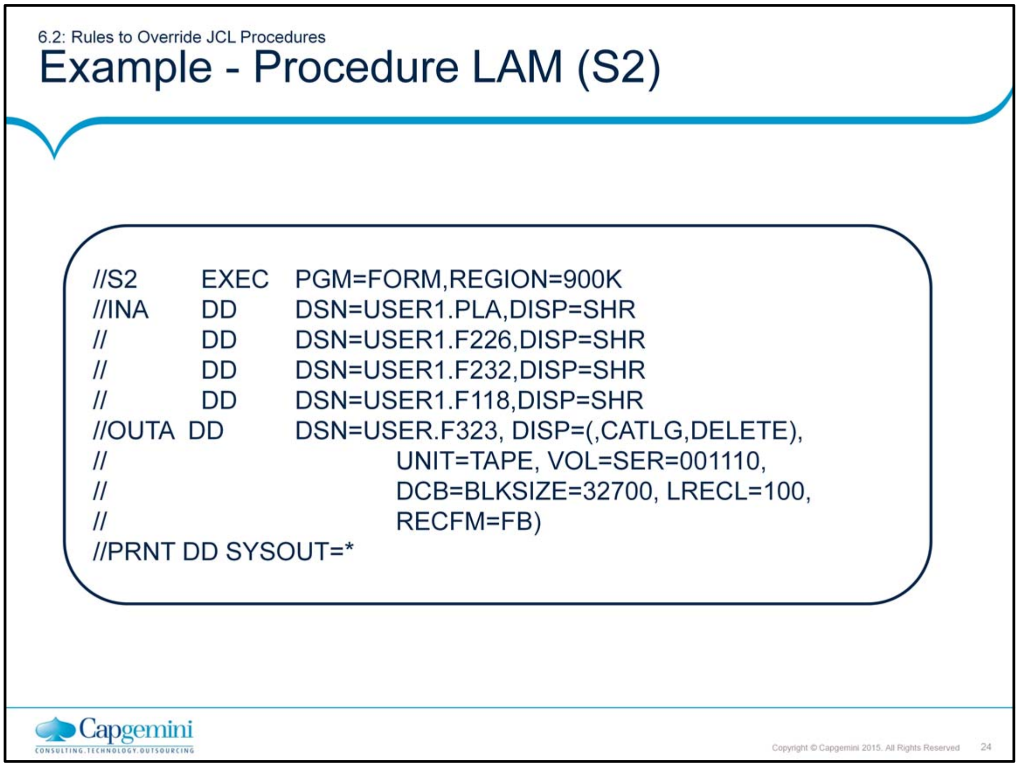
```
//S1            EXEC PGM=ED, PARM=(A,B,C,E)
//              REGION=900K, TIME=(5,30)
//STEPLIB       DD DSN=DEV.LOADLIB,DISP=SHR
//IN1           DD DSN=USER1.FILE2,DISP=SHR
//IN2           DD DSN=USER1.FILEX,DISP=OLD,
//              UNIT=TAPE, VOL=SER=000101
//REP           DD SYSOUT =*,
//OUT           DD DSN=USER1.PLA,DISP=(,CTLG,DELETE),
//                 UNIT=SYSDA,VOL=SER=BS3003,
//                 SPACE=(CYL,(20,5),DCB=(BLKSIZE=4000,
//                 LRECL=80, RECFM=FB)
```



6.2: Rules to Override JCL Procedures

Example

- Following is required in step S1:
  - PARM must be (A,B,C,D) and TIME nullified.
  - In IN1, DSN must be USER1.FILE3.
  - IN2 must retrieve USER1.FILEX as a cataloged dataset.
  - In OUT, BLKSIZE must be 23440.

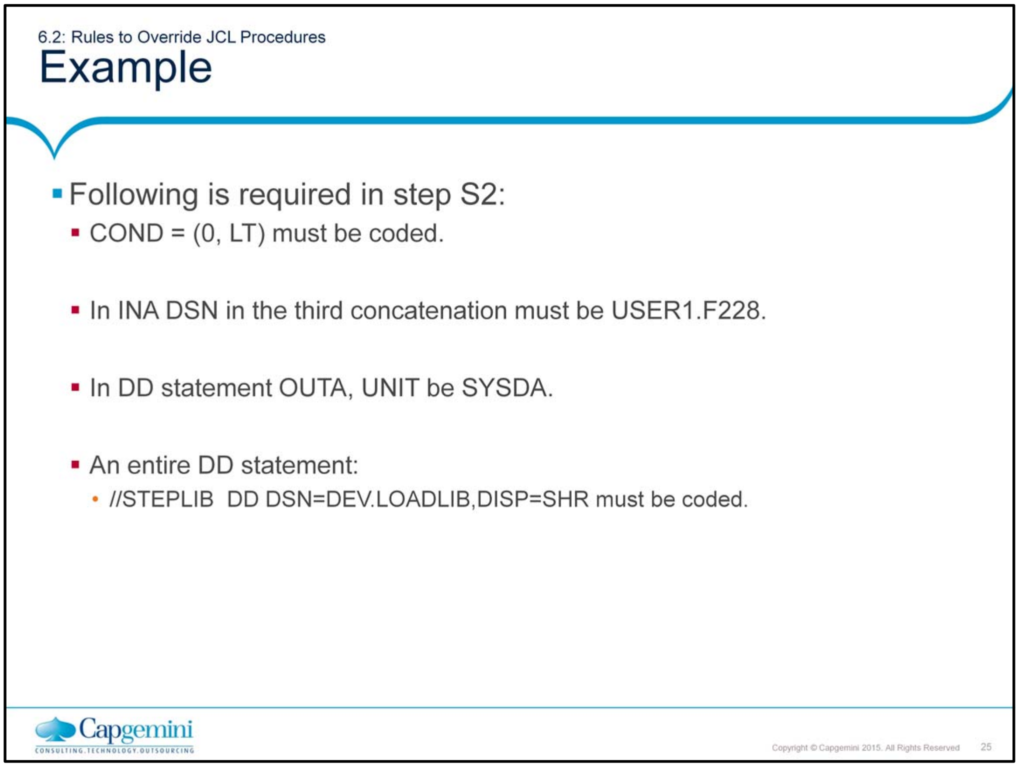


6.2: Rules to Override JCL Procedures

Example - Procedure LAM (S2)

```
//S2      EXEC   PGM=FORM,REGION=900K
//INA     DD     DSN=USER1.PLA,DISP=SHR
//        DD     DSN=USER1.F226,DISP=SHR
//        DD     DSN=USER1.F232,DISP=SHR
//        DD     DSN=USER1.F118,DISP=SHR
//OUTA  DD       DSN=USER.F323, DISP=(,CATLG,DELETE),
//                       UNIT=TAPE, VOL=SER=001110,
//                       DCB=BLKSIZE=32700, LRECL=100,
//                       RECFM=FB)
//PRNT DD SYSOUT=*
```

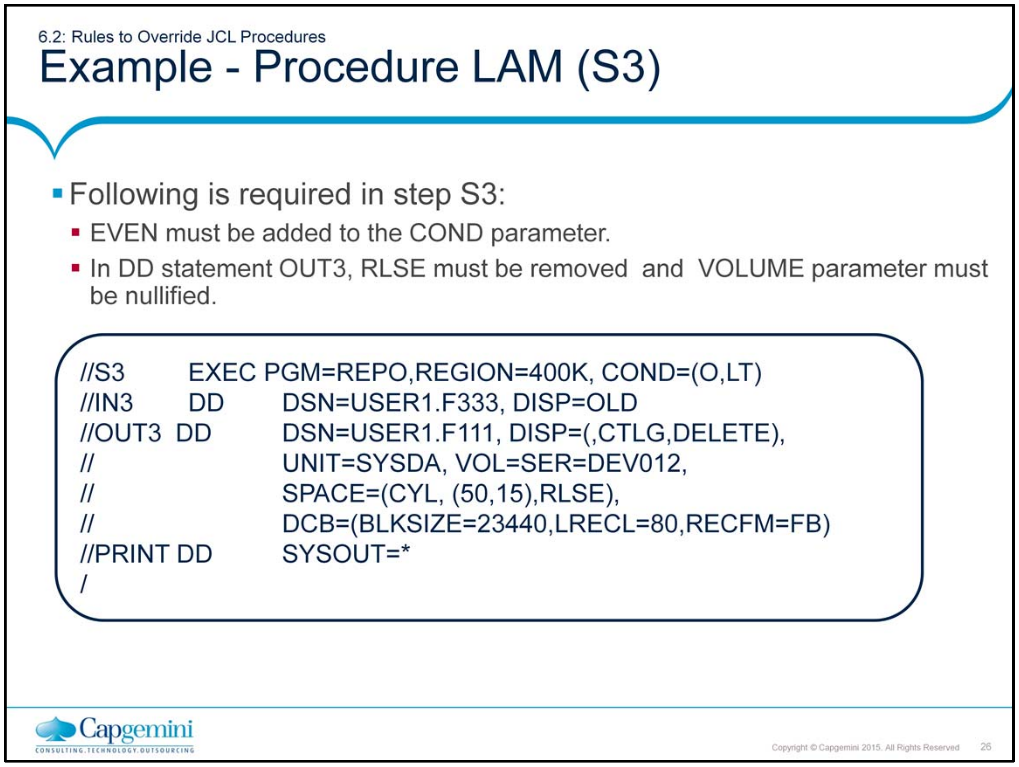
Copyright © Capgemini 2015. All Rights Reserved 24



6.2: Rules to Override JCL Procedures

Example

- Following is required in step S2:
  - COND = (0, LT) must be coded.
  - In INA DSN in the third concatenation must be USER1.F228.
  - In DD statement OUTA, UNIT be SYSDA.
  - An entire DD statement:
    - //STEPLIB DD DSN=DEV.LOADLIB,DISP=SHR must be coded.

