CA Datacom®/DB

AutoScope User Guide for z/OS

Version 14.02

CA Technologies Product References

This document references the following CA Technologies products:

- CA Datacom®/DB
- CA Datacom® Datadictionary™
- CA Datacom® Server
- CA Datacom® SQL
- CA Dataquery™ for CA Datacom® (CA Dataquery)
- CA SYSVIEW® Performance Management CA Datacom® Option (CA SYSVIEW Datacom Option)

Contact CA Technologies

Contact CA Support

For your convenience, CA Technologies provides one site where you can access the information that you need for your Home Office, Small Business, and Enterprise CA Technologies products. At http://ca.com/support, you can access the following resources:

- Online and telephone contact information for technical assistance and customer services
- Information about user communities and forums
- Product and documentation downloads
- CA Support policies and guidelines
- Other helpful resources appropriate for your product

Providing Feedback About Product Documentation

If you have comments or questions about CA Technologies product documentation, you can send a message to techpubs@ca.com.

To provide feedback about CA Technologies product documentation, complete our short customer survey which is available on the CA Support website at http://ca.com/docs.

Contents

Chapter 1: Introduction

The AutoScope functions are a set of tools for the CA Datacom/DB environment that are based on a statistics methodology. They use standard supported CA Datacom APIs to collect information from the CA Datacom Dynamic System Tables and present that information in a usable form.

The purpose of this guide is to provide the user with a set of basic principles and examples for how to use the AutoScope tools.

All of the AutoScope executable functions are performed using the standard CA Datacom/DB DBUTLTY batch utility. For detailed information about the execution and use of each of the DBUTLTY functions, see the *CA Datacom/DB DBUTLTY User Guide*.

You can use the AutoScope tools and information to help you in your work on the following tasks:

- System management
- Database tuning
- Application tuning
- Hardware and software performance verification

The following are the AutoScope functions:

AutoCollect (AUTOCOLL)

Extracts information from selected Dynamic System Tables (DSTs) in snapshots and records these snapshots in Snapshot rows in tables in a permanent database. Since the counters in the DSTs continually increase as resources are used, each Snapshot will represent the total resources consumed since MUF start-up. Users can run reports to determine what Snapshots have been collected, then when ready, process the Snapshots into Deltas. A Delta is calculated by taking two Snapshots from the same MUF instance and subtracting the earlier one from the later one to create the difference (or Delta). Users then analyze the Delta results to determine performance metrics and make tuning decisions. AutoCollect operations are typically focused on a given MUF. In other words, while you can use AutoCollect to collect data on multiple MUFs, the built-in data analysis functions are only used against one MUF at a time.

To use AutoCollect, you must have an active MUF environment with the AutoCollect databases (default 1019 and 1020) defined. For information about how to create these databases in a new or upgraded CA Datacom r12 system, see the *CA Datacom Installation and Maintenance Guide*.

AutoStatus (AUTOSTAT)

Extracts task activity information in snapshots from the Dynamic System Table MUF_ACTIVE_TASKS, from one or more MUFs on a given LPAR and records these snapshots in a permanent database so you can run reports and analyze the results. AutoStatus information is typically used to track task activity on one or more MUFs looking for possible interactions, locking problems, and so on.

To use AutoStatus, you must have an active MUF environment with the AutoStatus database (default 1018) defined. For information about how to create this database in a new or upgraded CA Datacom r12 system, see the *CA Datacom Installation and Maintenance Guide.*

AutoInfo (AUTOINFO)

Extracts a single informational snapshot from Dynamic System Tables and combines it with other available information to provide the user with a simple way to gather multi-user environmental information such as MUF Start-UP options, system statistics, task information, and so forth and produce a report with the information. In addition to the report, an optional sequential file can be created that can be sent (using FTP) to the CA Support staff to help in problem debugging. AutoInfo will provide the user a simple way to document the MUF environment set-up and statistics since the MUF was last enabled. To get the full use AutoInfo, the user must have the Dynamic System Tables activated in the MUF.

Hardware and Software Requirements

The CA Datacom environment must have the Dynamic Systems Table (DST) database activated.

Required Maintenance Level

You must have CA Datacom r12 or higher installed prior to using the AutoScope functions.

Restrictions

The SNAPSHOT options of the AUTOCOLL, AUTOSTAT, and the AUTOINFO function, must be executed on the same LPAR as the source MUF environment. These functions may not be executed cross LPAR using XCF or CCI connectivity.

Limitations

For counter columns in Dynamic System tables where large numeric values have been encountered, we have increased these column values in r12 to s9(11) which will support values up to 99,999,999,999. In other counter columns where we do not expect large numbers, the columns are defined as integers and can handle values to slightly over two billion.

If a very active MUF remains active for very long periods of time, these columns may reach their maximum value. If that occurs, their values either freeze at the high water mark or may reset back to zero. Statistical analysis of these columns after this occurs is not effective. We do not expect this to occur in normal circumstances.

While processing Dynamic System Tables (DSTs) into Snapshot and Delta rows, the large numeric values from DSTs are stored in columns defined as s9(15). This allows the Snapshot and Delta tables to support substantially larger values needed for total rows and Delta summarization functions. However, if you use the AutoCollect functions to summarize a large amount of MUF activity (such as for a year) into a single set of Delta rows, it may exceed the capacity of these numeric columns.

Chapter 2: AutoCollect Process

For business requirements, there is a need to collect systemwide statistics on production MUF environments. These statistics can be invaluable tools in measuring performance, detecting growth situations, debugging problems, and performance tuning.

Having good baseline statistics provides you with the ability to note changes and evaluate effects of new applications, new hardware, new operating systems, and even new versions of CA Datacom.

CA Datacom provides a wealth of information through the Dynamic Systems Tables (DST) interface. This information is real-time information. In many cases, it is the same information that can be found in reports and screens throughout the CA Datacom environment.

The strength of the DSTs lies with the availability of MUF environmental and statistical data available real-time as a set of SQL accessible tables. This data is readily accessible anytime the MUF is enabled. The weakness to DSTs is that the data is not persistent and is lost when a given MUF ends. Also since the data is always accumulating, there is no way to go back in time to see a particular DST data value.

Creating the Snapshot Rowsets

The AutoCollect process provides a specific methodology to take snapshots of selected DST tables and record them in a permanent database for analysis and reporting.

Multiple DST tables are collected in a given snapshot and the number of rows collected from each table vary. The AutoCollect functions assign a unique identifier (key value) to each row of a given snapshot. In this way, all rows from the different tables can easily be found. To make it easier to refer to the collection of rows that make up a snapshot, we have created the term ROWSET. A rowset refers to all the rows (in all tables) that make up a given snapshot or delta. Delta rowsets are discussed later in this section.

For AutoCollect to collect as much data as possible, verify that the collection is executed one last time just before the MUF terminates.

The purpose of AutoScope is to collect and analyze performance statistics over given periods. The occasional missed snapshot collection should not reduce its overall value to you.

Creating the Delta Rowsets

Once the snapshots are collected and stored in the AutoCollect snapshot database, execute an AutoCollect function to convert the Snapshot rowsets into Delta rowsets. A Delta rowset represents the resources consumed by the MUF between two snapshots. The delta creation process creates a delta for each interval (between two snapshots) and one delta from the beginning of MUF to the last snapshot. Because these deltas are created as part of the standard AutoCollect system, they are called system created deltas.

In addition to the system created delta rowsets, you can create additional delta rowsets that are created by combining or averaging existing deltas (both system and user created). Each of these user created delta rowsets are discussed in detail next.

The AutoCollect process also includes functions to extract the delta data into a file format acceptable as input to various spreadsheet products such as MS Excel.

Printing and Deleting Rowsets

The AutoCollect functions include options to print the available snapshot, delta rowsets, and a program that can be used to delete rowsets. In the AutoCollect system, you are responsible for deleting unwanted or no longer needed rowsets.

Setting Up the AutoCollect Environment

This section discusses the setup of the AutoCollect environment. The AutoCollect database definitions are prepopulated in all new installations and are added as part of the upgrade process for Version 12.0. Even though each MUF environment can house a complete AutoCollect environment, determine which MUFs to target for snapshot collection and which MUFs to store and manipulate the snapshots in.

Determine the Location of the Source and Repository MUFs

AutoCollect provides you with the ability to collect performance data from a MUF, known as the **source** MUF, and store or manipulate it in a different MUF, known as the **repository** MUF.

- Source MUF communication uses a special navigation process that is only available to Datacom system processes. The snapshot function must be executed on the same LPAR as the source MUF. The snapshot function uses a dynamic navigation process to locate the source MUF and collect the appropriate DST data into memory. The snapshot function only collects one snapshot from one MUF per execution.
- Repository MUF communication uses the standard navigation process. The Snapshot function uses the standard DBSIDPR navigation process to communicate to the repository MUF. This process allows communication to MUFs through SVC (same LPAR), XCF (different LPAR, but same Sysplex), or CCI (different Sysplex). Once communication to the repository MUF is established, the DST information that is collected in memory is written to the snapshot database on the repository MUF as a snapshot rowset.

By using the two forms of communication, AutoCollect can easily access DSTs from one MUF in your environment and store the results as Snapshots on a different MUF in your environment. This is especially helpful when you set up a Complex implementation.

Simple Implementation

You can use the same MUF environment as both the source and the repository for AutoCollect functions. For most cases, this is the typical implementation.

Alternate implementation

At certain sites, such as those with a heavily loaded production system, you may want to limit the overhead of AutoCollect processing on the source environment. In these sites, you can use a different MUF for storing or managing the Snapshot data (repository databases).

Complex implementation

Since AutoCollect attaches the source MUFNAME (required to be unique per installation) to each row that is stored in the repository databases, you can choose to collect data from multiple source MUFs and store it all in one set of repository databases. The AutoCollect programs have been coded to work with this possibility.

Allocating the AutoCollect Repository Database Datasets

The data collected by the AutoCollect process varies widely depending on your usage and historical storage criteria. Therefore, determining the appropriate size of the two index areas (IXX1019 and IXX1020) and the two data areas (SNP1019 and DEL1020) takes some experience at each site.

Depending on the number of captured Snapshot rowsets or Delta rowsets and the length of time they are kept, the amount of data can be significant.

The default AutoCollect database definitions have DYNAMIC EXTEND set to YES for both index and data areas. However, verify that there is adequate space on the selected volumes to allow the extend process.

Monitor the growth of these AutoCollect databases and add space as needed.

For more information about the number of rows and row content of each of the AutoCollect tables, see Snapshot Database (see page 85) and Delta Database (see page 107) respectively.

Starting Allocations

The following are the suggested initial allocations:

Name	Job code	Allocation
IXX1019	highlevl.AUTOCOLL.IXX1019	270 Tracks
IXX1020	highlevl.AUTOCOLL.IXX1020	290 Tracks
SNP1019	highlevl.AUTOCOLL.SNP1019	750 Tracks
DEL1020	highlevl.AUTOCOLL.DEL1020	850 Tracks

Selecting a MUF as a Source for Collection

You can select one or more source MUF environments per repository environment. The source and repository can both be in the same MUF environment.

To target a MUF environment as the source for data collection, you must know its MUFNAME. The MUFNAME value is established during MUF initiation. If the MUF startup option *MUF mufname* is specified, the MUFNAME value is required. This value must be a unique 8-byte name.

If the *MUF mufname* start-up option is not specified, the MUFNAME value is created using the z/OS batch job name or started task information.

To help ensure accuracy, all MUF source environments used for the AutoCollect process, should be uniquely identified using the *MUF mufname* startup option. For more information about specifying the MUFNAME, see the *CA Datacom/DB Database and System Administration Guide*.

Note: If you have the CA SYSVIEW Datacom option, you can use its DCLIST function to list all MUFs on a given LPAR, select a MUF job, and then issue a DCMUFS command to get the MUF identity information which includes the active MUFNAME.

AutoCollect and MUFPLEX (Including Shadow MUF)

In certain complex MUFPLEX environments, the MUFNAME parameter can be specified as an asterisk (*). This means that the MUFNAME is assigned at MUF startup from a list of values stored in the DBSIDPR module.

The Snapshot function still requires you to specify a MUFNAME parameter. In these cases, if the MUFNAME used by the MUF you want to target can change, submit a DBUTLTY Snapshot function for each of the possible MUFNAMEs. SNAPSHOTs that use a MUFNAME that is not active get a DBUTLTY return code error and do not affect the AutoCollect tables. Those that find an active MUFNAME produce the Snapshot rowset. You can use the DBUTLTY SET OPTION1=ON-ERROR-CONTINUE to bypass the snapshot functions that received the DBUTLTY error code.

The OUTNAME= parameter is provided in the Snapshot function to allow you to reassign the MUFNAME used in the Snapshot rowset when it is being generated. This allows you to select data from a MUF that may have multiple different MUFNAMEs, yet still build the rowsets with a consistent MUFNAME. For more information, see the OUTNAME parameter in the AUTOCOLL OPTION=SNAPSHOT function in the *CA Datacom/DB DBUTLTY Reference Guide*.

Example

Suppose you have a PRODMUF with a shadow MUF and you would typically migrate the workload from the active MUF to the shadow MUF once a week at midnight on Saturday.

After the shadow MUF is converted to the active MUF, you allow the new MUF to remain in place for the following week of processing. Then you restart a second MUF which becomes the shadow MUF for this active MUF. At the next weekly boundary, you repeat the process.

Now, you are interested in the collection of statistics from switchover on Sunday morning until 9 P.M. on Saturday.

The sequence of events is as follows:

```
8/05-8/12       Active MUF was PRODMF1A

8/12-8/19       Active MUF was PRODMF1B

8/19-8/26       Active MUF was PRODMF1A

8/26-9/02       Active MUF was PRODMF1B
```

To be able to capture these statistics as a single representation of the same source MUF within AutoCollect, the Snapshot function could be executed each Saturday for both possible MUF names with the OUTNAME being PRODMUF1.

The result would be as follows:

```
8/12 Snapshot from PRODMUF1A creates rowsets as PRODMUF1
8/12 Snapshot from PRODMUF1B fails with RC=16

8/19 Snapshot from PRODMUF1A fails with RC=16
8/19 Snapshot from PRODMUF1B creates rowsets as PRODMUF1

8/26 Snapshot from PRODMUF1A creates rowsets as PRODMUF1
8/26 Snapshot from PRODMUF1B fails with RC=16

9/02 Snapshot from PRODMUF1A fails with RC=16
9/02 Snapshot from PRODMUF1B creates rowsets as PRODMUF1
```

The AutoCollect system would have 4 Snapshot rowsets, 1 for each week with the MUFNAME of PRODMUF1.

More complex MUFPLEX usage including full data sharing may require additional study and implementations using the OUTNAME parameter which is not included with this guide at this time.

Examples

In the following section, we discuss some possible AutoCollect implementations. In one example of a complex implementation, we discuss how the OUTNAME parameter can be used to record two sets of AutoCollect snapshots from the same MUF with different names for collection and reporting purposes.

Determining a Schedule for Snapshot Collection

After you have selected a source MUF, decide on the schedule of snapshots that will best serve your needs.

For first time users of the AutoCollect system, we recommend that you take either a weekly or at most a daily snapshot.

As you become more experienced, you may want to do additional snapshots to get more granular information. However, the more snapshots taken, the more data that is collected and needs to be managed.

Example 1

You want to build a set of weekly statistics to compare weekly activities and to evaluate various tuning changes. This is the recommended schedule for the beginner doing Business Value Metrics (BVM) measurements and tuning.

- The MUF is cycled every Saturday evening at midnight.
- You use a batch job to submit the MUF EOJ request.

For this example, the solution is to insert one additional step in front of the MUF EOJ step that requests the snapshot to be taken. If the snapshot fails for some reason, the MUFEOJ step should be skipped. This allows the MUF to stay up and for the snapshot attempt to be re-issued to verify that the data is collected before the MUF is ended.

Example 2

You want to build a set of weekly statistics to compare weekly activities and to evaluate various tuning changes.

- The MUF is cycled every Saturday evening at midnight.
- You use a console command to submit the MUF EOJ request.

For this example, the solution is to require the operator to submit the Snapshot function batch job prior to issuing the EOJ command.

An alternative would be that in normal occurrences, the workload from Saturday at 17:00 until the MUF is cycled is not significant. You might decide to just have a scheduled snapshot that runs every Saturday at 17:00. This way you can compare each week as the period from when MUF comes up until 17:00 on Saturday and would not require special attention from the operator.

Example 3

You want to build a set of weekly statistics but MUF is cycled on a monthly, or on an *as-needed* basis. This means that it may not be predictable which day of the week the MUF is cycled.

- The MUF cycles as needed.
- You use a console command to submit the MUF EOJ request.

For this example, the solution is to have an automated job that runs Saturday at 17:00 to create a snapshot. In addition, a final snapshot must be taken for the end of MUF statistics just prior to MUF EOJ.

Each week that includes a MUF cycle would actually have two or more snapshots that must be added together to get that week's statistics.

You can use the AUTOCOLL OPTION=SUMMARY function to combine these multiple rowset periods (for a given week) into a single new SUMMARY rowset providing the ability to combine the full week's data into a single rowset for comparison purposes. For information about the AUTOCOLL OPTION=SUMMARY function, see the *CA Datacom/DB DBUTLTY Reference Guide*.

Example 4

You want to build a set of weekly statistics, but also want to see daily activity between 9:00 and 17:00. This would allow you to compare important times each day and statistics.

- The MUF is cycled every Saturday night at midnight.
- You use a console command to submit the MUF EOJ request.

For this example, the solution is to have an automated job that runs the Snapshot function each day at 9:00 and at 17:00. In addition, a final snapshot must be taken for the end of week total. This final snapshot is either at 24:00 on Saturday or just prior to MUF EOJ.

As stated in the preceding section, the first time users for AutoCollect should choose a simpler schedule of Snapshot functions to meet their needs and then add additional snapshots as they become more familiar with the system.

Example 5 (Complex Situation) – building two sets of statistics from one MUF

In certain complex environments, the MUFNAME parameter that was discussed earlier in regards to MUFPLEX (including Shadow MUF) environments can be used as a means to collect snapshot data with two different goals in mind.

You want to have weekly snapshots to use for weekly comparisons as suggested in Business Value Metrics and to have snapshots for critical processing done during the daylight hours of Monday through Friday.

In this case, consider using the OUTNAME parameter as a way to segregate the two sets of snapshots. The schedule should be as follows. The rows for Sunday through Friday are daily snapshots and the row for Saturday is the weekly snapshot. Because the snapshot rowsets always contain accumulation values you get the full statistics for the weekly snapshots even though there were intervening daily snapshots.

```
Sun    8/6     00:00   MUF is initiated
Mon    8/7     09:00   Snapshot with OUTNAME PRODMUFD
Mon    8/7     17:00   Snapshot with OUTNAME PRODMUFD
Tue    8/8     09:00   Snapshot with OUTNAME PRODMUFD
Tue    8/8     17:00   Snapshot with OUTNAME PRODMUFD
Wed    8/9     09:00   Snapshot with OUTNAME PRODMUFD
Wed    8/9     17:00   Snapshot with OUTNAME PRODMUFD
Thu    8/10    09:00   Snapshot with OUTNAME PRODMUFD
Thu    8/10    17:00   Snapshot with OUTNAME PRODMUFD
Fri    8/11    09:00   Snapshot with OUTNAME PRODMUFD
Fri    8/11    17:00   Snapshot with OUTNAME PRODMUFD
Sat    8/12    24:00   Snapshot with OUTNAME PRODMUFW
```

AutoCollect provides the ability to process Snapshot rowsets by MUFNAME. This allows you to create spreadsheets and BVM data based on either the daily information for PRODMUFD or the weekly data for PRODMUFW.

Example 6 (Complex Situation) – building two sets of statistics from one MUF but storing the data separately in two repositories

Another alternative is to define two different MUF environments as repositories for a given source MUF. When executing the Snapshot function, you can point to the appropriate repository for that Snapshot rowset using DBSIDPR.

For example:

- Source MUF is MUFPROD.
- Defined repository databases in MUFPROD for weekly (tuning) snapshots.
- Defined repository databases in MUFSYSP for daily (debugging) snapshots.

Each day snapshots are taken every three hours and directed to the MUFSYSP repository.

- The AUTOCOLL OPTION=SNAPSHOT function is executed in the same LPAR as MUFPROD.
- Step or job libraries load a DBSIDPR that points to the MUFSYSP MUF.

Once a week on Saturday at 24:00, weekly snapshots are taken and directed to the MUFPROD repository.

- The AUTOCOLL OPTION=SNAPSHOT function is executed in the same LPAR as MUFPROD.
- Step or job libraries load a DBSIDPR that points to the MUFPROD MUF.

Note: The flexibility of the snapshot schedule coupled with the ability to assign MUFNAMEs and select different repository MUFs gives the user a wide choice of options for implementing AutoCollect. The previous six examples are just a few of the possible scenarios.

Chapter 3: AutoCollect – Creating SNAPSHOT and System DELTA Rowsets

This chapter covers the collection and storage of Snapshot rowsets and the creation of the corresponding Delta rowsets.

Snapshots of Selected DST Tables

The AUTOCOLL OPTION=SNAPSHOT function pulls information from selected Dynamic System Tables (DST) and stores it in the SNAPSHOT database (DBID 1019) in the repository MUF.

The DST information is copied to the SNAPSHOT tables with additional columns added to the front of each row to uniquely identify each Snapshot rowset.

- S_MUF_ENABLE—Date/time this execution of MUF was enabled
- S_MUF_DATETIME—Date/time of the Snapshot
- S_MUF_TOTALROW—Differentiate between normal and total rows (see the following)

The columns can be combined with the MUF_NAME and other columns for each table definition to uniquely identify the rows in a given Snapshot rowset.

The rest of the row's content is copied intact from the DST row. When the DST table being processed has one row, such as the MUF_SYSTEM_STATS table, there is one detail Snapshot row created. When the DST table has multiple rows, such as the MUF_AREA_STATS table, the Snapshot table will have a corresponding set of detail rows.

For multiple-row DST tables, the AutoCollect process also creates one or more total rows in the Snapshot table. The total rows are indicated by having the S_MUF_TOTALROW flag set to Y, detail rows have this flag set to N. The total row's value columns is the total of values collected from detail rows for this table, in this Snapshot rowset.

For more information about the number and contents of each Snapshot rowset, see SNAPSHOT Database (see page 85).

Executing the AUTOCOLL SNAPSHOT Function

The SNAPSHOT function must be executed on the same LPAR as the source MUF. When executing the function, the STEPLIB concatenation must include the following:

- Library concatenation that provides a DBSIDPR module that points to the repository MUF, that is, the MUF that contains DBID 1019 and 1020.
  - If the repository MUF is on a different LPAR from the source MUF, the DBSIDPR communication must be set up to use XCF or CCI to connect to the repository MUF.
- Complete set of CA-Datacom execution libraries.

The SNAPSHOT function performs the following:

- Uses a standard URT to connect to the repository MUF through DBSIDPR.
- Uses the dynamic navigation process to connect to the source MUF.
- Issues the appropriate commands to retrieve the DST rows from the source MUF.
- Formats and stores the DST rows as snapshot rows using RAAT ADDIT commands to the SNAPSHOT tables on the repository MUF
- Produces a report providing the following:
  - Connection status
  - Snapshot row counts
  - Errors

Selection Parameters

For the SNAPSHOT function, select the source MUF to process into a snapshot rowset.

`MUFNAME=mufname`

Specifies the MUFNAME to be processed.

Output Parameters

`OUTNAME`

Place your 1- to 8-byte character MUFNAME that will be used to overlay the MUFNAME of the selected the source MUF. If the OUTNAME is not specified, the rows are created with the MUFNAME from the source MUF.

The expected return code is 0 when executing the SNAPSHOT function. For details about the AUTOCOLL CA Datacom/DB *DBUTLTY Reference Guide*.

Reporting Snapshot Rowsets

You can produce reports on specific Snapshot rowsets with the AUTOCOLL OPTION=SNAPRPT function. The report provides information about each of the rowsets and the number of rows in each SNAPSHOT table that is part of that rowset.

The rowset report is used to determine what snapshots are available to be processed into deltas. The report does not list the various rows and their contents.

To see snapshot row contents you need to use one of the various query tools provided in CA Datacom (CA Dataquery, DBSQLPR, DB Reporter option, and so on). While user programs can also be written using the standard RAAT, SAAT and SQL APIs, we recommend that they are not used to manipulate the snapshot rows data content as this can create problems for the rest of the AutoCollect functions.

The expected return code is 0 when executing the SNAPRPT function. For details about the AUTOCOLL OPTION=SNAPRPT syntax, parameters, and sample output report see the *CA Datacom/DB DBUTLTY Reference Guide.*

Deleting Snapshot Rowsets

In some cases, you may want to remove snapshot rowsets once they have been processed into delta rowsets. In other cases, you may want to remove an improperly built snapshot rowset before building the delta rowsets.

To clear out old data or remove incorrect SNPASHOT rowsets, you can use the SNAPDEL function.

SNAPDEL provides various selection parameters to limit which rowsets are deleted. Before running SNAPDEL, you should run a SNAPRPT using the same selection criteria to verify that only the unneeded rowsets are deleted. As a precaution, you can also decide to back up the Snapshot database (1019) before running the delete.

Use the AUTOCOLL OPTION=SNAPDEL function to delete a set of Snapshot rowsets. The snapshot rowset delete function removes all rows in each table that match the selection criteria. It provides a report of the rows that were deleted from each table.

The expected return code is 0 when executing the SNAPDEL function. For details about the AUTOCOLL OPTION=SNAPDEL syntax, parameters, and sample output report see the *CA Datacom/DB DBUTLTY Reference Guide.*

Process Snapshot Rowsets into Delta Rowsets

The AUTOCOLL OPTION=DELTACRE function processes snapshot rowsets into Delta rowsets. The function can be started at a specific point in time (in the snapshot database) and can be targeted to a specific MUFNAME. The function processes the values from two Snapshot rowsets from the same MUF instance execution (MUFNAME and MUF Enable time are the same) and create a Delta rowset with the following:

- **Identity columns**. These columns represent Delta rowset identification information.
- **Fixed columns.** These columns represent Delta identification that is built from non-counter snapshot columns.
- **Calculated columns.** These are columns that are built from the calculation of ending snapshot value minus beginning snapshot value.

The AutoCollect DELTA Database (see page 107) chapter provides the contents of each Delta table. The column information includes a column information flag that indicates if the column was built as part of the rowset identification, copied from the starting snapshot, copied from the ending snapshot or calculated by subtracting the beginning snapshot value from the ending snapshot value.

Once the Delta row is built, it is added into the appropriate DELTA table.

For each Delta period processed, the DELTACRE function adds one row to the BVM and BVD DELTA ratio tables. These tables represent selected information collected and summarized from matching rows found in the other DELTA tables.

Note: In most shops, the information from BVM and BVD is used to report system performance and the information in the other DELTA tables is used to *drill down* as needed.

A report is generated providing Delta row generation counts and any error information.

The generation of Delta rows can be done multiple times. All selected Snapshot rowsets are processed and the corresponding Delta rowsets are created.

The DELTACRE function attempts to add all newly built Delta rowsets. If a Delta rowset already exists with the same identifying columns, the new Delta rowset is ignored. The DELTACRE function report indicates the number of rowsets ignored as a duplicate. This ignore process prevents you from overlaying a previously created Delta rowset. If you determine that the rowset should be overlaid, you need to first delete the rowset using the Delta rowset delete function and then re-run the DELTACRE function.

The DELTACRE function validates the calculated values for each Delta row. Calculated columns cannot have a negative value. For example, while the I/O count for a given duration can be zero (that is no I/O was done), it cannot be a negative value. The DELTACRE function reports an error for any Delta rows that have a negative value and those rows are *not* added to the Delta rowset.

Delta rowsets are not automatically deleted. Therefore, if you remove a Snapshot rowset, the Delta rowset remains. You can use the AUTOCOLL OPTION=DELTADEL function to remove unwanted Delta rowsets.

Delta Row Format

The information loaded into the Delta row is very similar to the corresponding DST table row with additional columns added to the front of the row to uniquely identify the Delta rowset.

- D_MUF_ENABLE—Date and time this execution of MUF was enabled
- D_MUF_START_DATETIME—Start date and time from the starting Snapshot
- D_MUF_END_DATETIME—End date and time of the ending Snapshot
- D_MUF_TYPE—Type of DELTA, either INTERVAL or LAST
- D_MUF_TOTALROW—Type of row, either Y for a total row or N for a normal row
- D_MUF_DURATION—Duration in hours that this Delta represents

By combining the first four fields with the MUF_NAME column found on each table, each Delta rowset can be uniquely identified.

Delta Processing Rules

Delta rowsets are created from Snapshot rowsets. Processing rules are as follows. See the following diagram for a sample.

First Snapshot Rowset for a Given MUF Execution (MUFNAME and S_MUF_ENABLE)

Create a Delta row with fixed values from the first Snapshot row since MUF was enabled.

Row	**Value**
D_MUF_ENABLE	from S_MUF_ENABLE
D_MUF_START_DATETIME	from S_MUF_ENABLE
D_MUF_END_DATETIME	from S_MUF_DATETIME
D_MUFTYPE	INTERVAL
D_MUF_TOTALROW	from S_MUF_TOTALROW
D_MUF_DURATION	from S_MUF_DATETIME minus S_MUF_ ENABLE

Copy corresponding Snapshot row counter values intact to Delta row.

Next Snapshot for a Given MUF Execution

Create a Delta row with fixed values from the corresponding Snapshot row.

Row	Value
D_MUF_ENABLE	from S_MUF_ENABLE (ending Snapshot)
D_MUF_START_DATETIME	from S_MUF_DATETIME (starting Snapshot)
D_MUF_END_DATETIME	from S_MUF_DATETIME (ending Snapshot)
D_MUFTYPE	INTERVAL
D_MUF_TOTALROW	from S_MUF_TOTALROW (ending Snapshot)
D_MUF_DURATION	from D_MUF_START_DATETIME minus D_MUF_END_ DATETIME

Calculate counter values by subtracting the beginning Snapshot column value from the ending Snapshot value.

Last Snapshot for a given MUF execution

Generate one interval rowset as defined in the preceding section.

The DELTACRE function generates a "last" delta rowset. Each last Delta row is created with fixed values from the Snapshot row and counter values intact from the Snapshot row.

This last delta rowset represents the MUF processing done from Startup to last recorded Snapshot.

Row	Value
D_MUF_ENABLE	from S_MUF_ENABLE
D_MUF_START_DATETIME	from S_MUF_ENABLE
D_MUF_END_DATETIME	from S_MUF_DATETIME
D_MUFTYPE	LAST
D_MUF_TOTALROW	from S_MUF_TOTALROW

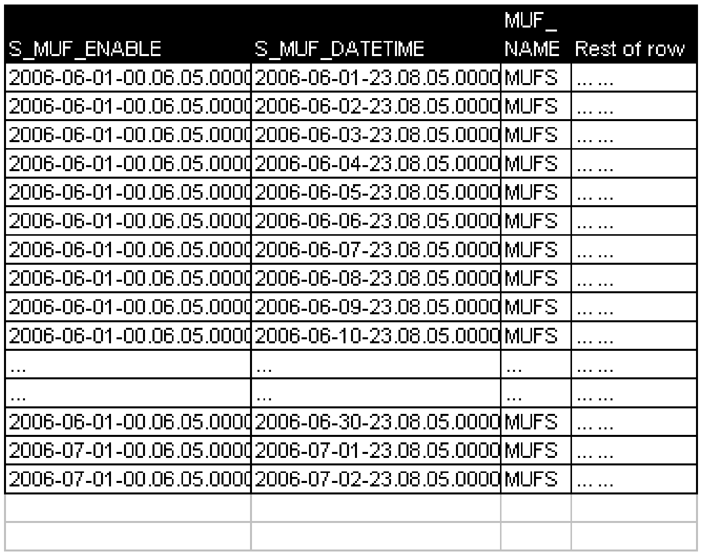
Row	Value
D_MUF_DURATION	from S_MUF_DATETIME minus S_MUF_ ENABLE

Copy corresponding Snapshot row counter values intact to Delta row.

Sample of Delta Rows

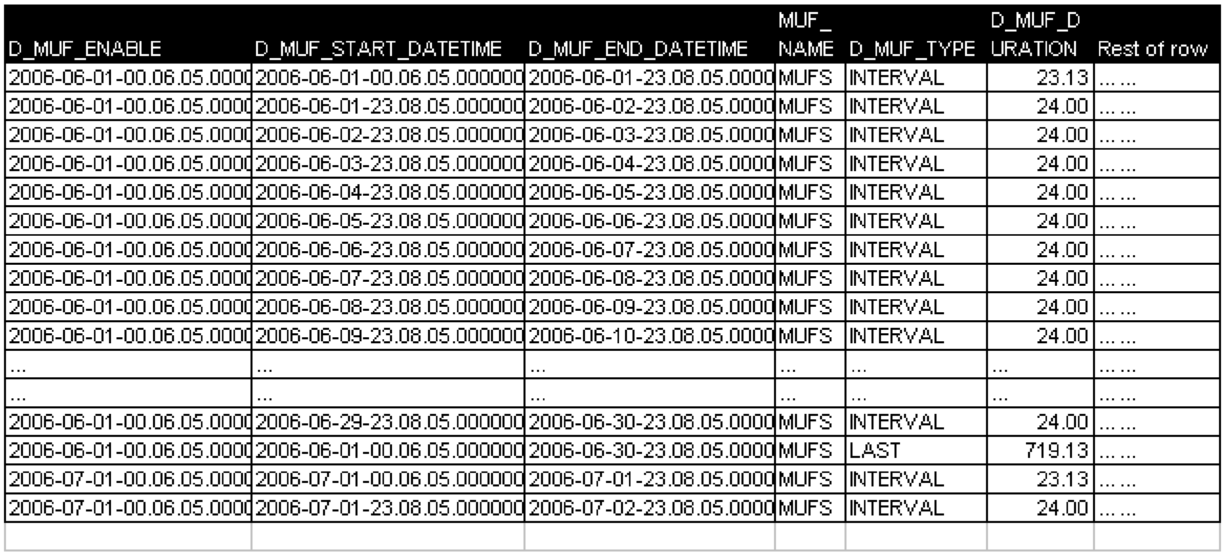
Delta rows created from a SNAPSHOT database that has a Snapshot taken every day at 23:08. MUF is ended at 23:10 on the last day of the month without any Snapshot. MUF is restarted at 00:00 on first day of the month.

