# EPV for DB2
## Preparing Input for a Demo

*IT Cost Under Control*

Supporting
**EPV for DB2 V7**
**EPV for DB2 Plus V7**

**May 2013**

**All the trademarks mentioned belong to their respective companies.**

**EPV Technologies contact details:**

EPV Technologies
Viale Angelico, 54
00195 Roma
Tel. 06 86210880
Fax. 06 86387461
E-mail: epvtech@epvtech.com
WEB: http://www.epvtech.com

# Contents

# About this manual

This manual is intended to help anyone who wants to provide the data needed to prepare an EPV for DB2 demo.

## Changes

Technical changes or additions to the text are indicated by a vertical line to the left of the change.

# Terminology

A “view” is an EPV report presented in an HTML page.

# 1 Introduction

The best way to evaluate the benefits provided by EPV for DB2 for customers is to have a demo based on their data in their own environment.
Providing the data needed to prepare a demo is a quick and easy task to perform.

In this manual, after a short description of EPV for DB2 input data, a simple four step process to do that is presented.
Sample JCLs are also provided.

# 2 Mandatory input data

Some SMF records data are mandatory in order to run EPV for DB2. If you don't provide them EPV will not produce any usable output.
They are:

- Record 30 subtype 2,3 (Address Space Interval activity);
- Record 70 and 72 (RMF CPU and Workload activity);
- Record 100 (DB2 Statistics);
- Record 101 (DB2 Accounting).

## 2.1 SMF 30 subtype 2, 3 records

SMF 30 subtype 2 and 3 records are not produced by default.

To activate SMF interval accounting using the global recording interval the following parameters have to be set in the SMFPRMxx member of the SYS1.PARMLIB library:

- INTVAL(mm) where mm is the interval duration; suggested values are 10 or 15 minutes;
- SYNCVAL(nn) where nn is the minute in the hour that starts the interval; suggested value is 00;

In addition the following parameter have to be set under SYS and SUBSYS sections:

- INTERVAL(SMF,SYNC).

Writing of these records has to be allowed in SMFPRMxx (under the TYPE sub parameter).

It's very important you synchronise SMF and RMF data; to do that you must set the following parameter in the ERBRMFxx member, used by RMF Monitor I, of your SYS1.PARMLIB library:

- SYNCH(SMF).

## 2.2 SMF 70 and 72 records

SMF 70 and 72 records are produced by default.
However the following parameters are normally explicitly specified in RMF monitor I (ERBRMFxx member of the SYS1.PARMLIB library):

- CPU, to produce CPU activity information;
- WKLD, to produce Workload activity information.

Writing of these records also has to be allowed in SMFPRMxx (under the TYPE sub parameter).

## 2.3 SMF 100 records

To produce SMF 100 the DB2 Statistic Trace, Class 1 has to be activated.

Writing of these records also has to be allowed in SMFPRMxx (under the TYPE sub parameter).

## 2.4 SMF 101 records

To produce SMF 101 the DB2 Accounting Trace, Class 1 has to be activated.

Writing of these records also has to be allowed in SMFPRMxx (under the TYPE sub parameter).

# 3 Other suggested input data

Using only the mandatory data will result in a subset of the EPV for DB2 views and analysis. So you are strongly advised to also provide additional info in SMF 100 and 101 records by activating also the following trace classes:

- Record 100, DB2 Statistic Trace, Class 3,5,6[1].
- Record 101, DB2 Accounting Trace, Class 2,3.

To get package CPU usage, accounting trace class 7 is needed.
To get information about package waits, accounting trace class 8 is needed.
Additional package information (such as the number of getpages) are available if trace class 10 is also activated[2].

[1] The class 6 only needed for DB2 releases prior than V9.
[2] Since DB2 V9; in prior releases this information is provided by class 7.

# 4 Optional input data

EPV can show detailed deadlock and timeout information collected from SMF 102 records; IFCID 105,172 and 196 have to be activated[3].

EPV shows database and index statistics collected querying the DB2 catalog. The needed information must be gathered daily from each subsystem.

EPV also shows DSNZPARM parameter settings collected running the IBM provided DSNWZP stored procedure. The needed information must be gathered daily from each subsystem.

[3] Statistic trace Class 3 has to be active.

# 5 Preparing data for a demo

To have a good demo, a few hours worth of data are enough. If you have more systems sharing resources the result will be better. If you had a bad day, with lot of problems, the EPV demo will probably help you understand what happened.
The following steps have to be performed in order to prepare input data for an EPV demo.