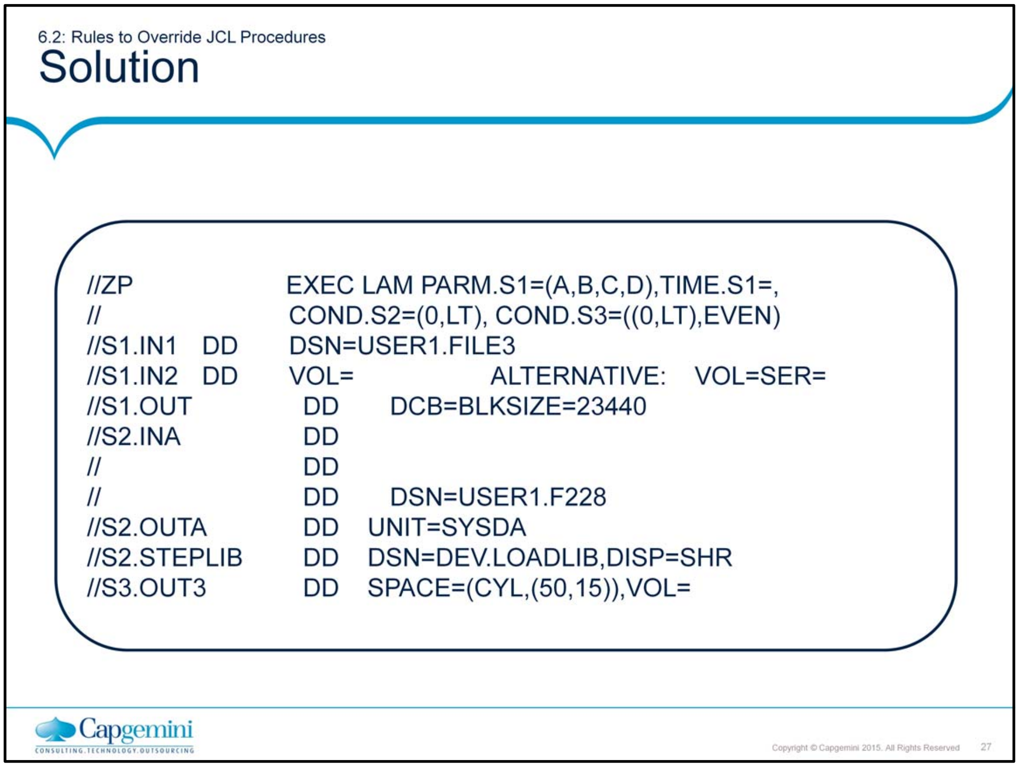


6.2: Rules to Override JCL Procedures

Example - Procedure LAM (S3)

- Following is required in step S3:
  - EVEN must be added to the COND parameter.
  - In DD statement OUT3, RLSE must be removed and VOLUME parameter must be nullified.

```
//S3        EXEC PGM=REPO,REGION=400K, COND=(O,LT)
//IN3       DD       DSN=USER1.F333, DISP=OLD
//OUT3  DD           DSN=USER1.F111, DISP=(,CTLG,DELETE),
//                   UNIT=SYSDA, VOL=SER=DEV012,
//                   SPACE=(CYL, (50,15),RLSE),
//                   DCB=(BLKSIZE=23440,LRECL=80,RECFM=FB)
//PRINT DD           SYSOUT=*
/
```

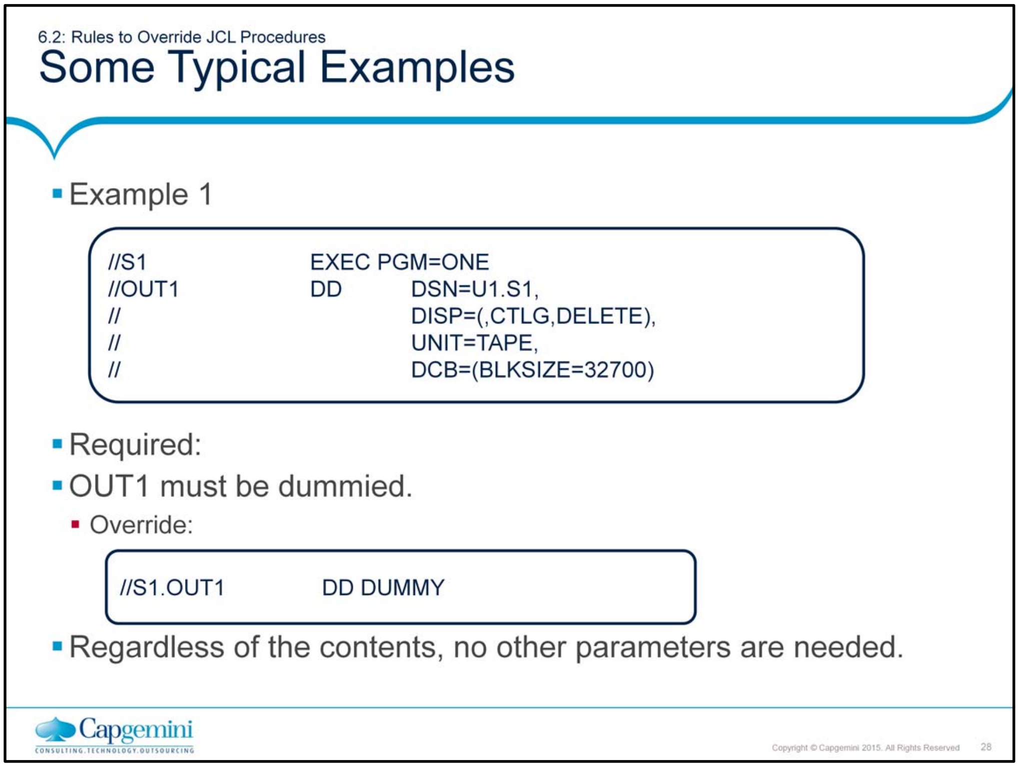


6.2: Rules to Override JCL Procedures

Solution

```
//ZP              EXEC LAM PARM.S1=(A,B,C,D),TIME.S1=,
//                 COND.S2=(0,LT), COND.S3=((0,LT),EVEN)
//S1.IN1   DD      DSN=USER1.FILE3
//S1.IN2   DD      VOL=              ALTERNATIVE:   VOL=SER=
//S1.OUT             DD      DCB=BLKSIZE=23440
//S2.INA             DD
//                   DD
//                   DD      DSN=USER1.F228
//S2.OUTA            DD    UNIT=SYSDA
//S2.STEPLIB         DD    DSN=DEV.LOADLIB,DISP=SHR
//S3.OUT3            DD    SPACE=(CYL,(50,15)),VOL=
```

27



6.2: Rules to Override JCL Procedures

Some Typical Examples

- Example 1

```
//S1              EXEC PGM=ONE
//OUT1            DD      DSN=U1.S1,
//                        DISP=(,CTLG,DELETE),
//                        UNIT=TAPE,
//                        DCB=(BLKSIZE=32700)
```

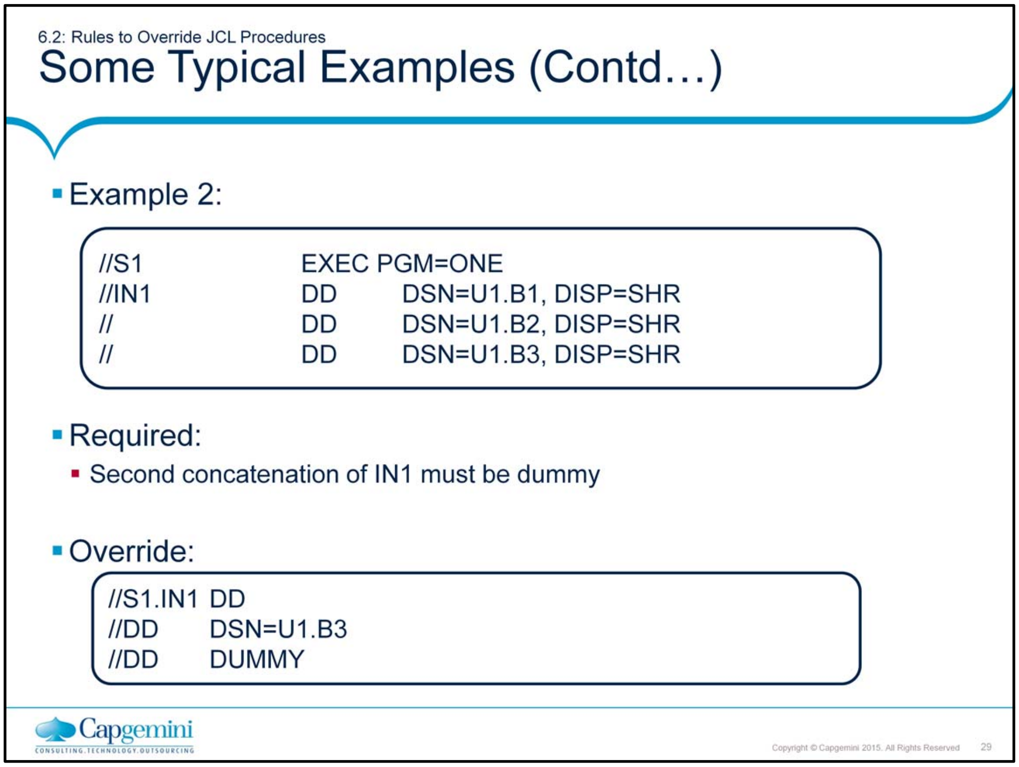
- Required:
- OUT1 must be dummied.
  - Override:

```
//S1.OUT1         DD DUMMY
```

- Regardless of the contents, no other parameters are needed.

