1 DB2 und DFSORT

Sources: Compilation of IBM Informational APAR II14047 and II14213 (august 2009), APAR II14296, APAR PK41899, excerpt of an essay of Mr. Kingsburys, IBM Lab San Jose, several IBM support notes, DB2 9 for z/OS - Using the Utilities Suite, and other technical documents. Covers DB2 V8 and V9.

1.1 Allowing db2 utilities to provide accurate information to DFSORT 2
1.2 Installing DFSORT if it is not your primary sort product 3
1.2.1 Installing the DFSORT SVC 3
1.2.2 Making the DFSORT Libraries Available 3
1.3 Tailoring DFSORT installation defaults to your environment 4
1.4 Verifying ACS routines do not allocate DFSORT work data sets to VIO 5
1.5 Region size (main memory) of the DB2 batch utility job 6
1.6 DFSORT work datasets 6
1.6.1 Limiting the number of parallel DFSORT tasks for db2 utilities 7
1.6.2 Excluding DFSORT work data sets from OEM product's use 7
1.6.3 Details about work dataset you should know 7
1.7 Removing the need to specify SORTNUM on DB2 utilities 7
1.7.1 The V9 enhancement in detail 8
1.8 How DB2 Utilities use DFSORT 10
1.8.1 REORG TABLESPACE Utility 10
1.8.2 LOAD Utility 11
1.8.3 DSNTIAUL Utility 11
1.9 DB2 Sort Feature 12
1.10 Important error messages 12
1.11 DFSORT support 13

The following DB2 Utilities use DFSORT exclusively[1] to perform their SORT and MERGE functions:
- LOAD
- REORG TABLESPACE and REORG INDEX
- REBUILD INDEX
- RUNSTATS
- CHECK DATA, CHECK INDEX, and CHECK LOB

IBM's DB2 and DFSORT Teams developed the interface (DFSORT R14 PTF UQ90054 which was incorporated into the base for z/OS DFSORT V1R5) between the two products in such a way that DB2 Utilities can use DFSORT regardless of whether or not you have a license for DFSORT on your system. For considerations in a 64-bit environment, refer to DFSORT INFO APAR II13495.

The information in this document is intended to assist in configuration and tuning of the DB2 and DFSORT options for improved performance and reliability. Some of the topics discussed include:

1. Allowing DB2 utilities to provide accurate information to DFSORT
    a. Keep RUNSTATS statistics updated
    b. Use SORTNUM to override DFSORT defaults
    c. Provide key count on the SORTKEYS parameter when LOAD input is not a sequential file on disk or is a file with varying length records
    d. Understand how parallel processing can be impacted whenSORTNUM is too high
2. Installing DFSORT if it is not your primary sort product
3. Tailoring DFSORT installation defaults to your environment
    a. DYNALOC - Set default number of work data sets and device type. Consider increasing from default number of 4.
    b. DSA - Set default amount of main storage requested for each sort.
    c. DYNAUTO=IGNWKDD to allow dynamic allocation of work data sets even if JCL work datasets are present
    d. SMF=SHORT or SMF=FULL to allow recording of DFSORT data to aid in analysis
4. Verifying ACS routines will not direct sort work datasets to VIO

[1] also see „DB2 Sort Feature“ on page 12 as an alternative to DFSORT.

   a. Use &MAXSIZE instead of &SIZE to determine size of allocation
   b. Exclude SORTWK*, STATWK*, DATAWK*, DAnnWK*, STnnWK* and SWnnWK* datasets from VIO
5. Exclude SORTWK*, STATWK*, DATAWK*, DAnnWK*, STnnWK* and SWnnWK* datasets from any OEM products that reduce space allocations.
6. Region size of the batch DB2 Utility job.
7. Limiting parallel DFSORT tasks for DB2 Utilities.
8. DFSORT Work datasets
9. Removing the the need to specify SORTNUM on DB2 utilites
10. How DB2 utilities use DFSORT
11. IBM's DB2 Sort feature
12. Important error messages
13. IBM's DFSORT support

1.1 Allowing db2 utilities to provide accurate information to DFSORT

Each DB2 Utility determines how to invoke DFSORT based on various factors and passes the needed information to DFSORT using a special interface. Multiple parallel invocations of DFSORT will be used when appropriate.

Recommendation: Keep the RUNSTATS statistics updated (specifically, CARDF, KEYCOUNTF, AVGROWLEN). These statistics are used to project the size of the sort for REORG and REBUILD INDEX. RUNSTATS statistics are not used for projecting the size of the sort for the LOAD utility. If the input data for the LOAD utility is a sequential file on disk, then the size of this file will be used for the sort calculations. If the input is not a sequential file on disk (i.e., it is on tape, a member of a PDS(E), or it is a cursor) then it is vital that a value be provided on the SORTKEYS parameter. See "Building indexes in parallel for LOAD" in the "Utility Guide and Reference" manual for calculating this value.

The most important parameters DB2 Utilities passes to DFSORT are related to storage usage and the number of work data sets. If appropriate, you can use the DB2 SORTNUM parameter to change the number of work data sets for each sort (e.g., SORTNUM=8 specifies 8 work data sets per sort). But you cannot use DFSORT's DFSPARM data set to change the MAINSIZE value (storage) or DYNALLOC n value (number of work data sets) used by DB2 Utilities at run-time because DB2 Utilities will not be aware of the changes in these values.

You need at least two sort work data sets for each sort. The SORTNUM value applies to each sort invocation in the utility. For example, if there are three indexes, SORTKEYS is specified, there are no constraints limiting parallelism, and SORTNUM is specified as 8, then a total of 24 sort work data sets will be allocated for the job.