## 5.1 Collect the data

Three JCLs have to be executed to collect all the needed data.

**JCL1**

The following JCL will select all the necessary SMF records.

Cut and paste it in your JCL library, and do the following customizations:

- CHANGE *smfinput* TO YOUR SMF INPUT FILE NAME
- CHANGE *smfpref* TO OUTPUT FILE PREFIX
- CHANGE *yyyyxxx* to the starting and ending Julian date you want to select
- CHANGE *hhmm* to the starting and ending hours you want to select

```
//SELSMF   EXEC PGM=IFASMFDP
//SYSPRINT DD SYSOUT=*
//INDD1    DD DSN=smfinput,DISP=SHR
//OUTDD1   DD DSN=smfpref.VBS,DISP=(,CATLG),
//         UNIT=SYSDA, SPACE=(CYL,(100,100),RLSE),
//         DCB=(LRECL=32760,BLKSIZE=27998,RECFM=VBS)
//SYSIN    DD *
 INDD(INDD1,OPTIONS(DUMP))
 OUTDD(OUTDD1,TYPE(30(2,3),70,72,100,101,102))
 DATE(yyyyxxx,yyyyxxx)
 START(hhmm)
 END(hhmm)
/*
```

**JCL2**

The following JCL will produce files containing DSNZPARM parameter settings calling the IBM provided DSNWZP stored procedure.

Cut and paste it in your JCL library, and do the following customizations:

- CHANGE *prefix* TO YOUR DB2 LIB PREFIX
- CHANGE *db2id* TO YOUR DB2 ID
- CHANGE *seqpref* TO OUTPUT FILE PREFIX

```
//STEP01   EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=*
//SYSIN    DD *
 DELETE seqpref.db2id.DSNZPARM
/*
//*
//STEP02   EXEC PGM=IEBGENER
//SYSPRINT DD SYSOUT=*
//SYSIN    DD DUMMY
//SYSUT1   DD DATA,DLM=EE
/* REXX */
PARSE ARG SSID
IF SSID = '' THEN DO
   SAY '*** ERROR - SSID NOT SET'
   EXIT (12)
END
"SUBCOM DSNREXX"
IF RC = 1
THEN S_RC = RXSUBCOM('ADD','DSNREXX','DSNREXX')
ADDRESS DSNREXX "CONNECT " SSID

IF SQLCODE <> 0 THEN DO
   SAY '*** ERROR - UNEXPECTED SQLCODE CONNECTING TO' SSID
   SAY '*** ERROR - SQLCODE  = ' SQLCODE
   EXIT (12)
END
/* SET UP SQLDA FOR CALL TO DSNWZP */
INSQLDA.SQLD = 1
INSQLDA.1.SQLTYPE = 449        /* VARCHAR */
INSQLDA.1.SQLLEN  = 32000
INSQLDA.1.SQLIND  = 0
INSQLDA.1.SQLDATA = ' '

/* CALL PROCEDURE DSNWZP */
ADDRESS DSNREXX
"EXECSQL CALL :DSNWZP USING DESCRIPTOR :INSQLDA  "
IF SQLCODE = 0 THEN DO
    RESULT = INSQLDA.1.SQLDATA
    PARMCOUNT = 1
    I = POS(X2C(404025),RESULT)
    DO WHILE ( I <> 0)
      RESULT = SUBSTR(RESULT,4)
      NI = POS(X2C(404025),RESULT)
      IF NI = 0 THEN NI = LENGTH(RESULT) + 1
      ZPARM.PARMCOUNT = SUBSTR(RESULT,1,NI-1)
      ZPARM.0 = PARMCOUNT
      PARMCOUNT = PARMCOUNT + 1
      RESULT = SUBSTR(RESULT,NI)
      I = POS(X2C(404025),RESULT)
    END

    DO I = 1 TO ZPARM.0
      ROWVALU = ZPARM.I
      OROWVALU = ROWVALU
```

```
      L = POS('/',ROWVALU)
      W1 = SUBSTR(LEFT(ROWVALU,L-1),1,8)
      WR = SUBSTR(ROWVALU,L+1)
      L = POS('/',WR)
      W2 = SUBSTR(LEFT(WR,L-1),1,8)
      WR = SUBSTR(WR,L+1)
      L = POS('/',WR)
      W3 = SUBSTR(LEFT(WR,L-1),1,24)
      WR = SUBSTR(WR,L+1)
      L = POS('/',WR)
      W4 = SUBSTR(LEFT(WR,L-1),1,8)
      WR = SUBSTR(WR,L+1)
      L = POS('/',WR)
      W5 = SUBSTR(LEFT(WR,L-1),1,8)
      WR = SUBSTR(WR,L+1)
      L = POS('/',WR)
      W6 = SUBSTR(LEFT(WR,L-1),1,40)
      WR = SUBSTR(WR,L+1,40)
      IF I = ZPARM.0 THEN DO
         WW = '';
         DO K=1 TO 40
            XX=SUBSTR(WR,K,1)
            IF C2X(XX) = '00' THEN DO
               XX = ' '
            END
            SAY WW
            WW = WW!!XX
         END
         WR = WW
      END
      OUT.I = W2 W3 WR W6 W4 W5
    END
    ADDRESS
    "EXECIO * DISKW PARMOUT (STEM OUT. FINIS"
    IF RC <> 0 THEN DO
       SAY '*** ERROR - CAN NOT WRITE OUTPUT FILE'
       EXIT (12)
       END
    EXIT (0)
END
ELSE DO
   SAY '*** ERROR - UNEXPECTED SQLCODE CALLING DSNWZP'
   SAY '*** ERROR - SQL CODE = ' SQLCODE
   EXIT (12)
END
EE
//*
//SYSUT2    DD DSN=&&PDS(REXXDSNZ),UNIT=SYSDA,
//          DISP=(NEW,PASS,DELETE),
//          SPACE=(TRK,(1,1,1)),
//          DCB=(LRECL=80,BLKSIZE=3120,RECFM=FB,DSORG=PO)
//*
//STEP03    EXEC PGM=IRXJCL,PARM='REXXDSNZ db2id'
//STEPLIB   DD DSN=prefix.SDSNLOAD,DISP=SHR
//SYSEXEC   DD DSN=&&PDS,DISP=(OLD,DELETE,DELETE)
```

```
//SYSTSPRT  DD SYSOUT=*
//SYSPRINT  DD SYSOUT=*
//SYSUDUMP  DD SYSOUT=*
//PARMOUT   DD DISP=(,CATLG),DSN=seqpref.db2id.DSNZPARM,
//          UNIT=SYSDA,SPACE=(CYL,(1,1)),
//          DCB=(LRECL=255,RECFM=FB,BLKSIZE=0)
//SYSTSIN   DD DUMMY
/*
//*
//STEP04    EXEC PGM=IEBGENER
//SYSPRINT  DD SYSOUT=*
//SYSIN     DD DUMMY
//SYSUT1    DD DATA,DLM=EE
/* REXX */
TRACE 'O'
CVTADDR = C2D(STORAGE(10,4))   /* CVTADDR FROM PSA + X'10'  */
AMB = STORAGE(D2X(CVTADDR+X2D('154')),8)
DD = DATE()
R.0=1
R.1='EPV ==> SYSNAME' AMB DD
"EXECIO * DISKW OUT1 (STEM R. FINIS"
IF RC <> 0 THEN EXIT 8
EXIT 0
EE
//*
//SYSUT2    DD DSN=&&PDS(SYS),UNIT=SYSDA,
//          DISP=(NEW,PASS,DELETE),
//          SPACE=(TRK,(1,1,1)),
//          DCB=(LRECL=80,BLKSIZE=3120,RECFM=FB,DSORG=PO)
//*
//STEP05    EXEC PGM=IRXJCL,PARM='SYS'
//SYSEXEC   DD DSN=&&PDS,DISP=(OLD,DELETE,DELETE)
//SYSTSPRT  DD SYSOUT=*
//SYSPRINT  DD SYSOUT=*
//SYSUDUMP  DD SYSOUT=*
//OUT1      DD DISP=MOD,DSN=seqpref.db2id.DSNZPARM
//SYSTSIN   DD DUMMY
/*
```

## JCL3

The following JCL will produce files containing database statistics collected querying the DB2 catalog.