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

28



6.2: Rules to Override JCL Procedures

Some Typical Examples (Contd…)

- Example 2:

```
//S1        EXEC PGM=ONE
//IN1       DD    DSN=U1.B1, DISP=SHR
//          DD    DSN=U1.B2, DISP=SHR
//          DD    DSN=U1.B3, DISP=SHR
```

- Required:
  - Second concatenation of IN1 must be dummy

- Override:

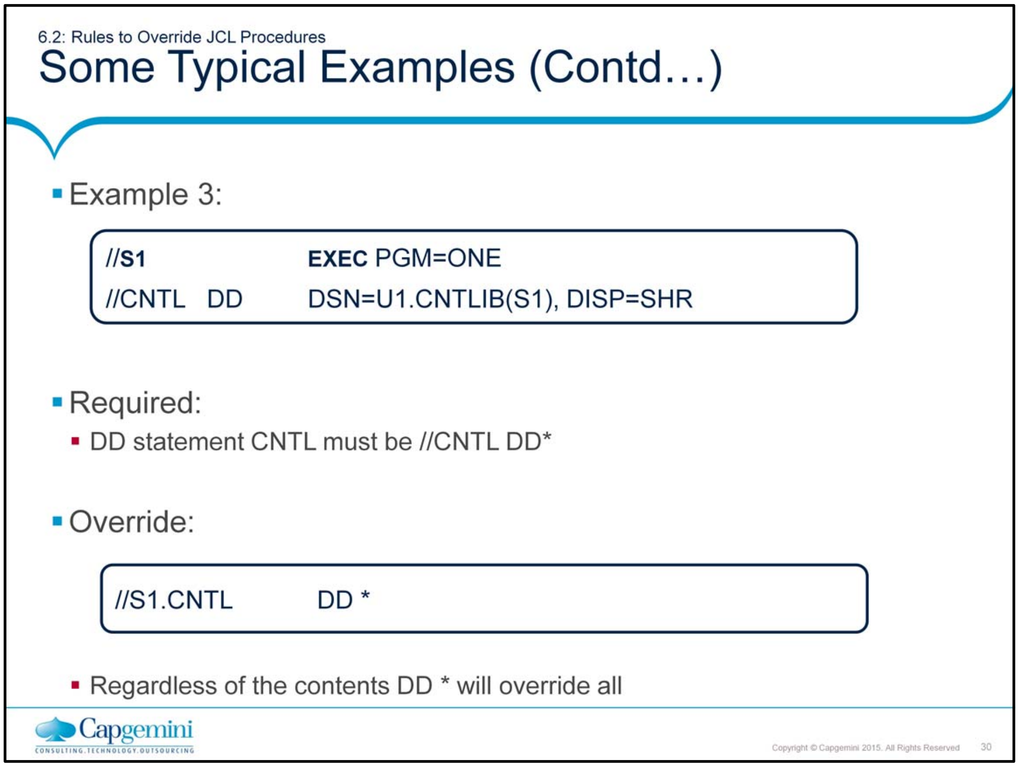
```
//S1.IN1 DD
//DD     DSN=U1.B3
//DD     DUMMY
```

Capgemini CONSULTING.TECHNOLOGY.OUTSOURCING

29

Overriding and dummying the second concatenation causes the third concatenation to also act as DUMMY.

```
//S1.IN1        DD
//              DD          DUMMY
```



6.2: Rules to Override JCL Procedures

Some Typical Examples (Contd…)

- Example 3:

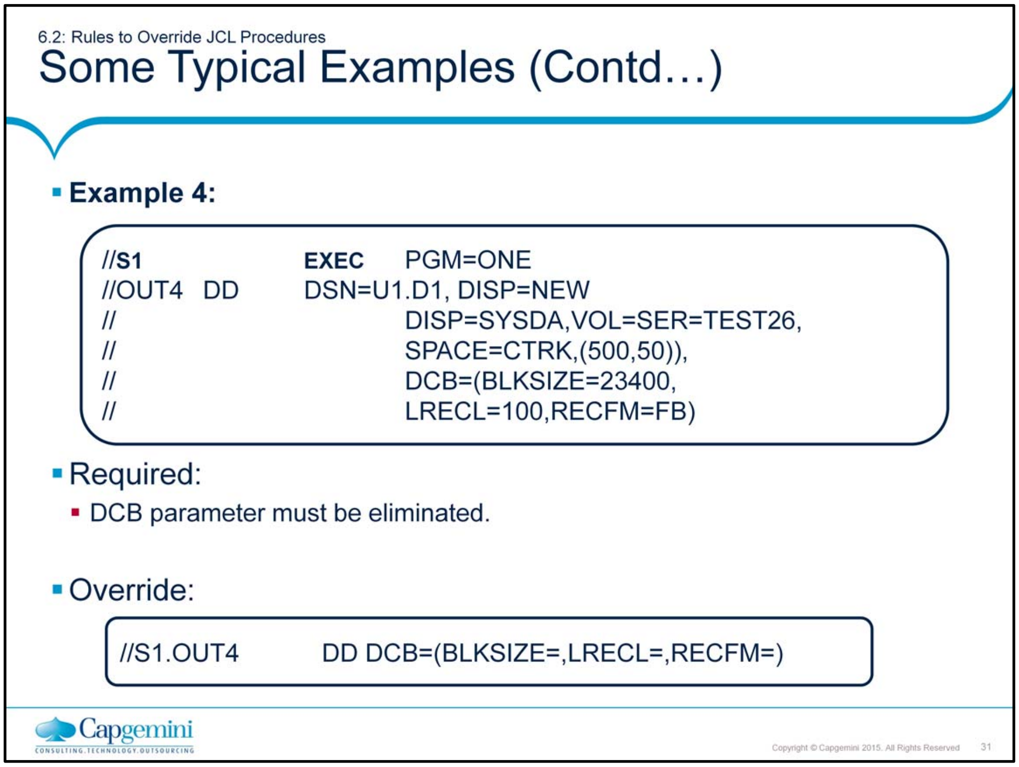
```
//S1            EXEC PGM=ONE
//CNTL   DD     DSN=U1.CNTLIB(S1), DISP=SHR
```

- Required:
  - DD statement CNTL must be //CNTL DD*

- Override:

```
//S1.CNTL       DD *
```

  - Regardless of the contents DD * will override all



6.2: Rules to Override JCL Procedures

Some Typical Examples (Contd…)

- **Example 4:**

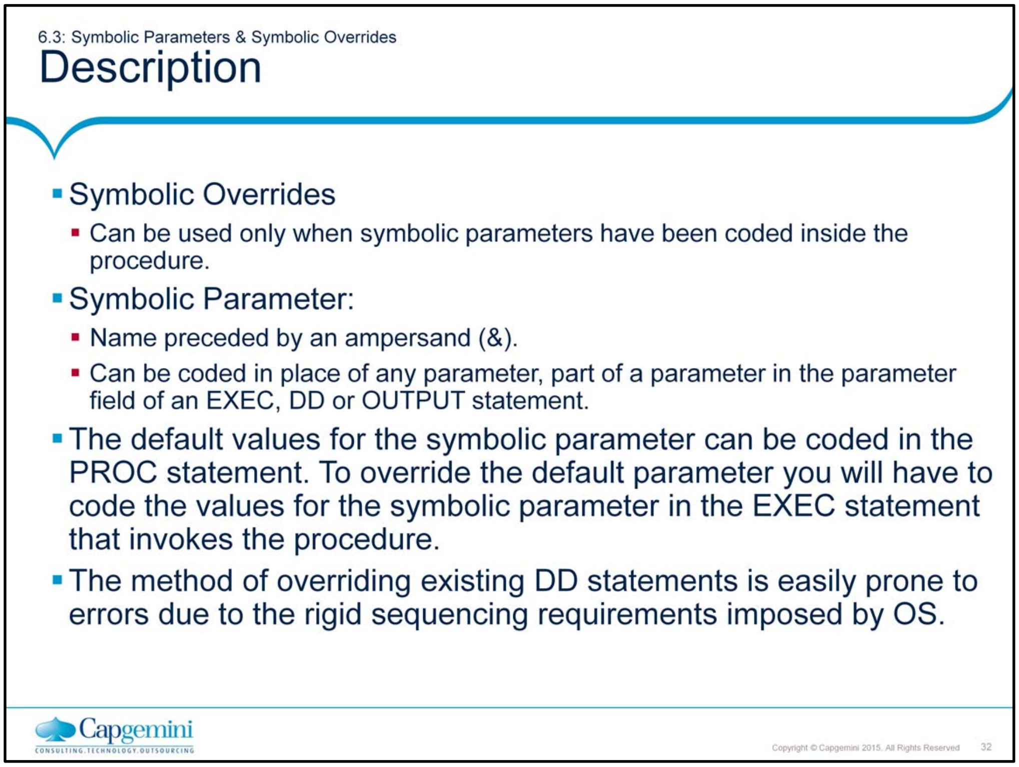
```
//S1                 EXEC     PGM=ONE
//OUT4   DD          DSN=U1.D1, DISP=NEW
//                            DISP=SYSDA,VOL=SER=TEST26,
//                            SPACE=CTRK,(500,50)),
//                            DCB=(BLKSIZE=23400,
//                            LRECL=100,RECFM=FB)
```

- Required:
  - DCB parameter must be eliminated.

- Override:

```
//S1.OUT4         DD DCB=(BLKSIZE=,LRECL=,RECFM=)
```

31



6.3: Symbolic Parameters & Symbolic Overrides

Description

- Symbolic Overrides
  - Can be used only when symbolic parameters have been coded inside the procedure.
- Symbolic Parameter:
  - Name preceded by an ampersand (&).
  - Can be coded in place of any parameter, part of a parameter in the parameter field of an EXEC, DD or OUTPUT statement.
- The default values for the symbolic parameter can be coded in the PROC statement. To override the default parameter you will have to code the values for the symbolic parameter in the EXEC statement that invokes the procedure.
- The method of overriding existing DD statements is easily prone to errors due to the rigid sequencing requirements imposed by OS.

32

An alternative approach is to anticipate which JCL parameters may change, and to define these as SYMBOLIC parameters in the procedure.

Symbolic parameters are used to override parameters on the DD statements & used both in Catalogued & In-stream PROC.

The same JCL can be used by different users to implement common task, such as opening, reading, writing of datasets.

A value assigned to a symbolic parameter may be overridden by another value, as long as the redefinition is within the same job.If it is not overridden then the same value will be assigned to it each time it is called.