Each sort work data set consumes both above the line and below the line virtual storage, so if you specify too high a value for SORTNUM, the utility may decrease the degree of parallelism due to virtual storage constraints, and possibly decreasing the degree down to one, meaning no parallelism. To determine the size of the sort datasets, search for "Calculating the size of the sort work data sets" under each utility that performs sorts in the "DB2 Utility Guide and Reference."

You can override other DFSORT parameters, such as the device type of the work data sets or the ddname of DFSORT's message data set, using the DFSPARM data set, e.g.,

```
//DFSPARM DD *
 OPTION DYNALLOC=WORK,MSGDDN=SRTMSGS
```

Note that you can also use the DB2 SORTDEVT parameter to change the device type of the work data sets (e.g. SORTDEVT=WORK specifies WORK as the device type).

Note: DFSORT APAR PK01155 (R14 DFSORT PTF UK03170 or z/OS DFSORT V1R5 PTF UK03171) uses the MAINSIZE and DYNALLOC n values passed by DB2 Utilities even if the user attempts to override them from other sources (e.g., DFSPARM).

SICELINK and SORTLPA libraries must be APF-authorized. Use the PROGxx parmlib member to authorize these libraries. For more information, see "z/OS Initialization and Tuning Reference". See the following for some information on tuning DFSORT for DB2 Utilities:

http://www-03.ibm.com/servers/storage/support/software/sort/mvs/tuning/pdf/srtmltun.pdf.

1.2 Installing DFSORT if it is not your primary sort product

If DFSORT is installed as your primary z/OS sort product, no additional actions are required to make DFSORT available to DB2 Utilities and the rest of this document does not apply to you. If DFSORT is not installed as your primary z/OS sort product, the rest of this document applies to you.

If you are not licensed to use DFSORT, do not define DFSORT as a licensed feature in SYS1.PARMLIB member IFAPRDxx. DB2 Utilities can use DFSORT regardless of whether or not you have a license for DFSORT on your system.

Note that DB2 Utilities uses only the SORT and MERGE functions of DFSORT. If you want to use DFSORT for any other uses outside of this limited DB2 support, you must separately order and license DFSORT. Use of DFSORT by DB2 Utilities does not interfere with the use of another sort product for other purposes.

You do not need to change any aliases for DFSORT or for your primary sort product. DB2 Utilities calls DFSORT using the new DFSORT-only aliases of ICEDFSRT and ICEDFSRB instead of ICEMAN and ICEBLDX.

DFSORT'S SVC is used to write SMF records: The DFSORT SVC is called to write SMF type-16 records. Correct DFSORT SVC for this release must be loaded in LPA or MLPA to write either SHORT or FULL SMF type-16 records. The DFSORT SVC is not required for the DFSORT installation default of SMF=NO. SORT installation default of SMF=NO.
If you receive an ABENDS16D-08 please refer to the Technote "Setting up the DFSORT SVC" (S1002196).

Cache Fast Write (CFW): If you would like to use Cache Fast Write, the SVC is required.

1.2.1 Installing the DFSORT SVC

If you require DFSORT's SVC, it will need to be installed. Also, if necessary, it is possible for the DFSORT SVC and the Syncsort SVC to coexist.

The DFSORT installation default is to use SVC=(109). If you choose to use the installation default, you need to load (add SICELPA) the SVC into LPA or MLPA.

Alternately, the installation default for the SVC can be changed to use:
- an alternate SVC 109 (IGX00038), or
- to use an user SVC (e.g., specified as SVC=(109,ALT) or SVC=239).

If you choose to use the alternate SVC or a user SVC, you will need to do the following and load (add SICELPA) the SVC into LPA or MLPA.

a. Create a copy of SVC 109 using the sample jobs ICESVREC and ICESVAPP. Ensure the FMID is correct (HSM1G00 for DFSORT R14, or HSM1H00 for DFSORT V1R5). Change ICESVC to the new module name (for example, IGX00038 for the alternate SVC or IGC0023I for SVC 239). Use the same SMP target zone and options you use for installing z/OS products.
b. Make the new copy of the SVC available on your system by placing the updated SICELPA library where you have the SVCs on your system.
c. Change the DFSORT SVC installation default value to use your selected SVC number. Instructions for modifying your DFSORT installation default values are given below, under Customization. For example, use SVC=(109,ALT) for the alternate SVC or SVC=239 for SVC 239.

You can find more information on installing SVCs in "z/OS MVS Initialization and Tuning Guide".

1.2.2 Making the DFSORT Libraries Available

Even though DFSORT is not licensed, the DFSORT libraries are shipped with z/OS. When you install z/OS, you also install the DFSORT libraries. DFSORT should go into the same target and distribution zones as z/OS. However, to make DFSORT available for DB2's use, you must make these libraries part of the system's search order for programs and available to all LPARs where DB2 calls DFSORT.

Where you choose to place DFSORT in the system's search order really depends on how heavily you use DB2 Utilities and your interest in optimizing central storage usage and access time. If DB2 Utilities are heavily used, DFSORT will be called many times on a system, so placing DFSORT SORTLPA into LPA and DFSORT SICELINK into the LINKLIST is recommended for maximum efficiency.
.
To enable DB2 Utilities to use DFSORT, and avoid causing problems with the operation of your primary sort product, you must ensure DFSORT will be found in the system search order after your primary sort product. It is important to determine where your primary sort product modules (SORT, ICEMAN and ICEBLDX) are located in your search order. If these primary sort product modules are in your link pack area, then DFSORT's SORTLPA can also be added to the link pack area list after your primary sort product library(s).
If your primary sort product modules SORT, ICEMAN and ICEBLDX are in the link list, then both DFSORT's SICELINK and SORTLPA libraries (in that order) must be added to the link list after your primary sort product library(s) or found by way of STEPLIB/JOBLIB.