Snapshot Rowsets



S_MUF_ENABLE	S_MUF_DATETIME	MUF_ NAME	Rest of row
2006-06-01-00.06.05.0000	2006-06-01-23.08.05.0000	MUFS	... ...
2006-06-01-00.06.05.0000	2006-06-02-23.08.05.0000	MUFS	... ...
2006-06-01-00.06.05.0000	2006-06-03-23.08.05.0000	MUFS	... ...
2006-06-01-00.06.05.0000	2006-06-04-23.08.05.0000	MUFS	... ...
2006-06-01-00.06.05.0000	2006-06-05-23.08.05.0000	MUFS	... ...
2006-06-01-00.06.05.0000	2006-06-06-23.08.05.0000	MUFS	... ...
2006-06-01-00.06.05.0000	2006-06-07-23.08.05.0000	MUFS	... ...
2006-06-01-00.06.05.0000	2006-06-08-23.08.05.0000	MUFS	... ...
2006-06-01-00.06.05.0000	2006-06-09-23.08.05.0000	MUFS	... ...
2006-06-01-00.06.05.0000	2006-06-10-23.08.05.0000	MUFS	... ...
...	...	...	... ...
...	...	...	... ...
2006-06-01-00.06.05.0000	2006-06-30-23.08.05.0000	MUFS	... ...
2006-07-01-00.06.05.0000	2006-07-01-23.08.05.0000	MUFS	... ...
2006-07-01-00.06.05.0000	2006-07-02-23.08.05.0000	MUFS	... ...

Delta Rowsets



D_MUF_ENABLE	D_MUF_START_DATETIME	D_MUF_END_DATETIME	MUF_ NAME	D_MUF_TYPE	D_MUF_D URATION	Rest of row
2006-06-01-00.06.05.0000	2006-06-01-00.06.05.000000	2006-06-01-23.08.05.0000	MUFS	INTERVAL	23.13	... ...
2006-06-01-00.06.05.0000	2006-06-01-23.08.05.000000	2006-06-02-23.08.05.0000	MUFS	INTERVAL	24.00	... ...
2006-06-01-00.06.05.0000	2006-06-02-23.08.05.000000	2006-06-03-23.08.05.0000	MUFS	INTERVAL	24.00	... ...
2006-06-01-00.06.05.0000	2006-06-03-23.08.05.000000	2006-06-04-23.08.05.0000	MUFS	INTERVAL	24.00	... ...
2006-06-01-00.06.05.0000	2006-06-04-23.08.05.000000	2006-06-05-23.08.05.0000	MUFS	INTERVAL	24.00	... ...
2006-06-01-00.06.05.0000	2006-06-05-23.08.05.000000	2006-06-06-23.08.05.0000	MUFS	INTERVAL	24.00	... ...
2006-06-01-00.06.05.0000	2006-06-06-23.08.05.000000	2006-06-07-23.08.05.0000	MUFS	INTERVAL	24.00	... ...
2006-06-01-00.06.05.0000	2006-06-07-23.08.05.000000	2006-06-08-23.08.05.0000	MUFS	INTERVAL	24.00	... ...
2006-06-01-00.06.05.0000	2006-06-08-23.08.05.000000	2006-06-09-23.08.05.0000	MUFS	INTERVAL	24.00	... ...
2006-06-01-00.06.05.0000	2006-06-09-23.08.05.000000	2006-06-10-23.08.05.0000	MUFS	INTERVAL	24.00	... ...
...	...	...	...	...	...	... ...
...	...	...	...	...	...	... ...
2006-06-01-00.06.05.0000	2006-06-29-23.08.05.000000	2006-06-30-23.08.05.0000	MUFS	INTERVAL	24.00	... ...
2006-06-01-00.06.05.0000	2006-06-01-00.06.05.000000	2006-06-30-23.08.05.0000	MUFS	LAST	719.13	... ...
2006-07-01-00.06.05.0000	2006-07-01-00.06.05.000000	2006-07-01-23.08.05.0000	MUFS	INTERVAL	23.13	... ...
2006-07-01-00.06.05.0000	2006-07-01-23.08.05.000000	2006-07-02-23.08.05.0000	MUFS	INTERVAL	24.00	... ...

Executing AUTOCOLL OPTION=DELTACRE

The DELTACRE function executes against the SNAPSHOT and DELTA database in the repository MUF environment only. To execute the DELTACRE function, the STEPLIB concatenation must include the following:

- Library concatenation that provides a DBSIDPR module that points to the repository MUF, that is, the MUF that contains DBID 1019 and 1020.
- Complete set of CA Datacom execution libraries

The DELTACRE function performs the following:

- Use a standard URT to connect to the repository MUF through DBSIDPR
- Process the Snapshot rowsets selected by the input parameters and generate Delta rowsets
- Produce a report providing the following:
  - Snapshot rowsets processed and any errors
  - Delta rowsets created
  - Duplicate Delta rowsets that were ignored
  - Delta rows that had errors and were not written
  - Error listing with detailed information about error rows

Selection Parameters

For the DELTACRE functions you can use the following parameters to select the snapshot rowsets used for input to the processing.

`MUFNAME=mufname`

Specifies to restrict processing to Snapshot rowsets from this selected MUF only. If MUFNAME is not specified all snapshot rowsets are available to be processed.

`DATEC=ccyymmdd`

Select to start processing at Snapshot rowsets that have a S_MUF_DATETIME data value that is equal to or greater than this parameter value. If DATEC is not specified all snapshot rowsets are available to be processed.

The expected return code is 0 when executing the DELTACRE function. For details about the AUTOCOLL OPTION=DELTACRE syntax, parameters and output report see the *CA Datacom/DB DBUTLTY Reference Guide*.

Reporting Delta Rowsets

You can produce reports about specific Delta rowsets with the AUTOCOLL OPTION=DELTARPT function. The report provides information about each of the rowsets and the number of rows in each DELTA table that is part of that rowset.

The rowset report is used to determine what deltas are available to be processed into other deltas or selected for extraction into output formats for loading into performance spreadsheets. The report does not list the various rows and their contents.

To see delta row contents, use one of the various query tools provided in CA Datacom (CA Dataquery, DBSQLPR, DB Reporter option, and so on.). While user programs can also be written using the standard RAAT, SAAT and SQL APIs, we recommend that they are not used to manipulate the delta rows' data content as this can create problems for the rest of the AutoCollect functions.

The expected return code is 0 when executing the DELTARPT function. For details about the AUTOCOLL OPTION=DELTARPT syntax, parameters, and sample output report see the *CA Datacom/DB DBUTLTY Reference Guide*.

Deleting Delta Rowsets

In some cases, you may want to delete DELTA rowsets that have been created. For example, you no longer need data from two years ago, or you have created a SUMMARY rowset but picked the wrong input rowsets. To clear out old data or remove incorrect rowsets before recreating the correct rowsets, you can use the DELTADEL function.

DELTADEL provides various selection parameters to limit which rowsets are deleted. Before running DELTADEL, you should run a DELTARPT using the same selection criteria to verify that only the unneeded rowsets are deleted. As a precaution, you may also decide to back up the Delta database (1020) before running the delete.

The expected return code is 0 when executing the DELTADEL function. For details about the AUTOCOLL OPTION=DELTADEL syntax, parameters, and sample output report see the *CA Datacom/DB DBUTLTY Reference Guide*.

Chapter 4: AutoCollect Delta Tables

The data stored in the Delta tables provides period-by-period execution information for a given source MUF.

The DELTA database contains 15 detail tables and two business value metric tables. A detailed layout of each of these tables is presented in DELTA Database (see page 107).

Detail Delta Tables

The Delta detail tables contain the identification columns followed by columns calculated from the snapshots of the DST tables. The format of each Delta table matches the format of the corresponding Snapshot table.

Business Value Metric Delta Tables

Currently, there are two business value metric tables. These tables provide key performance metrics or ratios:

- D_BUSINESS_VALUE_DETAILS (BVD)
- D_BUSINESS_VALUE_METRICS (BVM)

The D_BUSINESS_VALUE_DETAILS (BVD) table provides a full list of key statistics and calculated performance ratios used in detailed MUF analysis. This set of metrics was created from known common DBA practices that have been developed of the years to measure CA Datacom performance.

The D_BUSINESS_VALUE_METRICS (BVM) table provides a short collection of high-level statistics and calculated performance ratios as related to the Business Value Metrics spreadsheet for system measurement and performance. These business value metrics provide a simple way for you to measure the overall performance (or business value) of a given MUF. Business value metrics have been introduced in recent years as a common way for you to measure and compare performance of your MUFs.

These ratio tables are built by collecting information from all detail rows for a given Delta rowset and summarizing them into a single row. For each Delta rowset, one row is added to the BVM and BVD tables.

The purpose of this summarization is to provide precalculated measurement statistics for each of the Delta periods that could be used to reduce the amount of day-to-day review that is performed on a given set of Delta statistics. You can use these summarization and measurement statistics to provide the primary performance reporting for a given MUF. Only when a problem or concern is detected would additional drill-down into the detail Delta tables be needed.

Unlike the detail Delta tables, the BVM and BVD tables are not modeled after any specific DST table. However, we strived to maintain a consistency between the column name and use in the BVM and BVD tables with that in the Delta and DST tables. In most cases, the column in the BVM and BVD tables represents the total for that column from all rows in a selected Delta rowset.

Chapter 5: AutoCollect - Creating User Defined DELTA Rowsets

The data stored in the system Delta rowsets provides period-by-period execution information for a given source MUF.

You can use additional AUTOCOLL functions to create additional user created DELTA rowsets.

Manipulating Delta Rowsets - User Created Delta Rowsets

The AutoCollect process recognizes that there is a need to manipulate the data stored in the Delta tables. In most cases, you want to do this manipulation based on a given rowset or group of rowsets.

There are three AUTOCOLL options to perform basic Delta rowset manipulation functions. The rowset functions are the following:

- SUMMARY is used to combine Delta rowsets into a single new Summary rowset for comparison purposes. These rowsets can be used to summarize smaller delta periods into larger delta periods. For example, daily deltas into weekly deltas or weekly delta into monthly deltas.
- BASELINE is used to combine Delta rowsets into a given new rowset and then divide the counter columns by the number of input rowsets to get a calculated baseline. These baseline rowsets could be used for comparison purposes such as comparing an average week from one year to an average week to another year.
- AVGPERF is used to create special comparison rowsets. These rowsets are built by adding selected rowsets together and then dividing by the total duration hours to get average performance per single duration hour. These rowsets are used to compare the performance of two periods where the duration of the periods is very different. For example, the average hourly performance of a selected week to the average hourly performance of a selected month.

As stated, you can use the DELTARPT function to help decide what rowsets are available for SUMMARY, BASELINE, or AVGPERF processing.

Once a user created Delta rowset is generated by the Summary/AVGPERF/BASELINE options, the rowset is available for further user Delta creation processing. For example, you create several weekly Summary rowsets from the matching seven days of daily (Interval) rowsets. These Summary rowsets are now available to be input to another four-week Summary rowset or a Baseline rowset.

Tagging Rows

As discussed above, you have the flexibility of overriding the MUFNAME value using the OUTNAME parameter. This functionality allows you to either combine or separate snapshots by MUFNAME before processing them into delta rowsets.

The AutoCollect OUTTAG parameter is valid for OPTION=BASELINE, OPTION=SUMMARY and OPTION=AVGPERF. The OUTTAG parameter provides a simple way to identify or tag Delta rowsets created with the SUMMARY, BASELINE, and AVGPERF functions. Once the rowsets are tagged with this addition identifier, future AutoCollect functions can use this value as part of their selection criteria. The OUTTAG parameter is useful in situations where you have a lot of data in the DELTA database that you want to keep, but would like to mark certain rowsets.

Since the tag information could be used to both select the input to a BASELINE, AVGPERF or SUMMARY function and to tag the output of the function, we have implemented a second parameter that is used when selecting rows by the tag information. This parameter is named USERTAG.

USERTAG is used to select delta rowsets that are being processed by the BASELINE, AVGPERF, SUMMARY, DELTARPT, and DELTADEL functions.

Example:

Summary rows are created from daytime intervals and nighttime intervals. You want to keep these summaries separate so that different spreadsheets can be generated. You could use the TIMETO and TIMEFROM values during the data extract (DSVOUT) function to do this. But in some cases the times may be varying and coding the time selection parameters may not be possible. In these cases, you can use the OUTTAG parameter during the OPTION=SUMMARY rowset creation to mark the rowsets for NIGHT versus DAY.

Since the USERTAG column, like all Delta columns, can be updated by user programs, you can add your own routines to mark selected rowsets to create special reports, generate spreadsheets, and so on. While user programs can be written using the standard RAAT, SAAT and SQL APIs, we recommend that they are not used to manipulate the delta rows' data column content as this may create problems for the rest of the AutoCollect functions or lead to improper performance reporting.

This OUTTAG/USERTAG functionality is only needed when you move into more complex processing situations.

Executing AUTOCOLL OPTION=BASELINE/SUMMARY/AVGPERF

When executing the AUTOCOLL function, the STEPLIB concatenation must include the following:

- Library with the DBSIDPR module that points to the repository MUF, that is, the MUF that contains DBID 1019 and 1020
- Complete set of CA Datacom execution libraries

The AUTOCOLL function performs the following:

- Use a standard URT to connect to the repository MUF through DBSIDPR
- Open the Delta database
- Process the Delta rowsets selected by the input parameters
- Generate a new Delta rowset
- Produce a report providing
  - Delta rows read in and processed
  - Output duplicate Delta rowset that was ignored
  - Output Delta rows that had errors and were not written
  - Error listing with detailed information about error rows

Command Option Parameter

Determines which OPTION= function should be executed. Only one function can be chosen per function DBUTLTY AUTOCOLL command.

OPTION=SUMMARY

Specifies to add the duration and counter columns from all Delta rowsets that are selected and build a new Summary Delta rowset. A Summary Delta rowset has its D_MUF_TYPE column set to SUMMARY.

The D_MUF_START_DATETIME is copied from the earliest selected Delta rowset. The D_MUF_END_DATETIME is copied from the latest Delta rowset.

OPTION=BASELINE

Specifies to add the duration and counter columns from all Delta rowsets that are selected and then divide the total value by the number of Delta rowsets selected. Use these values to build a new Baseline Delta rowset. A Baseline rowset has its D_MUF_TYPE column set to BASELINE.

The D_MUF_START_DATETIME is copied from the earliest selected Delta rowset. The D_MUF_END_DATETIME is copied from the latest Delta rowset.

OPTION=AVGPERF

Specifies to add the duration and counter columns from all selected Delta rowsets that are selected and then divide the total number of duration hours of Delta rowsets selected. Use these values to build a new Average performance (for one duration hour) Delta rowset. An Average Performance rowset has its D_MUF_TYPE column set to AVGPERF, and its D_MUF_DURATION set to 1.00.

The S_MUF_START_DATETIME is copied from the earliest selected Delta rowset. The S_MUF_END_DATETIME is copied from the latest Delta rowset.

Selection Parameters

For the BASELINE/SUMMARY/AVGPERF functions, you can use the following parameters to select the delta rowsets used for input to the processing.

MUFNAME=mufname

Specifies to restrict processing to Delta rowsets from this selected MUF only. Therefore, to create new rowsets for more than one MUF, execute this job once for each MUF you want to target.

TYPE=type

Specifies to restrict the Delta rowsets that you want selected as input as follows:

AVGPERF

Selects only Delta rowsets built by the AUTOCOLL OPTION=AVGPERF function

BASELINE

Selects only Delta rowsets built by the AUTOCOLL OPTION=BASELINE function

INTERVAL

Selects only Interval Delta rowsets built by the AUTOCOLL OPTION=DELTACRE function

LAST

Selects only Last Delta rowsets built by the AUTOCOLL OPTION=DELTACRE function

SUMMARY

Selects only Delta rowsets that built by the AUTOCOLL OPTION=DELTACRE function

DATERG

Select only Delta rowsets that have their D_MUF_END_DATETIME date value within the range of dates specified in this parameter.

The format of the entry is ccyymmddccyymmdd where the first 8 bytes are the starting date and the last 8 bytes are the ending date. The starting and ending date can be in the range of 19000101 to 39991231. The ending date must be equal to or greater than the starting date. If you do not specify a date range, the date value is not used when selecting a rowset.

TIMERG

Select only Delta rowsets that have their D_MUF_END_DATETIME time value within the range of times specified in this parameter.

The format of the entry is hhmmsshhmmss where the first 6 bytes are the starting time and the last 6 bytes are the ending time. The starting and ending time can be in the range of 000000 to 240000. The ending time must be equal to or greater than the starting time. If you do not specify a time range, the time value is not used when selecting a rowset.

USERTAG

Select only Delta rowsets that have this value in their USERTAG column. If you do not specify a USERTAG, the USERTAG value is not used when selecting a rowset.

Output Parameters

OUTTAG

Place your 1- to 16-byte character tag in the column USERTAG on each Delta row produced by this execution. The OUTTAG is an optional feature that allows you to mark the newly created Delta rowsets so they can easily be selected for future processing. If the OUTTAG is not specified, the rows are created with a blank USERTAG column.

The expected return code is 0 when executing the BASELINE/SUMMARY/AVGPERF function. For details about the AUTOCOLL OPTION=BASELINE/SUMMARY/AVGPERF syntax, parameters, and sample output report see the *CA Datacom/DB DBUTLTY Reference Guide*.

How Date and Time Selection Is Processed

The DATERG and TIMERG parameters are applied to the rowset selection as individual selection pairs.

As each rowset is evaluated, it is first verified that the date portion of the D_MUF_END_DATETIME fits within the specified DATERG value. If the rowset qualifies for date selection, then the time portion of the D_MUF_END_DATETIME is verified that it fits within the TIMERG value.

The purpose of this process is to allow you to select certain time slices from certain days for processing.

For example, if you collect two snapshots each day at 0800 and 1700, the processing would generate Delta rowsets for the period from 1700 through the next day at 0800 and from 0800 through 1700. In this case, the Delta rowsets would represent the online day, 0800 to 1700, and the overnight processing, 1700 to 0800 (on the next day).

When creating SUMMARY, BASELINE, and AVGPERF rowsets, you can include the online day, the overnight processing, or both.

To process the online day only, the TIMERG would be 163000173000. This would allow for the 1700 (5 PM) snapshot to have occurred slightly before or after 1700. This selection would not select the time slices that ended at 0800 (8 AM).

To process only the overnight time slices (those that end at 0800), the TIMERG value would be specified as 073000083000.

To include both periods, specify TIMERG as 000000240000.

When processing Delta rowsets, the row must pass all selection criteria specified to be selected for processing. As mentioned earlier with delete processing for Delta rowsets, you may consider running a DELTARPT with the same selection criteria before you run the SUMMARY/BASELINE/AVGPERF function. The report would show the periods being selected and allow you to verify the selection criteria before creating the new Delta rowset.

Examples of Creating Delta Summary Rowsets

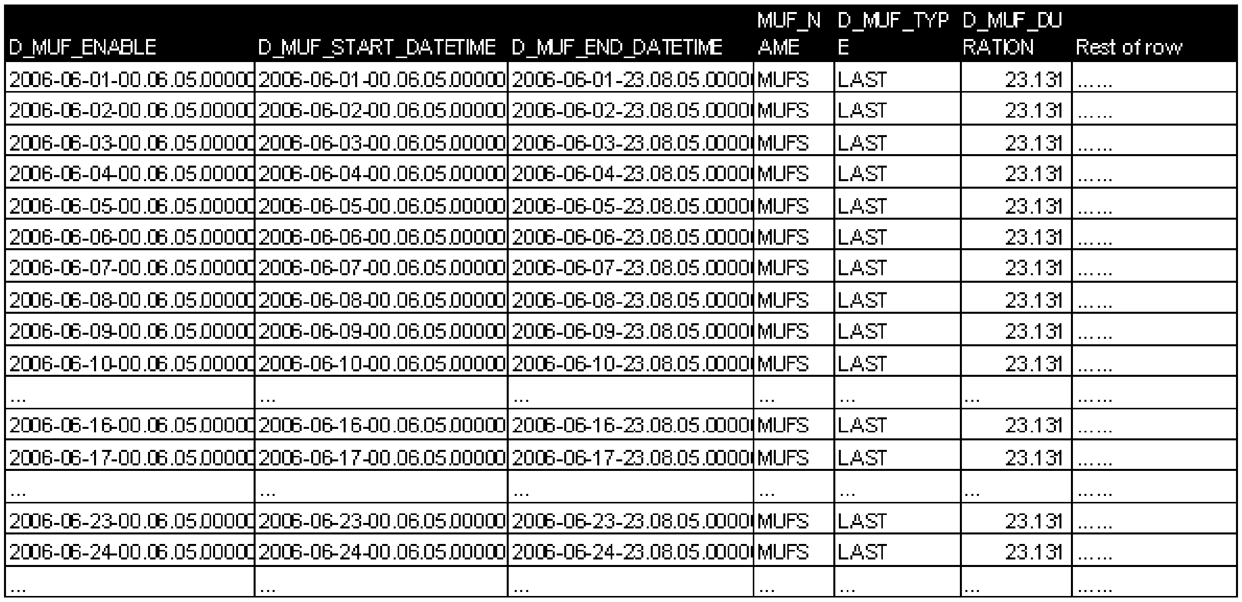
The following examples explain the use of Summary rowsets:

SUMMARY Function Example 1

In this example, the user has an automated process that creates one Snapshot rowset just before MUF is cycled every night. The user executes the AUTOCOLL OPTION=DELTACRE function and the corresponding Delta rowsets have been created.

However, for a given month, there are 30 Delta rowsets which would be too much data to look at and the time slices are too small to easily detect trends. It would be easier if the data was represented as weekly statistics.

Delta Rowsets



D_MUF_ENABLE	D_MUF_START_DATETIME	D_MUF_END_DATETIME	MUF_NAME	D_MUF_TYPE	D_MUF_DURATION	Rest of row
2006-06-01-00.06.05.00000	2006-06-01-00.06.05.00000	2006-06-01-23.08.05.0000	MUFS	LAST	23.131	……
2006-06-02-00.06.05.00000	2006-06-02-00.06.05.00000	2006-06-02-23.08.05.0000	MUFS	LAST	23.131	……
2006-06-03-00.06.05.00000	2006-06-03-00.06.05.00000	2006-06-03-23.08.05.0000	MUFS	LAST	23.131	……
2006-06-04-00.06.05.00000	2006-06-04-00.06.05.00000	2006-06-04-23.08.05.0000	MUFS	LAST	23.131	……
2006-06-05-00.06.05.00000	2006-06-05-00.06.05.00000	2006-06-05-23.08.05.0000	MUFS	LAST	23.131	……
2006-06-06-00.06.05.00000	2006-06-06-00.06.05.00000	2006-06-06-23.08.05.0000	MUFS	LAST	23.131	……
2006-06-07-00.06.05.00000	2006-06-07-00.06.05.00000	2006-06-07-23.08.05.0000	MUFS	LAST	23.131	……
2006-06-08-00.06.05.00000	2006-06-08-00.06.05.00000	2006-06-08-23.08.05.0000	MUFS	LAST	23.131	……
2006-06-09-00.06.05.00000	2006-06-09-00.06.05.00000	2006-06-09-23.08.05.0000	MUFS	LAST	23.131	……
2006-06-10-00.06.05.00000	2006-06-10-00.06.05.00000	2006-06-10-23.08.05.0000	MUFS	LAST	23.131	……
…	…	…	…	…	…	……
2006-06-16-00.06.05.00000	2006-06-16-00.06.05.00000	2006-06-16-23.08.05.0000	MUFS	LAST	23.131	……
2006-06-17-00.06.05.00000	2006-06-17-00.06.05.00000	2006-06-17-23.08.05.0000	MUFS	LAST	23.131	……
…	…	…	…	…	…	……
2006-06-23-00.06.05.00000	2006-06-23-00.06.05.00000	2006-06-23-23.08.05.0000	MUFS	LAST	23.131	……
2006-06-24-00.06.05.00000	2006-06-24-00.06.05.00000	2006-06-24-23.08.05.0000	MUFS	LAST	23.131	……
…	…	…	…	…	…	……

In this case, the user would like to have the data summarized into four weeks of data: weeks ending 6/9, 6/16, 6/23, and 6/30.

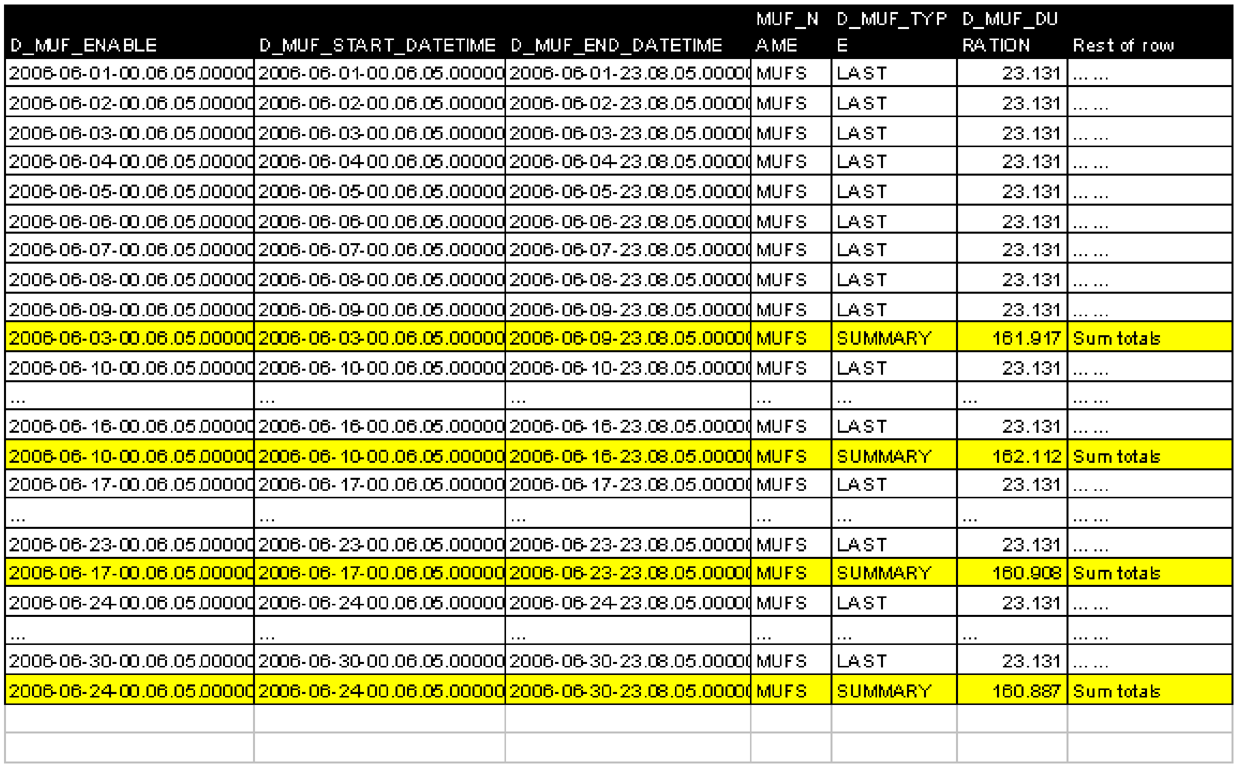
By running one AUTOCOLL OPTION=SUMMARY for each week, the user can create four separate Summary rowsets, providing the weekly totals.

The following are the execution parameters for the four separate OPTION=SUMMARY functions:

	Summary for Week 1	**Summary for Week 2**	**Summary for Week 3**	**Summary for Week 4**
OPTION	SUMMARY	SUMMARY	SUMMARY	SUMMARY
MUFNAME	MUFS	MUFS	MUFS	MUFS
TYPE	LAST	LAST	LAST	LAST
DATERG	20060603 20060609	20060610 20060616	20060617 20060623	20060624 20060630
TIMERG	000000 240000	000000 240000	000000 240000	000000 240000

After the fourth function completes, the Delta database now contains four Summary rowsets. Each rowset represents the sum of the selected LAST rowsets. The SUMMARY function does not remove or delete any information being summarized.

Equation 1: Example of Creating Delta Summary Rowsets (SUMMARY Function with 4 rowsets)



D_MUF_ENABLE	D_MUF_START_DATETIME	D_MUF_END_DATETIME	MUF_NAME	D_MUF_TYPE	D_MUF_DURATION	Rest of row
2006-06-01-00.06.05.00000	2006-06-01-00.06.05.00000	2006-06-01-23.08.05.0000	MUFS	LAST	23.131	... ...
2006-06-02-00.06.05.00000	2006-06-02-00.06.05.00000	2006-06-02-23.08.05.0000	MUFS	LAST	23.131	... ...
2006-06-03-00.06.05.00000	2006-06-03-00.06.05.00000	2006-06-03-23.08.05.0000	MUFS	LAST	23.131	... ...
2006-06-04-00.06.05.00000	2006-06-04-00.06.05.00000	2006-06-04-23.08.05.0000	MUFS	LAST	23.131	... ...
2006-06-05-00.06.05.00000	2006-06-05-00.06.05.00000	2006-06-05-23.08.05.0000	MUFS	LAST	23.131	... ...
2006-06-06-00.06.05.00000	2006-06-06-00.06.05.00000	2006-06-06-23.08.05.0000	MUFS	LAST	23.131	... ...
2006-06-07-00.06.05.00000	2006-06-07-00.06.05.00000	2006-06-07-23.08.05.0000	MUFS	LAST	23.131	... ...
2006-06-08-00.06.05.00000	2006-06-08-00.06.05.00000	2006-06-08-23.08.05.0000	MUFS	LAST	23.131	... ...
2006-06-09-00.06.05.00000	2006-06-09-00.06.05.00000	2006-06-09-23.08.05.0000	MUFS	LAST	23.131	... ...
2006-06-03-00.06.05.00000	2006-06-03-00.06.05.00000	2006-06-09-23.08.05.0000	MUFS	SUMMARY	161.917	Sum totals
2006-06-10-00.06.05.00000	2006-06-10-00.06.05.00000	2006-06-10-23.08.05.0000	MUFS	LAST	23.131	... ...
...	...	...	...	...	...	... ...
2006-06-16-00.06.05.00000	2006-06-16-00.06.05.00000	2006-06-16-23.08.05.0000	MUFS	LAST	23.131	... ...
2006-06-10-00.06.05.00000	2006-06-10-00.06.05.00000	2006-06-16-23.08.05.0000	MUFS	SUMMARY	162.112	Sum totals
2006-06-17-00.06.05.00000	2006-06-17-00.06.05.00000	2006-06-17-23.08.05.0000	MUFS	LAST	23.131	... ...
...	...	...	...	...	...	... ...
2006-06-23-00.06.05.00000	2006-06-23-00.06.05.00000	2006-06-23-23.08.05.0000	MUFS	LAST	23.131	... ...
2006-06-17-00.06.05.00000	2006-06-17-00.06.05.00000	2006-06-23-23.08.05.0000	MUFS	SUMMARY	160.908	Sum totals
2006-06-24-00.06.05.00000	2006-06-24-00.06.05.00000	2006-06-24-23.08.05.0000	MUFS	LAST	23.131	... ...
...	...	...	...	...	...	... ...
2006-06-30-00.06.05.00000	2006-06-30-00.06.05.00000	2006-06-30-23.08.05.0000	MUFS	LAST	23.131	... ...
2006-06-24-00.06.05.00000	2006-06-24-00.06.05.00000	2006-06-30-23.08.05.0000	MUFS	SUMMARY	160.887	Sum totals

Using AUTOCOLL OPTION=DSVOUT or another available tool, the user can easily select just the summary rowsets for processing into a weekly statistical report.

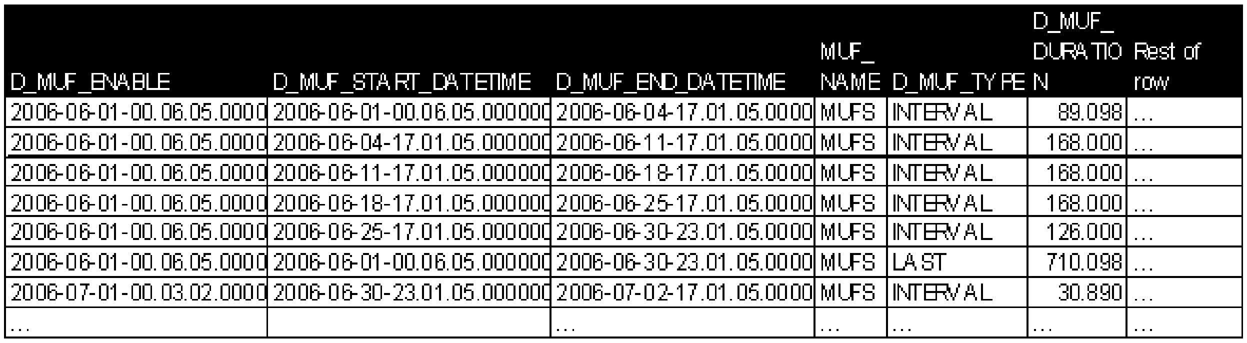
SUMMARY Function Example 2

A user has a source MUF that comes down once a month on the last day of the month. For four weeks of the month, they have one Delta rowset INTERVAL that was created on Sunday at 5 PM, but for the end of the month week, they have two rowsets from different MUF executions that represent the *weekly* period.

In this example, rather than summarizing that one week where there are two rowsets, it would be simpler for later reporting and selection purposes for the user to build one Summary rowset for each week, even though it would be the same as the INTERVAL rowset for three of the four weeks.

Note: In this example, the user excluded the LAST rowset to avoid creating a summary with data counted twice.