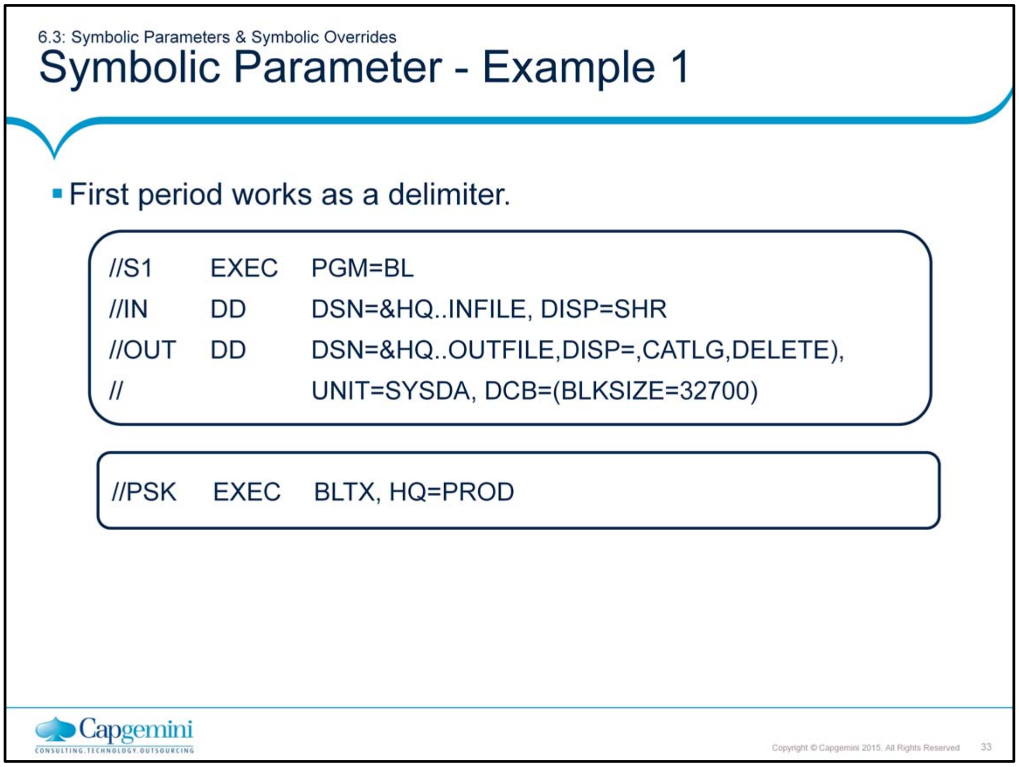
If Positional parameters are coded as symbolic then a period should be inserted between them.

Symbolic overrides can be used only when symbolic parameters have been coded inside the procedure.

```
//MYPROC PROC A=LP2WK1,B=SYSDA
//STEP1  EXEC PGM=IEFBR14
//DD1    DD DSN=MAINUSR.ABC.INPUT,DISP=SHR,VOL=SER=&A,UNIT=&B
//       PEND
//STEPX     EXEC MYPROC,A=LP1WK1,B=SYSSQ
AT RUNTIME
//STEP1 EXEC PGM=IEFBR14
//DD1     DD DSN=MAINUSR.ABC.INPUT,DISP=SHR,
//             VOL=SER=LP1WK1,UNIT=SYSSQ
```

In the above example, A and B are symbolic parameters. Instead of hard-coding the values for the VOL and UNIT parameters, they have been assigned the values contained in the symbolic parameters.

In case we do not change the value of the Symbolic parameters when the Procedure is called, the default values specified at the Procedure-declaration statement (PROC) are taken.



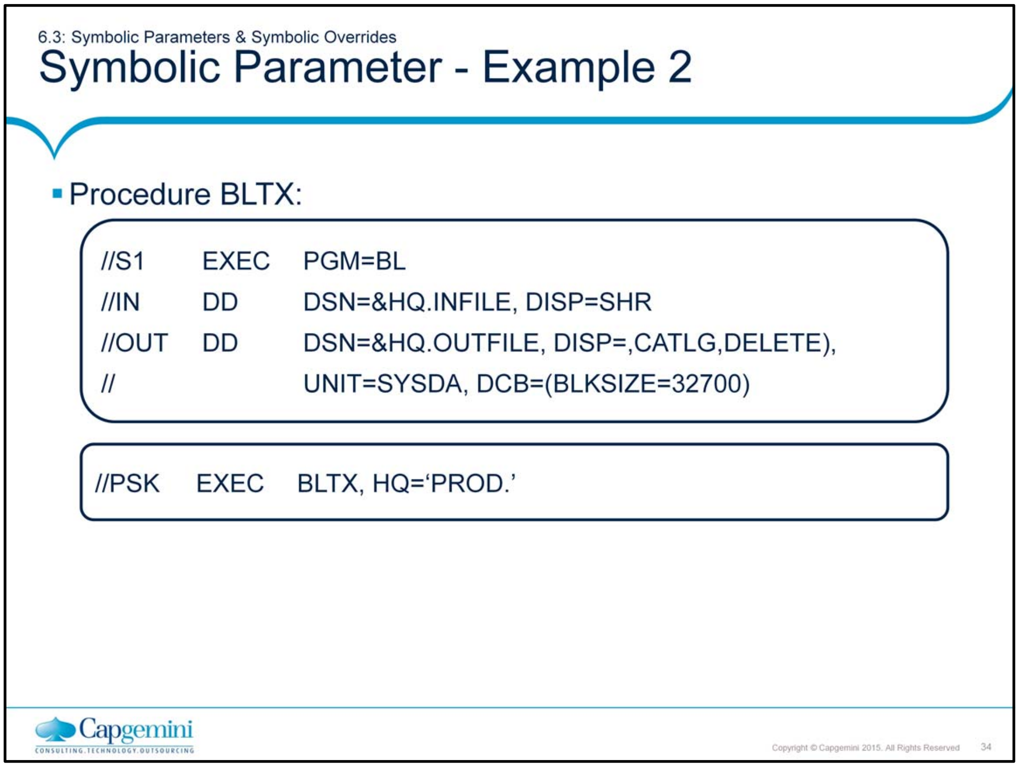
6.3: Symbolic Parameters & Symbolic Overrides

Symbolic Parameter - Example 1

- First period works as a delimiter.

```
//S1      EXEC    PGM=BL
//IN      DD      DSN=&HQ..INFILE, DISP=SHR
//OUT     DD      DSN=&HQ..OUTFILE,DISP=,CATLG,DELETE),
//                UNIT=SYSDA, DCB=(BLKSIZE=32700)
```

```
//PSK     EXEC    BLTX, HQ=PROD
```



6.3: Symbolic Parameters & Symbolic Overrides

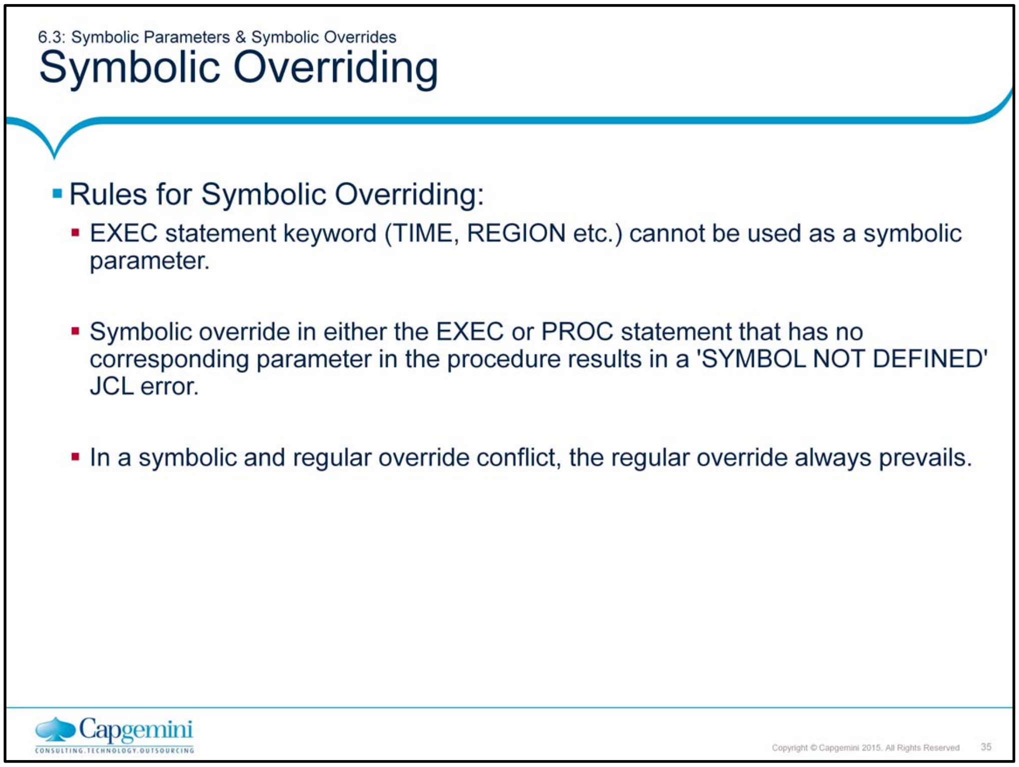
Symbolic Parameter - Example 2

- Procedure BLTX:

```
//S1     EXEC   PGM=BL
//IN     DD     DSN=&HQ.INFILE, DISP=SHR
//OUT    DD     DSN=&HQ.OUTFILE, DISP=,CATLG,DELETE),
//              UNIT=SYSDA, DCB=(BLKSIZE=32700)
```

```
//PSK   EXEC   BLTX, HQ='PROD.'
```

Copyright © Capgemini 2015. All Rights Reserved 34

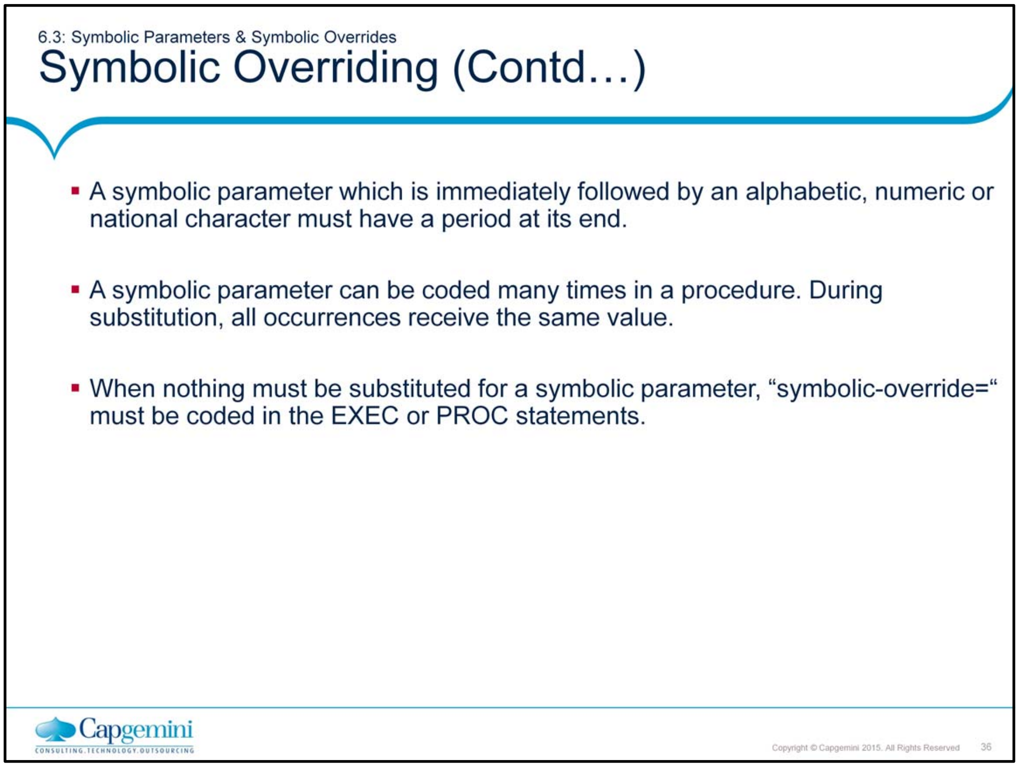


6.3: Symbolic Parameters & Symbolic Overrides

Symbolic Overriding

- Rules for Symbolic Overriding:
  - EXEC statement keyword (TIME, REGION etc.) cannot be used as a symbolic parameter.
  - Symbolic override in either the EXEC or PROC statement that has no corresponding parameter in the procedure results in a 'SYMBOL NOT DEFINED' JCL error.
  - In a symbolic and regular override conflict, the regular override always prevails.