To accomplish this, first determine how your primary sort product is made available (specifically modules SORT, ICEMAN and ICEBLDX), for example, in the link pack area, in the link list, or in JOBLIB/STEPLIB. Then select an appropriate method, as described in the bullets below, that puts DFSORT in the system search order after your primary sort product.

For example, if your primary sort product libraries are in the link pack area and in the link list, you could place your DFSORT SORTLPA library in the link pack area after your primary sort product library in the link pack area, and place your DFSORT SICELINK library in the link list after your primary sort product library in the link list. Alternatively, you could place your DFSORT SICELINK library and your DFSORT SORTLPA library (in that order) in the link list after your primary sort product library in the link list. As appropriate, do one or more of the following:

- Add the DFSORT SORTLPA library to the link pack area list and add the DFSORT SICELINK library to the link list
- Add the DFSORT SICELINK library and then the SORTLPA library, in that order, to the link list (the order is important).
- Add the DFSORT SICELINK library and then the SORTLPA library, in that order (the order is important), to the JOBLIB DD statements you use for your DB2 Utility jobs.
- Add the DFSORT SICELINK library and then the SORTLPA library, in that order (the order is important), to the STEPLIB DD statements you use for your DB2 Utility jobs.
- Add the DFSORT modules to a private library that is equivalent to one of the above configurations. In this case, ensure for the above JOBLIB and STEPLIB configurations, that all of the DFSORT SICELINK modules are before the DFSORT SORTLPA modules in the system search order.

Important: If your primary sort product is run from a private library, you must use a JOBLIB or STEPLIB for your DB2 Utility jobs.

1.3 Tailoring DFSORT installation defaults to your environment

DFSORT supports various options for four invocation environments (JCL, INV, TSO and TSOINV) and four time-of-day environments (TD1-4). With DFSORT as your primary sort product, you can customize the DFSORT options for these environments as appropriate. But when you have another primary sort product and are just using DFSORT for DB2 Utilities, the only environment you can change is the INV environment. DFSORT is shipped with default options for this environment that generally provide efficient operation. However, you may want to change some of the INV options to better match your own environment for running DB2 Utilities. In particular:

- you may want to increase the DSA value above the shipped default of 64 (megabytes) for z/OS DFSORT V1R5 or 32 (megabytes) for DFSORT R14, e.g., DSA=128
- you may want to increase the number of dynamically allocated work data sets above the shipped default of 4, e.g., DYNALOC=(SYSDA,8). A "best practice" here is to set this number to a value that satisfies 80-90% of your sorts, and then use the SORTNUM parameter on those utilities that require a different value.

- you may want to change the shipped default of DYNAUTO=YES to DYNAUTO=IGNWKDD to allow DFSORT to automatically use dynamically allocated work data sets instead of only the JCL work data sets.
- you may want to change the shipped default of SMF=NO to SMF=SHORT or SMF=FULL to record DFSORT SMF type 16 records

See "z/OS DFSORT Installation and Customization" and "z/OS DFSORT Tuning Guide" for more information. You can access these and the other DFSORT books on-line using the "Publications" link on the DFSORT website at: http://www.ibm.com/storage/dfsort

To modify your DFSORT installation options, write a usermod to modify ICEMAC. The following instructions illustrate the process for changing the DSA value to 128 (megabytes) for use by DB2 Utilities.

The sample library SICESAMP contains the following members:
- ICEOPREC does the RECEIVE
- ICEOPAPP does the APPLY

Update the ICEMAC INV statement in the ICEAM2 module within the ICEOPREC sample job

```
++ SRCUPD (ICEAM2) DISTMOD (AICELIB) DISTLIB (AICESRCE)
/* The ICEAM2 member in the AICESRCE library contains the
default lines for creating the ICEAM2 module. A model ICEMAC
INV statement for customizing the ICEAM2 module is shown
below. Replace it with an ICEMAC INV statement that
specifies the parameters you want to use for ICEAM2.
You MUST use the correct sequence numbers in columns 73-80
to replace the appropriate lines in the ICEAM2 member.
*/ .
./ CHANGE NAME=ICEAM2 73
ICEMAC INV,DSA=128                                          00550000
```

You can remove the ICEAM1, ICEAM3, ICEAM4 updates in the ICEOPREC sample job. Ensure the FMID is correct (HSM1G00 for DFSORT R14, or HSM1H00 for DFSORT V1R5). Also ensure you use the same SMP target zone and options you use for installing z/OS products. Then do the RECEIVE using the modified ICEOPREC sample job and the APPLY using the ICEOPAPP sample job. When you install a new release of z/OS, this usermod has to be RECEIVEd and APPLYed to the new release.

1.4 Verifying ACS routines do not allocate DFSORT work data sets to VIO

The use of Virtual I/O (VIO) for DFSORT work data sets is NOT recommended. VIO=NO is DFSORT's shipped installation default and recommended setting for the VIO parameter. If your site's ACS routines allocate DFSORT work data sets to VIO, it is recommended that you change your ACS routines to avoid using VIO for SORTWK*, STATWK*, DATAWK*, DAnnWK*, STnnWK* and SWnnWK* data definitions. This is because DFSORT already performs storage hierarchy management (the efficient balancing of processor memory and disk storage), and mixing this with VIO which uses processor memory results in sub-optimal efficiency.