Delta Rowsets:



D_MUF_ENABLE	D_MUF_START_DATETIME	D_MUF_END_DATETIME	MUF_ NAME	D_MUF_TYPE	D_MUF_ DURATION	Rest of row
2006-06-01-00.06.05.0000	2006-06-01-00.06.05.000000	2006-06-04-17.01.05.0000	MUFS	INTERVAL	89.098	…
2006-06-01-00.06.05.0000	2006-06-04-17.01.05.000000	2006-06-11-17.01.05.0000	MUFS	INTERVAL	168.000	…
2006-06-01-00.06.05.0000	2006-06-11-17.01.05.000000	2006-06-18-17.01.05.0000	MUFS	INTERVAL	168.000	…
2006-06-01-00.06.05.0000	2006-06-18-17.01.05.000000	2006-06-25-17.01.05.0000	MUFS	INTERVAL	168.000	…
2006-06-01-00.06.05.0000	2006-06-25-17.01.05.000000	2006-06-30-23.01.05.0000	MUFS	INTERVAL	126.000	…
2006-06-01-00.06.05.0000	2006-06-01-00.06.05.000000	2006-06-30-23.01.05.0000	MUFS	LAST	710.098	…
2006-07-01-00.03.02.0000	2006-06-30-23.01.05.000000	2006-07-02-17.01.05.0000	MUFS	INTERVAL	30.890	…
…		…	…	…	…	…

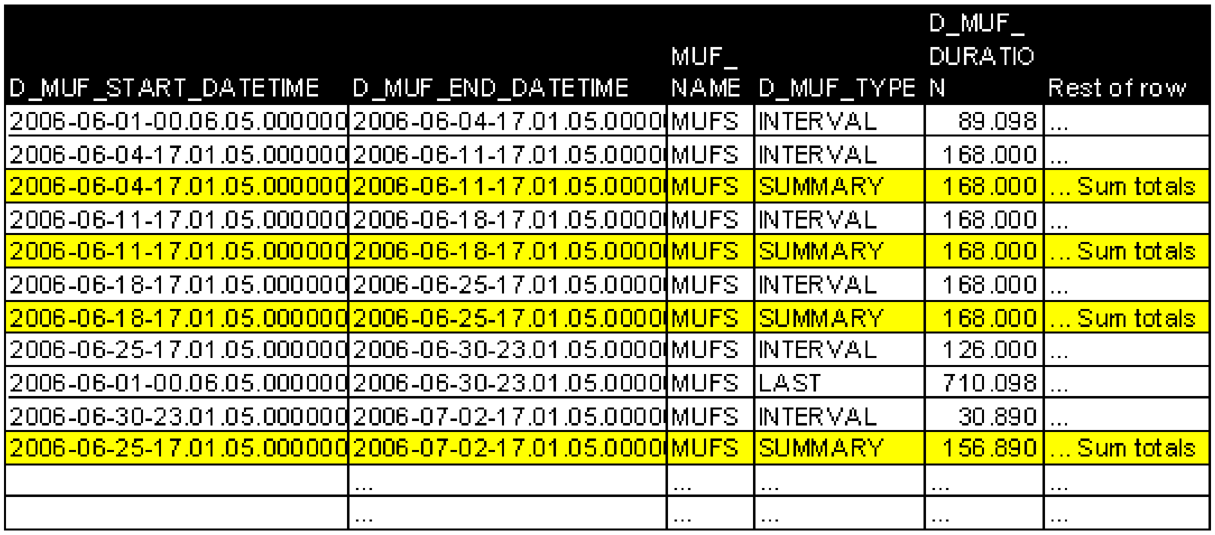
For this example, the user would like to have the data summarized into four weeks of data: weeks ending on 6/11, 6/18, 6/25, and 7/2.

Running one summary pass for each week, we could create four equal weekly Summary rowsets.

The following are the execution parameters:

	Summary for Week 1	**Summary for Week 2**	**Summary for Week 3**	**Summary for Week 4**
OPTION	SUMMARY	SUMMARY	SUMMARY	SUMMARY
MUFNAME	MUFS	MUFS	MUFS	MUFS
TYPE	INTERVAL	INTERVAL	INTERVAL	INTERVAL
DATERG	20060604 20060611	20060611 20060618	20060618 20060625	20060625 20060702
TIMERG	000000 240000	000000 240000	000000 240000	000000 240000

After the fourth execution, the Delta rowsets would now contain four Summary rowsets. Each Summary rowset would represent the sum and totals of the selected rows. The SUMMARY function does not remove or delete any of the information being summarized.



D_MUF_START_DATETIME	D_MUF_END_DATETIME	MUF_ NAME	D_MUF_TYPE	D_MUF_ DURATIO N	Rest of row
2006-06-01-00.06.05.000000	2006-06-04-17.01.05.0000	MUFS	INTERVAL	89.098	...
2006-06-04-17.01.05.000000	2006-06-11-17.01.05.0000	MUFS	INTERVAL	168.000	...
2006-06-04-17.01.05.000000	2006-06-11-17.01.05.0000	MUFS	SUMMARY	168.000	... Sum totals
2006-06-11-17.01.05.000000	2006-06-18-17.01.05.0000	MUFS	INTERVAL	168.000	...
2006-06-11-17.01.05.000000	2006-06-18-17.01.05.0000	MUFS	SUMMARY	168.000	... Sum totals
2006-06-18-17.01.05.000000	2006-06-25-17.01.05.0000	MUFS	INTERVAL	168.000	...
2006-06-18-17.01.05.000000	2006-06-25-17.01.05.0000	MUFS	SUMMARY	168.000	... Sum totals
2006-06-25-17.01.05.000000	2006-06-30-23.01.05.0000	MUFS	INTERVAL	126.000	...
2006-06-01-00.06.05.000000	2006-06-30-23.01.05.0000	MUFS	LAST	710.098	...
2006-06-30-23.01.05.000000	2006-07-02-17.01.05.0000	MUFS	INTERVAL	30.890	...
2006-06-25-17.01.05.000000	2006-07-02-17.01.05.0000	MUFS	SUMMARY	156.890	... Sum totals
	...	...	...	...	...
	...	...	...	...	...

Using AUTOCOLL OPTION=DSVOUT or another available tool, the user can now easily select just the summary rows for processing into a weekly statistical report.

Creating Delta Baseline Rowsets

AutoCollect also provides a facility to create Baseline rowsets. Baseline rowsets provide a different function from Summary rowsets.

As you were collecting and displaying data for performance monitoring, you have seen that it is useful to have a performance baseline to compare results.

In the preceding examples, we saw how a user could use the SUMMARY function to combine various Delta rowsets into weekly Summary rowsets for week-to-week comparison.

Once you have built a number of Delta rowsets that represent the summary of a weekly processing for normal periods. You may want to create a baseline of weekly performance to use for future comparisons. A baseline is typically an average of a set of similar time periods.

In this case, the baseline would represent a weekly baseline.

Note: When creating a baseline, the rows that are selected to be averaged must represent like periods of time. Selecting a set of Delta rowsets representing unequal periods of time may create a baseline that is not truly representative.

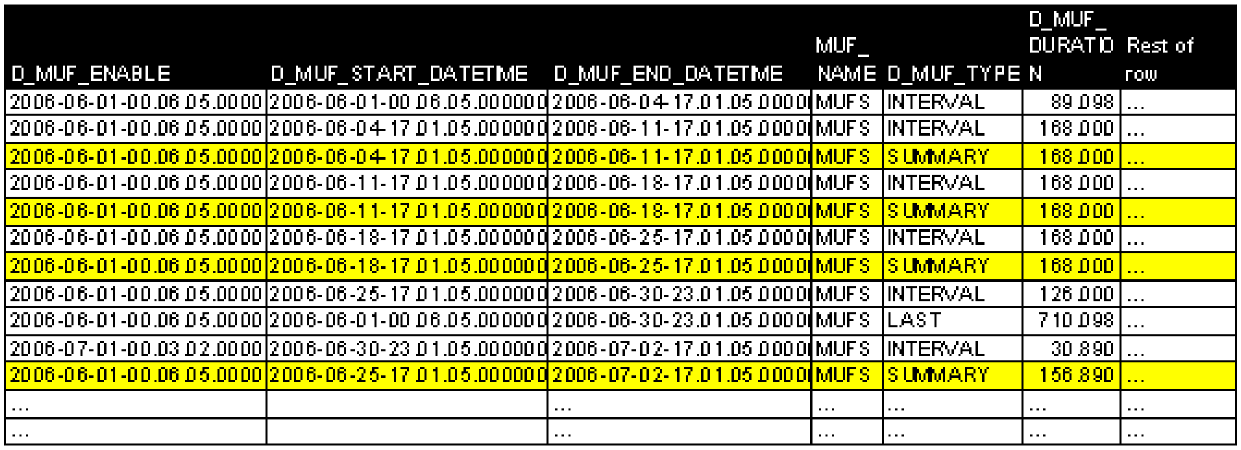
Once a baseline is created, it can be compared to new weekly statistics as they are created.

When doing performance tuning, making changes to the MUF, it is important to compare the performance after the change to the baseline of the performance from before the change.

In a different case, you may want to track the weekly performance, a baseline, from one month to the next month, or even one year to the next year. Summary and baseline rowsets can provide these functions without significant user interaction.

BASELINE Function Example

In the preceding SUMMARY Example 2, summary records were constructed for each of the four weeks in June, ending on July 2.

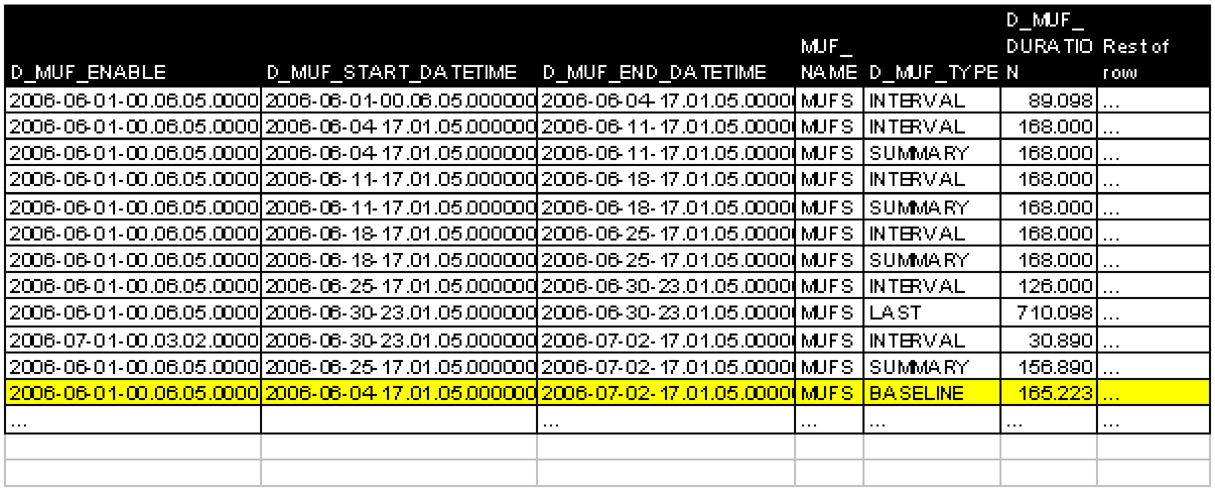


D_MUF_ENABLE	D_MUF_START_DATETIME	D_MUF_END_DATETIME	MUF_ NAME	D_MUF_TYPE	D_MUF_ DURATION	Rest of row
2006-06-01-00.06.05.0000	2006-06-01-00.06.05.000000	2006-06-04-17.01.05.00000	MUFS	INTERVAL	89.098	...
2006-06-01-00.06.05.0000	2006-06-04-17.01.05.000000	2006-06-11-17.01.05.00000	MUFS	INTERVAL	168.000	...
2006-06-01-00.06.05.0000	2006-06-04-17.01.05.000000	2006-06-11-17.01.05.00000	MUFS	SUMMARY	168.000	...
2006-06-01-00.06.05.0000	2006-06-11-17.01.05.000000	2006-06-18-17.01.05.00000	MUFS	INTERVAL	168.000	...
2006-06-01-00.06.05.0000	2006-06-11-17.01.05.000000	2006-06-18-17.01.05.00000	MUFS	SUMMARY	168.000	...
2006-06-01-00.06.05.0000	2006-06-18-17.01.05.000000	2006-06-25-17.01.05.00000	MUFS	INTERVAL	168.000	...
2006-06-01-00.06.05.0000	2006-06-18-17.01.05.000000	2006-06-25-17.01.05.00000	MUFS	SUMMARY	168.000	...
2006-06-01-00.06.05.0000	2006-06-25-17.01.05.000000	2006-06-30-23.01.05.00000	MUFS	INTERVAL	126.000	...
2006-06-01-00.06.05.0000	2006-06-01-00.06.05.000000	2006-06-30-23.01.05.00000	MUFS	LAST	710.098	...
2006-07-01-00.03.02.0000	2006-06-30-23.01.05.000000	2006-07-02-17.01.05.00000	MUFS	INTERVAL	30.890	...
2006-06-01-00.06.05.0000	2006-06-25-17.01.05.000000	2006-07-02-17.01.05.00000	MUFS	SUMMARY	156.890	...
...		...	...	...	...	...
...		...	...	...	...	...

We can create a baseline rowset for the four weeks using the BASELINE function.

Name	Value
OPTION	BASELINE
MUFNAME	MUFS
TYPE	SUMMARY
DATERG	2006060420060702
TIMERG	000000240000

After completing the BASELINE run, you would have a new rowset inserted into the Delta tables.



D_MUF_ENABLE	D_MUF_START_DATETIME	D_MUF_END_DATETIME	MUF_ NAME	D_MUF_TYPE	D_MUF_ DURATIO N	Rest of row
2006-06-01-00.06.05.0000	2006-06-01-00.06.05.000000	2006-06-04-17.01.05.0000	MUFS	INTERVAL	89.098	...
2006-06-01-00.06.05.0000	2006-06-04-17.01.05.000000	2006-06-11-17.01.05.0000	MUFS	INTERVAL	168.000	...
2006-06-01-00.06.05.0000	2006-06-04-17.01.05.000000	2006-06-11-17.01.05.0000	MUFS	SUMMARY	168.000	...
2006-06-01-00.06.05.0000	2006-06-11-17.01.05.000000	2006-06-18-17.01.05.0000	MUFS	INTERVAL	168.000	...
2006-06-01-00.06.05.0000	2006-06-11-17.01.05.000000	2006-06-18-17.01.05.0000	MUFS	SUMMARY	168.000	...
2006-06-01-00.06.05.0000	2006-06-18-17.01.05.000000	2006-06-25-17.01.05.0000	MUFS	INTERVAL	168.000	...
2006-06-01-00.06.05.0000	2006-06-18-17.01.05.000000	2006-06-25-17.01.05.0000	MUFS	SUMMARY	168.000	...
2006-06-01-00.06.05.0000	2006-06-25-17.01.05.000000	2006-06-30-23.01.05.0000	MUFS	INTERVAL	126.000	...
2006-06-01-00.06.05.0000	2006-06-30-23.01.05.000000	2006-06-30-23.01.05.0000	MUFS	LAST	710.098	...
2006-07-01-00.03.02.0000	2006-06-30-23.01.05.000000	2006-07-02-17.01.05.0000	MUFS	INTERVAL	30.890	...
2006-06-01-00.06.05.0000	2006-06-25-17.01.05.000000	2006-07-02-17.01.05.0000	MUFS	SUMMARY	156.890	...
2006-06-01-00.06.05.0000	2006-06-04-17.01.05.000000	2006-07-02-17.01.05.0000	MUFS	BASELINE	165.223	...
...		...	...	...	...	...

Creating AVGPERF Rowsets

AVGPERF rowsets provide a processing average based on a standard duration of one hour. Each value is calculated on the amount of work done divided by the total duration hours.

AVGPERF records provide a specific function and only be created when trying to compare average performance from two different durations or work periods.

Note: AVGPERF records should not be generated from data that represents a small duration period (less than an hour). For example, when building an AVGPERF from a duration of .001 hours, the work values would be divided by one one-thousandth which actually results in the values being multiplied by a 1000. With such a large multiplication value, the relative performance values have significant variance possibilities.

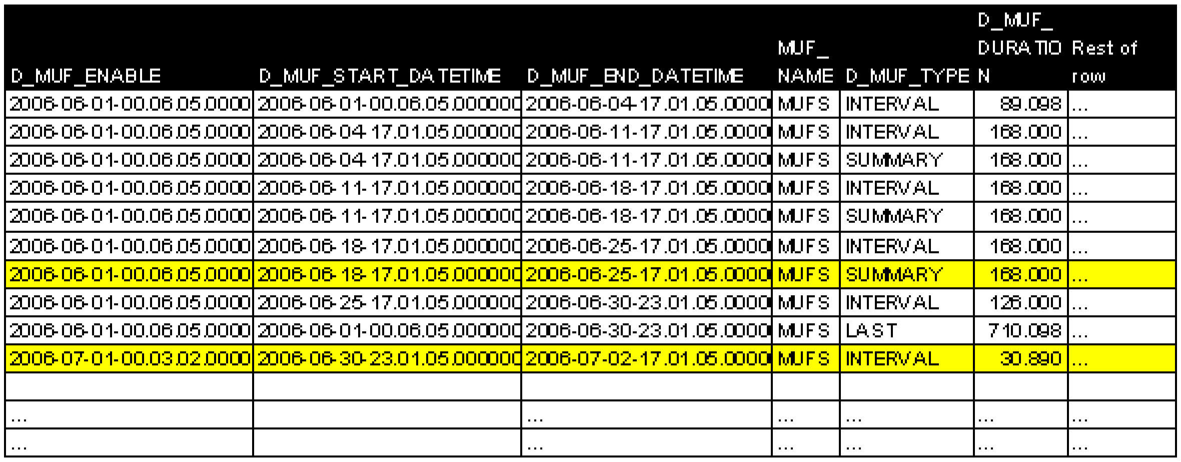
After you have collected and displayed data for performance monitoring from varying duration lengths, it may be useful to have average performance AVGPERF rowsets.

In the examples, we saw how you could use the SUMMARY function to combine various Delta rowsets into weekly Summary rowset for week-to-week comparison.

In the following example, we build average performance rowsets from a two day period versus the average performance from a previous week.

AVGPERF Function Example

In the previous SUMMARY Function Example 2, we had Summary records constructed for each of the four weeks in June and various INTERVAL and LAST rowsets.

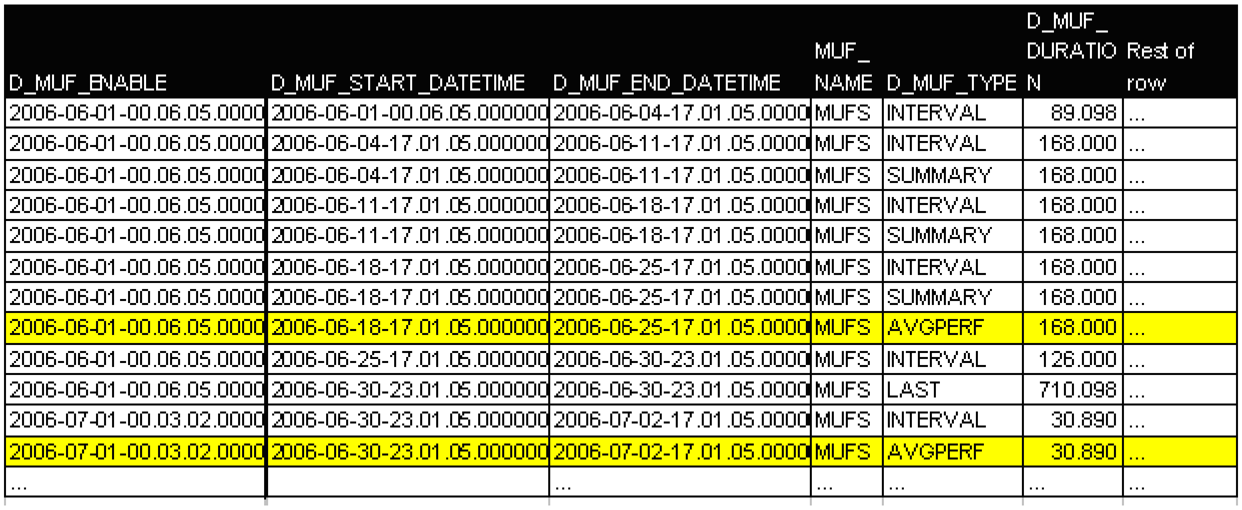


D_MUF_ENABLE	D_MUF_START_DATETIME	D_MUF_END_DATETIME	MUF_ NAME	D_MUF_TYPE	D_MUF_ DURATIO N	Rest of row
2006-06-01-00.06.05.0000	2006-06-01-00.06.05.000000	2006-06-04-17.01.05.0000	MUFS	INTERVAL	89.098	...
2006-06-01-00.06.05.0000	2006-06-04 17.01.05.000000	2006-06-11-17.01.05.0000	MUFS	INTERVAL	168.000	...
2006-06-01-00.06.05.0000	2006-06-04 17.01.05.000000	2006-06-11-17.01.05.0000	MUFS	SUMMARY	168.000	...
2006-06-01-00.06.05.0000	2006-06-11-17.01.05.000000	2006-06-18-17.01.05.0000	MUFS	INTERVAL	168.000	...
2006-06-01-00.06.05.0000	2006-06-11-17.01.05.000000	2006-06-18-17.01.05.0000	MUFS	SUMMARY	168.000	...
2006-06-01-00.06.05.0000	2006-06-18-17.01.05.000000	2006-06-25-17.01.05.0000	MUFS	INTERVAL	168.000	...
2006-06-01-00.06.05.0000	2006-06-18-17.01.05.000000	2006-06-25-17.01.05.0000	MUFS	SUMMARY	168.000	...
2006-06-01-00.06.05.0000	2006-06-25-17.01.05.000000	2006-06-30-23.01.05.0000	MUFS	INTERVAL	126.000	...
2006-06-01-00.06.05.0000	2006-06-01-00.06.05.000000	2006-06-30-23.01.05.0000	MUFS	LAST	710.098	...
2006-07-01-00.03.02.0000	2006-06-30-23.01.05.000000	2006-07-02-17.01.05.0000	MUFS	INTERVAL	30.890	...
...		...	...	...	...	...
...		...	...	...	...	...

We can create an AVGPERF rowset for the Summary rowset from June 25 and the INTERVAL ending on July 2.

	AVGPERF for 0625 Summary	**AVGPERF for 0702 Summary**
OPTION	AVERGPERF	AVERGPERF
MUFNAME	MUFS	MUFS
TYPE	SUMMARY	INTERVAL
DATERG	2006062520060625	2006070220060702
TIMERG	160000185900	160000185900

After completing the AVGPERF runs, you would have two new rowsets inserted into the Delta tables.



D_MUF_ENABLE	D_MUF_START_DATETIME	D_MUF_END_DATETIME	MUF_ NAME	D_MUF_TYPE	D_MUF_ DURATIO N	Rest of row
2006-06-01-00.06.05.0000	2006-06-01-00.06.05.000000	2006-06-04-17.01.05.0000	MUFS	INTERVAL	89.098	...
2006-06-01-00.06.05.0000	2006-06-04-17.01.05.000000	2006-06-11-17.01.05.0000	MUFS	INTERVAL	168.000	...
2006-06-01-00.06.05.0000	2006-06-04-17.01.05.000000	2006-06-11-17.01.05.0000	MUFS	SUMMARY	168.000	...
2006-06-01-00.06.05.0000	2006-06-11-17.01.05.000000	2006-06-18-17.01.05.0000	MUFS	INTERVAL	168.000	...
2006-06-01-00.06.05.0000	2006-06-11-17.01.05.000000	2006-06-18-17.01.05.0000	MUFS	SUMMARY	168.000	...
2006-06-01-00.06.05.0000	2006-06-18-17.01.05.000000	2006-06-25-17.01.05.0000	MUFS	INTERVAL	168.000	...
2006-06-01-00.06.05.0000	2006-06-18-17.01.05.000000	2006-06-25-17.01.05.0000	MUFS	SUMMARY	168.000	...
2006-06-01-00.06.05.0000	2006-06-18-17.01.05.000000	2006-06-25-17.01.05.0000	MUFS	AVGPERF	168.000	...
2006-06-01-00.06.05.0000	2006-06-25-17.01.05.000000	2006-06-30-23.01.05.0000	MUFS	INTERVAL	126.000	...
2006-06-01-00.06.05.0000	2006-06-30-23.01.05.000000	2006-06-30-23.01.05.0000	MUFS	LAST	710.098	...
2006-07-01-00.03.02.0000	2006-06-30-23.01.05.000000	2006-07-02-17.01.05.0000	MUFS	INTERVAL	30.890	...
2006-07-01-00.03.02.0000	2006-06-30-23.01.05.000000	2006-07-02-17.01.05.0000	MUFS	AVGPERF	30.890	...
...		...	...	...	...	...

Reporting Rowsets

As needed, a Delta rowset report can be created by executing AUTOCOLL OPTION=DELTARPT. The DELTARPT report provides a list of each of the rowsets and the number of rows in each Delta table that is part of that rowset. The Delta rowset report includes SUMMARY, AVGPERF, and BASELINE function rowsets.

Removing Rowsets

As needed, a Delta rowset can be deleted using the AUTOCOLL OPTION=DELTADEL. In some cases, you may want to rebuild the Summary, AVGPERF and Baseline Delta rowsets. In these cases, you would delete the Delta Summary, AVGPERF, or Baseline rowsets before re executing the functions to rebuild the Delta rowsets.

DELTADEL provides various selection parameters to limit which rowsets are deleted. Before running DELTADEL, you, should run a DELTARPT using the same selection criteria to verify that only the unneeded rowsets are deleted. As a precaution you may also decide to back up the Delta database (1020) before running the delete.

Extracting Delta Rowsets into DSV Output Datasets

You can select Delta rowsets based on various selection criteria and output the rowsets' content as a Delimiter Separated Value (DSV) file, then use available FTP protocols to download the sequential output files from the mainframe to the PC as text files. These text files can be imported into a PC-based spreadsheet product such as Microsoft Excel.

During the OPTION=DSVOUT execution, as the utility processes each of the Delta tables, it attempts to open a matching sequential dataset for each of the tables. The sequential dataset DDname is specified as:

```
OUTxxx
```

The *xxx* is the three character table name in the Delta database.

If the DDNAME is missing from the JCL, the printed report indicates that no output dataset could be opened and skips the DSV processing for that table. The utility attempts to process all 17 tables generating output for those datasets that are defined.

To allow you to place output from multiple DSVOUT functions into a single set of OUT*xxx* sequential files, you can use the DISP=(MOD...) JCL parameter. MOD allows each DSVOUT function to add its selected output to the "end" of the OUT*xxx* datasets.

Note: Before running the DBUTLTY OPTION=DSVOUT function with the DISP=MOD JCL, remove any unwanted data in the OUT*xxx* datasets or delete them entirely.

Two Delta rowset tables that provide immediate benefit to your site for the MUF performance analysis are the Business Value Metrics (BVM) and Business Value Details (BVD) tables. These tables provide key performance measurement ratios.

Once loaded into a spreadsheet, you can perform a variety of editing, sorting, and summarization tasks to manipulate the data into a usable form for your site.

Selection Parameters

For the DSVOUT function, you can use the following parameters to select the Delta rowsets used for input to the creation of the sequential output file.

MUFNAME=mufname

Specifies to restrict processing to Delta rowsets from this selected MUF only. Therefore, to select rowsets for more than one MUF, execute this function once for each MUF you want to target.

TYPE=type

Specifies to restrict the Delta rowsets that you want selected as input as follows:

AVGPERF

Selects only Delta rowsets built by the AUTOCOLL OPTION=AVGPERF function

BASELINE

Selects only Delta rowsets built by the AUTOCOLL OPTION=BASELINE function

INTERVAL

Selects only Interval Delta rowsets built by the AUTOCOLL OPTION=DELTACRE function

LAST

Selects only Last Delta rowsets built by the AUTOCOLL OPTION=DELTACRE function

SUMMARY

Selects only Delta rowsets built by the AUTOCOLL OPTION=DELTACRE function

DATERG

Select only Delta rowsets that have their D_MUF_END_DATETIME date value within the range of dates specified in this parameter.

The format of the entry is ccyymmddccyymmdd where the first 8 bytes are the starting date and the last 8 bytes are the ending date. The starting and ending date can be in the range of 19000101 to 39991231. The ending date must be equal to or greater than the starting date. If you do not specify a date range, the date value is not used when selecting a rowset.

TIMERG

Select only Delta rowsets that have their D_MUF_END_DATETIME time value within the range of times specified in this parameter.

The format of the entry is hhmmsshhmmss where the first 6 bytes are the starting time and the last 6 bytes are the ending time. The starting and ending time can be in the range of 000000 to 240000. The ending time must be equal to or greater than the starting time. If you do not specify a time range, the time value is not used when selecting a rowset.

USERTAG

Select only Delta rowsets that have this value in their USERTAG column. If you do not specify a USERTAG, the USERTAG value is not used when selecting a rowset.

Parameter Specifications

DECIMAL

Specifies the character that is used to designate the decimal point in DSV column output. It cannot be the same as the delimiter value. Suggested value for USA is a period (.), for Europe, it is a comma (,).

DELIMITER

Specifies the character that is used to separate columns within the DSV output. Only one character can be chosen for all DSV output generated in a given DSVOUT function execution. Choose a character value that would not normally occur in the DST, Snapshot, and Delta rows. Suggested value is a semi-colon (;).

HEADER

Specifying YES (default) tells the DSVOUT function to produce a column header as the first row in the DSV output. NO indicates that the column headers are to be suppressed and only data values are written to the DSV output files.

The expected return code is 0 when executing the DSVOUT function. For details on the AUTOCOLL OPTION=DSVOUT syntax, parameters, JCL and sample output report see the *CA Datacom/DB DBUTLTY Reference Guide*.

Sample DSV Output

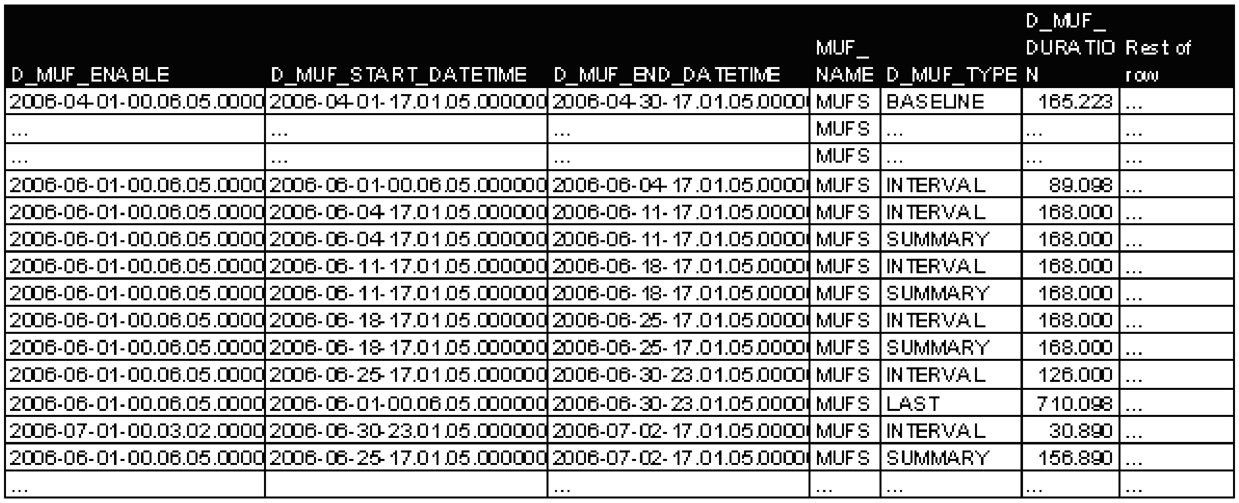
Each of the DSV output datasets are created and saved according to your JCL.

The output file is created with the column headers across the top (1st row) and the data rows going down (2nd row through nnnnn).

```
MUF_ENABLE;  MUF_START_DATETIME;  MUF_END_DATETIME;   MUF_TYPE;  MUF_DURATION;  MUF_NAME;…
20060727 19.25.47.000000;20060727 19.25.47.000000;20060727 19.32.08.027611;LAST;0.1058;MUFS;…
20060727 19.25.47.000000;20060727 19.25.47.000000;20060728 01.16.35.084569;LAST;5.8466;MUFS;…
20060728 01.17.56.000000;20060728 01.17.56.000000;20060728 01.20.30.629424;LAST;0.0428;MUFS;…
```

Example 1 – Creating DSVOUT from Summary Deltas

The following is a sample of Delta rowsets. The user would like to build a spreadsheet from the weekly summary records in this example.



D_MUF_ENABLE	D_MUF_START_DATETIME	D_MUF_END_DATETIME	MUF_NAME	D_MUF_TYPE	D_MUF_DURATION	Rest of row
2006-04-01-00.06.05.0000	2006-04-01-17.01.05.000000	2006-04-30-17.01.05.0000	MUFS	BASELINE	165.223	...
...	...	...	MUFS	...	...	...
...	...	...	MUFS	...	...	...
2006-06-01-00.06.05.0000	2006-06-01-00.06.05.000000	2006-06-04-17.01.05.0000	MUFS	INTERVAL	89.098	...
2006-06-01-00.06.05.0000	2006-06-04-17.01.05.000000	2006-06-11-17.01.05.0000	MUFS	INTERVAL	168.000	...
2006-06-01-00.06.05.0000	2006-06-04-17.01.05.000000	2006-06-11-17.01.05.0000	MUFS	SUMMARY	168.000	...
2006-06-01-00.06.05.0000	2006-06-11-17.01.05.000000	2006-06-18-17.01.05.0000	MUFS	INTERVAL	168.000	...
2006-06-01-00.06.05.0000	2006-06-11-17.01.05.000000	2006-06-18-17.01.05.0000	MUFS	SUMMARY	168.000	...
2006-06-01-00.06.05.0000	2006-06-18-17.01.05.000000	2006-06-25-17.01.05.0000	MUFS	INTERVAL	168.000	...
2006-06-01-00.06.05.0000	2006-06-18-17.01.05.000000	2006-06-25-17.01.05.0000	MUFS	SUMMARY	168.000	...
2006-06-01-00.06.05.0000	2006-06-25-17.01.05.000000	2006-06-30-23.01.05.0000	MUFS	INTERVAL	126.000	...
2006-06-01-00.06.05.0000	2006-06-01-00.06.05.000000	2006-06-30-23.01.05.0000	MUFS	LAST	710.098	...
2006-07-01-00.03.02.0000	2006-06-30-23.01.05.000000	2006-07-02-17.01.05.0000	MUFS	INTERVAL	30.890	...
2006-06-01-00.06.05.0000	2006-06-25-17.01.05.000000	2006-07-02-17.01.05.0000	MUFS	SUMMARY	156.890	...
...		...	...	...	...	...