35



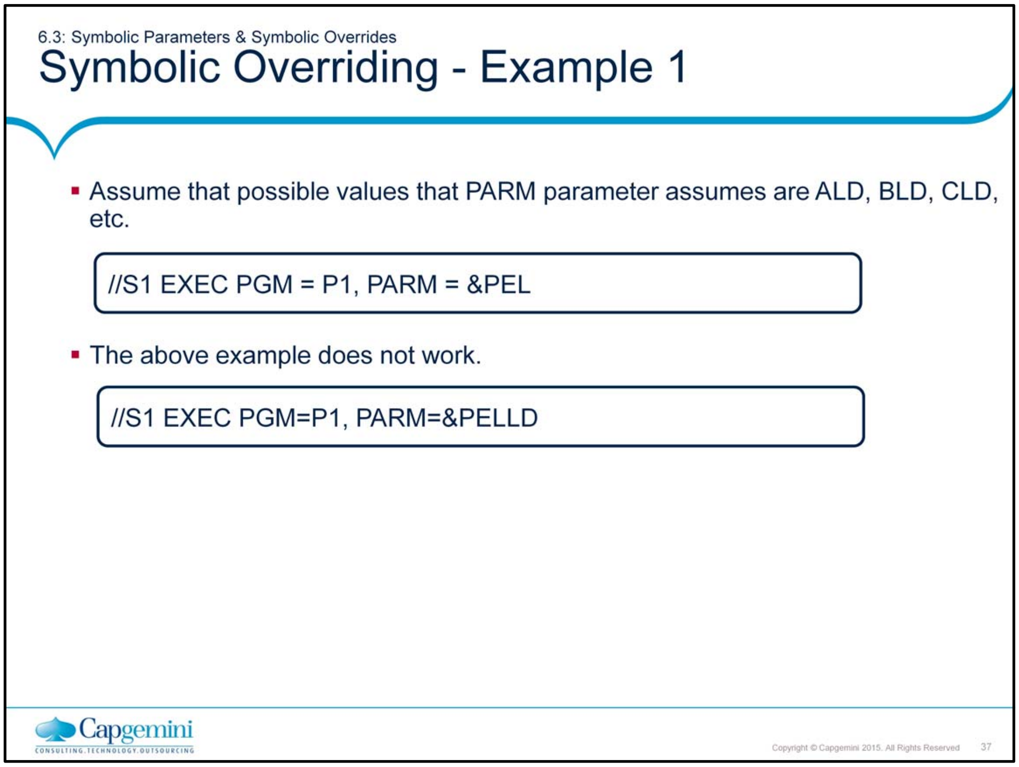
6.3: Symbolic Parameters & Symbolic Overrides

Symbolic Overriding (Contd…)

- A symbolic parameter which is immediately followed by an alphabetic, numeric or national character must have a period at its end.
- A symbolic parameter can be coded many times in a procedure. During substitution, all occurrences receive the same value.
- When nothing must be substituted for a symbolic parameter, “symbolic-override=“ must be coded in the EXEC or PROC statements.

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

36



6.3: Symbolic Parameters & Symbolic Overrides

Symbolic Overriding - Example 1

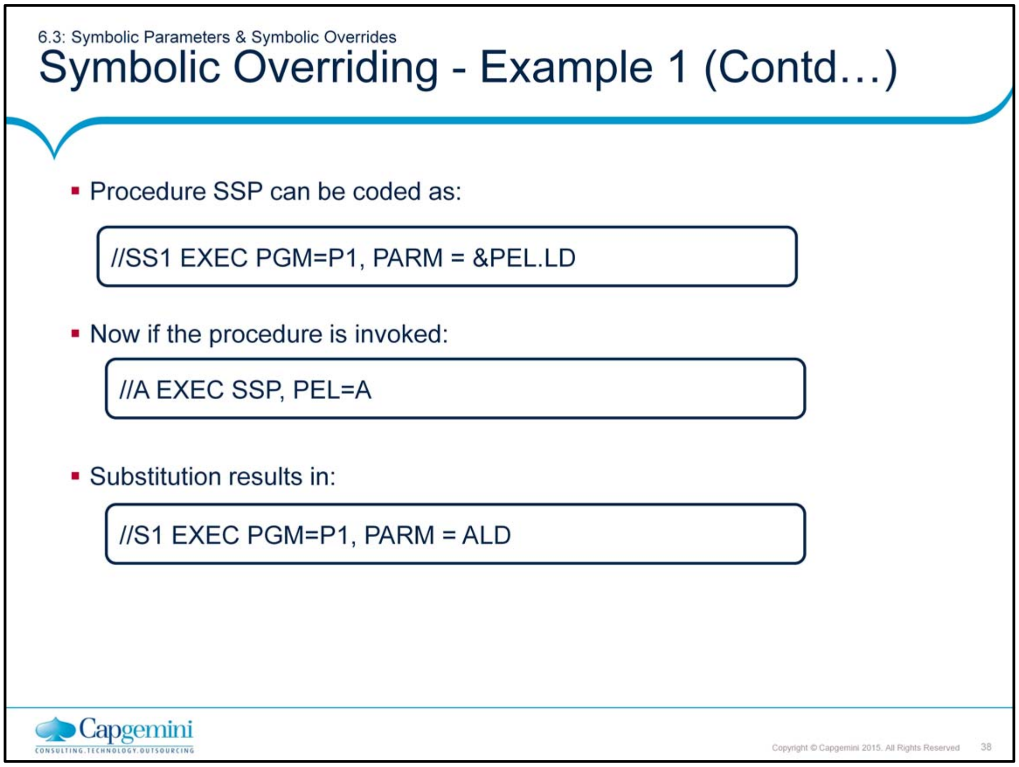
- Assume that possible values that PARM parameter assumes are ALD, BLD, CLD, etc.

```
//S1 EXEC PGM = P1, PARM = &PEL
```

- The above example does not work.

```
//S1 EXEC PGM=P1, PARM=&PELLD
```

37



6.3: Symbolic Parameters & Symbolic Overrides

Symbolic Overriding - Example 1 (Contd…)

- Procedure SSP can be coded as:

```
//SS1 EXEC PGM=P1, PARM = &PEL.LD
```

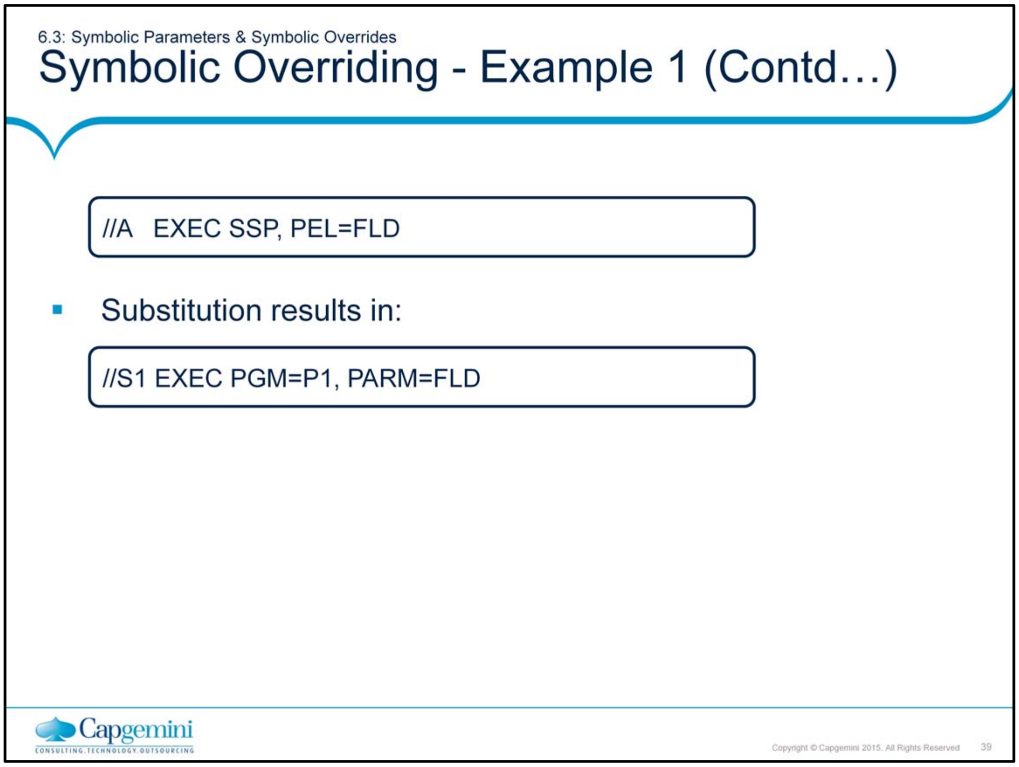
- Now if the procedure is invoked:

```
//A EXEC SSP, PEL=A
```