DFSORT exploits processor storage through the use of either Hiperspaces, Dataspaces or Memory Objects. Memory Object sorting will only be used if MEMLIMIT is non-zero. If MEMLIMIT is not set in your JCL then it defaults to the MEMLIMIT value specified in the SMFPRMxx member of SYS1.PARMLIB. If MEMLIMIT is not specified in SMFPRMxx then that default is zero. More information on the relationship between MEMLIMIT, REGION, IEFUSI and SMFPRMxx can be found in the z/OS MVS Programming: Extended Addressability Guide.

DFSORT uses a primary allocation of 0 tracks for the initial allocation of work data sets. This can impact the decisions made by DFSMS Automatic Class Selection (ACS) routines if the &size variable is being used to assign a unit (such as VIO) or select a storage class and group based on data set size. It is recommended that the &maxsize variable be used as a basis for decisions related to data set size. Since DFSORT work data set allocations contain a secondary space quantity, the &maxsize variable will allow for a more accurate decision on the assignment of unit, storage class, storage

group, etc.

While DFSORT performs the initial work data set allocations with a primary space quantity of 0, it will attempt to extend them once the accurate amount of required space is determined. Should a work data set reside on a volume which does not have sufficient space available, DFSORT invokes recovery routines that may result in the data set being deleted and re-allocated.

1.5 Region size (main memory) of the DB2 batch utility job

DFSORT may abend or process inefficiently for large sorts when there is not enough main memory available, which can occur especially when the Utility runs many sorts in parallel. For large sorts when main memory is constrained, DFSORT will compensate with intermediate merges that increases the need for sort work data sets. One method to avoid this is to ensure that there is sufficient main memory available for large sorts. The following information gives a rough estimate on the amount of memory that DFSORT would need to run efficiently:

Data size Estimated main memory use per DFSORT task

1 GB	10 MB
10 GB	30 MB
100 GB	70 MB

where data size is the number of bytes to be sorted, e.g., record count * record length.

Increasing the region size of the batch Utility job to accommodate the main memory needed for the parallel DFSORT tasks can allow the sorts to succeed.

1.6 DFSORT work datasets

Temporary datasets

- SORTWKnn: Temporary datasets for sort input and output when sorting keys if Parallel Index Build (PIB) is not used. nn identifies one or more data sets that are to be used by the sort task for LOAD, REBUILD, and REORG. The SORTWK01 data set is also used when collecting distribution statistics by RUNSTATS. The used message data set is UTPRINT.
- SWnnWKmm: Temporary datasets for sort input and output when sorting keys if Parallel Index Build (PIB) is used. nn identifies the subtask pair (one per index), and mm identifies one or more data sets that are to be used by that subtask pair when executing LOAD, REBUILD, or REORG. The used message data set is UTPRINnn.
- DATAWKnn: Temporary data sets for sort input and output used by REORG for the data sort. nn identifies one or more data sets that are to be used by the sort task. The used message data set is UTPRINT.
- DannWKmm: Temporary datasets used by REORG for unload parallelism. nn identifies the subtask pair (one per partition), and mm identifies one or more data sets that are to be used by that subtask pair. The used message data sets are DTPRINnn.
- STATWK01: Temporary datasets for sort input and output when collecting distribution statistics for columns groups (RUNSTATS and inline statistics). The used message data set is RNPRIN01.
- ST01WKnn: Temporary datasets for sort input and output when collecting frequency statistics on DPSIs or when TABLESPACE TABLE COLGROUP FREQVAL is specified (RUNSTATS and inline statistics). The used message data set is STPRIN01.
- ST02WKnn: Temporary datasets for sort input and output when collecting frequency statistics for column groups (RUNSTATS and inline statistics).

Informational output datasets

- UTPRINT: A dataset that contains messages from DFSORT. Required for the LOAD, REBUILD, and REORG utilities.
- UTPRINnn: Data sets that contain messages from DFSORT from utility subtasks pairs. These data sets are dynamically allocated when UTPRINT is allocated to SYSOUT.
- DTPRINnn: Datasets that contain messages from UNLOAD parallelism functions. These data sets are dynamically allocated when UTPRINT is allocated to SYSOUT.
- STPRIN01: A dataset that contains messages from DFSORT. This data set is required when frequency statistics are collected on DPSIs or when TABLESPACE TABLE COLGROUP FREQVAL is specified.

- RNPRIN01: A dataset that contains messages from DFSORT. This data set is required when distribution statistics are collected for column groups.

Other datasets:

- SORTDEVT: Gives DB2 the device type of where to allocate the sortwork data sets. The specification of this keyword allows DB2 to perform dynamic data set allocation, unless the user overrides it with their own DFSORT DYNALLOC parameter. For REORG unload/reload partition parallelism, DB2 requires the specification of SORTDEVT so that the optimal degree of parallelism can be determined.
  Note: Try to always specify the SORTDEVT keyword in the utility control statements. This allows for an optimal degree of parallelism.

1.6.1 Limiting the number of parallel DFSORT tasks for db2 utilities

The number of parallel DFSORT tasks for Utilities may be limited by allocating the UTPRINxx data sets to something other than SYSOUT *. E.g., if UTPRIN01, UTPRIN02, and UTPRIN03 are allocated to real data sets then at most 3 parallel sort tasks would be initiated by the Utility.

1.6.2 Excluding DFSORT work data sets from OEM product's use

Exclude SORTWK*, STATWK*, DATAWK*, DAnnWK*, STnnWK*, and SWnnWK* datasets from any OEM products that reduce space allocations.

While running concurrent utilities, if the percent free of your work pool is low, you can either reduce the number of concurrently running utilities or you can increase the number of volumes in your work pool. If you increase the number of volumes, you can then monitor utilization and bring that number back down if it's consistently underutilized. As a general rule, your work pool should average over 20% free to allow for optimal allocation of DFSORT work data sets and reduce the risk of failures due to lack of work space (ICE083A, ICE046A, etc.)