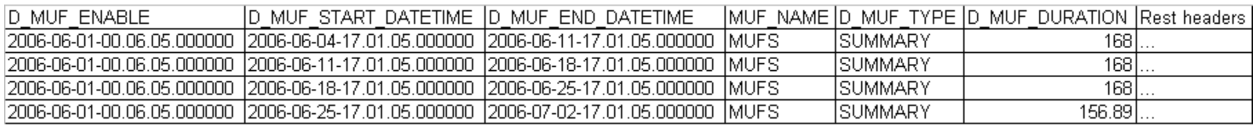
To build DSV output using the four summary records from June. The following are the parameters specified:

Parameter	**Value**
DELIMITER	";" (semicolon)
DECIMAL	"." (period)
MUFNAME	MFS
HEADER	YES
TYPE	SUMMARY
DATERG	2006061120060702
TIMERG	000000240000

The DSV output from the detail Delta tables would be as follows:

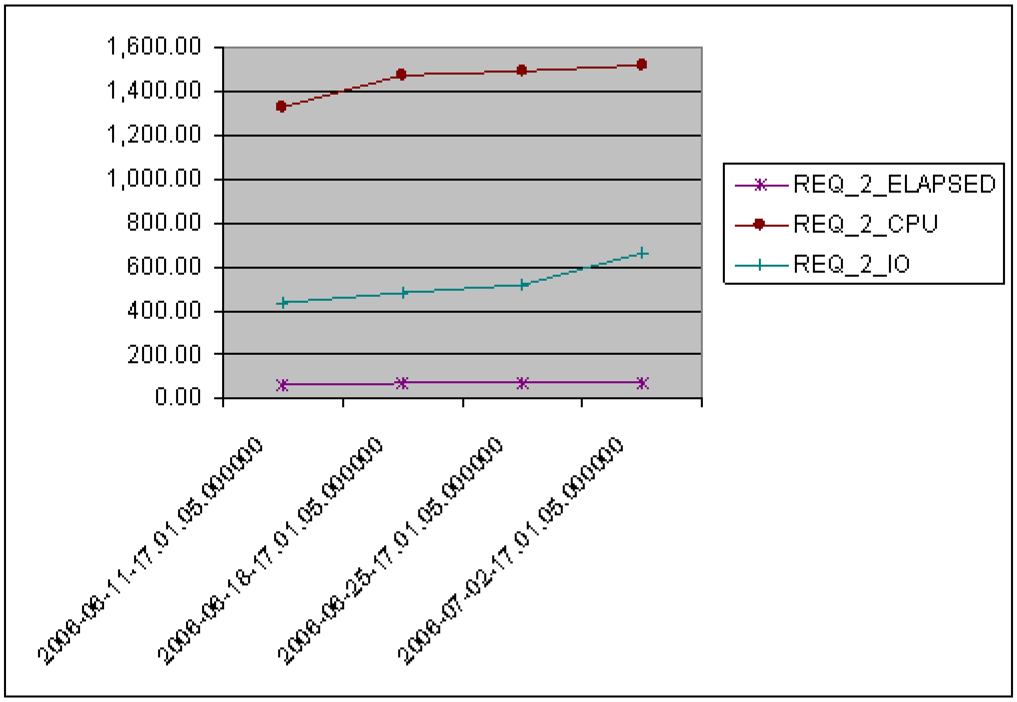
```
D_MUF_ENABLE;D_MUF_START_DATETIME;D_MUF_END_DATETIME;MUF_NAME;D_MUF_TYPE;D_MUF_DURATION;Rest
headers…
2006-06-01-00.06.05.000000;2006-06-04-17.01.05.000000;2006-06-11-17.01.05.000000;MUFS;SUMMARY;168
.000;…
2006-06-01-00.06.05.000000;2006-06-11-17.01.05.000000;2006-06-18-17.01.05.000000;MUFS;SUMMARY;168
.000;…
2006-06-01-00.06.05.000000;2006-06-18-17.01.05.000000;2006-06-25-17.01.05.000000;MUFS;SUMMARY;168
.000;…
2006-06-01-00.06.05.000000;2006-06-25-17.01.05.000000;2006-07-02-17.01.05.000000;MUFS;SUMMARY;156
.890;…
```

The file could then be imported into Excel:



D_MUF_ENABLE	D_MUF_START_DATETIME	D_MUF_END_DATETIME	MUF_NAME	D_MUF_TYPE	D_MUF_DURATION	Rest headers
2006-06-01-00.06.05.000000	2006-06-04-17.01.05.000000	2006-06-11-17.01.05.000000	MUFS	SUMMARY	168	...
2006-06-01-00.06.05.000000	2006-06-11-17.01.05.000000	2006-06-18-17.01.05.000000	MUFS	SUMMARY	168	...
2006-06-01-00.06.05.000000	2006-06-18-17.01.05.000000	2006-06-25-17.01.05.000000	MUFS	SUMMARY	168	...
2006-06-01-00.06.05.000000	2006-06-25-17.01.05.000000	2006-07-02-17.01.05.000000	MUFS	SUMMARY	156.89	...

From here, the user can use many of the functions of Excel to produce spreadsheets and graphs to meet the business' need. Using the preceding spreadsheet, the following graph could be generated.



Example 2 – Creating DSVOUT to compare weekly Summary rowsets with a Baseline rowset

In this example, the user wants to compare the weekly summary records to the performance baseline that was created for the month of April. In this case, the user needs to execute the DSVOUT function once to select the Baseline rowset and then execute the DSVOUT function a second time to pull the Summary rowsets.

Since the output of the DSVOUT function is simple sequential datasets, the output of multiple passes can easily be stored in the same dataset using dataset MOD capabilities.

To make the baseline and summary rowsets appear next to each other, we suppress the header row on the second execution (HEADER NO).

D_MUF_ENABLE	D_MUF_START_DATETIME	D_MUF_END_DATETIME	MUF_ NAME	D_MUF_TYPE	D_MUF_ DURATIO N	Rest of row
2006-04-01-00.06.05.0000	2006-04-01-17.01.05.000000	2006-04-30-17.01.05.0000	MUFS	BASELINE	165.223	...
...	...	...	MUFS	...	...	...
...	...	...	MUFS	...	...	...
2006-06-01-00.06.05.0000	2006-06-01-00.06.05.000000	2006-06-04-17.01.05.0000	MUFS	INTERVAL	89.098	...
2006-06-01-00.06.05.0000	2006-06-04-17.01.05.000000	2006-06-11-17.01.05.0000	MUFS	INTERVAL	168.000	...
2006-06-01-00.06.05.0000	2006-06-04-17.01.05.000000	2006-06-11-17.01.05.0000	MUFS	SUMMARY	168.000	...
2006-06-01-00.06.05.0000	2006-06-11-17.01.05.000000	2006-06-18-17.01.05.0000	MUFS	INTERVAL	168.000	...
2006-06-01-00.06.05.0000	2006-06-11-17.01.05.000000	2006-06-18-17.01.05.0000	MUFS	SUMMARY	168.000	...
2006-06-01-00.06.05.0000	2006-06-18-17.01.05.000000	2006-06-25-17.01.05.0000	MUFS	INTERVAL	168.000	...
2006-06-01-00.06.05.0000	2006-06-18-17.01.05.000000	2006-06-25-17.01.05.0000	MUFS	SUMMARY	168.000	...
2006-06-01-00.06.05.0000	2006-06-25-17.01.05.000000	2006-06-30-23.01.05.0000	MUFS	INTERVAL	126.000	...
2006-06-01-00.06.05.0000	2006-06-01-00.06.05.000000	2006-06-30-23.01.05.0000	MUFS	LAST	710.098	...
2006-07-01-00.03.02.0000	2006-06-30-23.01.05.000000	2006-07-02-17.01.05.0000	MUFS	INTERVAL	30.890	...
2006-06-01-00.06.05.0000	2006-06-25-17.01.05.000000	2006-07-02-17.01.05.0000	MUFS	SUMMARY	156.890	...
...		...	...	...	...	...

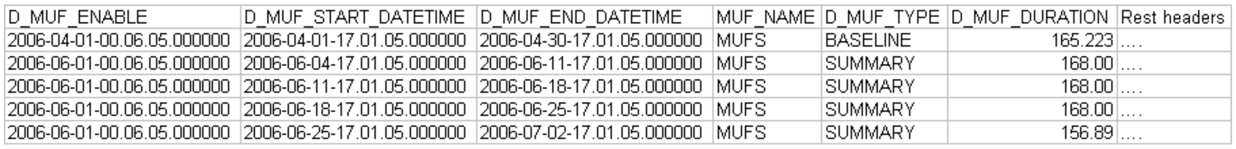
To build DSV output using the baseline rowset and the four summary rowsets, the following are the parameters specified:

Parameter	Value	Parameter	Value
AUTOCOLL	DSVOUT	AUTOCOL	DSVOUT
DELIMITER	";"	DELIMITER	";"
DECIMAL	""	DECIMAL	""
MUFNAME	MFS	MUFNAME	MFS
HEADER	YES	HEADER	NO
TYPE	BASELINE	TYPE	SUMMARY
DATERG	2006043020060430	DATERG	2006061120060702
TIMERG	000000240000	TIMERG	000000240000

The DSV output for all Delta tables would be as follows:

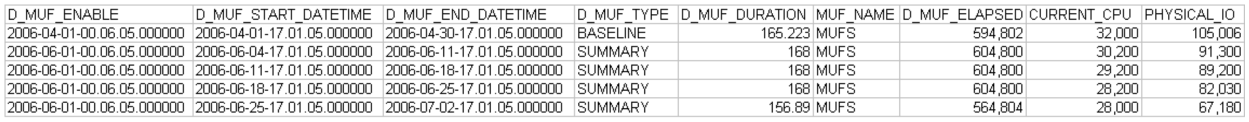
```
D_MUF_ENABLE;D_MUF_START_DATETIME;D_MUF_END_DATETIME;MUF_NAME;D_MUF_TYPE;D_MUF_DURATION;Rest headers
2006-04-01-00.06.05.000000;2006-04-01-17.01.05.000000;2006-04-30-17.01.05.000000;MUFS;BASELINE;165.223;…
2006-06-01-00.06.05.000000;2006-06-04-17.01.05.000000;2006-06-11-17.01.05.000000;MUFS;SUMMARY;168.000;…
2006-06-01-00.06.05.000000;2006-06-11-17.01.05.000000;2006-06-18-17.01.05.000000;MUFS;SUMMARY;168.000;…
2006-06-01-00.06.05.000000;2006-06-18-17.01.05.000000;2006-06-25-17.01.05.000000;MUFS;SUMMARY;168.000;…
2006-06-01-00.06.05.000000;2006-06-25-17.01.05.000000;2006-07-02-17.01.05.000000;MUFS;SUMMARY;156.890;…
```

The file could then be uploaded to Excel:

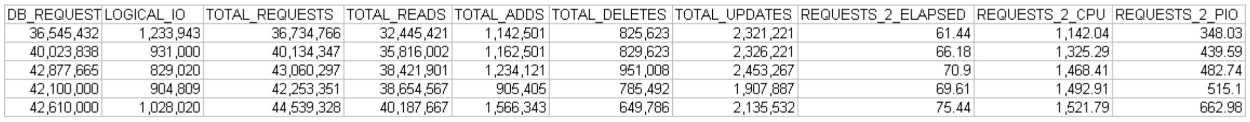


D_MUF_ENABLE	D_MUF_START_DATETIME	D_MUF_END_DATETIME	MUF_NAME	D_MUF_TYPE	D_MUF_DURATION	Rest headers
2006-04-01-00.06.05.000000	2006-04-01-17.01.05.000000	2006-04-30-17.01.05.000000	MUFS	BASELINE	165.223	….
2006-06-01-00.06.05.000000	2006-06-04-17.01.05.000000	2006-06-11-17.01.05.000000	MUFS	SUMMARY	168.00	….
2006-06-01-00.06.05.000000	2006-06-11-17.01.05.000000	2006-06-18-17.01.05.000000	MUFS	SUMMARY	168.00	….
2006-06-01-00.06.05.000000	2006-06-18-17.01.05.000000	2006-06-25-17.01.05.000000	MUFS	SUMMARY	168.00	….
2006-06-01-00.06.05.000000	2006-06-25-17.01.05.000000	2006-07-02-17.01.05.000000	MUFS	SUMMARY	156.89	….

The DSV output from the summary Delta (BVM and BVD) would be the same format as all the Delta datasets.



D_MUF_ENABLE	D_MUF_START_DATETIME	D_MUF_END_DATETIME	D_MUF_TYPE	D_MUF_DURATION	MUF_NAME	D_MUF_ELAPSED	CURRENT_CPU	PHYSICAL_IO
2006-04-01-00.06.05.000000	2006-04-01-17.01.05.000000	2006-04-30-17.01.05.000000	BASELINE	165.223	MUFS	594,802	32,000	105,006
2006-06-01-00.06.05.000000	2006-06-04-17.01.05.000000	2006-06-11-17.01.05.000000	SUMMARY	168	MUFS	604,800	30,200	91,300
2006-06-01-00.06.05.000000	2006-06-11-17.01.05.000000	2006-06-18-17.01.05.000000	SUMMARY	168	MUFS	604,800	29,200	89,200
2006-06-01-00.06.05.000000	2006-06-18-17.01.05.000000	2006-06-25-17.01.05.000000	SUMMARY	168	MUFS	604,800	28,200	82,030
2006-06-01-00.06.05.000000	2006-06-25-17.01.05.000000	2006-07-02-17.01.05.000000	SUMMARY	156.89	MUFS	564,804	28,000	67,180



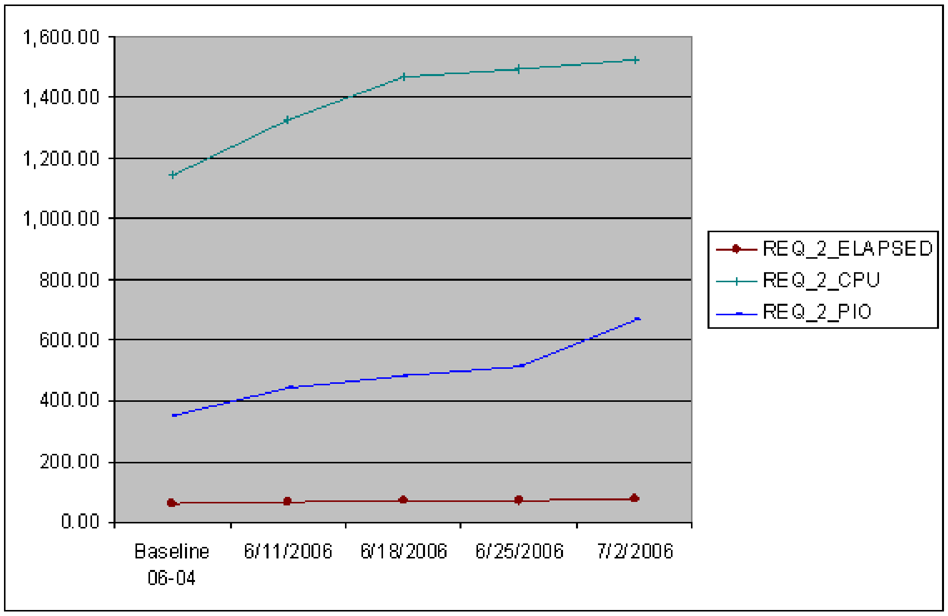
DB_REQUEST	LOGICAL_IO	TOTAL_REQUESTS	TOTAL_READS	TOTAL_ADDS	TOTAL_DELETES	TOTAL_UPDATES	REQUESTS_2_ELAPSED	REQUESTS_2_CPU	REQUESTS_2_PIO
36,545,432	1,233,943	36,734,766	32,445,421	1,142,501	825,623	2,321,221	61.44	1,142.04	348.03
40,023,838	931,000	40,134,347	35,816,002	1,162,501	829,623	2,326,221	66.18	1,325.29	439.59
42,877,665	829,020	43,060,297	38,421,901	1,234,121	951,008	2,453,267	70.9	1,468.41	482.74
42,100,000	904,809	42,253,351	38,654,567	905,405	785,492	1,907,887	69.61	1,492.91	515.1
42,610,000	1,028,020	44,539,328	40,187,667	1,566,343	649,786	2,135,532	75.44	1,521.79	662.98

Using the Excel Transpose Option

While the previous format is still very readable, the user can use available Excel facilities to transpose the BVM and BVD tables from landscape format to portrait format.

D_MUF_ENABLE	2006-04-01-00.06.05.000000	2006-06-01-00.06.05.000000	2006-06-01-00.06.05.000000	2006-06-01-00.06.05.000000	2006-06-01-00.06.05.000000
D_MUF_START_DATETIME	2006-04-01-17.01.05.000000	2006-06-04-17.01.05.000000	2006-06-11-17.01.05.000000	2006-06-18-17.01.05.000000	2006-06-25-17.01.05.000000
D_MUF_END_DATETIME	2006-04-30-17.01.05.000000	2006-06-11-17.01.05.000000	2006-06-18-17.01.05.000000	2006-06-25-17.01.05.000000	2006-07-02-17.01.05.000000
D_MUF_TYPE	BASELINE	SUMMARY	SUMMARY	SUMMARY	SUMMARY
D_MUF_DURATION	165.223	168	168	168	156.89
MUF_NAME	MUFS	MUFS	MUFS	MUFS	MUFS
D_MUF_ELAPSED	594,802	604,800	604,800	604,800	564,804
CURRENT_CPU	32,000	30,200	29,200	28,200	28,000
PHYSICAL_IO	105,006	91,300	89,200	82,030	67,180
DB_REQUESTS	36,545,432	40,023,838	42,877,665	42,100,000	42,610,000
LOGICAL_IO	1,233,943	931,000	829,020	904,809	1,028,020
TOTAL_REQUESTS	36,734,766	40,134,347	43,060,297	42,253,351	44,539,328
TOTAL_READS	32,445,421	35,816,002	38,421,901	38,654,567	40,187,667
TOTAL_ADDS	1,142,501	1,162,501	1,234,121	905,405	1,566,343
TOTAL_DELETES	825,623	829,623	951,008	785,492	649,786
TOTAL_UPDATES	2,321,221	2,326,221	2,453,267	1,907,887	2,135,532
REQUESTS_2_ELAPSED	61.44	66.18	70.9	69.61	75.44
REQUESTS_2_CPU	1,142.04	1,325.29	1,468.41	1,492.91	1,521.79
REQUESTS_2_PIO	348.03	439.59	482.74	515.1	662.98

Excel graphs can now be created comparing the baseline week with the four summary weeks.



Placing DSV files into Excel

This section covers the basic process used above to place DSV output into an Excel spreadsheet.

Create a Sample Empty Excel Sheet

To make the import as easy as possible, you can create a blank master for AutoCollect spreadsheets.

To create a sample empty Excel spreadsheet for MS-EXCEL 2003

1. Go to Excel and open a new spreadsheet.
2. Right-click the tab SHEET1 and rename this tab to BVM.
3. Right-click the tab SHEET2 and rename this tab to BVD.
4. Right-click the tab SHEET3 and rename this tab to MFS.
5. Click INSERT and choose WORKSHEET

   A new worksheet table is created.
6. Right-click the new tab and rename this tab to MFA.

Repeat as needed for each of the DSV output datasets you want to import from the mainframe.

To create a sample empty Excel spreadsheet for MS-EXCEL 2007

1. Go to Excel and open a new spreadsheet.
2. Right-click the tab SHEET1 and rename this tab to BVM.
3. Right-click the tab SHEET2 and rename this tab to BVD.
4. Right-click the tab SHEET3 and rename this tab to MFS.
5. Place your cursor on one of the worksheet tabs and right-click.
6. Click INSERT and choose WORKSHEET

   A new worksheet tab is created.
7. Right-click the new tab and rename this tab to MFA.

Repeat as needed for each of the DSV output datasets you want to import from the mainframe.

Save this Excel spreadsheet as your "empty" master.

FTP Mainframe Datasets to PC

Using an available FTP tool, download the selected DSV mainframe datasets to a work area on your PC. Use ".txt" for the PC file type.

Import Data into Excel

After you create a master spreadsheet, you can import data into it.

To import data into an Excel spreadsheet for MS-Excel 2003

1. Open the "empty" master spreadsheet saved earlier.
2. Save the spreadsheet with a new name.
3. Select BVM tab.
4. Click on cell A1.
5. Click on DATA > Import External Data > Import Data.

   A browse screen appears.
6. Use the browse screen to locate the BVM.txt file just downloaded and click on that file to import to the Excel worksheet.
7. Choose "Delimited", start input at row "1", then click NEXT.
8. Choose Delimiter as "semi-colon" (or whatever you used in the DSVOUT function), then click FINISH.
9. Select OK to insert at A1.
10. Repeat for each tab that you want to upload.

To import data into an Excel spreadsheet for MS-Excel 2007

1. Open the "empty" master spreadsheet saved earlier.
2. Save the spreadsheet with a new name.
3. Select BVM tab.
4. Click on cell A1.
5. Click on DATA > From text

   A browse screen appears.
6. Use the browse screen to locate the BVM.txt file just downloaded and click on that file to import to the Excel worksheet.
7. Choose "Delimited", start input at row "1", then click NEXT.
8. Choose Delimiter as "semi-colon" (or whatever you used in the DSVOUT function), then click FINISH.
9. Select OK to insert at A1.
10. Repeat for each tab that you want to upload.

Transposing an Excel Spreadsheet

As we mentioned in the second sample for the DSVOUT function, sometimes there is a need to rotate an Excel spreadsheet from a horizontal (landscape) to vertical (portrait) format. This can be accomplished using the copy and paste functions of Excel.

For MS-Excel 2003

- Highlight all of the active columns in the current spreadsheet.
- Issue a copy command to copy the cells into memory.
- Highlight a new worksheet or a new area on the existing spreadsheet.
- Choose PASTE SPECIAL.
  - From the PASTE SPECIAL menu, choose the TRANSPOSE option.
  - Press Enter.

The resulting pasted area will be transposed (rotated) as requested.

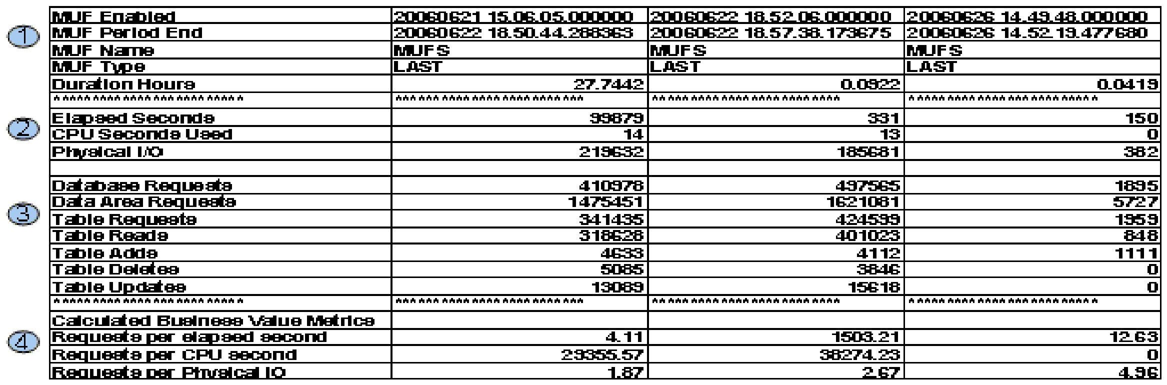
For MS-Excel 2007

- Highlight all of the active columns in the current spreadsheet.
- Issue a copy command to copy the cells into memory.
- Highlight a new worksheet or a new area on the existing spreadsheet.
- Click on the PASTE drop-down button
  - From the PASTE menu, choose the TRANSPOSE option.
  - Press Enter.

The resulting pasted area will be transposed (rotated) as requested.

Sample Excel Spreadsheets

Excel – BVM Business Value Metrics Tab



	MUF Enabled	20060621 15.06.05.000000	20060622 18.52.06.000000	20060626 14.49.48.000000
1	MUF Period End	20060622 18.50.44.288363	20060622 18.57.38.173675	20060626 14.52.19.477680
	MUF Name	MUFS	MUFS	MUFS
	MUF Type	LAST	LAST	LAST
	Duration Hours	27.7442	0.0922	0.0419
	************************	************************	************************	************************
2	Elapsed Seconds	99879	331	150
	CPU Seconds Used	14	13	0
	Physical I/O	219632	185681	382
	Database Requests	410978	497565	1895
	Data Area Requests	1475451	1621081	5727
3	Table Requests	341435	424599	1959
	Table Reads	318628	401023	848
	Table Adds	4633	4112	1111
	Table Deletes	5085	3846	0
	Table Updates	13089	15618	0
	************************	************************	************************	************************
	Calculated Business Value Metrics			
4	Requests per elapsed second	4.11	1503.21	12.63
	Requests per CPU second	29355.57	38274.23	0
	Requests per Physical IO	1.87	2.67	4.96

The sample spreadsheet shows the BVM tab after it has been transposed:

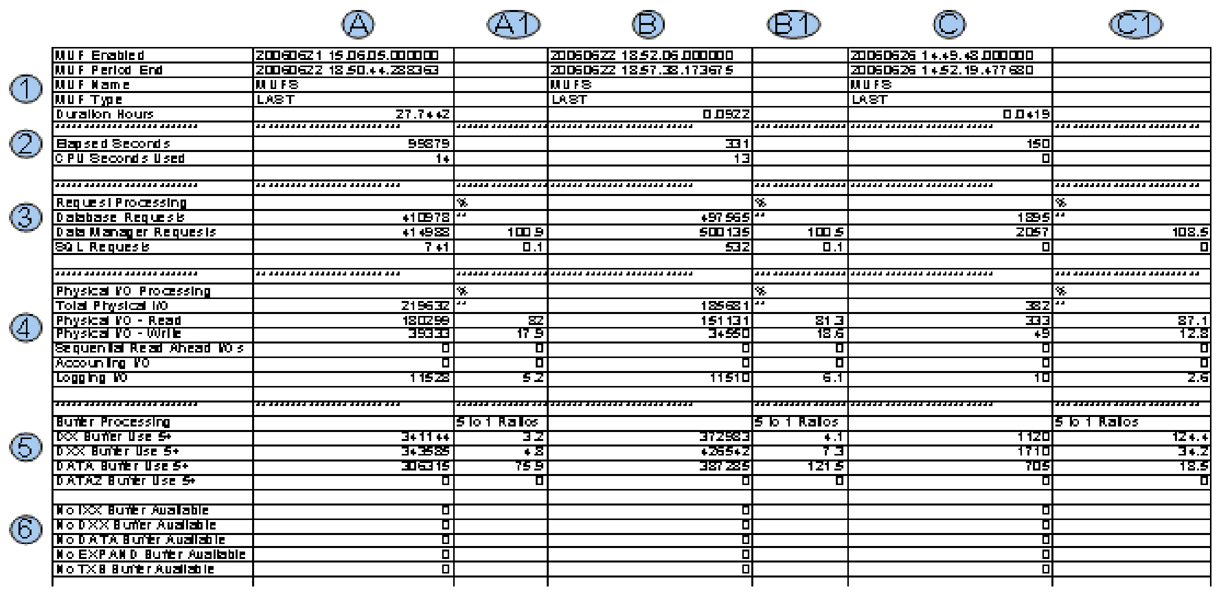
1 Identifies the processing MUF and processing period

2 Amount of resources used

3 Amount of work done

4 Calculation of Business Value Metrics

Excel – BVD Business Value Details Tab



		A	A1	B	B1	C	C1
	MUF Enabled	20060621 15.06.05.000000		20060622 18.52.06.000000		20060626 14.49.48.000000	
	MUF Period End	20060622 18.50.44.288363		20060622 18.57.38.173675		20060626 14.52.19.477680	
1	MUF Name	MUF8		MUF8		MUF8	
	MUF Type	LAST		LAST		LAST	
	Duration Hours	27.7442		0.0922		0.0419	
	----------------------------	----------------------------	----------------	----------------------------	----------------	----------------------------	----------------------------
2	Elapsed Seconds	99879		331		150	
	CPU Seconds Used	14		13		0	
	----------------------------	----------------------------	----------------	----------------------------	----------------	----------------------------	----------------------------
	Request Processing		%		%		%
3	Database Requests	410978	--	497565	--	1895	--
	Data Manager Requests	414988	100.9	500135	100.5	2057	108.5
	SQL Requests	741	0.1	532	0.1	0	0
	----------------------------	----------------------------	----------------	----------------------------	----------------	----------------------------	----------------------------
	Physical I/O Processing		%		%		%
	Total Physical I/O	219632	--	185681	--	382	--
4	Physical I/O - Read	180299	82	151131	81.3	333	87.1
	Physical I/O - Write	39333	17.9	34550	18.6	49	12.8
	Sequential Read Ahead I/O s	0	0	0	0	0	0
	Accounting I/O	0	0	0	0	0	0
	Logging I/O	11528	5.2	11510	6.1	10	2.6
	----------------------------	----------------------------	----------------	----------------------------	----------------	----------------------------	----------------------------
	Buffer Processing		5 to 1 Ratios		5 to 1 Ratios		5 to 1 Ratios
5	IXX Buffer Use 5+	3411 44	3.2	372983	4.1	1120	124.4
	DXX Buffer Use 5+	343585	4.8	426542	7.3	1710	34.2
	DATA Buffer Use 5+	306315	75.9	387285	121.5	705	18.5
	DATA2 Buffer Use 5+	0	0	0	0	0	0
	No IXX Buffer Available	0		0		0	
6	No DXX Buffer Available	0		0		0	
	No DATA Buffer Available	0		0		0	
	No EXPAND Buffer Available	0		0		0	
	No TXB Buffer Available	0		0		0	

The sample spreadsheet above shows the BVD tab after it has been transposed:

1 Identifies the processing MUF and processing period

2 Amount of CPU and elapsed seconds used

3 Amount of work done

4 Breakdown of physical IO

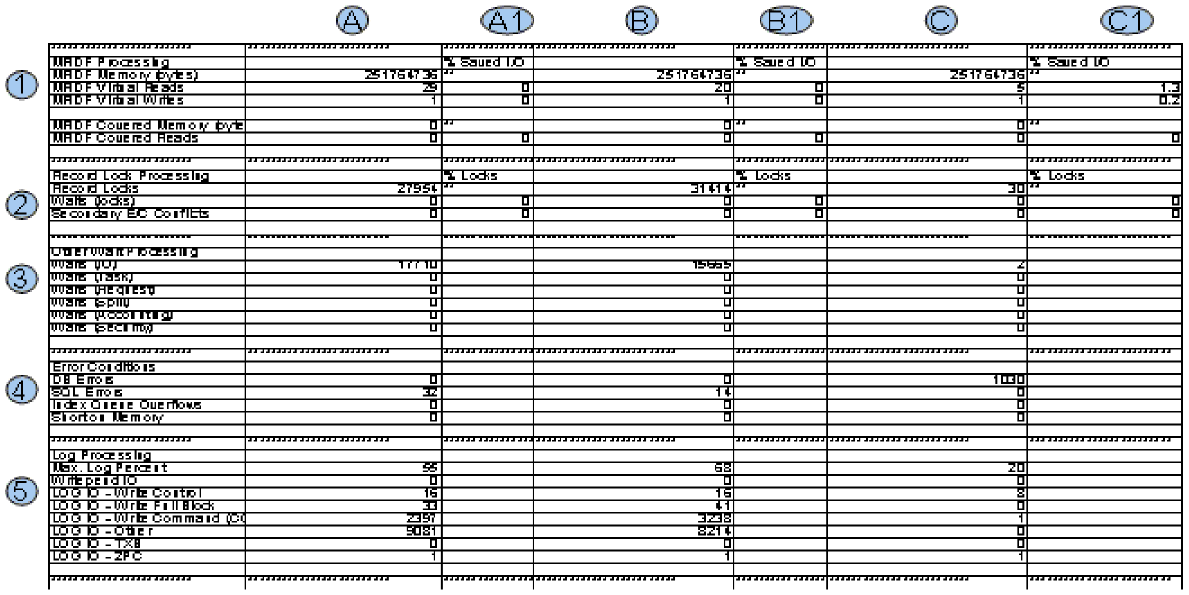
5 Buffer processing 5+ counts

6 No buffer available counts

A/B/C Counts for this period

A1/B1/C1 Key processing ratios

Excel – BVD Business Value Details Tab (Cont'd)


A
A1
B
B1
C
C1
1
2
3
4
5

	A	A1	B	B1	C	C1
MRDF Processing		% Saved I/O		% Saved I/O		% Saved I/O
MRDF Memory (bytes)	251764736	"	251764736	"	251764736	"
MRDF Virtual Reads	29	0	20	0	5	1.3
MRDF Virtual Writes	1	0	1	0	1	0.2
MRDF Covered Memory (byte	0	"	0	"	0	"
MRDF Covered Reads	0	0	0	0	0	0
Record Lock Processing		% Locks		% Locks		% Locks
Record Locks	27954	"	31414	"	30	"
Waits (locks)	0	0	0	0	0	0
Secondary E/C Conflicts	0	0	0	0	0	0
Other Wait Processing						
Waits (IO)	17710		15665		2	
Waits (Task)	0		0		0	
Waits (Request)	0		0		0	
Waits (Spill)	0		0		0	
Waits (Accounting)	0		0		0	
Waits (Security)	0		0		0	
Error Conditions						
DB Errors	0		0		1030	
SQL Errors	32		14		0	
Index Queue Overflows	0		0		0	
Short on Memory	0		0		0	
Log Processing						
Max. Log Percent	55		68		20	
Writepend IO	0		0		0	
LOG IO - Write Control	16		16		8	
LOG IO - Write Full Block	33		41		0	
LOG IO - Write Command (C	2397		3238		1	
LOG IO - Other	9081		8214		0	
LOG IO - TXB	0		0		0	
LOG IO - 2PC	1		1		1	

1 MRDF processing

2 Locks/Exclusive control

3 Waits

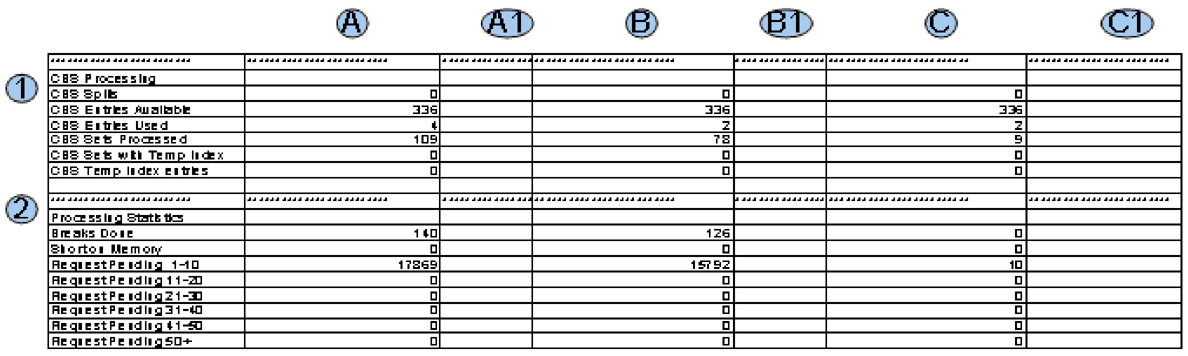
4 Errors

5 Log processing

A/B/C Counts for this period

A1/B1/C1 Key processing ratios

Excel – BVD Business Value Details Tab (Cont'd)



	A	A1	B	B1	C	C1
CBS Processing						
CBS Splits	0		0		0	
CBS Entries Available	336		336		336	
CBS Entries Used	4		2		2	
CBS Sets Processed	109		78		9	
CBS Sets with Temp Index	0		0		0	
CBS Temp Index entries	0		0		0	
Processing Statistics						
Breaks Done	140		126		0	
Short on Memory	0		0		0	
RequestPending 1-10	17869		15792		10	
RequestPending 11-20	0		0		0	
RequestPending 21-30	0		0		0	
RequestPending 31-40	0		0		0	
RequestPending 41-50	0		0		0	
RequestPending 50+	0		0		0	

1	CBS processing
2	Request processing
A/B/C	Counts for this period
A1/B1/C1	Key processing ratios

Processing Data Outside of the AUTOCOLL Functions

Reporting

The AutoCollect tables are standard CA Datacom tables. They can easily be used as input for various reporting tools.