- Substitution results in:

```
//S1 EXEC PGM=P1, PARM = ALD
```

38



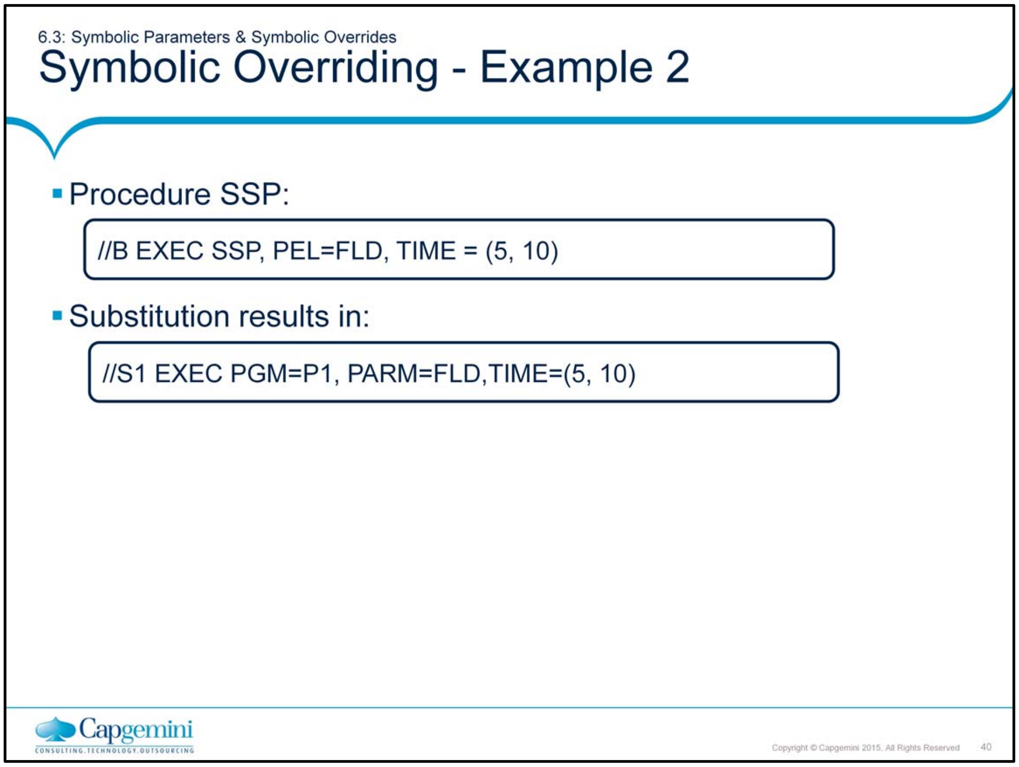
6.3: Symbolic Parameters & Symbolic Overrides

Symbolic Overriding - Example 1 (Contd…)

```
//A   EXEC SSP, PEL=FLD
```

- Substitution results in:

```
//S1 EXEC PGM=P1, PARM=FLD
```



6.3: Symbolic Parameters & Symbolic Overrides

Symbolic Overriding - Example 2

- Procedure SSP:

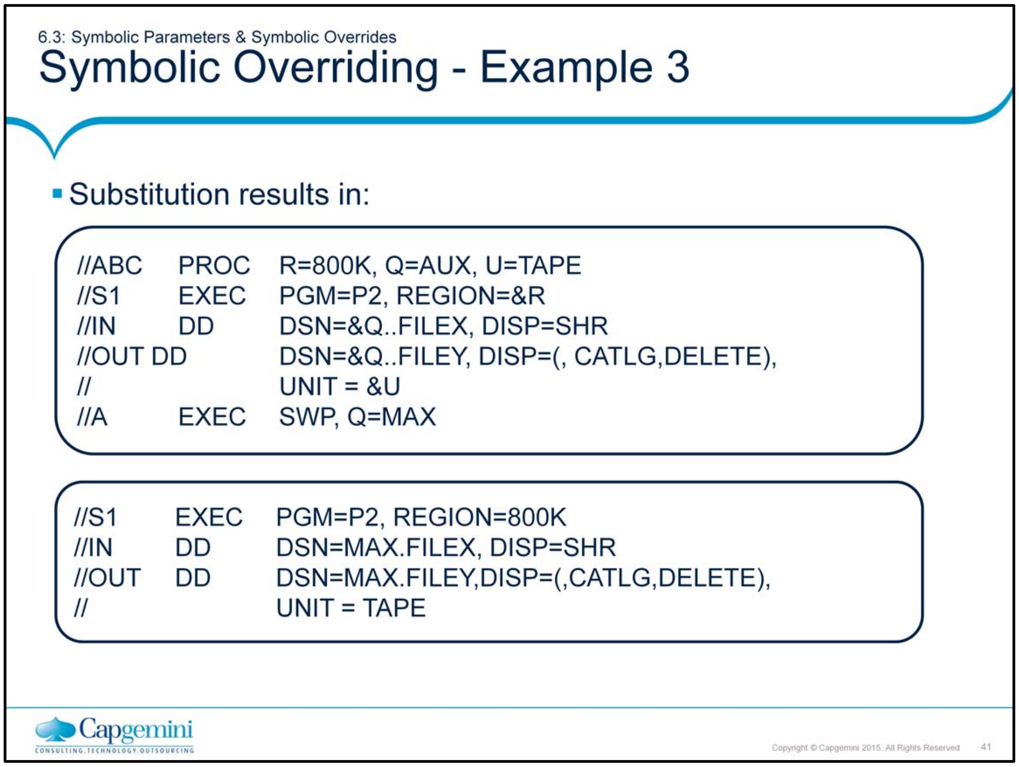
```
//B EXEC SSP, PEL=FLD, TIME = (5, 10)
```

- Substitution results in:

```
//S1 EXEC PGM=P1, PARM=FLD,TIME=(5, 10)
```

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

40



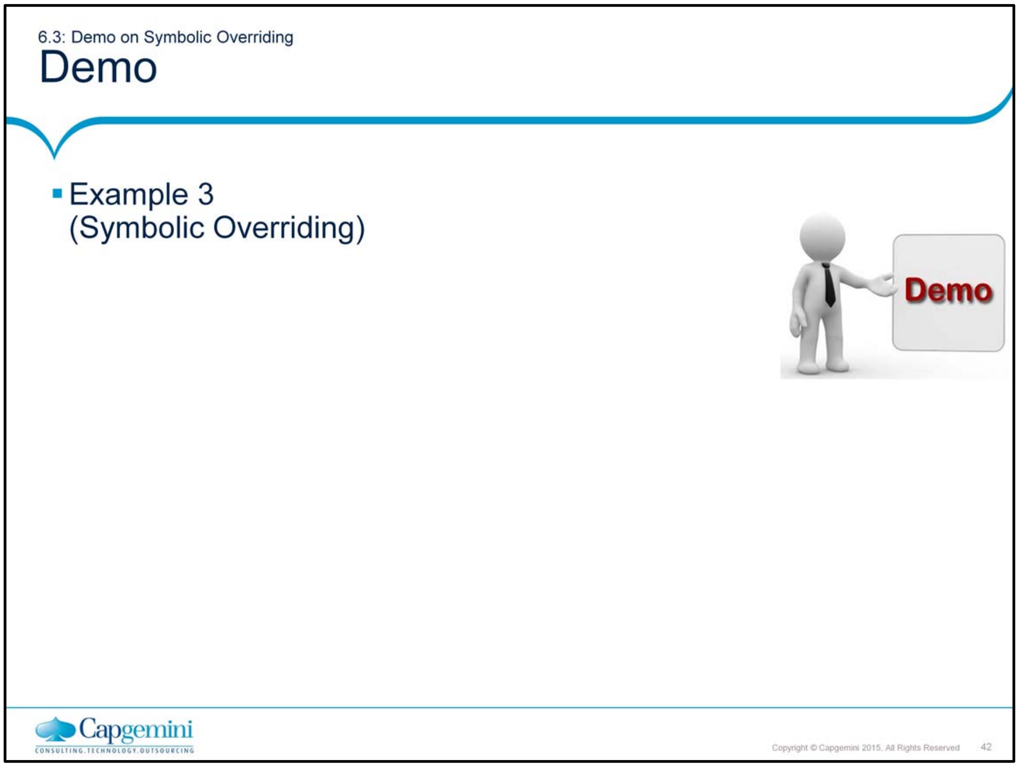
6.3: Symbolic Parameters & Symbolic Overrides

Symbolic Overriding - Example 3

- Substitution results in:

```
//ABC     PROC    R=800K, Q=AUX, U=TAPE
//S1      EXEC    PGM=P2, REGION=&R
//IN      DD      DSN=&Q..FILEX, DISP=SHR
//OUT DD          DSN=&Q..FILEY, DISP=(, CATLG,DELETE),
//                UNIT = &U
//A       EXEC    SWP, Q=MAX
```

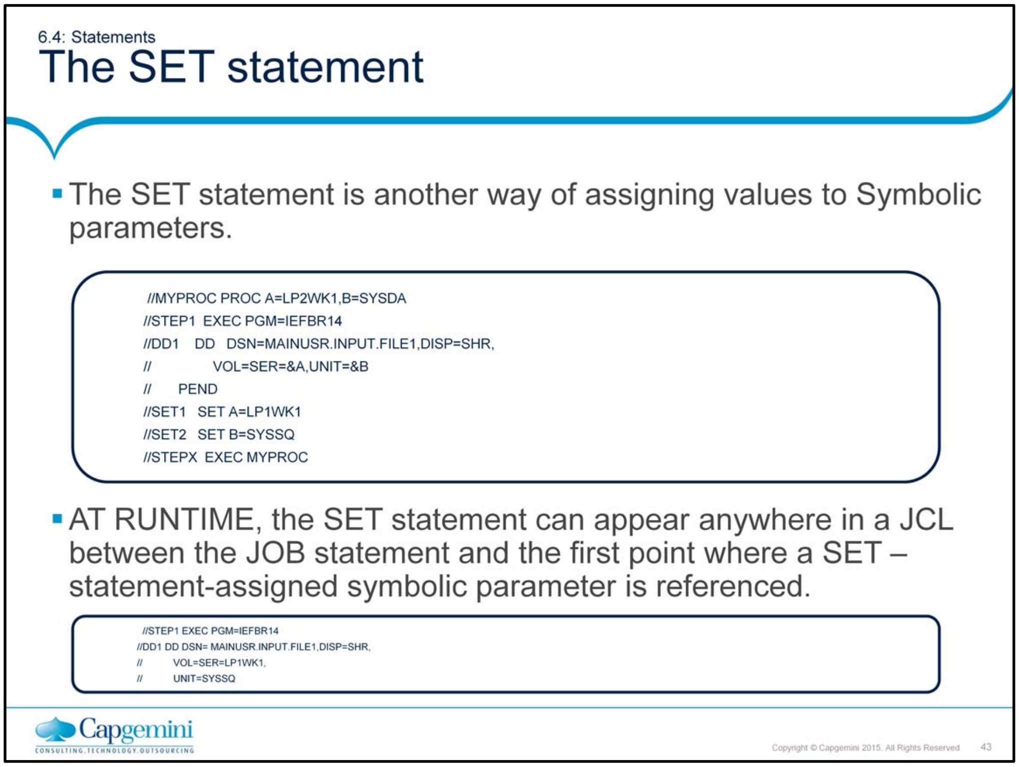
```
//S1      EXEC    PGM=P2, REGION=800K
//IN      DD      DSN=MAX.FILEX, DISP=SHR
//OUT     DD      DSN=MAX.FILEY,DISP=(,CATLG,DELETE),
//                UNIT = TAPE
```