1.6.3 Details about work dataset you should know

Sort work data sets cannot span volumes. Smaller volumes require more sort work data sets to sort the same amount of data; therefore, large volume sizes can reduce the number of needed sort work data sets. Since z/OS V1.7, it is possible to define volumes with more than 64,000 tracks. IBM recommends that at least 1.2 times the amount of data to be sorted be provided in sort work data sets on disk. Using two or three large SORTWKnn data sets are preferable to several small ones.

DFSORT runs most efficiently with a small number of data sets. A small number of data sets also benefits the maximum degree of parallelism for utilities, so you can likely improve the elapsed time, if you have large sort volumes to reduce the number of data sets for very large sorts.

When assigning a data class, it is important that you turn off the ability to allocate sort work data sets on multiple volumes (either directly defining them preventively as multi-volume or letting space constraint relief expand a data set to multiple volumes). DFSORT can only use the part of the sort work data set that resides on the first volume; all other parts on an additional volume will go unused by DFSOR, which wastes disk space and likely causes SORT CAPACITY EXCEEDED failures.

Ensure that sort work allocations are directed to those devices with the fastest available random I/O service times. Spread I/O for sort work files by assigning logical volumes on different physical devices to avoid cache and channel contention. In general, sort work EXCPs are large, which could have a noticeable impact on measured disconnect times for other applications, hence isolate sort work devices particularly from data with critical response time requirements.

SORTNUM tells DB2 how many sort work data sets to use for each sort invocation. This is only passed on to DFSORT in the DYNALLOC option statement and tells DFSORT into how many data sets the sort work space should be divided. DB2 allocated sort work data sets do not use this number; they are always allocated as large as possible to result in the smallest possible number of data sets.

1.7 Removing the need to specify SORTNUM on DB2 utilities

Factors that influence the success of utility sort processing are the ability to allocate sufficient sort work space and the correct estimation of the number of records that need to be sorted.

These issues are addressed by APAR PK45916 in DB2 V8 for z/OS, and APAR PK41899 in DB2 9 for z/OS. They add new functionality to pre-allocate sort work space before invoking DFSORT and to enhance the record estimation with the use of Real-Time statistics (RTS). IBM's recommendation to our customers for best practices regarding allocation of sort work space is to use the enhancements made by these APARs. For utility jobs experiencing SORT capacity errors, ABEND04E with RC00E40005 with MSGICE046A SORT CAPACITY EXCEEDED or ABENDB37 when DFSORT tries to extend the allocated data sets, customers should enable the enhancements made by these APARs to prevent the SORT capacity errors.

Allocation of sort work space within DB2 utility leaves it with full control over the used resources - utilities like REORG TABLESPACE, CHECK INDEX, and REBUILD INDEX need to take the used resources into account for the calculation of the possible degree of parallelism. For this DB2 needs to know the actual number of sort work data sets before the subtasks that do the work in parallel are attached.

This new behavior is activated by setting system parameter UTSORTAL=YES. Sort work data sets will then only be allocated by the utility when no SORTNUM option is specified on the utility statement AND if no sort work data sets were specified in the JCL. In addition system parameter IGNSORTN can be set to YES to override specified SORTNUM values on utility statements and to ensure DB2 controlled sort work space allocation.

For every invocation of DFSORT the utility will pass an estimated number of records that need to be sorted. For some utilities this number will be known exactly from previous phases, but most utilities have to estimate that number. Prior to this enhancement, that estimate was calculated using table space allocation information like high used RBA and RUNSTATS values like AVGROWLEN and NPAGES. However this technique depends on accurate statistics by periodically running the RUNSTATS utility.

By setting UTSORTAL=YES the estimation will now first look up the processed objects in Real-Time statistics tables which maintain current counts of rows or keys for every table space and every index. Only if the object can not be found in RTS or the count is set to NULL will the previous estimation logic based on RUNSTATS output be used. Thus the DB2 utilities enhancement enabled by UTSORTAL set to YES has a greater dependency on the accuracy of Real Time Statistics.

Note: It is expected that future versions of DB2 will retrieve and rely upon Real-Time statistics independent from the UTSORTAL parameter setting and that further utilities will exploit RTS information.

In order to make sure that information in Real-Time statistics is accurate and the utility jobs can rely on that information it is important that RTS values are initialized correctly and are then maintained without interruption. RTS columns that are used by DB2 utilities will be initialized by running REORG TABLESPACE, LOAD REPLACE, or REBUILD INDEX. RUNSTATS does NOT initialize the RTS values that are required by DB2 Utilities.

When DB2 objects are replaced by any method outside of DB2's control (e.g. by using DSN1COPY or DFSMS COPY), the information in the RTS tables may no longer matche the actual table space or index space. To indicate that those values should no longer be used the TOTALROWS column in SYSIBM.SYSTABLESPACESTATS or TOTALENTRIES column in SYSIBM.SYSINDEXSPACESTATS for DB2 9 should be set to NULL or the exact value if known. For DB2 V8 the same columns in SYSIBM.TABLESPACESTATS and SYSIBM.INDEXSPACESTATS would have to be set.

1.7.1 The V9 enhancement in detail

DB2 Utilities have been enhanced as follows:

(1) DB2 utilities CHECK DATA, CHECK INDEX, CHECK LOB, LOAD, REBUILD INDEX, REORG TABLESPACE, RUNSTATS have been enhanced to allocate sort work data sets dynamically before invoking DFSORT if the new ZPARM UTSORTAL is set to YES and no SORTNUM value is specified in the utility statement.