Cut and paste it in your JCL library, and do the following customizations:

- CHANGE prefix TO YOUR DB2 LIB PREFIX
- CHANGE db2id TO YOUR DB2 ID
- CHANGE seqpref TO OUTPUT FILE PREFIX
- CHANGE aulplan TO DSNTIAUL PLAN (default is DSNTIB81 in 8.1 or DSNTIB91 in 9.1)

```
//STEP01   EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=*
```

```
//SYSIN    DD *
 DELETE seqpref.db2id.DBCOUNT
/*
//*
//STEP02   EXEC PGM=IKJEFT01,DYNAMNBR=20
//STEPLIB  DD DISP=SHR,DSN=db2id.DB2.SDSNLOAD
//SYSTSPRT DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSPUNCH DD SYSOUT=*
//SYSTSIN  DD *
  DSN SYSTEM(db2id)
  RUN  PROGRAM(DSNTIAUL) PLAN(aulplan)  PARMS('SQL') -
       LIB('prefix.RUNLIB.LOAD')
/*
//SYSREC00 DD DISP=(,CATLG),DSN=seqpref.db2id.DBCOUNT,
//         UNIT=SYSDA,SPACE=(CYL,(1,1)),
//         DCB=(LRECL=133,RECFM=FB,BLKSIZE=0)
//SYSUDUMP DD DUMMY
//SYSIN    DD *
   SELECT * FROM (
   SELECT SUBSTR(CURRENT SERVER,1,4)          ,
          SUBSTR(';',1,1)                     ,
          SUBSTR(CURRENT MEMBER,1,4)          ,
          SUBSTR(';',1,1)                     ,
          SUBSTR(DBNAME,1,24)                 ,
          SUBSTR(';',1,1)                     ,
          SUBSTR(TSNAME,1,24)                 ,
          SUBSTR(';',1,1)                     ,
          DIGITS(PARTITION)                   ,
          SUBSTR(';',1,1)                     ,
          DIGITS(DECIMAL((CARDF)))            ,
          SUBSTR(';',1,1)                     ,
          SUBSTR(DIGITS(DECIMAL((SPACEF)/(1024),8,2)),1,6)
          !!'.'!!
          SUBSTR(DIGITS(DECIMAL((SPACEF)/(1024),8,2)),7,2),
          SUBSTR(';',1,1)                     ,
          (DATE(STATSTIME))
          FROM SYSIBM.SYSTABLEPART
          ) AS TBDUM;
/*
//*
//STEP03   EXEC PGM=IEBGENER
//SYSPRINT DD SYSOUT=*
//SYSIN    DD DUMMY
//SYSUT1   DD DATA,DLM=EE
/* REXX */
TRACE 'O'
CVTADDR = C2D(STORAGE(10,4))   /* CVTADDR FROM PSA + X'10'  */
AMB = STORAGE(D2X(CVTADDR+X2D('154')),8)
DD = DATE()
R.0=1
R.1='EPV ==> SYSNAME' AMB DD
"EXECIO * DISKW OUT2 (STEM R. FINIS"
IF RC <> 0 THEN EXIT 8
EXIT 0
```

```
EE
//*
//SYSUT2   DD  DSN=&&PDS(SYS),UNIT=SYSDA,
//         DISP=(NEW,PASS,DELETE),
//         SPACE=(TRK,(1,1,1)),
//         DCB=(LRECL=80,BLKSIZE=3120,RECFM=FB,DSORG=PO)
//*
//STEP04   EXEC PGM=IRXJCL,PARM='SYS'
//SYSEXEC  DD DSN=&&PDS,DISP=(OLD,DELETE,DELETE)
//SYSTSPRT DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//OUT2     DD DISP=MOD,DSN=seqpref.db2id.DBCOUNT
//SYSTSIN  DD DUMMY
/*
```

## JCL4

The following JCL will produce files containing index statistics collected querying the DB2 catalog.

Cut and paste it in your JCL library, and do the following customizations:

- CHANGE prefix TO YOUR DB2 LIB PREFIX
- CHANGE db2id TO YOUR DB2 ID
- CHANGE seqpref TO OUTPUT FILE PREFIX
- CHANGE aulplan TO DSNTIAUL PLAN (default is DSNTIB81 in 8.1 or DSNTIB91 in 9.1)

```
//STEP01   EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=*
//SYSIN    DD *
 DELETE seqpref.db2id. IXCOUNT
/*
//*
//STEP02   EXEC PGM=IKJEFT01,DYNAMNBR=20
//STEPLIB  DD DISP=SHR,DSN=db2id.DB2.SDSNLOAD
//SYSTSPRT DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSPUNCH DD SYSOUT=*
//SYSTSIN  DD *
  DSN SYSTEM(db2id)
  RUN  PROGRAM(DSNTIAUL) PLAN(aulplan)  PARMS('SQL') -
       LIB('prefix.RUNLIB.LOAD')
/*
//SYSREC00 DD DISP=(,CATLG),DSN=seqpref.db2id.IXCOUNT,
//         UNIT=SYSDA,SPACE=(CYL,(1,1)),
//         DCB=(LRECL=133,RECFM=FB,BLKSIZE=0)
//SYSUDUMP DD DUMMY
//SYSIN    DD *
   SELECT * FROM (
   SELECT SUBSTR(CURRENT SERVER,1,4)          ,
          SUBSTR(';',1,1)                     ,
          SUBSTR(CURRENT MEMBER,1,4)          ,
          SUBSTR(';',1,1)                     ,
```

```
          SUBSTR(DBNAME,1,24)                 ,
          SUBSTR(';',1,1)                     ,
          SUBSTR(INDEXSPACE,1,24)             ,
          SUBSTR(';',1,1)                     ,
          DIGITS(PARTITION)                   ,
          SUBSTR(';',1,1)                     ,
          DIGITS(DECIMAL((CARDF)))            ,
          SUBSTR(';',1,1)                     ,
          SUBSTR(DIGITS(DECIMAL((SPACEF)/(1024),8,2)),1,6)
          !!'.'!!
          SUBSTR(DIGITS(DECIMAL((SPACEF)/(1024),8,2)),7,2),
          SUBSTR(';',1,1)                     ,
          (DATE(STATSTIME))                   ,
          SUBSTR(';',1,1)                     ,
          (DATE(LASTUSED))
     FROM SYSIBM.SYSINDEXPART ,
    (SELECT DBNAME,TBNAME,INDEXSPACE,NAME AS XNAME,CREATOR AS XCREATOR
     FROM SYSIBM.SYSINDEXES ) AS TBX ,
    (SELECT DBNAME AS SDBNAME,NAME AS SNAME,CREATOR AS SCREATOR
     ,PARTITION AS SPARTITION,LASTUSED
     FROM SYSIBM.SYSINDEXSPACESTATS ) AS TBXS
     WHERE IXNAME    = XNAME
     AND   IXCREATOR = XCREATOR
     AND   IXNAME    = SNAME
     AND   IXCREATOR = SCREATOR
     AND   PARTITION = SPARTITION
     ) AS TBDUM;
/*
//*
//STEP03   EXEC PGM=IEBGENER
//SYSPRINT DD SYSOUT=*
//SYSIN    DD DUMMY
//SYSUT1   DD DATA,DLM=EE
/* REXX */
TRACE 'O'
CVTADDR = C2D(STORAGE(10,4))   /* CVTADDR FROM PSA + X'10'  */
AMB = STORAGE(D2X(CVTADDR+X2D('154')),8)
DD = DATE()
R.0=1
R.1='EPV ==> SYSNAME' AMB DD
"EXECIO * DISKW OUT2 (STEM R. FINIS"
IF RC <> 0 THEN EXIT 8
EXIT 0
EE
//*
//SYSUT2   DD  DSN=&&PDS(SYS),UNIT=SYSDA,
//         DISP=(NEW,PASS,DELETE),
//         SPACE=(TRK,(1,1,1)),
//         DCB=(LRECL=80,BLKSIZE=3120,RECFM=FB,DSORG=PO)
//*
//STEP04   EXEC PGM=IRXJCL,PARM='SYS'
//SYSEXEC  DD DSN=&&PDS,DISP=(OLD,DELETE,DELETE)
//SYSTSPRT DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
```

```
//OUT2     DD DISP=MOD,DSN=seqpref.db2id.IXCOUNT
//SYSTSIN  DD DUMMY
/*
```

## 5.2 Transfer data to PC

When transferring variable data (VB or VBS) from the mainframe to other platforms it is obviously important to do that without corrupting the logical structure of the records. The JCL to use differs slightly depending on the available File Transfer (FT) tools used.

**JCL5 – (you need to use the standard IBM FTP)**

The following JCLs will transfer the SMF files produced using the standard IBM FTP.
Records are read as if in undefined format in order to avoid FTP eliminating the VB and VBS headers and so corrupting the records. As stated in the comments it is essential not to change the RECFM and BLKSIZE parameters. It's also required that the transfer is done in binary mode.