6.3: Demo on Symbolic Overriding

Demo

- Example 3
  (Symbolic Overriding)



6.4: Statements

The SET statement

- The SET statement is another way of assigning values to Symbolic parameters.

```
 //MYPROC PROC A=LP2WK1,B=SYSDA
//STEP1  EXEC PGM=IEFBR14
//DD1    DD   DSN=MAINUSR.INPUT.FILE1,DISP=SHR,
//              VOL=SER=&A,UNIT=&B
//      PEND
//SET1   SET A=LP1WK1
//SET2   SET B=SYSSQ
//STEPX  EXEC MYPROC
```

- AT RUNTIME, the SET statement can appear anywhere in a JCL between the JOB statement and the first point where a SET – statement-assigned symbolic parameter is referenced.

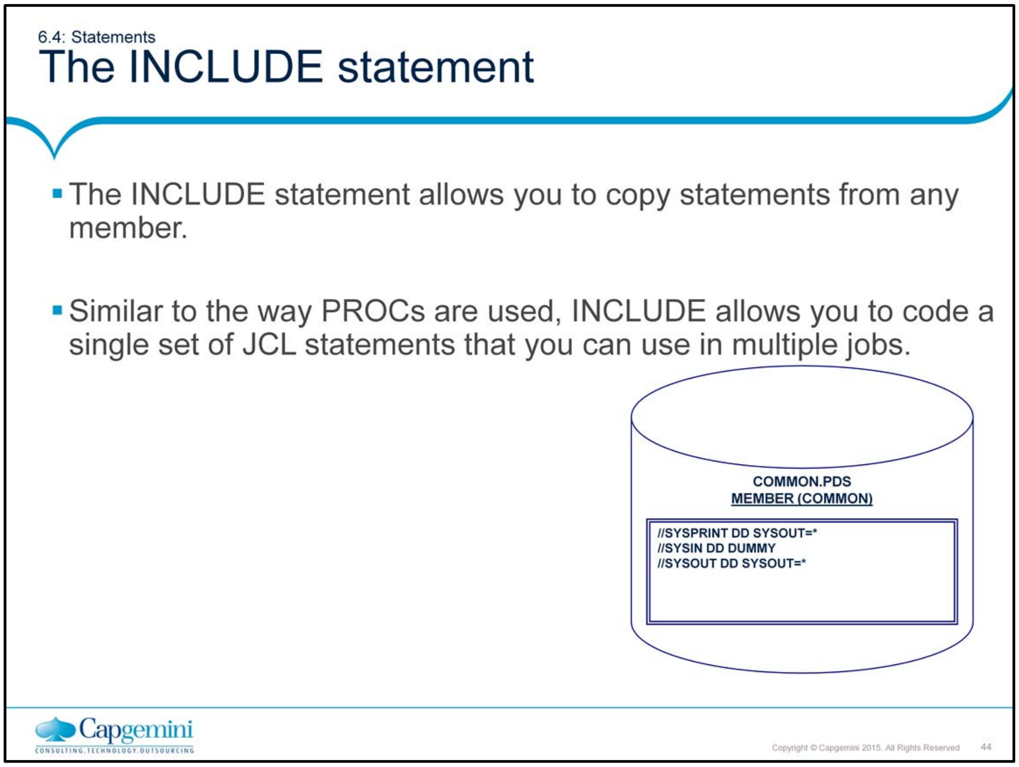
```
 //STEP1 EXEC PGM=IEFBR14
//DD1 DD DSN= MAINUSR.INPUT.FILE1,DISP=SHR,
//        VOL=SER=LP1WK1,
//        UNIT=SYSSQ
```

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

43

The SET Statement

```
//MCVT01A JOB NOTIFY=&SYSUID
//SETPROC PROC A=MCVT01
//STEP1 EXEC   PGM=IEFBR14
//DD1      DD DSN=MFCVT01.GTP74.SET6,
//            UNIT=SYSDA,VOL=SER=&A,
//            DCB=(LRECL=80,BLKSIZE=800,RECFM=FB),
//            SPACE=(TRK,(2,1)),DISP=(NEW,CATLG,DELETE)
//       PEND
//SET      SET  A=USER02
//STEP2   EXEC PROC=SETPROC
//SYSIN     DD DUMMY
//
```



6.4: Statements

The INCLUDE statement

- The INCLUDE statement allows you to copy statements from any member.
- Similar to the way PROCs are used, INCLUDE allows you to code a single set of JCL statements that you can use in multiple jobs.

COMMON.PDS
MEMBER (COMMON)

```
//SYSPRINT DD SYSOUT=*
//SYSIN DD DUMMY
//SYSOUT DD SYSOUT=*
```

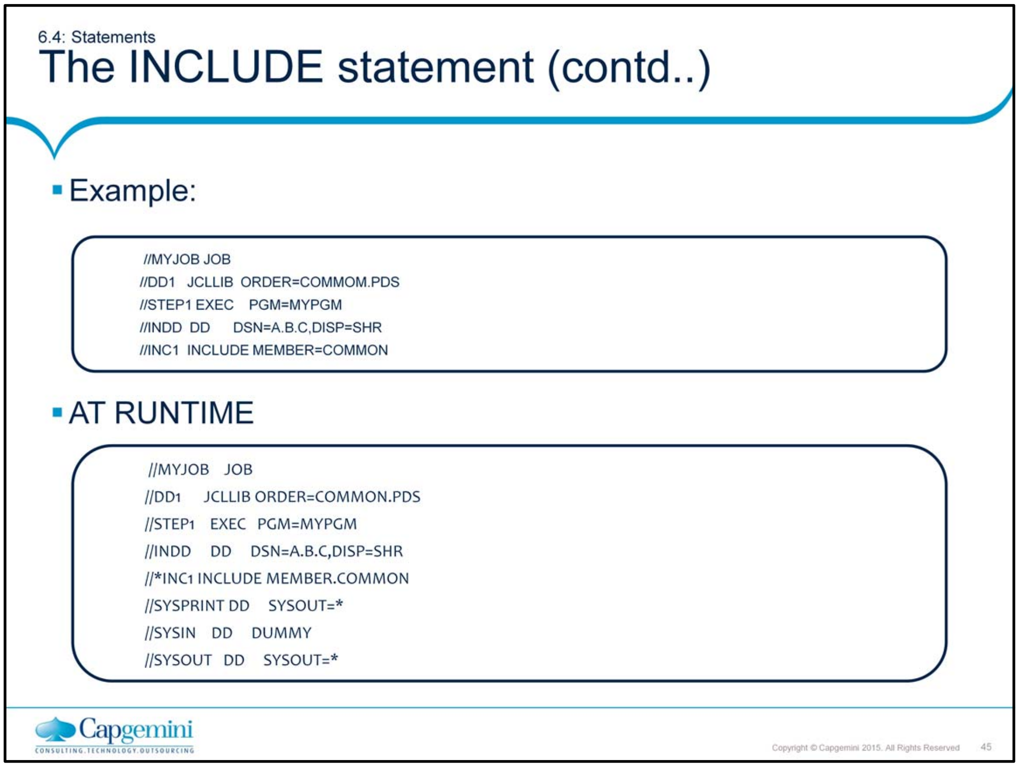
44

The Include Statement

```
//MCVT01A JOB     NOTIFY=&SYSUID
//DD1      JCLLIB  ORDER=(MCVT01.GTP74.SATYA)
//STEP1    EXEC    PGM=IEFBR14
//SYSPRINT  DD     SYSOUT=*
//SYSOUT    DD     SYSOUT=*
//DD1       DD      DSN=MCVT01.GTP.FILEX, DISP=(NEW,CATLG,DELETE),
//             VOL=SER=USER1,UNIT=SYSDA, SPACE=(TRK,(2,1)),
//            DCB=(LRECL=80,BLKSIZE=800,RECFM=FB)
//INC1     INCLUDE MEMBER=PDSSTAT
```

And **INCLUDE STATEMENTS(PDSSTAT)**

```
//SYSPRINT DD SYSOUT=*
//SYSIN DD DUMMY
//
```



6.4: Statements

The INCLUDE statement (contd..)