Setting new ZPARM IGNSORTN to YES will cause utilities to dynamically allocate sort work data sets even if SORTNUM parameter was specified in the utility statement. The specified SORTNUM

value will then be ignored.

The utility will never allocate sort work data sets dynamically if DFSORT related DD cards (SORTWKnn, SW01WKnn, DATAWKnn, DA01WKnn, STATWKnn, ST01WKnn) were already specified in the JCL.

When the utility allocates sort work data sets dynamically it will calculate the disk space requirements according to available record estimates and record length. Utilities will be trying to allocate the required disk space in two data sets for each sort task but will reduce allocation size if the requested amount is not available. Allocation of additional data sets continues until the required amount was fully allocated. Data sets will be allocated using DSNTYPE=LARGE if supported by the system.

(2) To estimate the number of records to be sorted during utility execution the processed object will be looked up in real-time statistics tables SYSIBM.SYSTABLESPACESTATS and SYSIBM.SYSINDEXSPACESTATS. If information is not available for the object (i.e. TOTALROWS or TOTALENTRIES is set to NULL), the current estimation logic based on RUNSTATS catalog statistics will be used.

It is recommended to set base values for each object that has counters containing NULL values in the real-time statistics tables by running LOAD REPLACE, REORG, or REBUILD INDEX. REORG and LOAD REPLACE will set base values both for table spaces and their indexes.
REBUILD INDEX only sets base values for the specified indexes.

When table spaces or indexes have been modified outside of DB2 (e.g. using DSN1COPY), the real-time statistics maintained by DB2 will no longer be in synch with the current object status. To avoid misleading estimates for such objects, the related information should be invalidated to keep DB2 utilities from using it. For table spaces set the column TOTALROWS in SYSIBM.SYSTABLESPACESTATS to NULL, for indexes set the column TOTALENTRIES in SYSIBM.SYSINDEXSPACESTATS to NULL. DB2 utilities will then use the current estimates based on RUNSTATS information.
For best results, real-time statistics should soon be re-initialized for such objects by running LOAD REPLACE, REORG, or REBUILD INDEX.

(3) The CHECK INDEX, REBUILD INDEX, LOAD and REORG TABLESPACE utilities take the number of allocated data sets into account for the calculation of the maximum number of parallel tasks. If the required disk space could be allocated in only a few large data sets this will increase the number of concurrent tasks and help in reducing elapsed time.

The parallel processing for these utilities will now assign indexes to sort tasks in a way to reduce the disk space overhead caused by padding smaller index keys to the maximum key length used in each sort task. This will reduce overall sort work disk space requirements as long as multiple sort tasks are possible. There will be no improvement in disk space usage for processes with a single sort task.

(4) CHECK INDEX was improved to use the same revised storage requirement values per data set and per task as the REBUILD INDEX utility when calculating the number of parallel tasks based on the available storage. See PK47212 for details on the REBUILD INDEX change.

(5) OPTIONS FILSZ can be used to specify the required sort work data set size for all sort tasks within a utility step. The size is specified in units of 1 MB and will be used for all sort tasks in the utility that follows the OPTIONS statement. Use this parameter as a bypass if the automatically calculated sort work data set size still causes SORT CAPACITY EXCEEDED failures.

To override the sort work data set size to 10 MB for CHECK INDEX you would use the following invocation:

```
OPTIONS FILESZ(10)
CHECK INDEX(myixname) SORTDEVT mydevt
```

1.8 How DB2 Utilities use DFSORT

DB2 utilities have progressed greatly since the days when an entire suite of vendor replacement utilities was a "must-have" for any substantial DB2 implementation. Intelligent interfacing of the utilities with DFSORT has been a key component of this improvement.

Similar general recommendations apply for DFSORT with large utility operation as with any other batch sort task; however, some special considerations warrant mentioning.

1.8.1 REORG TABLESPACE Utility

Three parameters for this utility relate to DFSORT performance.

SORTDATA

When specified, this parameter directs the utility to perform unload via a tablespace or partition scan and batch sort to clustering index sequence – as opposed to unloading using the clustering index, the equivalent of a SQL **SELECT * ... ORDER BY ...** statement.

Except where the underlying data clustering is very near to perfect, SORTDATA will provide a substantial - often radical - improvement in elapsed time. DB2 makes no decision to use SORTDATA dynamically, even if the CLUSTERRATIO column of the index in the DB2 catalog (SYSIBM.SYSINDEXES or SYSIBM.SYSINDEXSTATS) contains a low value.

Note, however, that if a clustering index is not **explicitly** defined on the table – as opposed to implicitly, by having the lowest OBID - DB2 will ignore the parameter if specified, and **always** unload via the **implicit** clustering index.

NOSYSREC

Ordinarily, REORG SORTDATA will unload the tablespace or partition to SYSREC, use DFSORT to perform a batch sort of this dataset, and reload the data by reading the sorted SYSREC.

NOSYSREC eliminates the I/O to SYSREC, by passing data directly to DFSORT in the UNLOAD phase of the utility, and retrieving data directly from DFSORT in the RELOAD phase.

One word of caution. If this parameter is used in a traditional (SHRLEVEL NONE) REORG execution, a SYSREC dataset is no longer available to re-start the utility should it fail during the RELOAD phase. The tablespace or partition must be RECOVERed - hence, a QUIESCE to establish a recovery point just prior to the REORG is mandatory. **SHRLEVEL NONE is the default.**

SORTKEYS

This parameter is specified without a key count estimate for REORG - this can be accurately determined by the utility itself. It directs REORG to pass **non-clustering index** key values directly from the RELOAD phase to a DFSORT subtask separate from that used for SORTDATA, rather than the traditional method of writing the key values to SYSUT1 for batch sorting into SORTOUT.