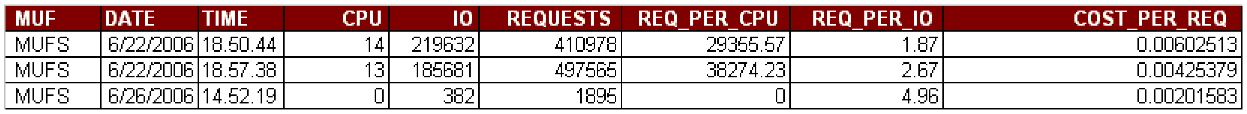
For example, CA Datacom Server can use ODBC/SQL to access the Delta tables and import the data directly into Excel spreadsheets. With the flexibility of Server and SQL, you could bypass the need for the DSVOUT function.

Sample SERVER/SQL

Sample SQL query against the Delta table D_BUSINESS_VALUE_METRICS selecting just TYPE=LAST rows and placing the output into an Excel spreadsheet.

```
SELECT MUF_NAME AS MUF,
       DATE(D_MUF_DATETIME) AS DATE,
       TIME(D_MUF_DATETIME) AS TIME,
       CURRENT_CPU AS CPU,
       PHYSICAL_IO AS IO,
       DB_REQUESTS AS REQUESTS,
       REQUESTS_2_CPU AS REQ_PER_CPU,
       REQUESTS_2_PIO AS REQ_PER_IO,
       ((CURRENT_CPU * 20)/DB_REQUESTS)+((PHYSICAL_IO*.01)/DB_REQUESTS) AS
       COST_PER_REQ
  FROM AUTOCOLLECT.D_BUSINESS_VALUE_METRICS
 WHERE D_MUF_TYPE = 'LAST';
```

Returned as an Excel spreadsheet:



MUF	DATE	TIME	CPU	IO	REQUESTS	REQ_PER_CPU	REQ_PER_IO	COST_PER_REQ
MUFS	6/22/2006	18.50.44	14	219632	410978	29355.57	1.87	0.00602513
MUFS	6/22/2006	18.57.38	13	185681	497565	38274.23	2.67	0.00425379
MUFS	6/26/2006	14.52.19	0	382	1895	0	4.96	0.00201583

There is a wide variety of site specific functionality that could be added by using available tools to build and deliver the AutoCollect data to the user's performance and tuning group.

Freeform Editing

We have documented various techniques to create and maintain Delta rowsets. However, in certain cases, you may need to do additional editing of the Snapshot or Delta rowsets.

Because the Snapshot and Delta tables are available for "open" access, you can use standard CA Datacom tools to update and or delete rows in these tables.

We recommend, that before any such action is taken by you that the Snapshot and Delta tables are backed up to prevent any inadvertent data loss. AutoCollect data can encompass information collected over weeks or months of execution. The data can only be recovered using DBUTLTY; it cannot be recreated (re-captured).

Rowset Concept

The key to successfully updating/deleting data in the AutoCollect system requires you to understand the Snapshot and Delta rowset concept. Any update activity that would change the content of one row in the rowset must be balanced with changes to other rows in the rowsets. Similarly, any deletion of rows must be done in consideration of other rows in the rowset.

Information about how to identify each of the AutoCollect rowsets is provided in the sections on database contents found later in this document.

Failure to make changes in a way that *matches* all corresponding rows in a rowset could invalidate the entire rowset and any performance measurement taken from that data.

Chapter 6: Accessing AutoCollect Data Using DBSQLPR

Products such as the CA Datacom Server can use SQL to directly access the Delta tables, apply various functions, and build result sets. With CA Datacom Server, these results can be loaded directly into a Microsoft Excel spreadsheet.

AutoCollect provides the ability to build Summary and Baseline rowsets so that later processing can be easier. The AutoCollect system also delivers a non-SQL batch option, OPTION=DSVOUT, to select data from the Delta tables that can be used to generate Microsoft Excel spreadsheets. However, in some cases, you may find it difficult to use the Delta table data being generated without some further summarization or editing.

The Delta tables are defined as a standard CA Datacom/DB database that can be accessed by any of the programming APIs (RAAT, SAAT, or SQL) and by the standard CA Datacom query tools, such as:

- CA Datacom Server
- CA Datacom SQL
  - DDOL ISQL Option
  - Batch DBSQLPR
- CA Datacom/DB Reporting Facility
- CA Dataquery

Users with any of these tools and an understanding of the Delta table layouts can easily extend their *view* of the AutoCollect data.

DBSQLPR Delimiter Separated Values Output

The DBSQLPR utility is described in the *CA Datacom/DB SQL User Guide*. It provides a simple way to dynamically execute most SQL statements in batch.

DBSQLPR accepts control parameters (sets execution variables) and SQL statements from SYSIN.

Each SQL statement is compiled and, if the compile is successful, the statement is executed. For SELECT statements, DBSQLPR generates a simple report file with the requested column headers and data.

In addition to the simple report feature, DBSQLPR allows you to change the output choice to a DSV output dataset instead of the generated report output for a given execution of DBSQLPR.

Executing DBSQLPR

DBSQLPR should be executed using the information (sample JCL) provided with the *CA Datacom/DB SQL User Guide*. For detailed information about each of the DBSQLPR functions and processing options, see the *CA Datacom/DB SQL User Guide*.

DBSQLPR Parameters

For DSV output, we recommend the following DBSQLPR processing options.

DATASEPARATOR=x

Directs DBSQLPR to create a DSV output dataset that is intended for downloading into a typical spreadsheet tool such as Microsoft Excel.

DBSQLPR output is directed to the DD name of STDOUT. For testing purposes, you may want to test your query using STDOUT as a SYSOUT dataset until you have the data formatted the way you want. Once you are ready, the DBSQLPR could be re-executed pointing STDOUT to a dataset that is suitable for transferring to the PC using FTP to your spreadsheet tool.

The *x* is any non-blank character that does not occur normally in your data values. Most systems use comma (,) as the value, but since some of the data in the Delta tables contains commas, we recommend using the semi-colon (;).

SQUISH

Directs DBSQLPR to remove unneeded spaces from column headers and data so that the output record is smaller.

When STDOUT is directed to create DSV output, the DSV columns are fixed in length to represent the longest value of the column header or the largest column value.

By combining SQUISH with the maximum value for PRTWIDTH, you are able to generate DSV output rows with a significant number of column values.

NOECHO

Indicates that DBSQLPR should not ECHO the processing options and SQL statement to STDOUT.

NOFORMFEED

Indicates that no form feed processing should be done. This prevents any additional *form feeds* to be placed in the output.

PAGELEN=999999999

Indicates that the output is treated as a single page of output.

NOTYPE

Indicates that the TYPE line generated by DBSQLPR to show the type and length of the column is to be suppressed.

PRTWIDTH=1500

Indicates that the output line can be up to 1500 characters. Output rows over 1500 characters cannot be used for DBSQLPR output files.

Sample Problem - 1

A user has existing AutoCollect spreadsheets, but would like a spreadsheet with a reduced view of the BVM table. All the required data is on the Delta LAST rowsets.

For *LAST* rowsets, list the following:

MUF

Display the MUF name.

DATE

Display the end date.

TIME

Display the end time.

CPU

Display the CPU seconds used.

IO

Display the I/O used.

REQUESTS

Display the requests processed.

REQ_PER_CPU

Display the requests per CPU second used.

REQ_PER_IO

Display the requests per I/O used.

COST_PER_REQ

Display the cost per request (new column).

DBSQLPR DSV Sample

Using standard DBSQLPR, execute a query that generates the requested report. Default options are chosen for most DBSQLPR parameters.

```
STDOUT directed to SYSOUT
```

Processing Options

```
AUTHID=SYSADM
PRTWIDTH=160
SQLMODE=DATACOM
WORKSPACE=128
```

```
INPUT STATEMENT:
   SELECT MUF_NAME AS MUF,
       DATE(D_MUF_DATETIME) AS DATE,
       TIME(D_MUF_DATETIME) AS TIME,
       CURRENT_CPU AS CPU,
       PHYSICAL_IO AS IO,
       DB_REQUESTS AS REQUESTS,
       REQUESTS_2_CPU AS REQ_PER_CPU,
       REQUESTS_2_PIO AS REQ_PER_IO,
       ((CURRENT_CPU * 20)/DB_REQUESTS)+((PHYSICAL_IO*.01)/DB_REQUESTS) AS
       COST_PER_REQ
  FROM AUTOCOLLECT.D_BUSINESS_VALUE_METRICS
 WHERE D_MUF_TYPE = 'LAST';
```

Generated Report

```
MUF          DATE       TIME                     CPU              IO       REQUESTS
CHAR(8) N.N. DATE N.N.  TIME N.N. DEC(11,0) N.N. DEC(11,0) N.N. DEC(11,0) N.N.

____________ __________ _________ ______________ ______________ ______________
MUFS         2006-06-22 18.50.44              14          219632         410978
MUFS         2006-06-22 18.57.38              13          185681         497565
MUFS         2006-06-26 14.52.19               0             382           1895
___ 3 rows returned ___
```

```
  REQ_PER_CPU     REQ_PER_IO                              COST_PER_REQ
 DEC(11,2) N.N. DEC(11,2) N.N.                       DEC(31,8) NOT NULL

 ______________ ______________ _________________________________
      29355.57           1.87                          0.00602513
      38274.23           2.67                          0.00425379
          0.00           4.96                          0.00201583
```

Use DBSQLPR with the following DSV output parameters to re-execute the query to generate a DSV file.

```
STDOUT directed to DSN  (//STDOUT DD DSN=seq.file,DISP=(NEW,CATLG),…
```

Processing Options

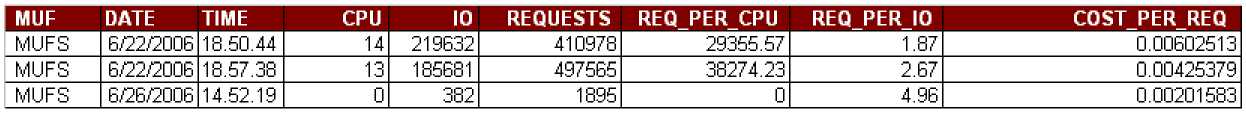
```
AUTHID=SYSADM
SQLMODE=DATACOM
WORKSPACE=128
PRTWIDTH=1500
NOECHO
NOFORMFEED
PAGELEN=999999999
NOTYPE
DATASEPARATOR=;
```

Generated DSV Output

Report generated:

```
MUF    ;DATE     ;TIME    ;     CPU;      IO;   REQUESTS;  REQ_PER_CPU; REQ_PER_IO;
COST_PER_REQ;
MUFS   ;2006-06-22;18.50.44;     14;  219632;     410978;     29355.57;      1.87;
0.00602513;
MUFS   ;2006-06-22;18.57.38;     13;  185681;     497565;     38274.23;      2.67;
0.00425379;
MUFS   ;2006-06-26;14.52.19;      0;     382;       1895;         0.00;      4.96;
0.00201583;
```

Converted to a Microsoft Excel spreadsheet:



MUF	DATE	TIME	CPU	IO	REQUESTS	REQ_PER_CPU	REQ_PER_IO	COST_PER_REQ
MUFS	6/22/2006	18.50.44	14	219632	410978	29355.57	1.87	0.00602513
MUFS	6/22/2006	18.57.38	13	185681	497565	38274.23	2.67	0.00425379
MUFS	6/26/2006	14.52.19	0	382	1895	0	4.96	0.00201583

Sample Problem - 2

The user wants a detailed list of logical and physical I/O per database for each of the LAST rowsets on file.

Using standard DBSQLPR, execute a query which generates the requested report. Default options chosen for most DBSQLPR parameters.

```
STDOUT directed to SYSOUT
```

Processing Options

```
AUTHID=SYSADM
AUTHID=SYSADM
PRTWIDTH=160
SQLMODE=DATACOM
WORKSPACE=128
NOTYPE


INPUT STATEMENT:
SELECT
  MUF_NAME,
  DATE (D_MUF_ENABLE) AS DATE,
  DBID,
  SUM(PHYSICAL_READS + PHYSICAL_WRITES) AS PHYS_IO,
  SUM(LOGICAL_READS + LOGICAL_WRITES) AS LOGIC_IO
FROM AUTOCOLLECT.D_MUF_AREA_STATS
  WHERE  D_MUF_TYPE = 'LAST'
 GROUP BY MUF_NAME, D_MUF_ENABLE,  DBID;
```

Generated Report

Report generated:

```
MUF_NAME DATE        DBID     PHYS_IO                         LOGIC_IO

________ __________ ______ ____________ _________________________________
MUFS     2006-06-21      0        12058                            30387
MUFS     2006-06-21      1           69                             5243
MUFS     2006-06-21      2       205160                          1427715
MUFS     2006-06-21      6            0                               28
MUFS     2006-06-21     15         2250                             9847
MUFS     2006-06-21     17            0                                6
MUFS     2006-06-21   1000            0                                0
MUFS     2006-06-21   1006            4                                4
MUFS     2006-06-21   1019           72                             2217
MUFS     2006-06-21   1020            4                                4
MUFS     2006-06-21   9999       219617                          1475451
MUFS     2006-06-22      0        11838                            28470
MUFS     2006-06-22      1           74                             5646
MUFS     2006-06-22      2       173086                          1583870
MUFS     2006-06-22      6            0                               19
MUFS     2006-06-22     15          640                             1898
MUFS     2006-06-22     17            0                                6
….
….
….
MUFS     2006-06-28     15         1119                             4080
MUFS     2006-06-28     17            0                               33
MUFS     2006-06-28   1000            0                                0
MUFS     2006-06-28   1006            4                                4
MUFS     2006-06-28   1019           33                             1398
MUFS     2006-06-28   1020          137                            15453
MUFS     2006-06-28   9999         5984                            39301
 ___ 43 rows returned ___
```

Using DBSQLPR with DSV output parameters to re-execute the query to generate a DSV file.

`STDOUT directed to DSN  (//STDOUT DD DSN=seq.file,DISP=(NEW,CATLG),…`

Processing Options

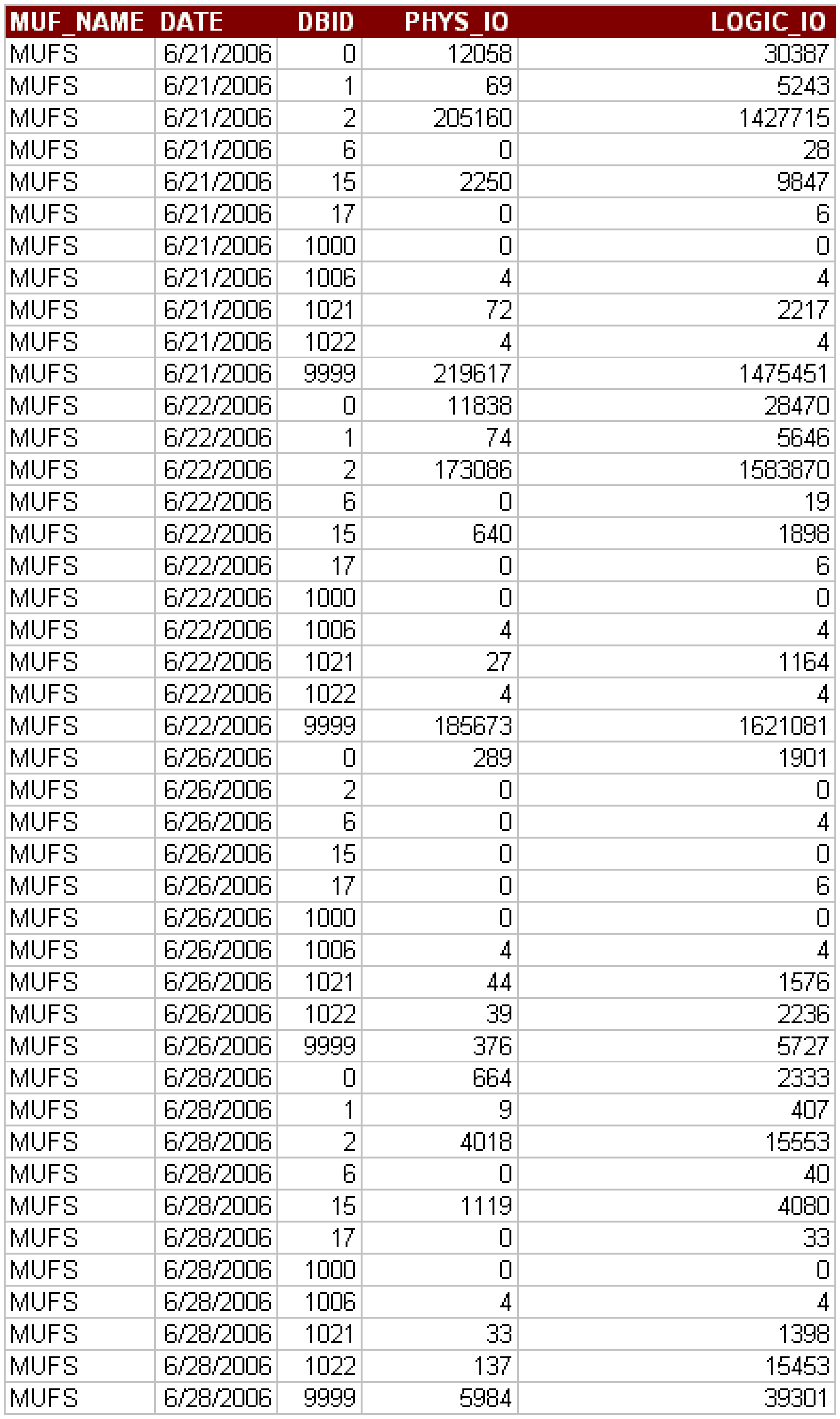
```
AUTHID=SYSADM
AUTHID=SYSADM
SQLMODE=DATACOM
WORKSPACE=128
PRTWIDTH=1500
NOECHO
NOFORMFEED
PAGELEN=999999999
NOTYPE
DATASEPARATOR=;
```

Generated DSV Output

Sequential file generated:

```
MUF_NAME;DATE      ;  DBID;     PHYS_IO;                         LOGIC_IO;
MUFS    ;2006-06-21;     0;       12058;                            30387;
MUFS    ;2006-06-21;     1;          69;                             5243;
MUFS    ;2006-06-21;     2;      205160;                          1427715;
MUFS    ;2006-06-21;     6;           0;                               28;
MUFS    ;2006-06-21;    15;        2250;                             9847;
MUFS    ;2006-06-21;    17;           0;                                6;
MUFS    ;2006-06-21;  1000;           0;                                0;
MUFS    ;2006-06-21;  1006;           4;                                4;
MUFS    ;2006-06-21;  1019;          72;                             2217;
MUFS    ;2006-06-21;  1020;           4;                                4;
MUFS    ;2006-06-21;  9999;      219617;                          1475451;
MUFS    ;2006-06-22;     0;       11838;                            28470;
MUFS    ;2006-06-22;     1;          74;                             5646;
MUFS    ;2006-06-22;     2;      173086;                          1583870;
MUFS    ;2006-06-22;     6;           0;                               19;
MUFS    ;2006-06-22;    15;         640;                             1898;
         ……
MUFS    ;2006-06-28;  1019;          33;                             1398;
MUFS    ;2006-06-28;  1020;         137;                            15453;
MUFS    ;2006-06-28;  9999;        5984;                            39301;
```

Converted into as a Microsoft Excel spreadsheet:



MUF_NAME	DATE	DBID	PHYS_IO	LOGIC_IO
MUFS	6/21/2006	0	12058	30387
MUFS	6/21/2006	1	69	5243
MUFS	6/21/2006	2	205160	1427715
MUFS	6/21/2006	6	0	28
MUFS	6/21/2006	15	2250	9847
MUFS	6/21/2006	17	0	6
MUFS	6/21/2006	1000	0	0
MUFS	6/21/2006	1006	4	4
MUFS	6/21/2006	1021	72	2217
MUFS	6/21/2006	1022	4	4
MUFS	6/21/2006	9999	219617	1475451
MUFS	6/22/2006	0	11838	28470
MUFS	6/22/2006	1	74	5646
MUFS	6/22/2006	2	173086	1583870
MUFS	6/22/2006	6	0	19
MUFS	6/22/2006	15	640	1898
MUFS	6/22/2006	17	0	6
MUFS	6/22/2006	1000	0	0
MUFS	6/22/2006	1006	4	4
MUFS	6/22/2006	1021	27	1164
MUFS	6/22/2006	1022	4	4
MUFS	6/22/2006	9999	185673	1621081
MUFS	6/26/2006	0	289	1901
MUFS	6/26/2006	2	0	0
MUFS	6/26/2006	6	0	4
MUFS	6/26/2006	15	0	0
MUFS	6/26/2006	17	0	6
MUFS	6/26/2006	1000	0	0
MUFS	6/26/2006	1006	4	4
MUFS	6/26/2006	1021	44	1576
MUFS	6/26/2006	1022	39	2236
MUFS	6/26/2006	9999	376	5727
MUFS	6/28/2006	0	664	2333
MUFS	6/28/2006	1	9	407
MUFS	6/28/2006	2	4018	15553
MUFS	6/28/2006	6	0	40
MUFS	6/28/2006	15	1119	4080
MUFS	6/28/2006	17	0	33
MUFS	6/28/2006	1000	0	0
MUFS	6/28/2006	1006	4	4
MUFS	6/28/2006	1021	33	1398
MUFS	6/28/2006	1022	137	15453
MUFS	6/28/2006	9999	5984	39301

Chapter 7: Suggested AutoCollect Implementation

The following is a suggested implementation for the first time user of AutoCollect.

The user must first do the following research and testing at his site.

1. Determine which MUFs are selected for AutoCollect processing.

   ■ Determine which MUFs are Source MUFs

     Verify that the Source MUFs have the Dynamic System Tables activated.

   ■ Determine which MUFsare the Repository MUF

   For the first time user we suggest that they select one MUF and use it as both the Source and Repository MUF.

2. Verify that the repository MUFs have the Snapshot and Delta databases installed in the CXX. A simple CXX report should be able to validate their presence and also provide the current allocation sizes.

   ■ If the databases are not present, check the r12 installation process for the steps to install the databases. If you cannot determine why the databases were not installed, contact CA Support.

   ■ If the dataset sizes are not large enough or if you have not allocated and initialized the two datasets for each database, follow the instructions earlier in this guide to allocate and initialize the AutoCollect databases.

   ■ Before starting data collection you should run a DBUTLTY LOAD FORMAT=NONE to verify the databases are empty and ready to receive AutoCollect data.

   Important! We strongly recommend that you implement the AutoCollect databases as 1019 (Snapshot) and 1020 (Delta). If you choose to use other DBIDs for these databases, the various AutoCollect functions of DBUTLTY need to include the DBIDs on every AutoCollect function execution.

3. Create a set of DBUTLTY jobs to execute and test the various AutoCollect functions:
   - Execute OPTION=SNAPSHOT several times.
   - Execute OPTION=SNAPRPT to report on the created Snapshot rowsets.
   - Execute OPTION=DELTACRE to create the Delta rowsets.
   - Execute OPTION=DELTARPT to report on the created Delta rowsets.
   - Execute OPTION=SUMMARY to summarize all the INTERVAL delta rowsets.
   - Execute OPTION=BASELINE to summary and baseline all the INTERVAL delta rowsets.
   - Execute OPTION=AVGPERF to summary and average all the INTERVAL delta rowsets.
   - Execute OPTION=DELTARPT to report on the created Delta rowsets included the user created Delta rowsets.
   - Execute OPTION=DSVOUT to create an output datasets for SUMMARY and BASELINE delta rowsets.
   - View the sequential output datasets through TSO or Roscoe.
   - Use available FTP protocol to transfer the files from the mainframe to .txt files on your PC. Allow EBCDIC to ASCII translation to occur.
   - Follow the instructions provided to import the .txt files into the excel spreadsheet.
   - (For sites with SQL) Following the examples for using DBSQLPR to print a report of the data in a Delta rowset.
   - (For sites with SQL) Following the examples for using DBSQLPR to create sequential output dataset for uploaded to a PC .txt file.
   - Use the SNAPRPT and SNAPDEL functions to report on selected Snapshot rowsets and then delete them. Practice using selection criteria such as DATERG, TIMERG and MUFNAME.
   - Use the DELTARPT and DELTADEL functions to report on selected Delta rowsets and then delete them. Practice using selection criteria such as TYPE, DATERG, TIMERG and MUFNAME.

   When done testing, execute a DBUTLTY LOAD FORMAT=NONE to clear out the test data. Save the various JCLs so that you have working examples for your site.

Once the initial testing phase has been completed, the user should now begin to plan on his AutoCollect snapshot collection process. Key to this process is the time periods in which the Source MUF is normally executing. As we have seen in the various examples in chapter 2, the timing of the MUF startup and shutdown will affect when we do over Snapshot collection points.

The process below provides a simple implementation for collection Snapshots of weekly data.

- Select a weekly Snapshot collection:
  - If the MUF comes down weekly, do the Snapshot just before the MUF cycle.
  - If the MUF stays up for longer periods of time, select a specific quiet time during the weekend. Execute the Snapshot at the same time each week.
  - If the MUF is cycled at non-weekly increments, combine the two points above of doing a snapshot at a fixed time each week and one right before the MUF is shu tdown.
- Collect 4 weeks of *normal* activity Snapshots.
- Create Delta rowsets.
- Produce Snapshot and Delta reports.
- Create Summary rowsets for each week, if needed.
- Create a weekly baseline record from the four weeks Summary rowsets.
- Create the DSVOUT output files.
- Download the DSV files into Microsoft Excel spreadsheets.

- Review the spreadsheets for any performance opportunities:
  - Look for trouble spots, such as low resource, waits, buffer not available, and so on.
  - Look for ways to improve performance, such as buffers, covered, virtual, and so on.
  - Build a list of suggested changes.
- Implement first performance change.
- Wait for a week to collect new performance information
- Create new weekly summary Delta rowset.
- Create a new DSV output with all summary rowsets
- Update to an Excel spreadsheet.
  - Compare all statistics:
    - Statistics where we expected change
    - Other statistics that changed
    - Determine if the change was positive.
    - Decide to keep or reset the change.
- Repeat the process until all noted changes have been made.
- Continue weekly review process.
  - Each week, statistics should be reviewed for the following:
- Overall performance
- Changing workloads
- Error conditions

Chapter 8: AutoCollect SNAPSHOT Database

Each of the Snapshot tables has the same basic column layout as the corresponding Dynamic System Table (DST).

Snapshot Rowsets

For every Snapshot taken, one Snapshot rowset is created with the following rows being added to Snapshot tables.

SQL NAME	Table	Detail Rows	Total Rows
S_MUF_ACCOUNTING	MFY	1 per acct table*	1
S_MUF_AREA_STATS	MFA	1 per area*	1
S_MUF_CBS	MCB	1	0
S_MUF_CONFIG	MCF	1 per configuration parameter*	0
S_MUF_COVEREDVIRTUAL	MFC	1 per definition*	2
S_MUF_INTERNAL_STATS	MFV	1	0
S_MUF_OPTIONS	MFO	1	0
S_MUF_RETURN_CODES	MFU	1 per DB return code*	1
S_MUF_SMP_STATS	MFW	1	0
S_MUF_SRB_ZIIP	MCF	1 per SRB*	1
S_MUF_SYSTEM_STATS	MFS	1	1
S_MUF_TABLE_STATS	MFT	1 per table*	1
S_MUF_TCB_OR_SRB	MTC	1 per subtask*	1
S_SQL_MISC_STATS	SQM	1	0
S_SQL_SQLCODES	SQQ	1 per SQL code*	1

Note: Asterisk (*) indicates that the number of rows is dependent on the active number of DST rows at the time the Snapshot was taken. Once an entry becomes active in DST it remains there for the duration of this execution of MUF.

The following is an example of Snapshot rowsets created:

- A user creates one Snapshot a day at 5 PM and one before the MUF is terminated.
- The source MUF environment is cycled weekly.
- The following maximum number of active items:
  - Areas = 100
  - Tables = 200
  - Accounting tables = 5
  - Virtual/Covered = 5
  - DB return codes = 10
  - SQL return codes = 10
- Expected rowsets created per Snapshot will be 342.
- 8 Snapshots = 2736 Snapshot rowsets created per week.

Snapshot Identification Columns

In addition to the DST columns, each table has three identification columns included at the front of each row.

Column	Description
S_MUF_ENABLE	Displays the timestamp that the source MUF of the Snapshot was enabled.
S_MUF_DATETIME	Displays the timestamp when the Snapshot was recorded.
S_MUF_TOTALROW	Indicates if this is a total row.

Snapshot Tables

Column headings in this section are abbreviated as follows:

- CLS = Class
- TYP = Type
- SGN = Numeric Sign
- LNG = Length
- NUL = Column can have a Null value
- REP = Number of times this column repeats (always 1)
- RDF = Does this column redefine another column (always N)
- DSP = Displacement in the row

S_MUF_ACCOUNTING (MFY)

SQL NAME	SQL TYPE	CLS	TYP	SGN	LNG	NUL	REP	RDF	DSP
S_MUF_ENABLE	SQL-STMP	S	B	N	10	N	1	N	0
S_MUF_DATETIME	SQL-STMP	S	B	N	10	N	1	N	10
S_MUF_TOTALROW	CHAR	S	C	N	1	N	1	N	20
MUF_NAME	CHAR	S	C	N	8	N	1	N	21
DBID	SMALLINT	S	B	Y	2	N	1	N	29
TABLE_NAME	CHAR	S	C	N	3	N	1	N	31
BUFFER_SIZE	INTEGER	S	B	Y	4	N	1	N	34
TABLE_STATUS	CHAR	S	C	N	1	N	1	N	38
THRESHOLD	INTEGER	S	B	Y	4	N	1	N	39
SPILLING	CHAR	S	C	N	1	N	1	N	43
TOTAL_REQUESTS	DECIMAL	S	D	Y	8	N	1	N	44
PHYSICAL_READWRITE	DECIMAL	S	D	Y	8	N	1	N	52
LOCATES	DECIMAL	S	D	Y	8	N	1	N	60
TOTAL_ADDS	DECIMAL	S	D	Y	8	N	1	N	68
TOTAL_UPDATES	DECIMAL	S	D	Y	8	N	1	N	76
UNAVAILABLE	DECIMAL	S	D	Y	8	N	1	N	84

SQL NAME	SQL TYPE	CLS	TYP	SGN	LNG	NUL	REP	RDF	DSP
CONFLICTS	DECIMAL	S	D	Y	8	N	1	N	92
SKIPPED	DECIMAL	S	D	Y	8	N	1	N	100
ENTRIES	DECIMAL	S	D	Y	8	N	1	N	108
RECLAIMED	DECIMAL	S	D	Y	8	N	1	N	116
LOCATES_STARTED	DECIMAL	S	D	Y	8	N	1	N	124
SPILLS_THRESHOLD	DECIMAL	S	D	Y	8	N	1	N	132
SPILLS_INTERNAL	DECIMAL	S	D	Y	8	N	1	N	140

Primary KEY columns are as follows:

- MUF_NAME
- DBID
- TABLE_NAME
- S_MUF_ENABLE
- S_MUF_DATETIME

Secondary KEY columns are as follows:

- MUF_NAME
- S_MUF_ENABLE
- S_MUF_DATETIME

S_MUF_AREA_STATS (MFA)

SQL NAME	SQL TYPE	CLS	TYP	SGN	LNG	NUL	REP	RDF	DSP
S_MUF_ENABLE	SQL-STMP	S	B	N	10	N	1	N	0
S_MUF_DATETIME	SQL-STMP	S	B	N	10	N	1	N	10
S_MUF_TOTALROW	CHAR	S	C	N	1	N	1	N	20
MUF_NAME	CHAR	S	C	N	8	N	1	N	21
BEGIN_TIME	SQL-STMP	S	B	N	10	N	1	N	29
CURRENT_DATETIME	SQL-STMP	S	B	N	10	N	1	N	39
DBID	SMALLINT	S	B	Y	2	N	1	N	49
AREA_NAME	CHAR	S	C	N	3	N	1	N	51
PHYSICAL_READS	DECIMAL	S	D	Y	8	N	1	N	54

SQL NAME	SQL TYPE	CLS	TYP	SGN	LNG	NUL	REP	RDF	DSP
PHYSICAL_WRITES	DECIMAL	S	D	Y	8	N	1	N	62
LOGICAL_READS	DECIMAL	S	D	Y	8	N	1	N	70
LOGICAL_WRITES	DECIMAL	S	D	Y	8	N	1	N	78

Primary KEY columns are as follows:

- MUF_NAME
- DBID
- AREA_NAME
- S_MUF_ENABLE
- S_MUF_DATETIME

Secondary KEY columns are as follows:

- MUF_NAME
- S_MUF_ENABLE
- S_MUF_DATETIME

S_MUF_CBS (MCB)

SQL NAME	SQL TYPE	CLS	TYP	SGN	LNG	NUL	REP	RDF	DSP
S_MUF_ENABLE	SQL-STMP	S	B	N	10	N	1	N	0
S_MUF_DATETIME	SQL-STMP	S	B	N	10	N	1	N	10
S_MUF_TOTALROW	CHAR	S	C	N	1	N	1	N	20
MUF_NAME	CHAR	S	C	N	8	N	1	N	21
BUF_CUR_PERC_FULL	SMALLINT	S	B	N	2	N	1	N	29
BUF_MAX_PERC_FULL	SMALLINT	S	B	N	2	N	1	N	31
CUR_USED_SET_MEM	INTEGER	S	B	N	4	N	1	N	33
MAX_USED_SET_MEM	INTEGER	S	B	N	4	N	1	N	37
SETS_DEL_MAX_AGE	DECIMAL	S	D	Y	8	N	1	N	41
SET FETCHED	DECIMAL	S	D	Y	8	N	1	N	49
SETS PROCESSED	DECIMAL	S	D	Y	8	N	1	N	57
SET_SPILLED	DECIMAL	S	D	Y	8	N	1	N	65
SETS_TEMP_INDEX	DECIMAL	S	D	Y	8	N	1	N	73