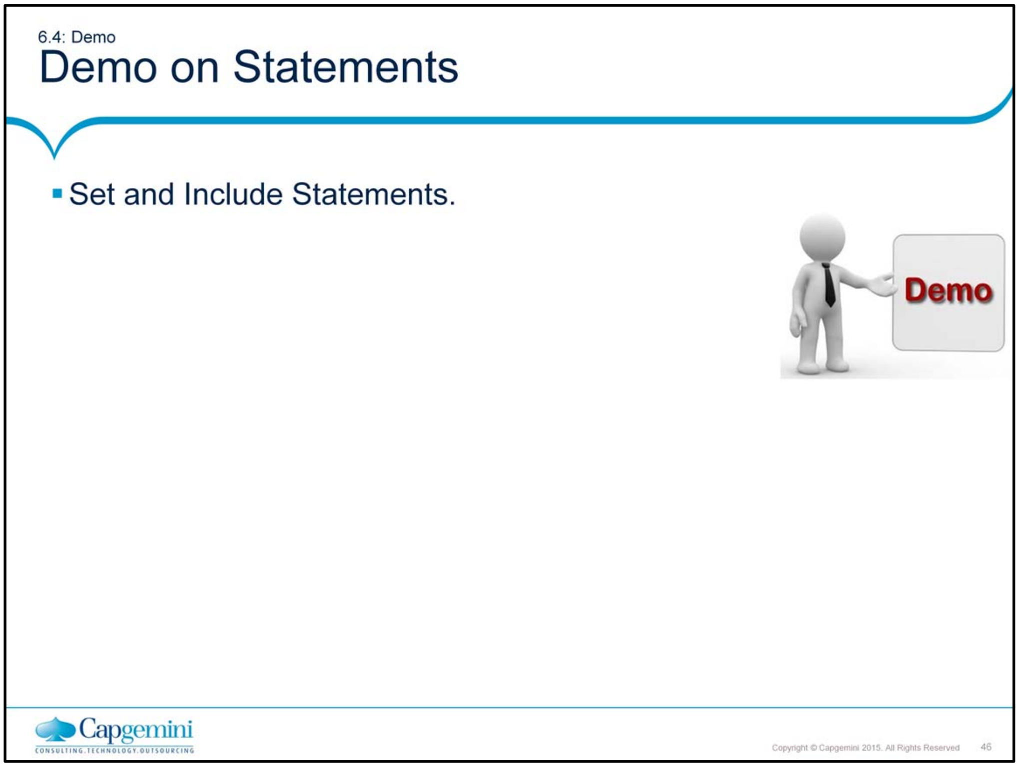
- Example:

```
 //MYJOB JOB
//DD1   JCLLIB  ORDER=COMMOM.PDS
//STEP1 EXEC    PGM=MYPGM
//INDD  DD      DSN=A.B.C,DISP=SHR
//INC1  INCLUDE MEMBER=COMMON
```

- AT RUNTIME

```
 //MYJOB    JOB
//DD1      JCLLIB ORDER=COMMON.PDS
//STEP1    EXEC   PGM=MYPGM
//INDD     DD     DSN=A.B.C,DISP=SHR
//*INC1 INCLUDE MEMBER.COMMON
//SYSPRINT DD     SYSOUT=*
//SYSIN    DD     DUMMY
//SYSOUT   DD     SYSOUT=*
```

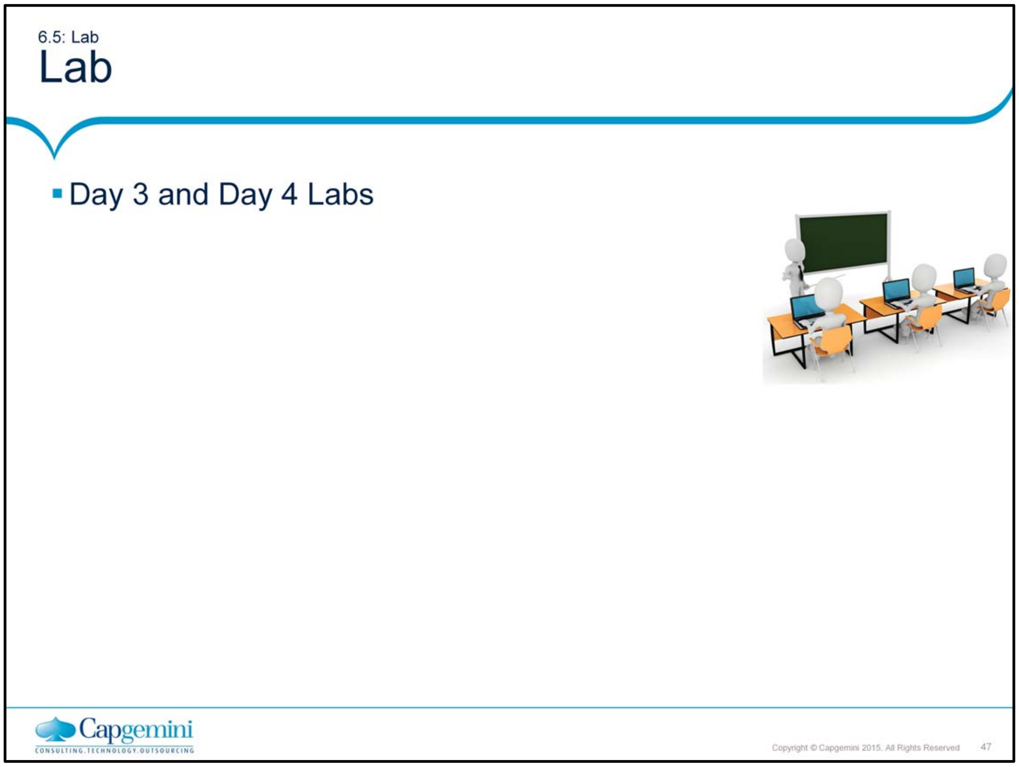
Copyright © Capgemini 2015. All Rights Reserved 45



6.4: Demo

Demo on Statements

- Set and Include Statements.

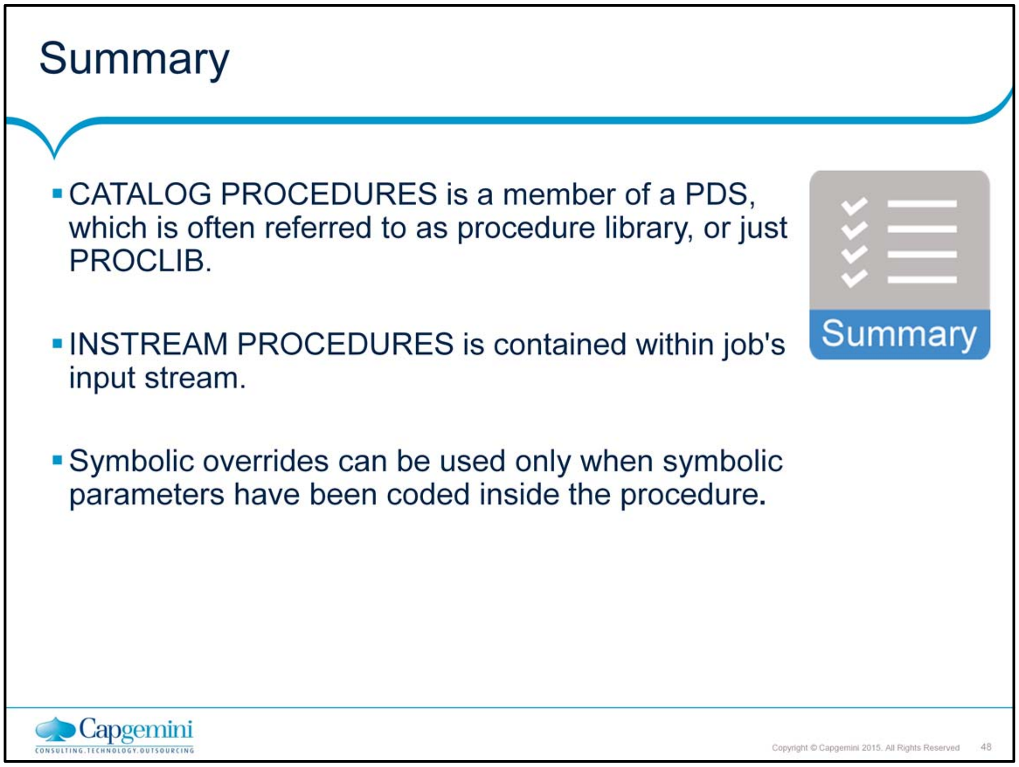


6.5: Lab

Lab

- Day 3 and Day 4 Labs

47

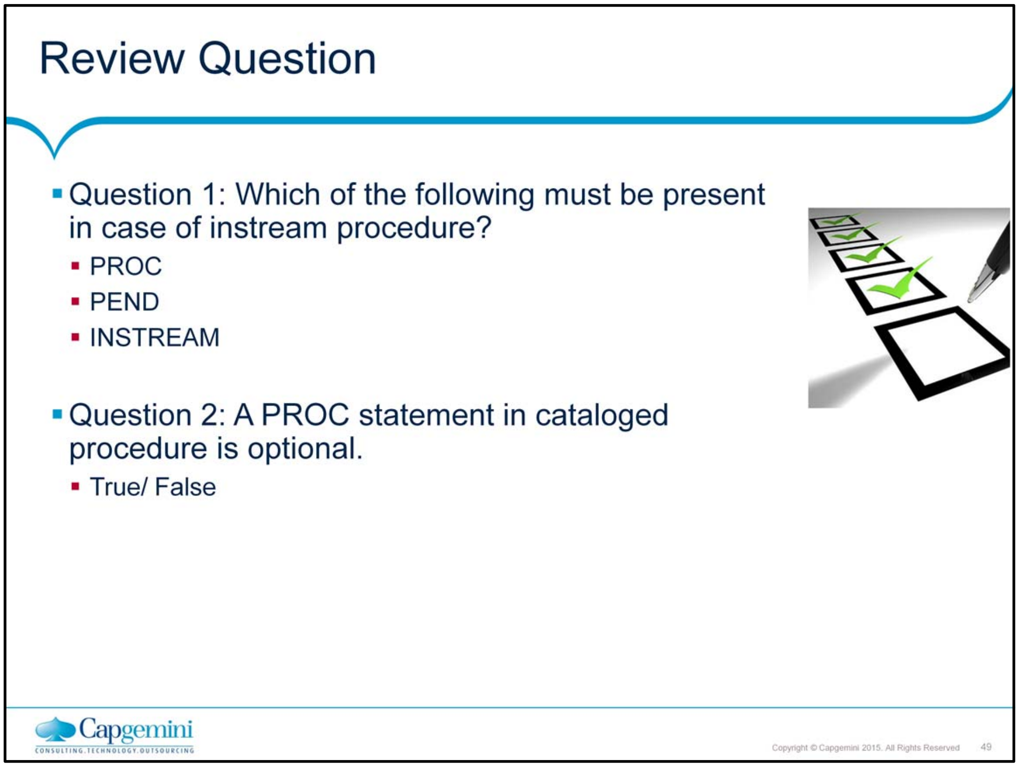


Summary

- CATALOG PROCEDURES is a member of a PDS, which is often referred to as procedure library, or just PROCLIB.
- INSTREAM PROCEDURES is contained within job's input stream.
- Symbolic overrides can be used only when symbolic parameters have been coded inside the procedure.

Summary
Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

48



Review Question

- Question 1: Which of the following must be present in case of instream procedure?
  - PROC
  - PEND
  - INSTREAM
- Question 2: A PROC statement in cataloged procedure is optional.
  - True/ False