The index BUILD phase then retrieves the sorted key values directly from DFSORT, eliminating all I/O to SYSUT1 and SORTOUT. SYSUT1 and SORTOUT must still be present in the utility JCL, but may have minimal space allocation.

From Version 6 of DB2 on, this index building is accomplished by multiple concurrent sort tasks.

Note that this parameter is only of benefit for indexes **additional to** the partitioning or (explicit or implicit) clustering index - the UNLOAD phase has already ordered the key values for these indexes, and an index SORT phase is not executed.

Recommendations

- Always define the clustering index **explicitly** for any table of substantial size. Note, however, that this is always implied for the partitioning index of a partitioned tablespace.

- Where appropriate parameters are not specified to eliminate I/O altogether, or are inappropriate, specify DFSMS striping for SYSREC, SYSUT1 and SORTOUT. For REORG INDEX, always specify striping for SYSUT1.
- When the tablespace or partition contains variable-length columns, use the DFSPARM data set to specify an accurate tablespace or partition row count via the OPTION parameter FILSZ=Un. (DB2 assumes in its estimate to DFSORT that all variable-length columns are at maximum length.)
- Execute both LOAD and REORG at the partition level to reduce the granularity of the DFSORT tasks. Concurrent partition-level utility tasks can, however, present contention problems when NPIs are defined on the tablespace.

1.8.2 LOAD Utility

The LOAD utility has a SORTKEYS parameter similar to that for REORG. Again, the parameter requests LOAD to pass index key and foreign key values directly from the RELOAD phase to a DFSORT subtask - rather than using SYSUT1/SORTOUT - and then directly to the BUILD and ENFORCE phases.

The following points are worth noting:

- If **SORTKEYS** is specified with no key count estimate, or with an estimate of zero, sorting takes place via SYSUT1 and SORTOUT as usual.
- If the tablespace or partition data to be loaded is presented to LOAD in pre-sorted key sequence, and only a single index exists on the table, the SORT phase is eliminated altogether.

Recommendations

When LOADing large partitioned tables with only a partitioning index:

- Pre-sort the input file using DFSORT
- Use DFSORT's multiple output function (OUTFIL) to split the DFSORT output by partitioning key
- Execute multiple concurrent partition-level LOAD utility tasks

For large segmented tablespaces with only a single index, the input file should be similarly pre-sorted.

- Specify SORTKEYS nnnnnnnn, calculating nnnnnnnn using the formulae stipulated in the DB2 Utilities Guide and Reference
- If sorting using WORKDDN files cannot be avoided, specify DFSMS striping for SYSREC, SYSUT1 and SORTOUT.

1.8.3 DSNTIAUL Utility

DB2 (up to version 5) does not provide a standard "unload" utility; facilities are limited to sample program DSNTIAUL which simply executes dynamic SQL to produce a default-format (or user-specified) unload dataset.

Note that with version 6, the REORG utility provides this facility via the FORMAT EXTERNAL option, but without the full power of SQL.

As a general rule, executing a table unload via a SQL **SELECT ... ORDER BY** has always been preferable to unloading via a tablespace scan and post-sorting the unload dataset. Two exceptions are notable, however:

- For very large tablespaces, DSNDB07 may well be overwhelmed by a very large ORDER BY or GROUP BY key set. At the minimum, the sorting process may not scale well.
- Access paths may be a concern. A non-matching index scan may be selected, instead of a tablespace scan, or the degree of parallelism selected by DB2 may be less than that desired or possible.

One possible improvement is to:

- Unload the tablespace at partition level using DSNTIAUL, selecting and ordering or aggregating data by partition. A partition scan access path should be engineered by specifying partitioning key column values explicitly as literals.
- Use DFSORT MERGE to produce the final ordered dataset from the partition-level unloads, or use SUM to produce a finally aggregated file.

Note that parallel unload of data need not be restricted to partitioned tablespaces; the unload of any substantial segmented tablespace may be divided in this manner.

1.9 DB2 Sort Feature

IBM DB2 Sort for z/OS is a new DB2 tool that provides high-speed utility sort processing for data stored in DB2 for z/OS. DB2 Sort for z/OS can help improve sorting performance while optimizing overall system efficiency through reduced elapsed time and CPU consumption, and improve memory management by exploiting the advanced facilities of the z/OS operating system and System z server.

DB2 Sort for z/OS:

- Helps reduce elapsed time and CPU consumption by IBM utilities sort
- Allows flexibility to optimize utility sort processing
- Improves application availability by reducing the batch window for utility maintenance

DB2 Sort for z/OS includes a 64-bit API that facilitates information sharing between the sorting software and IBM products that invoke it. The API implements a handshaking algorithm that validates authorized callers and provides improved memory and sort workspace management. IBM DB2 Utilities Suite for z/OS V9.1 and V8.1 are supported by the API.

This product is not designed to be a replacement for general purpose sorting products that are invoked through the standard z/OS system sort interface.

1.10 Important error messages

- **ICE083A** RESOURCES WERE UNAVAILABLE FOR DYNAMIC ALLOCATION OF WORK DATA SETS (xxxx)
  - If xxxx is 064K:
    The work space needed for each work data set exceeds the limit of 64K tracks for a single, physical data set. Dynamic allocation could not be attempted.
  - If xxxx is not 064K:
    DFSORT attempted to dynamically allocate work data sets using the DYNALOC, DYNSPC, and DYNALLOC values in effect. Dynamic allocation failed; xxxx indicates the failure code returned by the system's dynamic allocation facility. See z/OS MVS Programming: Authorized Assembler Services Guide for an explanation of the failure code.