SQL NAME	SQL TYPE	CLS	TYP	SGN	LNG	NUL	REP	RDF	DSP
TEMP_INDEX_ENTRIES	DECIMAL	S	D	Y	8	N	1	N	81
SPILLED_JOB_NAME	CHAR	S	C	N	8	Y	1	N	90
SPILLED_UIB	CHAR	S	C	N	32	Y	1	N	99
DURATION_AT_SPILL	CHAR	S	C	N	6	Y	1	N	132

Primary KEY columns are as follows:

- MUF_NAME
- S_MUF_ENABLE
- S_MUF_DATETIME

Secondary KEY columns are as follows:

- MUF_NAME
- S_MUF_ENABLE
- S_MUF_DATETIME

S_MUF_CONFIG (MFS)

SQL NAME	SQL TYPE	CLS	TYP	SGN	LNG	NUL	REP	RDF	DSP
S_MUF_ENABLE	SQL-STMP	S	B	N	10	N	1	N	0
S_MUF_DATETIME	SQL-STMP	S	B	N	10	N	1	N	10
S_MUF_TOTALROW	CHAR	S	C	N	1	N	1	N	20
MUF_NAME	CHAR	S	C	N	8	N	1	N	21
CONFIG_COMMAND	CHAR	S	C	N	25	N	`	N	29
CONFIG_PARMS	CHAR	S	C	N	65	N	1	N	54

Primary KEY columns are as follows:

- MUF_NAME
- CONFIG_COMMAND
- S_MUF_ENABLE
- S_MUF_DATETIME

Secondary KEY columns are as follows:

- MUF_NAME
- S_MUF_ENABLE
- S_MUF_DATETIME

S_MUF_COVEREDVIRTUAL (MFC)

SQL NAME	SQL TYPE	CLS	TYP	SGN	LNG	NUL	REP	RDF	DSP
S_MUF_ENABLE	SQL-STMP	S	B	N	10	N	1	N	0
S_MUF_DATETIME	SQL-STMP	S	B	N	10	N	1	N	10
S_MUF_TOTALROW	CHAR	S	C	N	1	N	1	N	20
MUF_NAME	CHAR	S	C	N	8	N	1	N	21
DBID	SMALLINT	S	B	Y	2	N	1	N	29
AREA_NAME	CHAR	S	C	N	3	N	1	N	31
VIRTUAL_COVERED	CHAR	S	C	N	1	N	1	N	34
FIRST_ACTIVE	CHAR	S	C	N	1	Y	1	N	36
DATASPACE	CHAR	S	C	N	1	Y	1	N	38
MEMORY_SIZE	DECIMAL	S	D	Y	6	N	1	N	39
BLOCKS_MAXIMUM	INTEGER	S	B	Y	4	Y	1	N	46
BLOCKS_CURRENT	INTEGER	S	B	Y	4	Y	1	N	51
TOTAL_READS	DECIMAL	S	D	Y	8	Y	1	N	56
MRDF_READS	DECIMAL	S	D	Y	8	Y	1	N	65
VIRTUAL_WRITES	DECIMAL	S	D	Y	8	Y	1	N	74
ACTIVE_READS	DECIMAL	S	D	Y	8	Y	1	N	83
NONFIRST_READS	DECIMAL	S	D	Y	8	Y	1	N	92
HIGH_VIRTUAL_BLOCK	INTEGER	S	B	Y	4	Y	1	N	101
BEGIN_BLOCK	SMALLINT	S	B	Y	2	Y	1	N	106

Primary KEY columns are as follows:

- MUF_NAME
- DBID
- AREA_NAME
- VIRTUAL_COVERED
- S_MUF_ENABLE
- S_MUF_DATETIME

Secondary KEY columns are as follows:

- MUF_NAME
- S_MUF_ENABLE
- S_MUF_DATETIME

S_MUF_INTERNAL_STATS (MFV)

SQL NAME	SQL TYPE	CLS	TYP	SGN	LNG	NUL	REP	RDF	DSP
S_MUF_ENABLE	SQL-STMP	S	B	N	10	N	1	N	0
S_MUF_DATETIME	SQL-STMP	S	B	N	10	N	1	N	10
S_MUF_TOTALROW	CHAR	S	C	N	1	N	1	N	20
MUF_NAME	CHAR	S	C	N	8	N	1	N	21
TIMES_READ_1	DECIMAL	S	D	Y	8	N	1	N	29
TIMES_READ_2_8	DECIMAL	S	D	Y	8	N	1	N	37
TIMES_READ_9_32	DECIMAL	S	D	Y	8	N	1	N	45
TIMES_READ_33_99	DECIMAL	S	D	Y	8	N	1	N	53
TIMES_WRITE_1	DECIMAL	S	D	Y	8	N	1	N	61
TIMES_WRITE_2_12	DECIMAL	S	D	Y	8	N	1	N	69
TIMES_WRITE_13_24	DECIMAL	S	D	Y	8	N	1	N	77
TIMES_WRITE_25_48	DECIMAL	S	D	Y	8	N	1	N	85
BREAKS_DONE	DECIMAL	S	D	Y	8	N	1	N	93
IO_MEM_SHORT	DECIMAL	S	D	Y	8	N	1	N	101

Primary KEY columns are as follows:

- MUF_NAME
- S_MUF_ENABLE
- S_MUF_DATETIME

Secondary KEY columns are as follows:

- MUF_NAME
- S_MUF_ENABLE
- S_MUF_DATETIME

S_MUF_OPTIONS (MFO)

SQL NAME	SQL TYPE	CLS	TYP	SGN	LNG	NUL	REP	RDF	DSP
S_MUF_ENABLE	SQL-STMP	S	B	N	10	N	1	N	0
S_MUF_DATETIME	SQL-STMP	S	B	N	10	N	1	N	10
S_MUF_TOTALROW	CHAR	S	C	N	1	N	1	N	20
MUF_NAME	CHAR	S	C	N	8	N	1	N	21
ACCTPRM_PRM_DBID	SMALLINT	S	B	Y	2	N	1	N	29
ACCTPRM_ANN_DBID	SMALLINT	S	B	Y	2	N	1	N	31
ACCTPRM_CPU_TIME	CHAR	S	C	N	1	N	1	N	33
AGENT	CHAR	S	C	N	1	N	1	N	34
AGENT_LOG_LEVEL	SMALLINT	S	B	Y	2	N	1	N	35
AGENT_USE_CONFIG	CHAR	S	C	N	1	N	1	N	37
BREAK	INTEGER	S	B	Y	4	N	1	N	38
CBS_DBID_TEMP	SMALLINT	S	B	Y	2	N	1	N	42
CBS_BUFFER	INTEGER	S	B	Y	4	N	1	N	44
CBS_MAXSTEN	INTEGER	S	B	Y	4	N	1	N	48
CBS_MAXSTIO	INTEGER	S	B	Y	4	N	1	N	52
CBS_TIME_DELETE	SMALLINT	S	B	Y	2	N	1	N	56
CBS_DBID_HEURISTIC	SMALLINT	S	B	Y	2	N	1	N	58
CSAFREE	CHAR	S	C	N	1	N	1	N	60
DATAPOOL_DATALN	SMALLINT	S	B	Y	2	N	1	N	61
DATAPOOL_DATANO	INTEGER	S	B	Y	4	N	1	N	63

SQL NAME	SQL TYPE	CLS	TYP	SGN	LNG	NUL	REP	RDF	DSP
DATAPOOL_DATALN2	SMALLINT	S	B	Y	2	N	1	N	67
DATAPOOL_DATANO2	INTEGER	S	B	Y	4	N	1	N	69
DATASPACE_SIZE	INTEGER	S	B	Y	4	N	1	N	73
DATASPACE_CONNECT	CHAR	S	C	N	1	N	1	N	77
DATETIME_DATE	CHAR	S	C	N	3	N	1	N	78
DATETIME_TIME	CHAR	S	C	N	3	N	1	N	81
DICTIONARY_DD_DBID	SMALLINT	S	B	Y	2	N	1	N	84
DICTIONARY_DDDDBID	SMALLINT	S	B	Y	2	N	1	N	86
EXCTLNO	INTEGER	S	B	Y	4	N	1	N	88
EXPAND_LENGTH	INTEGER	S	B	Y	4	N	1	N	92
EXPAND_NUMBER	INTEGER	S	B	Y	4	N	1	N	96
FLEXPOOL_IXXNO	INTEGER	S	B	Y	4	N	1	N	100
FLEXPOOL_DXXNO	INTEGER	S	B	Y	4	N	1	N	104
FLEXPOOL_DATANO	INTEGER	S	B	Y	4	N	1	N	108
FLEXPOOL_DATANO2	INTEGER	S	B	Y	4	N	1	N	112
IOTASK_MAXIMUM	SMALLINT	S	B	Y	2	N	1	N	116
IOTASK_CURRENT	SMALLINT	S	B	Y	2	N	1	N	118
LOGOPTION_OPN_CLS	CHAR	S	C	N	1	N	1	N	120
LOGPEND	SMALLINT	S	B	Y	2	N	1	N	121
LOGPOOL	SMALLINT	S	B	Y	2	N	1	N	123
LOGRCV	CHAR	S	C	N	5	N	1	N	125
LOGRSYS	SMALLINT	S	B	Y	2	N	1	N	130
LOGSPILL_A	SMALLINT	S	B	Y	2	N	1	N	132
LOGSPILL_B	SMALLINT	S	B	Y	2	N	1	N	134
LOGSPILL_C	SMALLINT	S	B	Y	2	N	1	N	136
LOGSPILL_D	SMALLINT	S	B	Y	2	N	1	N	138
LOGSPILL_E	SMALLINT	S	B	Y	2	N	1	N	140
MAXELRQ	SMALLINT	S	B	Y	2	N	1	N	142
MUF_RUN_UNIT_GROUP	SMALLINT	S	B	Y	2	N	1	N	144
MUF_ENDED_NO_LOG	CHAR	S	C	N	1	N	1	N	146
MUFMSG_A	CHAR	S	C	N	1	N	1	N	147

SQL NAME	SQL TYPE	CLS	TYP	SGN	LNG	NUL	REP	RDF	DSP
MUFMSG_B	CHAR	S	C	N	1	N	1	N	148
MUFMSG_C	CHAR	S	C	N	1	N	1	N	149
MUFPLEX_NAME	CHAR	S	C	N	8	N	1	N	150
MUFPLEX_NUMBER	SMALLINT	S	B	Y	2	N	1	N	158
MUFPLEX_LOCKS	INTEGER	S	B	Y	4	N	1	N	160
MUFPLEX_MAX_TASKS	SMALLINT	S	B	Y	2	N	1	N	164
NONSWAP	CHAR	S	C	N	1	N	1	N	166
PLANSEC_CHECKWHO	CHAR	S	C	N	8	N	1	N	167
PLANSEC_CHECKWHEN	CHAR	S	C	N	8	N	1	N	175
PLANSEC_CHECKPLAN	CHAR	S	C	N	1	N	1	N	183
PROCEDURE_CACHE	INTEGER	S	B	Y	4	N	1	N	184
PROCEDURE_NESTS	SMALLINT	S	B	Y	2	N	1	N	188
PROCEDURE_TCBS	SMALLINT	S	B	Y	2	N	1	N	190
PXX_STATS	CHAR	S	C	N	6	N	1	N	192
READAHD	CHAR	S	C	N	8	N	1	N	198
RESTART_IGNORE	CHAR	S	C	N	1	N	1	N	206
RESTART_ACCESS_OFF	CHAR	S	C	N	1	N	1	N	207
RRS	CHAR	S	C	N	8	N	1	N	208
RXX_UNIT_NAME	CHAR	S	C	N	8	N	1	N	216
RXX_VOLUME_COUNT	SMALLINT	S	B	Y	2	N	1	N	224
RXX_UNIT_COUNT	SMALLINT	S	B	Y	2	N	1	N	226
RXXROLLBACK	CHAR	S	C	N	1	N	1	N	228
SMPTASK_MAXIMUM	SMALLINT	S	B	Y	2	N	1	N	229
SMPTASK_CURRENT	SMALLINT	S	B	Y	2	N	1	N	231
SMPTASK_READY_TASK	SMALLINT	S	B	Y	2	N	1	N	233
SQLDEFAULT_DBID	SMALLINT	S	B	Y	2	N	1	N	235
SQLDEFAULT_NAME	CHAR	S	C	N	32	N	1	N	237
SQLOPTION_OPTION	CHAR	S	C	N	1	N	1	N	269
SQLOPTION_TTMID	SMALLINT	S	B	Y	2	N	1	N	270
SQLOPTION_MODE	CHAR	S	C	N	7	N	1	N	272
SQLOPTION_TIMEOUT	SMALLINT	S	B	Y	2	N	1	N	279

SQL NAME	SQL TYPE	CLS	TYP	SGN	LNG	NUL	REP	RDF	DSP
SQLOPTION_VIEW	CHAR	S	C	N	1	N	1	N	281
SQLOPTION_BOTH	CHAR	S	C	N	1	N	1	N	282
STAR_TASKS	SMALLINT	S	B	Y	2	N	1	N	283
STAR_NODE	CHAR	S	C	N	8	N	1	N	285
STAR_SIZE	INTEGER	S	B	Y	4	N	1	N	293
STAR_MAX_NODES	SMALLINT	S	B	Y	2	N	1	N	297
STAR_TIME_BATCH	SMALLINT	S	B	Y	2	N	1	N	299
STAR_TIME_CICS	SMALLINT	S	B	Y	2	N	1	N	301
STATBFR	INTEGER	S	B	Y	4	N	1	N	303
SUBTASK	CHAR	S	C	N	8	N	1	N	307
SYSOUT_CLASS	CHAR	S	C	N	1	N	1	N	315
SYSPOOL_CXXNO	SMALLINT	S	B	Y	2	N	1	N	316
SYSPOOL_IXXNO	INTEGER	S	B	Y	4	N	1	N	318
SYSPOOL_DXXNO	INTEGER	S	B	Y	4	N	1	N	322
SYSPOOL_INDEX	SMALLINT	S	B	Y	2	N	1	N	326
SYSTEMDBID_DBID	SMALLINT	S	B	Y	2	N	1	N	328
SYSTEMDBID_SYSOUT	CHAR	S	C	N	1	N	1	N	330
TASKS_NUMBER	SMALLINT	S	B	Y	2	N	1	N	331
TASKS_SIZE	INTEGER	S	B	Y	4	N	1	N	333
TASKS_VAE_NUMBER	SMALLINT	S	B	Y	2	N	1	N	337
TASKS_VM_NUMBER	SMALLINT	S	B	Y	2	N	1	N	339
TASKS_XCF_NUMBER	SMALLINT	S	B	Y	2	N	1	N	341

Primary KEY columns are as follows:

- MUF_NAME
- S_MUF_ENABLE
- S_MUF_DATETIME

Secondary KEY columns are as follows:

- MUF_NAME
- S_MUF_ENABLE
- S_MUF_DATETIME

S_MUF_RETURN_CODES (MFU)

SQL NAME	SQL TYPE	CLS	TYP	SGN	LNG	NUL	REP	RDF	DSP
S_MUF_ENABLE	SQL-STMP	S	B	N	10	N	1	N	0
S_MUF_DATETIME	SQL-STMP	S	B	N	10	N	1	N	10
S_MUF_TOTALROW	CHAR	S	C	N	1	N	1	N	20
MUF_NAME	CHAR	S	C	N	8	N	1	N	21
EXTERNAL_CODE	CHAR	S	C	N	2	N	1	N	29
INTERNAL_CODE	CHAR	S	C	N	3	N	1	N	31
COUNT_CODES	DECIMAL	S	D	Y	8	N	1	N	34

Primary KEY columns are as follows:

- MUF_NAME
- EXTERNAL_CODE
- INTERNAL_CODE
- S_MUF_ENABLE
- S_MUF_DATETIME

Secondary KEY columns are as follows:

- MUF_NAME
- S_MUF_ENABLE
- S_MUF_DATETIME

S_MUF_SRB_ZIIP (MZI)

SQL NAME	SQL TYPE	CLS	TYP	SGN	LNG	NUL	REP	RDF	DSP
S_MUF_ENABLE	SQL-STMP	S	B	N	10	N	1	N	0
S_MUF_DATETIME	SQL-STMP	S	B	N	10	N	1	N	10
S_MUF_TOTALROW	CHAR	S	C	N	1	N	1	N	20
MUF_NAME	CHAR	S	C	N	8	N	1	N	21
SEQUENCE_NUMBER	SMALLINT	S	B	N	2	N	1	N	29

SQL NAME	SQL TYPE	CLS	TYP	SGN	LNG	NUL	REP	RDF	DSP
ENCLAVE_SECONDS	DECIMAL	S	D	Y	8	N	1	N	31
ZIIP_ON_CP_SECONDS	DECIMAL	S	D	Y	8	N	1	N	39
ZIIP_QUAL_SECONDS	DECIMAL	S	D	Y	8	N	1	N	47
ZIIP SECONDS	DECIMAL	S	D	Y	8	N	1	N	55
ZIIP_PERCENT	SMALLINT	S	B	N	2	N	1	N	63

Primary KEY columns are as follows:

- MUF_NAME
- SEQUENCE_NUMBER
- S_MUF_ENABLE
- S_MUF_DATETIME

Secondary KEY columns are as follows:

- MUF_NAME
- S_MUF_ENABLE
- S_MUF_DATETIME

S_MUF_SYSTEM_STATS (MFS)

SQL NAME	SQL TYPE	CLS	TYP	SGN	LNG	NUL	REP	RDF	DSP
S_MUF_ENABLE	SQL-STMP	S	B	N	10	N	1	N	0
S_MUF_DATETIME	SQL-STMP	S	B	N	10	N	1	N	10
S_MUF_TOTALROW	CHAR	S	C	N	1	N	1	N	20
MUF_NAME	CHAR	S	C	N	8	N	1	N	21
BEGIN_TIME	SQL-STMP	S	B	N	10	N	1	N	29
CURRENT_DATETIME	SQL-STMP	S	B	N	10	N	1	N	39
CURRENT_CPU	DECIMAL	S	D	Y	8	N	1	N	49
RQ_PENDING_1_10	DECIMAL	S	D	Y	8	N	1	N	57
RQ_PENDING_11_20	DECIMAL	S	D	Y	8	N	1	N	65
RQ_PENDING_21_30	DECIMAL	S	D	Y	8	N	1	N	73
RQ_PENDING_31_40	DECIMAL	S	D	Y	8	N	1	N	81

SQL NAME	SQL TYPE	CLS	TYP	SGN	LNG	NUL	REP	RDF	DSP
RQ_PENDING_41_50	DECIMAL	S	D	Y	8	N	1	N	89
RQ_PENDING_51_60	DECIMAL	S	D	Y	8	N	1	N	97
RQ_PENDING_61_70	DECIMAL	S	D	Y	8	N	1	N	105
RQ_PENDING_71_80	DECIMAL	S	D	Y	8	N	1	N	113
RQ_PENDING_81_90	DECIMAL	S	D	Y	8	N	1	N	121
RQ_PENDING_91_100	DECIMAL	S	D	Y	8	N	1	N	129
RQ_PENDING_GT_100	DECIMAL	S	D	Y	8	N	1	N	137
DB_REQUESTS	DECIMAL	S	D	Y	8	N	1	N	145
SQL_REQUESTS	DECIMAL	S	D	Y	8	N	1	N	153
DATA_MGR_REQUESTS	DECIMAL	S	D	Y	8	N	1	N	161
PHYSICAL_READS	DECIMAL	S	D	Y	8	N	1	N	169
PHYSICAL_WRITES	DECIMAL	S	D	Y	8	N	1	N	177
RECORD_LOCKS	DECIMAL	S	D	Y	8	N	1	N	185
WAIT_LOCK	DECIMAL	S	D	Y	8	N	1	N	193
KEY_ELEMENT_REORG	DECIMAL	S	D	Y	8	N	1	N	201
NO_IXX_BUFFER	DECIMAL	S	D	Y	8	N	1	N	209
NO_DXX_BUFFER	DECIMAL	S	D	Y	8	N	1	N	217
NO_DATA_BUFFER	DECIMAL	S	D	Y	8	N	1	N	225
NO_EXPAND_BUFFER	DECIMAL	S	D	Y	8	N	1	N	233
IXX_SEQUENCE	DECIMAL	S	D	Y	8	N	1	N	241
DXX_SEQUENCE	DECIMAL	S	D	Y	8	N	1	N	249
DATA_SEQUENCE	DECIMAL	S	D	Y	8	N	1	N	257
EXPAND_SEQUENCE	DECIMAL	S	D	Y	8	N	1	N	265
IXX_USED_1	DECIMAL	S	D	Y	8	N	1	N	273
IXX_USED_2	DECIMAL	S	D	Y	8	N	1	N	281
IXX_USED_3	DECIMAL	S	D	Y	8	N	1	N	289
IXX_USED_4	DECIMAL	S	D	Y	8	N	1	N	297
IXX_USED_5	DECIMAL	S	D	Y	8	N	1	N	305
DXX_USED_1	DECIMAL	S	D	Y	8	N	1	N	313
DXX_USED_2	DECIMAL	S	D	Y	8	N	1	N	321
DXX_USED_3	DECIMAL	S	D	Y	8	N	1	N	329

SQL NAME	SQL TYPE	CLS	TYP	SGN	LNG	NUL	REP	RDF	DSP
DXX_USED_4	DECIMAL	S	D	Y	8	N	1	N	337
DXX_USED_5	DECIMAL	S	D	Y	8	N	1	N	345
DATA_USED_1	DECIMAL	S	D	Y	8	N	1	N	353
DATA_USED_2	DECIMAL	S	D	Y	8	N	1	N	361
DATA_USED_3	DECIMAL	S	D	Y	8	N	1	N	369
DATA_USED_4	DECIMAL	S	D	Y	8	N	1	N	377
DATA_USED_5	DECIMAL	S	D	Y	8	N	1	N	385
SEQ_READ_AHEAD	DECIMAL	S	D	Y	8	N	1	N	393
LOG_PERCENT_FULL	INTEGER	S	B	Y	4	N	1	N	401
SECONDARY_CONFLICT	DECIMAL	S	D	Y	8	N	1	N	405
WAIT_TASK	DECIMAL	S	D	Y	8	N	1	N	413
WAIT_REQUEST	DECIMAL	S	D	Y	8	N	1	N	421
WAIT_SPILL	DECIMAL	S	D	Y	8	N	1	N	429
WAIT_ACCOUNTING	DECIMAL	S	D	Y	8	N	1	N	437
WAIT_SECURITY	DECIMAL	S	D	Y	8	N	1	N	445
INDEX_Q_PROCESSED	DECIMAL	S	D	Y	8	N	1	N	453
INDEX_Q_OVERFLOW	DECIMAL	S	D	Y	8	N	1	N	461
SPLIT_IXX	DECIMAL	S	D	Y	8	N	1	N	469
SPLIT_DXX	DECIMAL	S	D	Y	8	N	1	N	477
DELETED_BLOCKS	DECIMAL	S	D	Y	8	N	1	N	485
CBS_SPILLS	DECIMAL	S	D	Y	8	N	1	N	493
CBS_ENTRIES	INTEGER	S	B	Y	4	N	1	N	501
CBS_ENTRIES_USED	INTEGER	S	B	Y	4	N	1	N	505
SETS_PROCESSED	DECIMAL	S	D	Y	8	N	1	N	509
SETS_WITH_T_INDEX	DECIMAL	S	D	Y	8	N	1	N	517
TEMP_INDEX_ENTRIES	DECIMAL	S	D	Y	8	N	1	N	525
CBS_MEMORY_ALLOCS	DECIMAL	S	D	Y	8	N	1	N	533
LOGPEND_WRITES	DECIMAL	S	D	Y	8	N	1	N	541
DATASECURE_ONE	INTEGER	S	B	Y	4	N	1	N	549
DATASECURE_TWO	INTEGER	S	B	Y	4	N	1	N	553
REEXPANDS	DECIMAL	S	D	Y	8	N	1	N	557

SQL NAME	SQL TYPE	CLS	TYP	SGN	LNG	NUL	REP	RDF	DSP
NO_TXB_BUFFER	DECIMAL	S	D	Y	8	N	1	N	565
INDIRECT_WAIT_IO	DECIMAL	S	D	Y	8	N	1	N	573
LOG_WRITE_CONTROL	DECIMAL	S	D	Y	8	N	1	N	581
LOG_WRITE_FULL	DECIMAL	S	D	Y	8	N	1	N	589
LOG_WRITE_COMMAND	DECIMAL	S	D	Y	8	N	1	N	597
LOG_WRITE_OTHER	DECIMAL	S	D	Y	8	N	1	N	605
LOG_WRITE_TXB	DECIMAL	S	D	Y	8	N	1	N	613
LOG_WRITE_2_PHASE	DECIMAL	S	D	Y	8	N	1	N	621
DATA2_USED_1	DECIMAL	S	D	Y	8	N	1	N	629
DATA2_USED_2	DECIMAL	S	D	Y	8	N	1	N	637
DATA2_USED_3	DECIMAL	S	D	Y	8	N	1	N	645
DATA2_USED_4	DECIMAL	S	D	Y	8	N	1	N	653
DATA2_USED_5	DECIMAL	S	D	Y	8	N	1	N	661

Primary KEY columns are as follows:

- MUF_NAME
- S_MUF_ENABLE
- S_MUF_DATETIME

Secondary KEY columns are as follows:

- MUF_NAME
- S_MUF_ENABLE
- S_MUF_DATETIME

S_MUF_TABLE_STATS (MFT)

SQL NAME	SQL TYPE	CLS	TYP	SGN	LNG	NUL	REP	RDF	DSP
S_MUF_ENABLE	SQL-STMP	S	B	N	10	N	1	N	0
S_MUF_DATETIME	SQL-STMP	S	B	N	10	N	1	N	10
S_MUF_TOTALROW	CHAR	S	C	N	1	N	1	N	20
MUF_NAME	CHAR	S	C	N	8	N	1	N	21

SQL NAME	SQL TYPE	CLS	TYP	SGN	LNG	NUL	REP	RDF	DSP
BEGIN_TIME	SQL-STMP	S	B	N	10	N	1	N	29
CURRENT_DATETIME	SQL-STMP	S	B	N	10	N	1	N	39
DBID	SMALLINT	S	B	Y	2	N	1	N	49
TABLE_NAME	CHAR	S	C	N	3	N	1	N	51
AREA_NAME	CHAR	S	C	N	3	N	1	N	54
TOTAL_REQUESTS	DECIMAL	S	D	Y	8	N	1	N	57
TOTAL_READS	DECIMAL	S	D	Y	8	N	1	N	65
TOTAL_ADDS	DECIMAL	S	D	Y	8	N	1	N	73
TOTAL_DELETES	DECIMAL	S	D	Y	8	N	1	N	81
TOTAL_UPDATES	DECIMAL	S	D	Y	8	N	1	N	89

Primary KEY columns are as follows:

- MUF_NAME
- DBID
- TABLE_NAME
- S_MUF_ENABLE
- S_MUF_DATETIME

Secondary KEY columns are as follows:

- MUF_NAME
- S_MUF_ENABLE
- S_MUF_DATETIME

S_MUF_TCB_OR_SRB (MTC)

SQL NAME	SQL TYPE	CLS	TYP	SGN	LNG	NUL	REP	RDF	DSP
S_MUF_ENABLE	SQL-STMP	S	B	N	10	N	1	N	0
S_MUF_DATETIME	SQL-STMP	S	B	N	10	N	1	N	10
S_MUF_TOTALROW	CHAR	S	C	N	1	N	1	N	20
MUF_NAME	CHAR	S	C	N	8	N	1	N	21
TASK_TYPE	CHAR	S	C	N	4	N	1	N	29

SQL NAME	SQL TYPE	CLS	TYP	SGN	LNG	NUL	REP	RDF	DSP
PROGRAM_NAME	CHAR	S	C	N	8	N	1	N	33
SEQUENCE_NUMBER	SMALLINT	S	B	N	2	N	1	N	41
CPU_SECONDS	DECIMAL	S	D	Y	8	N	1	N	43
TIMES_USED	DECIMAL	S	D	Y	8	N	1	N	51
TIMES_POSTED	DECIMAL	S	D	Y	8	N	1	N	59
PHYSICAL_IO	DECIMAL	S	D	Y	8	N	1	N	67

Primary KEY columns are as follows:

- MUF_NAME
- PROGRAM_NAME
- SEQUENCE_NUMBER
- S_MUF_ENABLE
- S_MUF_DATETIME

Secondary KEY columns are as follows:

- MUF_NAME
- S_MUF_ENABLE
- S_MUF_DATETIME

S_SQL_MISC_STATS (SQM)

SQL NAME	SQL TYPE	CLS	TYP	SGN	LNG	NUL	REP	RDF	DSP
S_MUF_ENABLE	SQL-STMP	S	B	N	10	N	1	N	0
S_MUF_DATETIME	SQL-STMP	S	B	N	10	N	1	N	10
S_MUF_TOTALROW	CHAR	S	C	N	1	N	1	N	20
MUF_NAME	CHAR	S	C	N	8	N	1	N	21
PROC_CACHE_REUSE	DECIMAL	S	D	Y	8	N	1	N	29
PROC_FROM_CALL	DECIMAL	S	D	Y	8	N	1	N	37
PROC_FROM_TRIGGER	DECIMAL	S	D	Y	8	N	1	N	45
PROC_QUEUED	DECIMAL	S	D	Y	8	N	1	N	53
PROC_SQL_STMTS	DECIMAL	S	D	Y	8	N	1	N	61

SQL NAME	SQL TYPE	CLS	TYP	SGN	LNG	NUL	REP	RDF	DSP
PROC_FAILURES	DECIMAL	S	D	Y	8	N	1	N	69
PROC_NESTS	DECIMAL	S	D	Y	8	N	1	N	77

Primary KEY columns are as follows:

- MUF_NAME
- S_MUF_ENABLE
- S_MUF_DATETIME

Secondary KEY columns are as follows:

- MUF_NAME
- S_MUF_ENABLE
- S_MUF_DATETIME

S_SQL_SQLCODES (SQQ)

SQL NAME	SQL TYPE	CLS	TYP	SGN	LNG	NUL	REP	RDF	DSP
S_MUF_ENABLE	SQL-STMP	S	B	N	10	N	1	N	0
S_MUF_DATETIME	SQL-STMP	S	B	N	10	N	1	N	10
S_MUF_TOTALROW	CHAR	S	C	N	1	N	1	N	20
MUF_NAME	CHAR	S	C	N	8	N	1	N	21
SQL_CODE	INTEGER	S	B	Y	4	N	1	N	29
CODE_COUNT	DECIMAL	S	D	Y	8	N	1	N	33

Primary KEY columns are as follows:

- MUF_NAME
- SQL_CODE
- S_MUF_ENABLE
- S_MUF_DATETIME

Secondary KEY columns are as follows:

- MUF_NAME
- S_MUF_ENABLE
- S_MUF_DATETIME

Chapter 9: AutoCollect DELTA Database

Delta tables are constructed from data found on the Snapshot tables. Additionally, each Delta table has a set of identifying columns that are added to the front of each row to identify the source of the Delta data, such as MUF, time period, and so on.

The DELTA database contains one detail table that corresponds to the tables in the Snapshot database. The DELTA database also contains two ratio tables (the BVM and the BVD) that are used to summarize the key data statistics found in the Delta rowsets and present the statistics as a set of ratios.

Delta Rowsets

For each Delta rowset created, one or more rows are added to the Delta tables. Delta tables that have multiple rows per rowset will have one or more total rows also created.

SQL Name	Table	Detail Rows	Total Rows
D_BUSINESS_VALUE_DETAILS	BVD	1	0
D_BUSINESS_VALUE_METRICS	BVM	1	0
D_MUF_ACCOUNTING	MFY	1 per acct table*	1
D_MUF_AREA_STATS	MFA	1 per area*	1
D_MUF_COVEREDVIRTUAL	MFC	1 per definition*	2
D_MUF_CBS	MCB	1	0
D_MUF_CONFIG	MFC	1 peer configuration parameter*	0
D_MUF_INTERNAL_STATS	MFV	1	0
D_MUF_OPTIONS	MFO	1	0
D_MUF_RETURN_CODES	MFU	1 per DB return code*	1
D_MUF_SMP_STATS	MFW	1	0
D_MUF_SRB_ZIIP	MFC	1 per SRB*	1
D_MUF_SYSTEM_STATS	MFS	1	1
D_MUF_TABLE_STATS	MFT	1 per table*	1
D_MUF_TCB_OR_SRB	MTC	1 per subtask*	1
D_SQL_MISC_STATS	SQM	1	0

SQL Name	Table	Detail Rows	Total Rows
D_SQL_SQLCODES	SQQ	1 per SQL code*	1

Note: Asterisk (*) indicates that the number of rows is dependent on the active number of DST participants at the time the Snapshot (used to build the Deltas) was taken.

Using the example above where the customer created 8 Snapshot rowsets in a given week, the Delta rows would be as follows:

- One Delta rowset for every Snapshot rowset
- One Delta rowset per MUF termination
    - Each Delta rowset also includes 1 BVM and 1 BVD row
    - Rowsets = 344 rows
- 9 Delta rowsets = 3096 Delta rows created per week.

Delta Identification Columns

In addition to the DST columns, each table has identification columns included at the front of each row. For more information about the contents of the Delta row identification columns, see the earlier sections of this guide.

- D_MUF_ENABLE
- D_MUF_START_DATETIME
- D_MUF_END_DATETIME
- D_MUF_TYPE
- D_MUF_TOTALROW
- D_MUF_DURATION

Delta Column information

Following the identification columns, the balance of the Delta rows contains columns extracted from the Snapshot tables. In most cases, these columns have the same column name and the same definition as the columns in the DST tables. For a DST row, the value for the column is based off the activity since the beginning of the MUF enable. For a Delta row, the column value is specific to the time period that the Delta represents.

In the table layouts provided next, each column is prefixed with an indicator. These indicators tell you how each Delta column's content was constructed.

I

Identification column built during Delta row creation as follows:

- For the Interval or Last rowset, the value is copied from the ending Snapshot row.
- For the Summary, AVGPERF or Baseline rowset, the value is copied from the last delta rowset selected for processing.

IS

Identification column built during Delta row creation as follows:

- For the Interval or Last rowset, the value is copied from the starting Snapshot row.
- For the Summary, AVGPERF or Baseline rowset, the value is copied from the earliest delta rowset selected for processing.

L

Fixed value taken from the ending input row as follows:

- For the Interval or Last rowset, the value is copied from the ending Snapshot row.
- For the Summary, AVGPERF or Baseline rowset, the value is copied from the last row selected for processing.

D

Value was a "delta value" calculated as follows:

- For the first Interval rowset (created from first Snapshot after MUF enable), the delta values are calculated by subtracting "0" from the Snapshot value.
- For each Interval rowset after the "first" rowset processing, the value is calculated by subtracting the beginning Snapshot column/row value from the ending Snapshot column/row value.
- For the Last rowset (created for last Snapshot per MUF execution and is to represent the entire MUF execution), the delta values are calculated by subtracting "0" from the Snapshot column/row value.