Cut and paste it in your JCL library, and do the following customizations:

- CHANGE *smfpref* TO INPUT AND OUTPUT FILE PREFIX
- CHANGE FTP parameters (*your.ftp.host.address, user and password*) to appropriate values

```
//* DO NOT CHANGE RECFM=U NOR BLKSIZE=32760 ON //DDSMF
//FTPSMF   EXEC PGM=FTP,PARM='(EXIT'
//SYSPRINT DD SYSOUT=*
//OUTPUT   DD SYSOUT=*
//DDSMF    DD DSN=smfpref.VBS,DCB=RECFM=U,BLKSIZE=32760,DISP=SHR
//INPUT    DD *
your.ftp.host.address
user password
quote PASV
bin
put //DD:DDSMF /smfpref.smf
close
quit
/*
```

**JCL 6**

The following JCLs will transfer the sequential files produced using the standard IBM FTP. *You only have to substitute JCL and commands in the second step to run any different FT tool.*

Cut and paste it in your JCL library, and do the following customizations:

- CHANGE *db2id* TO YOUR DB2 ID
- CHANGE *seqpref* TO OUTPUT FILE PREFIX
- CHANGE FTP parameters (*your.ftp.host.address, user and password*) to appropriate values

```
//FTPSTAT EXEC PGM=FTP,PARM='(EXIT'
//SYSPRINT DD SYSOUT=*
//OUTPUT   DD SYSOUT=*
```

```
//INPUT    DD *
your.ftp.host.address
user password
quote PASV
ascii
put 'seqpref.db2id.DSNZPARM'  db2id.DSNZPARM.TXT
put 'seqpref.db2id.DBCOUNT'  db2id.DBCOUNT.TXT
put 'seqpref.db2id.IXCOUNT'  db2id.IXCOUNT.TXT
close
quit
/*
```

## JCL5.1 – (alternative to JCL5 if ou don't use the standard IBM FTP)

The following JCL will transfer the files produced using tools other than the standard IBM FTP.
A first preliminary IEBGENER step changes the DSCB from VB and VBS to undefined format. A second step transfers the data. It's required that the transfer is done in binary mode.
*In the following examples we refer to IBM FTP, you have to substitute JCL and commands in the second step with the ones required to run your FT tool.*

Cut and paste it in your JCL library, and do the following customizations:

- CHANGE *smfpref* TO INPUT AND OUTPUT FILE PREFIX
- CHANGE FTP parameters (*your.ftp.host.address, user and password*) to appropriate values

```
//* DO NOT CHANGE RECFM=U NOR BLKSIZE=32760 ON BOTH DD
//UNDSMF   EXEC PGM=IEBGENER
//SYSPRINT DD SYSOUT=*
//SYSUT2   DD DSN=smfpref.VBS,DISP=MOD,
//         DCB=RECFM=U
//SYSUT1   DD DSN=NULLFILE,DCB=*.SYSUT2
//SYSIN    DD DUMMY
/*
//FTPSMF   EXEC PGM=FTP,PARM='(EXIT'
//SYSPRINT DD  SYSOUT=*
//OUTPUT   DD  SYSOUT=*
//INPUT DD *
your.ftp.host.address
user password
quote PASV
bin
put 'smfpref.VBS' /smfpref.smf
close
quit
/*
```

## 5.3 Compress the data

When the data is on PC you should compress it (the compression factor is usually very high).
Please be aware that compression tools may have limitations on the size of the file they can compress.

## 5.4 Send the data

You can send data to EPV Technologies in two main ways:

- Uploading the data to the EPV FTP server;
- Creating a CD/DVD and sending it to our local distributor or directly to EPV Technologies via a courier service.

It's always better before to send a small file with only one SMF record type by FTP or e-Mail, so we can confirm everything is correct before spending time sending large amounts of data.

# 6 Customer support

For any technical problem with or question about EPV for DB2 please write an email to:

epv.support@epvtech.com

For any other issue about EPV for DB2 please write an email to:

epv.info@epvtech.com

# Related documentation

The following manuals complement the information provided in this manual:

- *EPV for DB2 V7 Installation and Customization Guide*
- *EPV for DB2 Plus V7 Installation and Customization Guide*
- *EPV for DB2 V7 List of Views*
- *EPV for DB2 V7 Release Notes*
- *EPV for DB2 V7 DataBase Layout*
- *EPV V11 User Interface*