  Two commonly received return codes are:
  (1) 0218 The device type is defined to the system, but the requested device has insufficient space available. For example, the device type is defined to the system but no devices of that type have been installed, or all devices of that type are mounted as private.
  (2) 021C The device type is not defined to the system. For example, SYSDX was specified instead of SYSDA.

  However, in a DFSMS environment, the return code is usually 970C regardless of the specific error encountered.

  System Action: The program terminates.

  Programmer Response:
  - If xxxx is 064K:
    Use the DYNALLOC=(,n) parameter to increase the maximum number of work data sets (n) such that the work space needed for each work data set does not exceed 64K tracks.
  - If xxxx is not 064K:
    Ensure that work data sets can be dynamically allocated.

If message ICE118I was received, specify FILSZ=En with a reasonably accurate estimate of the number of records to be sorted. If you cannot specify FILSZ=En, use DYNSPC=n to decrease the primary space allocated.

- **ICE046A** SORT CAPACITY EXCEEDED - RECORD COUNT: n

  Explanation: Critical. DFSORT was not able to complete processing with the intermediate storage available (Hiperspace or disk work data sets).
  For work data sets with secondary allocation allowed, DFSORT overrides system B37 abends and continues processing; this message is issued only when no more space is available in Hiperspace or on any allocated work data set.

  Note: DFSORT uses only the first volume of multi-volume work data sets.

  The count n is either an approximation of the number of records or is the total number of records that DFSORT was able to read in before it used all of the available intermediate storage.

  The amount of intermediate storage required can vary depending on many factors including:
  - The amount of Hiperspace DFSORT is able to use at the time the sort is run
  - The amount of main storage available
  - The degree of randomness of the records to be sorted
  - The values specified (or defaulted) for options such as DYNALOC, DYNAUTO, DYNSPC, FILSZ/SIZE, AVGRLEN, or DYNALLOC
  - The amount of padding required for short records when VLSHRT is in effect.

  System Action: The program terminates.

  Programmer Response: Take one or more of the following actions:
  - If dynamic allocation was used, refer to message ICE254I (if issued) for more information.
  - If JCL work data sets were used, increase the amount of work space available to DFSORT.
  - If message ICE118I was received, refer to message ICE253I for more information
  - If you are sorting variable length records, refer to message ICE098I for more information.
  - If appropriate, increase the amount of main storage available to DFSORT using the options MAINSIZE/SIZE or the JCL option REGION.
    Increasing the amount of main storage available to DFSORT can help DFSORT use less intermediate storage. Avoid running a large sort in a small amount of main storage as this can degrade performance and increase intermediate storage requirements. In some cases, a small amount of main storage can force DFSORT to perform an intermediate merge as indicated by message ICE247I; refer to ICE247I for more information on the consequences of an intermediate merge and how to avoid it.
  - If VLSHRT was in effect and the total size of all control fields was significantly larger than the average LRECL for the data set, you may be able to reduce the amount of work space required by reducing the total size of the control fields.

1.11 DFSORT support

Even if you do not have a license for DFSORT, IBM will provide central service for DFSORT problems related to the use of DFSORT by DB2 Utilities. IBM already provides all DFSORT service (PTFs) to all z/OS customers. If a DFSORT defect is encountered, DFSORT service and any prerequisite PTFs will be required. You must be aware that you now "care" about DFSORT service and take appropriate actions. For example, IBM recommends that you specify DFSORT in the group of FMIDs that you want services. If you use Enhanced HOLDDATA, DFSORT HIPERs will show up in the REPORT ERRSYSMODS listing. If you've tailored notifications (like ASAP in ServiceLink) at the FMID level, you should add notifications for DFSORT HIPERs.

If you have a any problem with DFSORT (e.g., DFSORT messages ICE039A, or ICE046A, or you receive an abend), you should open a PMR against DB2. If necessary, DB2 support will transfer the problem to DFSORT. If the problem can be reproduced, add SORTDIAG and DFSPARM DD statements to the failing job and rerun it:

```
//SORTDIAG DD DUMMY
//DFSPARM DD *
```

```
DEBUG ABEND
/*
```

This will allow DFSORT to display additional diagnostic (ICE8xxx and ICE9xxx) messages, and ABEND if a DFSORT error message is issued. Send the complete JOBLOG and dump (if an error occurs) to the appropriate IBM service representative.

Recommended DFSORT service:

- APAR PK25047
  ICE046A can result when DFSORT uses multiple Hiperspaces and more than 2G of hiperspace is required to successfully complete the sort.
- APAR PK85856 and PK85889
  DFSORT has been modified to support DB2 V8.1 and V9.1 Utilities additional offload to zIIP processors when available. DFSORT will allow offload to zIIP processors in the memory object sort path for fixed length records when invoked by DB2 Utilities with support for additional zIIP offload. When this additional zIIP offload takes place, it will be indicated by DFSORT with ...

  ```
  MSGICE256I DFSORT CODE IS ELIGIBLE TO USE ZIIP FOR THIS DB2 UTILITY RUN.
  ```

  Exploitation of this enhancement requires DB2 APAR PK85889. If PK85889 is not applied then the enhancement delivered by this APAR is not enabled.
- APAR PM18196
  New Function - Optimization of DFSORT algorithms for selecting the sort technique that makes the best use of available storage.
  DFSORT has been enhanced by optimizing the algorithms for selecting the sort technique that makes the best use of available central storage resources. As a result the user may see more frequent use of Hiperspace sorting and less use of memory object sorting.
  Note: DFSORT will still use Memory object sorting when appropriate.

Additionally, there are a number of APARs available for both DFSORT and DB2 that affect the way they interact. You should ensure that you are current on both DFSORT and DB2 maintenance.