- For a Summary rowset, the value is calculated by adding the values from each of the Delta rows selected to be summarized into this new Delta rowset.
- For a Baseline rowset, the value is calculated by adding the values from each of the Delta rows selected to be processed. Once all rows are added in, the values are divided by the number of rowsets processed to build a Baseline average.
- For an AVGPERF, the value is calculated by adding the values from each of the Delta rows selected to be processed. Once all rows are added in, the value is divided by the duration total hours processed to build an AVGPERF rowset based on one standard duration hour.

R

Value is a ratio value built by dividing one column by another column. Ratio columns are only built in the two special Delta ratio tables.

U

Value is assigned by the user using the OUTTAG option on selected AUTOCOLL functions.

Delta Tables

This section contains the Delta tables.

Column headings in this section are abbreviated as follows:

- ID = Identification
- CL = Class
- TY = Type
- SI = Numeric Sign
- LE = Length
- NU = Column can contain a null value
- RE = Column Repeat value (always 1)
- RD = Column Redefines another column (always N)
- DI = Displacement in the row

D_MUF_ACCOUNTING (MFY)

ID	SQL NAME	SQL TYPE	CL	TY	SI	LN	NU	RE	RD	DI
I	D_MUF_ENABLE	SQL-STMP	S	B	N	10	N	1	N	0

ID	SQL NAME	SQL TYPE	CL	TY	SI	LN	NU	RE	RD	DI
IS	D_MUF_START_DATETIME	SQL-STMP	S	B	N	10	N	1	N	10
I	D_MUF_END_DATETIME	SQL-STMP	S	B	N	10	N	1	N	20
I	D_MUF_TYPE	CHAR	S	C	N	10	N	1	N	30
I	D_MUF_TOTALROW	CHAR	S	C	N	1	N	1	N	40
D	D_MUF_DURATION	DECIMAL	S	D	Y	6	N	1	N	41
L	MUF_NAME	CHAR	S	C	N	8	N	1	N	47
L	DBID	SMALLINT	S	B	Y	2	N	1	N	55
L	TABLE_NAME	CHAR	S	C	N	3	N	1	N	57
L	BUFFER_SIZE	INTEGER	S	B	Y	4	N	1	N	60
L	TABLE_STATUS	CHAR	S	C	N	1	N	1	N	64
L	THRESHOLD	INTEGER	S	B	Y	4	N	1	N	65
L	SPILLING	CHAR	S	C	N	1	N	1	N	69
D	TOTAL_REQUESTS	DECIMAL	S	D	Y	8	N	1	N	70
D	PHYSICAL_READWRITE	DECIMAL	S	D	Y	8	N	1	N	78
D	LOCATES	DECIMAL	S	D	Y	8	N	1	N	86
D	TOTAL_ADDS	DECIMAL	S	D	Y	8	N	1	N	94
D	TOTAL_UPDATES	DECIMAL	S	D	Y	8	N	1	N	102
D	UNAVAILABLE	DECIMAL	S	D	Y	8	N	1	N	110
D	CONFLICTS	DECIMAL	S	D	Y	8	N	1	N	118
D	SKIPPED	DECIMAL	S	D	Y	8	N	1	N	126
D	ENTRIES	DECIMAL	S	D	Y	8	N	1	N	134
D	RECLAIMED	DECIMAL	S	D	Y	8	N	1	N	142
D	LOCATES_STARTED	DECIMAL	S	D	Y	8	N	1	N	150
D	SPILLS_THRESHOLD	DECIMAL	S	D	Y	8	N	1	N	158
D	SPILLS_INTERNAL	DECIMAL	S	D	Y	8	N	1	N	166
U	D_MUF_USERTAG	CHAR	S	C	N	16	N	1	N	174

Primary KEY columns are as follows:

- MUF_NAME
- DBID
- TABLE_NAME
- D_MUF_ENABLE
- D_MUF_END_DATETIME
- D_MUF_TYPE
- D_MUF_USERTAG
- D_MUF_START_DATETIME

Secondary KEY columns are as follows:

- MUF_NAME
- D_MUF_ENABLE
- D_MUF_END_DATETIME
- D_MUF_TYPE
- D_MUF_USERTAG
- D_MUF_START_DATETIME
- D_MUF_DURATION

D_MUF_AREA_STATS (MFA)

ID	SQL NAME	SQL TYPE	CL	TY	SI	LE	NU	RE	RD	DI
I	D_MUF_ENABLE	SQL-STMP	S	B	N	10	N	1	N	0
IS	D_MUF_START_DATETIME	SQL-STMP	S	B	N	10	N	1	N	10
I	D_MUF_END_DATETIME	SQL-STMP	S	B	N	10	N	1	N	20
I	D_MUF_TYPE	CHAR	S	C	N	10	N	1	N	30
I	D_MUF_TOTALROW	CJAR	S	C	N	1	N	1	N	40
D	D_MUF_DURATION	DECIMAL	S	D	Y	6	N	1	N	41
L	MUF_NAME	CHAR	S	C	N	8	N	1	N	47
L	BEGIN_TIME	SQL-STMP	S	B	N	10	N	1	N	55
L	CURRENT_DATETIME	SQL-STMP	S	B	N	10	N	1	N	65
L	DBID	SMALLINT	S	B	Y	2	N	1	N	75

ID	SQL NAME	SQL TYPE	CL	TY	SI	LE	NU	RE	RD	DI
L	AREA_NAME	CHAR	S	C	N	3	N	1	N	77
D	PHYSICAL_READS	DECIMAL	S	D	Y	8	N	1	N	80
D	PHYSICAL_WRITES	DECIMAL	S	D	Y	8	N	1	N	88
D	LOGICAL_READS	DECIMAL	S	D	Y	8	N	1	N	96
U	LOGICAL_WRITES	CHAR	S	D	Y	8	N	1	N	104
I	D_MUF_USERTAG		S	C	N	16	N	1	N	112

Primary KEY columns are as follows:

- MUF_NAME
- DBID
- AREA_NAME
- D_MUF_ENABLE
- D_MUF_END_DATETIME
- D_MUF_TYPE
- D_MUF_USERTAG
- D_MUF_START_DATETIME

Secondary KEY columns are as follows:

- MUF_NAME
- D_MUF_ENABLE
- D_MUF_END_DATETIME
- D_MUF_TYPE
- D_MUF_USERTAG
- D_MUF_START_DATETIME
- D_MUF_DURATION

D_MUF_CBS (MCB)

ID	SQL NAME	SQL TYPE	CL	TY	SI	LE	NU	RE	RD	DI
I	D_MUF_ENABLE	SQL-STMP	S	B	N	10	N	1	N	0
IS	D_MUF_START_DATETIME	SQL-STMP	S	B	N	10	N	1	N	10

ID	SQL NAME	SQL TYPE	CL	TY	SI	LE	NU	RE	RD	DI
I	D_MUF_END_DATETIME	SQL-STMP	S	B	N	10	N	1	N	20
I	D_MUF_TYPE	CHAR	S	C	N	10	N	1	N	30
I	D_MUF_TOTALROW	CJAR	S	C	N	1	N	1	N	40
D	D_MUF_DURATION	DECIMAL	S	D	Y	6	N	1	N	41
L	MUF_NAME	CHAR	S	C	N	8	N	1	N	47
L	BUF_CUR_PERC_FULL	SMALLINT	S	B	N	2	N	1	N	55
L	BUT_MAX_PERC_FULL	SMALLINT	S	B	N	2	N	1	N	57
L	CUR_USED_SET_MEM	INTEGER	S	B	N	4	N	1	N	59
L	MAX_USED_SET_MEM	INTEGER	S	B	N	4	N	1	N	63
L	SETS_DEL_MAX_AGE	DECIMAL	S	D	Y	8	N	1	N	67
D	SETS_FETCHED	DECIMAL	S	D	Y	8	N	1	N	75
D	SETS_PROCESSED	DECIMAL	S	D	Y	8	N	1	N	83
D	SETS_SPILLED	DECIMAL	S	D	7	8	N	1	N	91
D	SETS_TEMP_INDEX	DECIMAL	S	D	Y	8	N	1	N	99
D	TEMP_INDEX_ENTRIES	DECIMAL	S	D	Y	8	N	1	N	107
L	SPILLED_JOB_NAME	CHAR	S	C	N	8	Y	1	N	116
L	SPILLED_UIB	CHAR	S	C	N	32	Y	1	N	125
L	DURATION_AT_SPILL	CHAR	S	C	N	6	Y	1	N	158
U	D_MUF_USERTAG	CHAR	S	C	N	16	N	1	N	164

Primary KEY columns are as follows:

- MUF_NAME
- D_MUF_ENABLE
- D_MUF_END_DATETIME
- D_MUF_TYPE
- D_MUF_USERTAG
- D_MUF_START_DATETIME

Secondary KEY columns are as follows:

- MUF_NAME
- D_MUF_ENABLE
- D_MUF_END_DATETIME
- D_MUF_TYPE
- D_MUF_USERTAG
- D_MUF_START_DATETIME
- D_MUF_DURATION

D_MUF_CONFIG (MCF)

ID	SQL NAME	SQL TYPE	CL	TY	SI	LE	NU	RE	RD	DI
I	D_MUF_ENABLE	SQL-STMP	S	B	N	10	N	1	N	0
IS	D_MUF_START_DATETIME	SQL-STMP	S	B	N	10	N	1	N	10
I	D_MUF_END_DATETIME	SQL-STMP	S	B	N	10	N	1	N	20
I	D_MUF_TYPE	CHAR	S	C	N	10	N	1	N	30
I	D_MUF_TOTALROW	CJAR	S	C	N	1	N	1	N	40
D	D_MUF_DURATION	DECIMAL	S	D	Y	6	N	1	N	41
L	MUF_NAME	CHAR	S	C	N	8	N	1	N	47
L	CONFIG_COMMAND	CHAR	S	C	N	25	N	1	N	55
L	CONFIG_PARMS	CHAR	S	C	N	65	N	1	N	80
U	D_MUF_USERTAG	CHAR	S	C	N	16	N	1	N	145

Primary KEY columns are as follows:

- MUF_NAME
- CONFIG_COMMAND
- D_MUF_ENABLE
- D_MUF_END_DATETIME
- D_MUF_TYPE
- D_MUF_USER_TAG
- D_MUF_START_DATETIME

Secondary KEY columns are as follows:

- MUF_NAME
- D_MUF_ENABLE
- D_MUF_END_DATETIME
- D_MUF_TYPE
- D_MUF_USERTAG
- D_MUF_START_DATETIME
- D_MUF_DURATION

D_MUF_COVEREDVIRTUAL (MFC)

ID	SQL NAME	SQL TYPE	CL	TY	SI	LE	NU	RE	RD	DI
I	D_MUF_ENABLE	SQL-STMP	S	B	N	10	N	1	N	0
IS	D_MUF_START_DATETIME	SQL-STMP	S	B	N	10	N	1	N	10
I	D_MUF_END_DATETIME	SQL-STMP	S	B	N	10	N	1	N	20
I	D_MUF_TYPE	CHAR	S	C	N	10	N	1	N	30
I	D_MUF_TOTALROW	CHAR	S	C	N	1	N	1	N	40
D	D_MUF_DURATION	DECIMAL	S	D	Y	6	N	1	N	41
L	MUF_NAME	CHAR	S	C	N	8	N	1	N	47
L	DBID	SMALLINT	S	B	Y	2	N	1	N	55
L	AREA_NAME	CHAR	S	C	N	3	N	1	N	57
L	VIRTUAL_COVERED	CHAR	S	C	N	1	N	1	N	60
L	FIRST_ACTIVE	CHAR	S	C	N	1	Y	1	N	62

ID	SQL NAME	SQL TYPE	CL	TY	SI	LE	NU	RE	RD	DI
L	DATASPACE	CHAR	S	C	N	1	Y	1	N	64
L	MEMORY_SIZE	DECIMAL	S	D	Y	6	N	1	N	65
L	BLOCKS_MAXIMUM	INTEGER	S	B	Y	4	Y	1	N	72
L	BLOCKS_CURRENT	INTEGER	S	B	Y	4	Y	1	N	77
D	TOTAL_READS	DECIMAL	S	D	Y	8	Y	1	N	82
D	MRDF_READS	DECIMAL	S	D	Y	8	Y	1	N	91
D	VIRTUAL_WRITES	DECIMAL	S	D	Y	8	Y	1	N	100
D	ACTIVE_READS	DECIMAL	S	D	Y	8	Y	1	N	109
D	NONFIRST_READS	DECIMAL	S	D	Y	8	Y	1	N	118
L	HIGH_VIRTUAL_BLOCK	INTEGER	S	B	Y	4	Y	1	N	127
L	BEGIN_BLOCK	SMALLINT	S	B	Y	2	Y	1	N	132
U	D_MUF_USERTAG	CHAR	S	C	N	16	N	1	N	134

Primary KEY columns are as follows:

- MUF_NAME
- DBID
- AREA_NAME
- VIRTUAL_COVERED
- D_MUF_ENABLE
- D_MUF_END_DATETIME
- D_MUF_TYPE
- D_MUF_USERTAG
- D_MUF_START_DATETIME

Secondary KEY columns are as follows:

- MUF_NAME
- D_MUF_ENABLE
- D_MUF_END_DATETIME
- D_MUF_TYPE
- D_MUF_USERTAG
- D_MUF_START_DATETIME
- D_MUF_DURATION

D_MUF_INTERNAL_STATS (MFV)

ID	SQL NAME	SQL TYPE	CL	TY	SI	LE	NU	RE	RD	DI
I	D_MUF_ENABLE	SQL-STMP	S	B	N	10	N	1	N	0
IS	D_MUF_START_DATETIME	SQL-STMP	S	B	N	10	N	1	N	10
I	D_MUF_END_DATETIME	SQL-STMP	S	B	N	10	N	1	N	20
I	D_MUF_TYPE	CHAR	S	C	N	10	N	1	N	30
I	D_MUF_TOTALROW	CHAR	S	C	N	1	N	1	N	40
D	D_MUF_DURATION	DECIMAL	S	D	Y	6	N	1	N	41
L	MUF_NAME	CHAR	S	C	N	8	N	1	N	47
D	TIMES_READ_1	DECIMAL	S	D	Y	8	N	1	N	55
D	TIMES_READ_2_8	DECIMAL	S	D	Y	8	N	1	N	63
D	TIMES_READ_9_32	DECIMAL	S	D	Y	8	N	1	N	71
D	TIMES_READ_33_99	DECIMAL	S	D	Y	8	N	1	N	79
D	TIMES_WRITE_1	DECIMAL	S	D	Y	8	N	1	N	87
D	TIMES_WRITE_2_12	DECIMAL	S	D	Y	8	N	1	N	95
D	TIMES_WRITE_13_24	DECIMAL	S	D	Y	8	N	1	N	103
D	TIMES_WRITE_25_48	DECIMAL	S	D	Y	8	N	1	N	111
D	BREAKS_DONE	DECIMAL	S	D	Y	8	N	1	N	119
D	IO_MEM_SHORT	DECIMAL	S	D	Y	8	N	1	N	127
U	D_MUF_USERTAG	CHAR	S	C	N	16	N	1	N	135

Primary KEY columns are as follows:

- MUF_NAME
- D_MUF_ENABLE
- D_MUF_END_DATETIME
- D_MUF_TYPE
- D_MUF_USERTAG
- D_MUF_START_DATETIME

Secondary KEY columns are as follows:

- MUF_NAME
- D_MUF_ENABLE
- D_MUF_END_DATETIME
- D_MUF_TYPE
- D_MUF_USERTAG
- D_MUF_START_DATETIME
- D_MUF_DURATION

D_MUF_OPTIONS (MFO)

ID	SQL NAME	SQL TYPE	CL	TY	SI	LE	NU	RE	RD	DI
I	D_MUF_ENABLE	SQL-STMP	S	B	N	10	N	1	N	0
IS	D_MUF_START_DATETIME	SQL-STMP	S	B	N	10	N	1	N	10
I	D_MUF_END_DATETIME	SQL-STMP	S	B	N	10	N	1	N	20
I	D_MUF_TYPE	CHAR	S	C	N	10	N	1	N	30
I	D_MUF_TOTALROW	CHAR	S	C	N	1	N	1	N	40
D	D_MUF_DURATION	DECIMAL	S	D	Y	6	N	1	N	41
L	MUF_NAME	CHAR	S	C	N	8	N	1	N	47
L	ACCTPRM_PRM_DBID	SMALLINT	S	B	Y	2	N	1	N	55
L	ACCTPRM_ANN_DBID	SMALLINT	S	B	Y	2	N	1	N	57
L	ACCTPRM_CPU_TIME	CHAR	S	C	N	1	N	1	N	59
L	AGENT	CHAR	S	C	N	1	N	1	N	60
L	AGENT_LOG_LEVEL	SMALLINT	S	B	Y	2	N	1	N	61

ID	SQL NAME	SQL TYPE	CL	TY	SI	LE	NU	RE	RD	DI
L	AGENT_USE_CONFIG	CHAR	S	C	N	1	N	1	N	63
L	BREAK	INTEGER	S	B	Y	4	N	1	N	64
L	CBS_DBID_TEMP	SMALLINT	S	B	Y	2	N	1	N	68
L	CBS_BUFFER	INTEGER	S	B	Y	4	N	1	N	70
L	CBS_MAXSTEN	INTEGER	S	B	Y	4	N	1	N	74
L	CBS_MAXSTIO	INTEGER	S	B	Y	4	N	1	N	78
L	CBS_TIME_DELETE	SMALLINT	S	B	Y	2	N	1	N	82
L	CBS_DBID_HEURISTIC	SMALLINT	S	B	Y	2	N	1	N	84
L	CSAFREE	CHAR	S	C	N	1	N	1	N	86
L	DATAPOOL_DATALN	SMALLINT	S	B	Y	2	N	1	N	87
L	DATAPOOL_DATANO	INTEGER	S	B	Y	4	N	1	N	89
L	DATAPOOL_DATALN2	SMALLINT	S	B	Y	2	N	1	N	93
L	DATAPOOL_DATANO2	INTEGER	S	B	Y	4	N	1	N	95
L	DATASPACE_SIZE	INTEGER	S	B	Y	4	N	1	N	99
L	DATASPACE_CONNECT	CHAR	S	C	N	1	N	1	N	103
L	DATETIME_DATE	CHAR	S	C	N	3	N	1	N	104
L	DATETIME_TIME	CHAR	S	C	N	3	N	1	N	107
L	DICTIONARY_DD_DBID	SMALLINT	S	B	Y	2	N	1	N	110
L	DICTIONARY_DDDDBID	SMALLINT	S	B	Y	2	N	1	N	112
L	EXCTLNO	INTEGER	S	B	Y	4	N	1	N	114
L	EXPAND_LENGTH	INTEGER	S	B	Y	4	N	1	N	118
L	EXPAND_NUMBER	INTEGER	S	B	Y	4	N	1	N	122
L	FLEXPOOL_IXXNO	INTEGER	S	B	Y	4	N	1	N	126
L	FLEXPOOL_DXXNO	INTEGER	S	B	Y	4	N	1	N	130
L	FLEXPOOL_DATANO	INTEGER	S	B	Y	4	N	1	N	134
L	FLEXPOOL_DATANO2	INTEGER	S	B	Y	4	N	1	N	138
L	IOTASK_MAXIMUM	SMALLINT	S	B	Y	2	N	1	N	142
L	IOTASK_CURRENT	SMALLINT	S	B	Y	2	N	1	N	144
L	LOGOPTION_OPN_CLS	CHAR	S	C	N	1	N	1	N	146
L	LOGPEND	SMALLINT	S	B	Y	2	N	1	N	147
L	LOGPOOL	SMALLINT	S	B	Y	2	N	1	N	149

ID	SQL NAME	SQL TYPE	CL	TY	SI	LE	NU	RE	RD	DI
L	LOGRCV	CHAR	S	C	N	5	N	1	N	151
L	LOGRSYS	SMALLINT	S	B	Y	2	N	1	N	156
L	LOGSPILL_A	SMALLINT	S	B	Y	2	N	1	N	158
L	LOGSPILL_B	SMALLINT	S	B	Y	2	N	1	N	160
L	LOGSPILL_C	SMALLINT	S	B	Y	2	N	1	N	162
L	LOGSPILL_D	SMALLINT	S	B	Y	2	N	1	N	164
L	LOGSPILL_E	SMALLINT	S	B	Y	2	N	1	N	166
L	MAXELRQ	SMALLINT	S	B	Y	2	N	1	N	168
L	MUF_RUN_UNIT_GROUP	SMALLINT	S	B	Y	2	N	1	N	170
L	MUF_ENDED_NO_LOG	CHAR	S	C	N	1	N	1	N	172
L	MUFMSG_A	CHAR	S	C	N	1	N	1	N	173
L	MUFMSG_B	CHAR	S	C	N	1	N	1	N	174
L	MUFMSG_C	CHAR	S	C	N	1	N	1	N	175
L	MUFPLEX_NAME	CHAR	S	C	N	8	N	1	N	176
L	MUFPLEX_NUMBER	SMALLINT	S	B	Y	2	N	1	N	184
L	MUFPLEX_LOCKS	INTEGER	S	B	Y	4	N	1	N	186
L	MUFPLEX_MAX_TASKS	SMALLINT	S	B	Y	2	N	1	N	190
L	NONSWAP	CHAR	S	C	N	1	N	1	N	192
L	PLANSEC_CHECKWHO	CHAR	S	C	N	8	N	1	N	193
L	PLANSEC_CHECKWHEN	CHAR	S	C	N	8	N	1	N	201
L	PLANSEC_CHECKPLAN	CHAR	S	C	N	1	N	1	N	209
L	PROCEDURE_CACHE	INTEGER	S	B	Y	4	N	1	N	210
L	PROCEDURE_NESTS	SMALLINT	S	B	Y	2	N	1	N	214
L	PROCEDURE_TCBS	SMALLINT	S	B	Y	2	N	1	N	216
L	PXX_STATS	CHAR	S	C	N	6	N	1	N	218
L	READAHD	CHAR	S	C	N	8	N	1	N	224
L	RESTART_IGNORE	CHAR	S	C	N	1	N	1	N	232
L	RESTART_ACCESS_OFF	CHAR	S	C	N	1	N	1	N	233
L	RRS	CHAR	S	C	N	8	N	1	N	234
L	RXX_UNIT_NAME	CHAR	S	C	N	8	N	1	N	242
L	RXX_VOLUME_COUNT	SMALLINT	S	B	Y	2	N	1	N	250

ID	SQL NAME	SQL TYPE	CL	TY	SI	LE	NU	RE	RD	DI
L	RXX_UNIT_COUNT	SMALLINT	S	B	Y	2	N	1	N	252
L	RXXROLLBACK	CHAR	S	C	N	1	N	1	N	254
L	SMPTASK_MAXIMUM	SMALLINT	S	B	Y	2	N	1	N	255
L	SMPTASK_CURRENT	SMALLINT	S	B	Y	2	N	1	N	257
L	SMPTASK_READY_TASK	SMALLINT	S	B	Y	2	N	1	N	259
L	SQLDEFAULT_DBID	SMALLINT	S	B	Y	2	N	1	N	261
L	SQLDEFAULT_NAME	CHAR	S	C	N	32	N	1	N	263
L	SQLOPTION_OPTION	CHAR	S	C	N	1	N	1	N	295
L	SQLOPTION_TTMID	SMALLINT	S	B	Y	2	N	1	N	296
L	SQLOPTION_MODE	CHAR	S	C	N	7	N	1	N	298
L	SQLOPTION_TIMEOUT	SMALLINT	S	B	Y	2	N	1	N	305
L	SQLOPTION_VIEW	CHAR	S	C	N	1	N	1	N	307
L	SQLOPTION_BOTH	CHAR	S	C	N	1	N	1	N	308
L	STAR_TASKS	SMALLINT	S	B	Y	2	N	1	N	309
L	STAR_NODE	CHAR	S	C	N	8	N	1	N	311
L	STAR_SIZE	INTEGER	S	B	Y	4	N	1	N	319
L	STAR_MAX_NODES	SMALLINT	S	B	Y	2	N	1	N	323
L	STAR_TIME_BATCH	SMALLINT	S	B	Y	2	N	1	N	325
L	STAR_TIME_CICS	SMALLINT	S	B	Y	2	N	1	N	327
L	STATBFR	INTEGER	S	B	Y	4	N	1	N	329
L	SUBTASK	CHAR	S	C	N	8	N	1	N	333
L	SYSOUT_CLASS	CHAR	S	C	N	1	N	1	N	341
L	SYSPOOL_CXXNO	SMALLINT	S	B	Y	2	N	1	N	342
L	SYSPOOL_IXXNO	INTEGER	S	B	Y	4	N	1	N	344
L	SYSPOOL_DXXNO	INTEGER	S	B	Y	4	N	1	N	348
L	SYSPOOL_INDEX	SMALLINT	S	B	Y	2	N	1	N	352
L	SYSTEMDBID_DBID	SMALLINT	S	B	Y	2	N	1	N	354
L	SYSTEMDBID_SYSOUT	CHAR	S	C	N	1	N	1	N	356
L	TASKS_NUMBER	SMALLINT	S	B	Y	2	N	1	N	357
L	TASKS_SIZE	INTEGER	S	B	Y	4	N	1	N	359
L	TASKS_VAE_NUMBER	SMALLINT	S	B	Y	2	N	1	N	363

ID	SQL NAME	SQL TYPE	CL	TY	SI	LE	NU	RE	RD	DI
L	TASKS_VM_NUMBER	SMALLINT	S	B	Y	2	N	1	N	365
L	TASKS_XCF_NUMBER	SMALLINT	S	B	Y	2	N	1	N	367
U	D_MUF_USERTAG	CHAR	S	C	N	16	N	1	N	369

Primary KEY columns are as follows:

- MUF_NAME
- D_MUF_ENABLE
- D_MUF_END_DATETIME
- D_MUF_TYPE
- D_MUF_USERTAG
- D_MUF_START_DATETIME

Secondary KEY columns are as follows:

- MUF_NAME
- D_MUF_ENABLE
- D_MUF_END_DATETIME
- D_MUF_TYPE
- D_MUF_USERTAG
- D_MUF_START_DATETIME
- D_MUF_DURATION

D_MUF_RETURN_CODES (MFU)

ID	SQL NAME	SQL TYPE	CL	TY	SI	LE	NU	RE	RD	DI
I	D_MUF_ENABLE	SQL-STMP	S	B	N	10	N	1	N	0
IS	D_MUF_START_DATETIME	SQL-STMP	S	B	N	10	N	1	N	10
I	D_MUF_END_DATETIME	SQL-STMP	S	B	N	10	N	1	N	20
I	D_MUF_TYPE	CHAR	S	C	N	10	N	1	N	30
I	D_MUF_TOTALROW	CHAR	S	C	N	1	N	1	N	40
D	D_MUF_DURATION	DECIMAL	S	D	Y	6	N	1	N	41

ID	SQL NAME	SQL TYPE	CL	TY	SI	LE	NU	RE	RD	DI
L	MUF_NAME	CHAR	S	C	N	8	N	1	N	47
L	EXTERNAL_CODE	CHAR	S	C	N	2	N	1	N	55
L	INTERNAL_CODE	CHAR	S	C	N	3	N	1	N	57
D	COUNT_CODES	DECIMAL	S	D	Y	8	N	1	N	60
U	D_MUF_USERTAG	CHAR	S	C	N	16	N	1	N	68

Primary KEY columns are as follows:

- MUF_NAME
- EXTERNAL_CODE
- INTERNAL_CODE
- D_MUF_ENABLE
- D_MUF_END_DATETIME
- D_MUF_TYPE
- D_MUF_USERTAG
- D_MUF_START_DATETIME

Secondary KEY columns are as follows:

- MUF_NAME
- D_MUF_ENABLE
- D_MUF_END_DATETIME
- D_MUF_TYPE
- D_MUF_USERTAG
- D_MUF_START_DATETIME
- D_MUF_DURATION

D_MUF_SMP_STATS (MFW)

ID	SQL NAME	SQL TYPE	CL	TY	SI	LE	NU	RE	RD	DI
I	D_MUF_ENABLE	SQL-STMP	S	B	N	10	N	1	N	0
IS	D_MUF_START_DATETIME	SQL-STMP	S	B	N	10	N	1	N	10
I	D_MUF_END_DATETIME	SQL-STMP	S	B	N	10	N	1	N	20

ID	SQL NAME	SQL TYPE	CL	TY	SI	LE	NU	RE	RD	DI
I	D_MUF_TYPE	CHAR	S	C	N	10	N	1	N	30
I	D_MUF_TOTALROW	CHAR	S	C	N	1	N	1	N	40
D	D_MUF_DURATION	DECIMAL	S	D	Y	6	N	1	N	41
L	MUF_NAME	CHAR	S	C	N	8	N	1	N	47
D	IXX_BUFFER	DECIMAL	S	D	Y	8	N	1	N	55
D	IXX_LRU	DECIMAL	S	D	Y	8	N	1	N	63
D	IXX_HASH	DECIMAL	S	D	Y	8	N	1	N	71
D	DXX_BUFFER	DECIMAL	S	D	Y	8	N	1	N	79
D	DXX_LRU	DECIMAL	S	D	Y	8	N	1	N	87
D	DXX_HASH	DECIMAL	S	D	Y	8	N	1	N	95
D	DXX_WRITE_PEND	DECIMAL	S	D	Y	8	N	1	N	103
D	DATA_BUFFER	DECIMAL	S	D	Y	8	N	1	N	111
D	DATA_LRU	DECIMAL	S	D	Y	8	N	1	N	119
D	DATA_HASH	DECIMAL	S	D	Y	8	N	1	N	127
D	DATA_WRITE_PEND	DECIMAL	S	D	Y	8	N	1	N	135
D	LOG_BUFFER	DECIMAL	S	D	Y	8	N	1	N	143
D	LOG_LRU	DECIMAL	S	D	Y	8	N	1	N	151
D	DATA_SPACE_BUFFER	DECIMAL	S	D	Y	8	N	1	N	159
D	DATA_SPACE_MGMT	DECIMAL	S	D	Y	8	N	1	N	167
D	POSTED_0	DECIMAL	S	D	Y	8	N	1	N	175
D	POSTED_1_5	DECIMAL	S	D	Y	8	N	1	N	183
D	POSTED_6_10	DECIMAL	S	D	Y	8	N	1	N	191
D	POSTED_11_23	DECIMAL	S	D	Y	8	N	1	N	199
D	POSTED_24_999	DECIMAL	S	D	Y	8	N	1	N	207
D	EXPAND_BUFFER	DECIMAL	S	D	Y	8	N	1	N	215
D	TSN	DECIMAL	S	D	Y	8	N	1	N	223
D	ADD_DELETE_FLEX	DECIMAL	S	D	Y	8	N	1	N	231
D	LOCK_LIST	DECIMAL	S	D	Y	8	N	1	N	239
D	ACCOUNTING_STATUS	DECIMAL	S	D	Y	8	N	1	N	247
D	ACCOUNTING_TABLE	DECIMAL	S	D	Y	8	N	1	N	255
D	SECURITY_CHECK	DECIMAL	S	D	Y	8	N	1	N	263

ID	SQL NAME	SQL TYPE	CL	TY	SI	LE	NU	RE	RD	DI
D	SQL_MISC	DECIMAL	S	D	Y	8	N	1	N	271
D	SQL_TTM	DECIMAL	S	D	Y	8	N	1	N	279
D	SQL_ATTACH	DECIMAL	S	D	Y	8	N	1	N	287
D	SQL_URT	DECIMAL	S	D	Y	8	N	1	N	295
D	SQL_PLAN	DECIMAL	S	D	Y	8	N	1	N	303
D	SQL_GLOBAL	DECIMAL	S	D	Y	8	N	1	N	311
D	SQL_STATUS	DECIMAL	S	D	Y	8	N	1	N	319
D	CBS_BUFFER	DECIMAL	S	D	Y	8	N	1	N	327
D	OPEN_TABLE	DECIMAL	S	D	Y	8	N	1	N	335
D	POSTED_LIST	DECIMAL	S	D	Y	8	N	1	N	343
D	MISCELLANEOUS	DECIMAL	S	D	Y	8	N	1	N	351
D	INDEX_QUEUE	DECIMAL	S	D	Y	8	N	1	N	359
D	TASK_AREA	DECIMAL	S	D	Y	8	N	1	N	367
D	STAR	DECIMAL	S	D	Y	8	N	1	N	375
D	WAIT_LIST	DECIMAL	S	D	Y	8	N	1	N	383
D	MEMORY_MANAGER	DECIMAL	S	D	Y	8	N	1	N	391
D	PXX_CONSOLE	DECIMAL	S	D	Y	8	N	1	N	399
D	USER_COMPRESSION	DECIMAL	S	D	Y	8	N	1	N	407
D	COVEREDVIRTUAL	DECIMAL	S	D	Y	8	N	1	N	415
D	CHECK_INTERLOCK	DECIMAL	S	D	Y	8	N	1	N	423
D	STAR_TASK	DECIMAL	S	D	Y	8	N	1	N	431
D	DSF_IN_MUF	DECIMAL	S	D	Y	8	N	1	N	439
D	WAIT_MUF_MUFPLEX	DECIMAL	S	D	Y	8	N	1	N	447
D	WAIT_COUPLER	DECIMAL	S	D	Y	8	N	1	N	455
D	PLEX_TSN_RETRY	DECIMAL	S	D	Y	8	N	1	N	463
D	PLEX_TSN_READ	DECIMAL	S	D	Y	8	N	1	N	471
D	WAIT_XCF	DECIMAL	S	D	Y	8	N	1	N	479
D	PLEX_LOG_RCD	DECIMAL	S	D	Y	8	N	1	N	487
D	PLEX_CLOSE_STAT	DECIMAL	S	D	Y	8	N	1	N	495
D	XES_MSG_REPLY	DECIMAL	S	D	Y	8	N	1	N	503
D	WAIT_PLEX_BASE	DECIMAL	S	D	Y	8	N	1	N	511

ID	SQL NAME	SQL TYPE	CL	TY	SI	LE	NU	RE	RD	DI
D	WAIT_PLEX_AREA	DECIMAL	S	D	Y	8	N	1	N	519
D	WAIT_PLEX_TABLE	DECIMAL	S	D	Y	8	N	1	N	527
D	PLEX_LOCK_DBYU	DECIMAL	S	D	Y	8	N	1	N	535
D	PLEX_LOCK_TSN	DECIMAL	S	D	Y	8	N	1	N	543
D	PLEX_LOCK_LOG_C	DECIMAL	S	D	Y	8	N	1	N	551
D	PLEX_LOCK_LOG_U	DECIMAL	S	D	Y	8	N	1	N	559
D	PLEX_LOCK_RUNUNIT	DECIMAL	S	D	Y	8	N	1	N	567
D	PLEX_LOCK_BUFFER	DECIMAL	S	D	Y	8	N	1	N	575
D	SQL_PROCEDURE	DECIMAL	S	D	Y	8	N	1	N	583
D	PLEX_REBUILD	DECIMAL	S	D	Y	8	N	1	N	591
D	LOGGING_CHAINS	DECIMAL	S	D	Y	8	N	1	N	599
D	LOGGING_FORCE	DECIMAL	S	D	Y	8	N	1	N	607
D	ACCOUNTING_SMP	DECIMAL	S	D	Y	8	N	1	N	615
D	SQL_MEM_MGR	DECIMAL	S	D	Y	8	N	1	N	623
D	OPEN_USER_EXIT	DECIMAL	S	D	Y	8	N	1	N	631
D	ACCESS_AREA	DECIMAL	S	D	Y	8	N	1	N	639
U	D_MUF_USERTAG	CHAR	S	C	N	16	N	1	N	647

Primary KEY columns are as follows:

- MUF_NAME
- D_MUF_ENABLE
- D_MUF_END_DATETIME
- D_MUF_TYPE
- D_MUF_USERTAG
- D_MUF_START_DATETIME

Secondary KEY columns are as follows:

- MUF_NAME
- D_MUF_ENABLE
- D_MUF_END_DATETIME
- D_MUF_TYPE
- D_MUF_USERTAG
- D_MUF_START_DATETIME
- D_MUF_DURATION

D_MUF_SRB_ZIIP (MZI)

ID	**SQL NAME**	**SQL TYPE**	**CL**	**TY**	**SI**	**LE**	**NU**	**RE**	**RD**	DI
I	D_MUF_ENABLE	SQL-STMP	S	B	N	10	N	1	N	0
IS	D_MUF_START_DATETIME	SQL-STMP	S	B	N	10	N	1	N	10
I	D_MUF_END_DATETIME	SQL-STMP	S	B	N	10	N	1	N	20
I	D_MUF_TYPE	CHAR	S	C	N	10	N	1	N	30
I	D_MUF_TOTALROW	CHAR	S	C	N	1	N	1	N	40
I	D_MUF_DURATION	DECIMAL	S	D	Y	6	N	1	N	41
L	MUF_NAME	CHAR	S	C	N	8	N	1	N	47
L	SEQUENCE_NUMBER	SMALLINT	S	B	N	2	N	1	N	55
D	ENCLAVE_SECONDS	DECIMAL	S	D	Y	8	N	1	N	57
D	ZIIP_ON_CP_SECONDS	DECIMAL	S	D	Y	8	N	1	N	65
D	ZIIP_QUAL_SECONDS	DECIMAL	S	D	Y	8	N	1	N	73
D	ZIIP_SECONDS	DECIMAL	S	D	Y	8	N	1	N	81

ID	SQL NAME	SQL TYPE	CL	TY	SI	LE	NU	RE	RD	DI
D	ZIIP_PERCENT	SMALLINT	S	B	N	2	N	1	N	89
U	D_MUF_USERTAG	CHAR	S	C	N	16	N	1	N	91

Primary KEY columns are as follows:

- MUF_NAME
- SEQUENCE_NUMBER
- D_MUF_ENABLE
- D_MUF_END_DATETIME
- D_MUF_TYPE
- D_MUF_USER_TAG
- D_MUF_START_DATETIME

Secondary KEY columns are as follows:

- MUF_NAME
- D_MUF_ENABLE
- D_MUF_END_DATETIME
- D_MUF_TYPE
- D_MUF_USERTAG
- D_MUF_START_DATETIME
- D_MUF_DURATION

D_MUF_SYSTEM_STATS (MFS)

ID	SQL NAME	SQL TYPE	CL	TY	SI	LE	NU	RE	RD	DI
I	D_MUF_ENABLE	SQL-STMP	S	B	N	10	N	1	N	0
IS	D_MUF_START_DATETIME	SQL-STMP	S	B	N	10	N	1	N	10
I	D_MUF_END_DATETIME	SQL-STMP	S	B	N	10	N	1	N	20
I	D_MUF_TYPE	CHAR	S	C	N	10	N	1	N	30
I	D_MUF_TOTALROW	CHAR	S	C	N	1	N	1	N	40
D	D_MUF_DURATION	DECIMAL	S	D	Y	6	N	1	N	41
L	MUF_NAME	CHAR	S	C	N	8	N	1	N	47

ID	SQL NAME	SQL TYPE	CL	TY	SI	LE	NU	RE	RD	DI
L	BEGIN_TIME	SQL-STMP	S	B	N	10	N	1	N	55
L	CURRENT_DATETIME	SQL-STMP	S	B	N	10	N	1	N	65
D	CURRENT_CPU	DECIMAL	S	D	Y	8	N	1	N	75
D	RQ_PENDING_1_10	DECIMAL	S	D	Y	8	N	1	N	83
D	RQ_PENDING_11_20	DECIMAL	S	D	Y	8	N	1	N	91
D	RQ_PENDING_21_30	DECIMAL	S	D	Y	8	N	1	N	99
D	RQ_PENDING_31_40	DECIMAL	S	D	Y	8	N	1	N	107
D	RQ_PENDING_41_50	DECIMAL	S	D	Y	8	N	1	N	115
D	RQ_PENDING_51_60	DECIMAL	S	D	Y	8	N	1	N	123
D	RQ_PENDING_61_70	DECIMAL	S	D	Y	8	N	1	N	131
D	RQ_PENDING_71_80	DECIMAL	S	D	Y	8	N	1	N	139
D	RQ_PENDING_81_90	DECIMAL	S	D	Y	8	N	1	N	147
D	RQ_PENDING_91_100	DECIMAL	S	D	Y	8	N	1	N	155
D	RQ_PENDING_GT_100	DECIMAL	S	D	Y	8	N	1	N	163
D	DB_REQUESTS	DECIMAL	S	D	Y	8	N	1	N	171
D	SQL_REQUESTS	DECIMAL	S	D	Y	8	N	1	N	179
D	DATA_MGR_REQUESTS	DECIMAL	S	D	Y	8	N	1	N	187
D	PHYSICAL_READS	DECIMAL	S	D	Y	8	N	1	N	195
D	PHYSICAL_WRITES	DECIMAL	S	D	Y	8	N	1	N	203
D	RECORD_LOCKS	DECIMAL	S	D	Y	8	N	1	N	211
D	WAIT_LOCK	DECIMAL	S	D	Y	8	N	1	N	219
D	KEY_ELEMENT_REORG	DECIMAL	S	D	Y	8	N	1	N	227
D	NO_IXX_BUFFER	DECIMAL	S	D	Y	8	N	1	N	235
D	NO_DXX_BUFFER	DECIMAL	S	D	Y	8	N	1	N	243
D	NO_DATA_BUFFER	DECIMAL	S	D	Y	8	N	1	N	251
D	NO_EXPAND_BUFFER	DECIMAL	S	D	Y	8	N	1	N	259
D	IXX_SEQUENCE	DECIMAL	S	D	Y	8	N	1	N	267
D	DXX_SEQUENCE	DECIMAL	S	D	Y	8	N	1	N	275
D	DATA_SEQUENCE	DECIMAL	S	D	Y	8	N	1	N	283
D	EXPAND_SEQUENCE	DECIMAL	S	D	Y	8	N	1	N	291
D	IXX_USED_1	DECIMAL	S	D	Y	8	N	1	N	299

ID	SQL NAME	SQL TYPE	CL	TY	SI	LE	NU	RE	RD	DI
D	IXX_USED_2	DECIMAL	S	D	Y	8	N	1	N	307
D	IXX_USED_3	DECIMAL	S	D	Y	8	N	1	N	315
D	IXX_USED_4	DECIMAL	S	D	Y	8	N	1	N	323
D	IXX_USED_5	DECIMAL	S	D	Y	8	N	1	N	331
D	DXX_USED_1	DECIMAL	S	D	Y	8	N	1	N	339
D	DXX_USED_2	DECIMAL	S	D	Y	8	N	1	N	347
D	DXX_USED_3	DECIMAL	S	D	Y	8	N	1	N	355
D	DXX_USED_4	DECIMAL	S	D	Y	8	N	1	N	363
D	DXX_USED_5	DECIMAL	S	D	Y	8	N	1	N	371
D	DATA_USED_1	DECIMAL	S	D	Y	8	N	1	N	379
D	DATA_USED_2	DECIMAL	S	D	Y	8	N	1	N	387
D	DATA_USED_3	DECIMAL	S	D	Y	8	N	1	N	395
D	DATA_USED_4	DECIMAL	S	D	Y	8	N	1	N	403
D	DATA_USED_5	DECIMAL	S	D	Y	8	N	1	N	411
D	SEQ_READ_AHEAD	DECIMAL	S	D	Y	8	N	1	N	419
L	LOG_PERCENT_FULL	INTEGER	S	B	Y	4	N	1	N	427
D	SECONDARY_CONFLICT	DECIMAL	S	D	Y	8	N	1	N	431
D	WAIT_TASK	DECIMAL	S	D	Y	8	N	1	N	439
D	WAIT_REQUEST	DECIMAL	S	D	Y	8	N	1	N	447
D	WAIT_SPILL	DECIMAL	S	D	Y	8	N	1	N	455
D	WAIT_ACCOUNTING	DECIMAL	S	D	Y	8	N	1	N	463
D	WAIT_SECURITY	DECIMAL	S	D	Y	8	N	1	N	471
D	INDEX_Q_PROCESSED	DECIMAL	S	D	Y	8	N	1	N	479
D	INDEX_Q_OVERFLOW	DECIMAL	S	D	Y	8	N	1	N	487
D	SPLIT_IXX	DECIMAL	S	D	Y	8	N	1	N	495
D	SPLIT_DXX	DECIMAL	S	D	Y	8	N	1	N	503
D	DELETED_BLOCKS	DECIMAL	S	D	Y	8	N	1	N	511
D	CBS_SPILLS	DECIMAL	S	D	Y	8	N	1	N	519
L	CBS_ENTRIES	INTEGER	S	B	Y	4	N	1	N	527
L	CBS_ENTRIES_USED	INTEGER	S	B	Y	4	N	1	N	531
D	SETS_PROCESSED	DECIMAL	S	D	Y	8	N	1	N	535

ID	SQL NAME	SQL TYPE	CL	TY	SI	LE	NU	RE	RD	DI
D	SETS_WITH_T_INDEX	DECIMAL	S	D	Y	8	N	1	N	543
D	TEMP_INDEX_ENTRIES	DECIMAL	S	D	Y	8	N	1	N	551
D	CBS_MEMORY_ALLOCS	DECIMAL	S	D	Y	8	N	1	N	559
D	LOGPEND_WRITES	DECIMAL	S	D	Y	8	N	1	N	567
L	DATASECURE_ONE	INTEGER	S	B	Y	4	N	1	N	575
L	DATASECURE_TWO	INTEGER	S	B	Y	4	N	1	N	579
D	REEXPANDS	DECIMAL	S	D	Y	8	N	1	N	583
D	NO_TXB_BUFFER	DECIMAL	S	D	Y	8	N	1	N	591
D	INDIRECT_WAIT_IO	DECIMAL	S	D	Y	8	N	1	N	599
D	LOG_WRITE_CONTROL	DECIMAL	S	D	Y	8	N	1	N	607
D	LOG_WRITE_FULL	DECIMAL	S	D	Y	8	N	1	N	615
D	LOG_WRITE_COMMAND	DECIMAL	S	D	Y	8	N	1	N	623
D	LOG_WRITE_OTHER	DECIMAL	S	D	Y	8	N	1	N	631
D	LOG_WRITE_TXB	DECIMAL	S	D	Y	8	N	1	N	639
D	LOG_WRITE_2_PHASE	DECIMAL	S	D	Y	8	N	1	N	647
D	DATA2_USED_1	DECIMAL	S	D	Y	8	N	1	N	655
D	DATA2_USED_2	DECIMAL	S	D	Y	8	N	1	N	663
D	DATA2_USED_3	DECIMAL	S	D	Y	8	N	1	N	671
D	DATA2_USED_4	DECIMAL	S	D	Y	8	N	1	N	679
D	DATA2_USED_5	DECIMAL	S	D	Y	8	N	1	N	687
U	D_MUF_USERTAG	CHAR	S	C	N	16	N	1	N	695

Primary KEY columns are as follows:

- MUF_NAME
- D_MUF_ENABLE
- D_MUF_END_DATETIME
- D_MUF_TYPE
- D_MUF_USERTAG
- D_MUF_START_DATETIME

Secondary KEY columns are as follows:

- MUF_NAME
- D_MUF_ENABLE
- D_MUF_END_DATETIME
- D_MUF_TYPE
- D_MUF_USERTAG
- D_MUF_START_DATETIME
- D_MUF_DURATION

D_MUF_TABLE_STATS (MFT)

ID	SQL NAME	SQL TYPE	CL	TY	SI	LE	NU	RE	RD	DI
I	D_MUF_ENABLE	SQL-STMP	S	B	N	10	N	1	N	0
IS	D_MUF_START_DATETIME	SQL-STMP	S	B	N	10	N	1	N	10
I	D_MUF_END_DATETIME	SQL-STMP	S	B	N	10	N	1	N	20
I	D_MUF_TYPE	CHAR	S	C	N	10	N	1	N	30
I	D_MUF_TOTALROW	CHAR	S	C	N	1	N	1	N	40
D	D_MUF_DURATION	DECIMAL	S	D	Y	6	N	1	N	41
L	MUF_NAME	CHAR	S	C	N	8	N	1	N	47
L	BEGIN_TIME	SQL-STMP	S	B	N	10	N	1	N	55
L	CURRENT_DATETIME	SQL-STMP	S	B	N	10	N	1	N	65
L	DBID	SMALLINT	S	B	Y	2	N	1	N	75
L	TABLE_NAME	CHAR	S	C	N	3	N	1	N	77
L	AREA_NAME	CHAR	S	C	N	3	N	1	N	80

ID	SQL NAME	SQL TYPE	CL	TY	SI	LE	NU	RE	RD	DI
D	TOTAL_REQUESTS	DECIMAL	S	D	Y	8	N	1	N	83
D	TOTAL_READS	DECIMAL	S	D	Y	8	N	1	N	91
D	TOTAL_ADDS	DECIMAL	S	D	Y	8	N	1	N	99
D	TOTAL_DELETES	DECIMAL	S	D	Y	8	N	1	N	107
D	TOTAL_UPDATES	DECIMAL	S	D	Y	8	N	1	N	115
U	D_MUF_USERTAG	CHAR	S	C	N	16	N	1	N	123

Primary KEY columns are as follows:

- MUF_NAME
- DBID
- TABLE_NAME
- D_MUF_ENABLE
- D_MUF_END_DATETIME
- D_MUF_TYPE
- D_MUF_USERTAG
- D_MUF_START_DATETIME

Secondary KEY columns are as follows:

- MUF_NAME
- D_MUF_ENABLE
- D_MUF_END_DATETIME
- D_MUF_TYPE
- D_MUF_USERTAG
- D_MUF_START_DATETIME
- D_MUF_DURATION

D_MUF_TCB_OR_SRB (MTC)

ID	SQL NAME	SQL TYPE	CL	TY	SI	LE	NU	RE	RD	DI
I	D_MUF_ENABLE	SQL-STMP	S	B	N	10	N	1	N	0
IS	D_MUF_START_DATETIME	SQL-STMP	S	B	N	10	N	1	N	10

ID	SQL NAME	SQL TYPE	CL	TY	SI	LE	NU	RE	RD	DI
I	D_MUF_END_DATETIME	SQL-STMP	S	B	N	10	N	1	N	20
I	D_MUF_TYPE	CHAR	S	C	N	10	N	1	N	30
I	D_MUF_TOTALROW	CHAR	S	C	N	1	N	1	N	40
D	D_MUF_DURATION	DECIMAL	S	D	Y	6	N	1	N	41
L	MUF_NAME	CHAR	S	C	N	8	N	1	N	47
L	TASK_TYPE	CHAR	S	C	N	4	N	1	N	55
L	PROGRAM_NAME	CHAR	S	C	N	8	N	1	N	59
L	SEQUENCE_NUMBER	SMALLINT	S	B	N	2	N	1	N	67
D	CPU_SECONDS	DECIMAL	S	D	Y	8	N	1	N	69
D	TIMES_USED	DECIMAL	S	D	Y	8	N	1	N	77
D	TIMES_POSTED	DECIMAL	S	D	Y	8	N	1	N	85
D	PHYSICAL_IO	DECIMAL	S	D	Y	8	N	1	N	93
U	D_MUF_USERTAG	CHAR	S	C	N	16	N	1	N	101

Primary KEY columns are as follows:

- MUF_NAME
- PROGRAM_NAME
- SEQUENCE_NUMBER
- D_MUF_ENABLE
- D_MUF_END_DATETIME
- D_MUF_TYPE
- D_MUF_USERTAG
- D_MUF_START_DATETIME

Secondary KEY columns are as follows:

- MUF_NAME
- D_MUF_ENABLE
- D_MUF_END_DATETIME
- D_MUF_TYPE
- D_MUF_USERTAG
- D_MUF_START_DATETIME
- D_MUF_DURATION

D_SQL_MISC_STATS (SQM)

ID	SQL NAME	SQL TYPE	CL	TY	SI	LE	NU	RE	RD	DI
I	D_MUF_ENABLE	SQL-STMP	S	B	N	10	N	1	N	0
IS	D_MUF_START_DATETIME	SQL-STMP	S	B	N	10	N	1	N	10
I	D_MUF_END_DATETIME	SQL-STMP	S	B	N	10	N	1	N	20
I	D_MUF_TYPE	CHAR	S	C	N	10	N	1	N	30
I	D_MUF_TOTALROW	CHAR	S	C	N	1	N	1	N	40
D	D_MUF_DURATION	DECIMAL	S	D	Y	6	N	1	N	41
L	MUF_NAME	CHAR	S	C	N	8	N	1	N	47
D	PROC_CACHE_REUSE	DECIMAL	S	D	Y	8	N	1	N	55
D	PROC_FROM_CALL	DECIMAL	S	D	Y	8	N	1	N	63
D	PROC_FROM_TRIGGER	DECIMAL	S	D	Y	8	N	1	N	71
D	PROC_QUEUED	DECIMAL	S	D	Y	8	N	1	N	79
D	PROC_SQL_STMTS	DECIMAL	S	D	Y	8	N	1	N	87
D	PROC_FAILURES	DECIMAL	S	D	Y	8	N	1	N	95
D	PROC_NESTS	DECIMAL	S	D	Y	8	N	1	N	103
U	D_MUF_USERTAG	CHAR	S	C	N	16	N	1	N	111

Primary KEY columns are as follows:

- MUF_NAME
- D_MUF_ENABLE
- D_MUF_END_DATETIME
- D_MUF_TYPE
- D_MUF_USERTAG
- D_MUF_START_DATETIME

Secondary KEY columns are as follows:

- MUF_NAME
- D_MUF_ENABLE
- D_MUF_END_DATETIME
- D_MUF_TYPE
- D_MUF_USERTAG
- D_MUF_START_DATETIME
- D_MUF_DURATION

D_SQL_SQLCODES (SQQ)

ID	SQL NAME	SQL TYPE	CL	TY	SI	LE	NU	RE	RD	DI
I	D_MUF_ENABLE	SQL-STMP	S	B	N	10	N	1	N	0
IS	D_MUF_START_DATETIME	SQL-STMP	S	B	N	10	N	1	N	10
I	D_MUF_END_DATETIME	SQL-STMP	S	B	N	10	N	1	N	20
I	D_MUF_TYPE	CHAR	S	C	N	10	N	1	N	30
I	D_MUF_TOTALROW	CHAR	S	C	N	1	N	1	N	40
D	D_MUF_DURATION	DECIMAL	S	D	Y	6	N	1	N	41
L	MUF_NAME	CHAR	S	C	N	8	N	1	N	47
L	SQL_CODE	INTEGER	S	B	Y	4	N	1	N	55
D	CODE_COUNT	DECIMAL	S	D	Y	8	N	1	N	59
U	D_MUF_USERTAG	CHAR	S	C	N	16	N	1	N	67

Primary KEY columns are as follows:

- MUF_NAME
- SQL_CODE
- D_MUF_ENABLE
- D_MUF_END_DATETIME
- D_MUF_TYPE
- D_MUF_USERTAG
- D_MUF_START_DATETIME

Secondary KEY columns are as follows:

- MUF_NAME
- D_MUF_ENABLE
- D_MUF_END_DATETIME
- D_MUF_TYPE
- D_MUF_USERTAG
- D_MUF_START_DATETIME
- D_MUF_DURATION

D_BUSINESS_VALUE_METRICS (BVM)

ID	SQL NAME	SQL TYPE	CL	TY	SI	LE	NU	RE	RD	DI
I	D_MUF_ENABLE	SQL-STMP	S	B	N	10	N	1	N	0
IS	D_MUF_START_DATETIME	SQL-STMP	S	B	N	10	N	1	N	10
I	D_MUF_END_DATETIME	SQL-STMP	S	B	N	10	N	1	N	20
I	D_MUF_TYPE	CHAR	S	C	N	10	N	1	N	30
I	D_MUF_TOTALROW	CHAR	S	C	N	1	N	1	N	40
D	D_MUF_DURATION	DECIMAL	S	D	Y	6	N	1	N	41
L	MUF_NAME	CHAR	S	C	N	8	N	1	N	47
D	ELAPSED_SECONDS	DECIMAL	S	D	Y	8	N	1	N	55
D	CPU_SECONDS	DECIMAL	S	D	Y	8	N	1	N	63
D	PHYSICAL_IO	DECIMAL	S	D	Y	8	N	1	N	71
D	DB_REQUESTS	DECIMAL	S	D	Y	8	N	1	N	79
D	LOGICAL_IO	DECIMAL	S	D	Y	8	N	1	N	87
D	TOTAL_REQUESTS	DECIMAL	S	D	Y	8	N	1	N	95
D	TOTAL_READS	DECIMAL	S	D	Y	8	N	1	N	103
D	TOTAL_ADDS	DECIMAL	S	D	Y	8	N	1	N	111
D	TOTAL_DELETES	DECIMAL	S	D	Y	8	N	1	N	119
D	TOTAL_UPDATES	DECIMAL	S	D	Y	8	N	1	N	127
R	REQUESTS_PER_SEC	DECIMAL	S	D	Y	8	N	1	N	135
R	REQUESTS_PER_CPU_SEC	DECIMAL	S	D	Y	8	N	1	N	143

ID	SQL NAME	SQL TYPE	CL	TY	SI	LE	NU	RE	RD	DI
R	REQUESTS_PER_PHYS_IO	DECIMAL	S	D	Y	8	N	1	N	151
U	D_MUF_USERTAG	CHAR	S	C	N	16	N	1	N	159

Primary KEY columns are as follows:

- MUF_NAME
- D_MUF_ENABLE
- D_MUF_END_DATETIME
- D_MUF_TYPE
- D_MUF_USERTAG
- D_MUF_START_DATETIME

Secondary KEY columns are as follows:

- MUF_NAME
- D_MUF_ENABLE
- D_MUF_END_DATETIME
- D_MUF_TYPE
- D_MUF_USERTAG
- D_MUF_START_DATETIME
- D_MUF_DURATION

D_BUSINESS_VALUE_DETAILS (BVD)

ID	SQL NAME	SQL TYPE	CL	TY	SI	LE	NU	RE	RD	DI
I	D_MUF_ENABLE	SQL-STMP	S	B	N	10	N	1	N	0
IS	D_MUF_START_DATETIME	SQL-STMP	S	B	N	10	N	1	N	10
I	D_MUF_END_DATETIME	SQL-STMP	S	B	N	10	N	1	N	20
I	D_MUF_TYPE	CHAR	S	C	N	10	N	1	N	30
I	D_MUF_TOTALROW	CHAR	S	C	N	1	N	1	N	40
D	D_MUF_DURATION	DECIMAL	S	D	Y	6	N	1	N	41
L	MUF_NAME	CHAR	S	C	N	8	N	1	N	47
D	ELAPSED_SECONDS	DECIMAL	S	D	Y	8	N	1	N	55

ID	SQL NAME	SQL TYPE	CL	TY	SI	LE	NU	RE	RD	DI
D	CPU_SECONDS	DECIMAL	S	D	Y	8	N	1	N	63
D	DB_REQUESTS	DECIMAL	S	D	Y	8	N	1	N	71
D	DATA_MGR_REQUESTS	DECIMAL	S	D	Y	8	N	1	N	79
D	REL_MGR_REQUESTS	DECIMAL	S	D	Y	8	N	1	N	87
D	SQL_REQUESTS	DECIMAL	S	D	Y	8	N	1	N	95
D	REL_SQL_REQUESTS	DECIMAL	S	D	Y	8	N	1	N	103
D	PHYSICAL_IO	DECIMAL	S	D	Y	8	N	1	N	111
D	PHYSICAL_READS	DECIMAL	S	D	Y	8	N	1	N	119
D	REL_PHYS_READS	DECIMAL	S	D	Y	8	N	1	N	127
D	PHYSICAL_WRITES	DECIMAL	S	D	Y	8	N	1	N	135
D	REL_PHYS_WRITES	DECIMAL	S	D	Y	8	N	1	N	143
D	SEQ_READ_AHEAD	DECIMAL	S	D	Y	8	N	1	N	151
D	REL_SEQ_READ_AHEAD	DECIMAL	S	D	Y	8	N	1	N	159
D	ACCOUNTING_IO	DECIMAL	S	D	Y	8	N	1	N	167
D	REL_ACCOUNTING_IO	DECIMAL	S	D	Y	8	N	1	N	175
D	LOGGING_IO	DECIMAL	S	D	Y	8	N	1	N	183
D	REL_LOGGING_IO	DECIMAL	S	D	Y	8	N	1	N	191
D	IXX_USED_5	DECIMAL	S	D	Y	8	N	1	N	199
R	IXX_RATIO_5_TO_1	DECIMAL	S	D	Y	8	N	1	N	207
D	DXX_USED_5	DECIMAL	S	D	Y	8	N	1	N	215
R	DXX_RATIO_5_TO_1	DECIMAL	S	D	Y	8	N	1	N	223
D	DATA_USED_5	DECIMAL	S	D	Y	8	N	1	N	231
R	DATA_RATIO_5_TO_1	DECIMAL	S	D	Y	8	N	1	N	239
D	DATA2_USED_5	DECIMAL	S	D	Y	8	N	1	N	247
R	DATA2_RATIO_5_TO_1	DECIMAL	S	D	Y	8	N	1	N	255
D	NO_IXX_BUFFER	DECIMAL	S	D	Y	8	N	1	N	263
D	NO_DXX_BUFFER	DECIMAL	S	D	Y	8	N	1	N	271
D	NO_DATA_BUFFER	DECIMAL	S	D	Y	8	N	1	N	279
D	NO_EXPAND_BUFFER	DECIMAL	S	D	Y	8	N	1	N	287
D	NO_TXB_BUFFER	DECIMAL	S	D	Y	8	N	1	N	295
L	MEMORY_SIZE_V	DECIMAL	S	D	Y	8	N	1	N	303

ID	SQL NAME	SQL TYPE	CL	TY	SI	LE	NU	RE	RD	DI
D	MRDF_READS_V	DECIMAL	S	D	Y	8	N	1	N	311
D	REL_VIRTUAL_READS	DECIMAL	S	D	Y	8	N	1	N	319
D	VIRTUAL_WRITES	DECIMAL	S	D	Y	8	N	1	N	327
D	REL_VIRTUAL_WRITES	DECIMAL	S	D	Y	8	N	1	N	335
L	MEMORY_SIZE_C	DECIMAL	S	D	Y	8	N	1	N	343
D	MRDF_READS_C	DECIMAL	S	D	Y	8	N	1	N	351
D	REL_COVERED_READS	DECIMAL	S	D	Y	8	N	1	N	359
D	RECORD_LOCKS	DECIMAL	S	D	Y	8	N	1	N	367
D	WAIT_LOCK	DECIMAL	S	D	Y	8	N	1	N	375
D	REL_WAIT_LOCK	DECIMAL	S	D	Y	8	N	1	N	383
D	SECONDARY_CONFLICT	DECIMAL	S	D	Y	8	N	1	N	391
D	REL_SEC_CONFLICT	DECIMAL	S	D	Y	8	N	1	N	399
D	INDIRECT_WAIT_IO	DECIMAL	S	D	Y	8	N	1	N	407
D	WAIT_TASK	DECIMAL	S	D	Y	8	N	1	N	415
D	WAIT_REQUEST	DECIMAL	S	D	Y	8	N	1	N	423
D	WAIT_SPILL	DECIMAL	S	D	Y	8	N	1	N	431
D	WAIT_ACCOUNTING	DECIMAL	S	D	Y	8	N	1	N	439
D	WAIT_SECURITY	DECIMAL	S	D	Y	8	N	1	N	447
D	DB_CODE_COUNT	DECIMAL	S	D	Y	8	N	1	N	455
D	SQL_CODE_COUNT	DECIMAL	S	D	Y	8	N	1	N	463
D	INDEX_Q_OVERFLOW	DECIMAL	S	D	Y	8	N	1	N	471
D	IO_MEM_SHORT	DECIMAL	S	D	Y	8	N	1	N	479
L	LOG_PERCENT_FULL	DECIMAL	S	D	Y	8	N	1	N	487
D	LOGPEND_WRITES	DECIMAL	S	D	Y	8	N	1	N	495
D	LOG_WRITE_CONTROL	DECIMAL	S	D	Y	8	N	1	N	503
D	LOG_WRITE_FULL	DECIMAL	S	D	Y	8	N	1	N	511
D	LOG_WRITE_COMMAND	DECIMAL	S	D	Y	8	N	1	N	519
D	LOG_WRITE_OTHER	DECIMAL	S	D	Y	8	N	1	N	527
D	LOG_WRITE_TXB	DECIMAL	S	D	Y	8	N	1	N	535
D	LOG_WRITE_2_PHASE	DECIMAL	S	D	Y	8	N	1	N	543
D	CBS_SPILLS	DECIMAL	S	D	Y	8	N	1	N	551

ID	SQL NAME	SQL TYPE	CL	TY	SI	LE	NU	RE	RD	DI
L	CBS_ENTRIES	DECIMAL	S	D	Y	8	N	1	N	559
L	CBS_ENTRIES_USED	DECIMAL	S	D	Y	8	N	1	N	567
D	SETS_PROCESSED	DECIMAL	S	D	Y	8	N	1	N	575
D	SETS_WITH_T_INDEX	DECIMAL	S	D	Y	8	N	1	N	583
D	TEMP_INDEX_ENTRIES	DECIMAL	S	D	Y	8	N	1	N	591
D	BREAKS_DONE	DECIMAL	S	D	Y	8	N	1	N	599
D	RQ_PENDING_1_10	DECIMAL	S	D	Y	8	N	1	N	607
D	RQ_PENDING_11_20	DECIMAL	S	D	Y	8	N	1	N	615
D	RQ_PENDING_21_30	DECIMAL	S	D	Y	8	N	1	N	623
D	RQ_PENDING_31_40	DECIMAL	S	D	Y	8	N	1	N	631
D	RQ_PENDING_41_50	DECIMAL	S	D	Y	8	N	1	N	639
D	RQ_PENDING_GT_50	DECIMAL	S	D	Y	8	N	1	N	647
U	D_MUF_USERTAG	CHAR	S	C	N	16	N	1	N	655

Primary KEY columns are as follows:

- MUF_NAME
- D_MUF_ENABLE
- D_MUF_END_DATETIME
- D_MUF_TYPE
- D_MUF_USERTAG
- D_MUF_START_DATETIME

Secondary KEY columns are as follows:

- MUF_NAME
- D_MUF_ENABLE
- D_MUF_END_DATETIME
- D_MUF_TYPE
- D_MUF_USERTAG
- D_MUF_START_DATETIME
- D_MUF_DURATION

Chapter 10: AutoStatus Process

The AUTOSTAT function creates a Snapshot of selected MUFs with their active tasks.

The AutoStatus function utilizes the same concept of a source MUF and repository MUF as covered previously for the AutoCollect process. AutoStatus utilizes a single database to house the captured task activity snapshots.

Similar to AutoCollect, the AutoStatus function opens the AutoStatus database in the repository MUF using the standard DBSIDPR connection. Next, it calls a dynamic interface to connect to the selected source MUFs and collect DST information about the tasks running on the MUFs. The repository MUF can also be a target MUF.

Unlike AutoCollect, the AutoStatus task activity snapshot can be executed against multiple MUFs at one time as long as they are in the same LPAR.

Selecting a MUF as a Repository for a Collection

To execute AutoStatus, the repository MUF must be at CA Datacom/DB Version 12 (or greater) environment with the AutoStatus database activated. However, AutoStatus Version 12 allows a target MUF to be in CA Datacom/DB r11 or greater.

This flexibility allows you to begin using AutoStatus as soon as you have a repository MUF upgraded or new installed at Version 12.

The AutoStatus database is installed as part of a Version 12 install or upgrade from r11 to Version 12. The default database ID is 1018. While you can decide to change the AutoStatus database to a different DBID, we recommend that the database be left as 1018.

Selecting MUFs as a Source for Collection

The process allows a single Snapshot to collect information from up to 100 active MUFs in the targeted LPAR.

Since AutoStatus uses the standard DBSIDPR to communicate to the repository MUF, the repository MUF can be on a different LPAR (assuming that you have the cross LPAR communication set-up in the repository MUF). For more information about setting up the repository MUF for cross LPAR access, see the section in the *CA Datacom/DB Database and System Administration Guide* for XCF and CCI communication.

When more than one MUF is selected as a target for an AutoStatus Snapshot, the data is collected from the target MUFs in serial fashion in alphabetic order.

For AutoStatus executions that have a significant number of MUFs, the actual process may run for a few seconds. To view all of this information as a single rowset, the same timestamp is loaded into each row created by this Snapshot execution (S_MUF_DATETIME).

Allocating the AutoStatus Repository Database Datasets

The data collected by the AutoStatus process varies widely depending on your usage and the extent of your defined criteria. Therefore, determining the appropriate sizing of the index area (IXX1018) and the data area (STA1018) takes some user experience at each site.

Depending on the number of Snapshot Status rows captured and the length of time they are kept, the amount of data could be significant.

Both the index and data areas have dynamic extend set to Yes. However, verify that there is adequate space on the selected volume or volumes to allow the extend process.

Monitor the growth of the AutoStatus database and add space as needed.

For more information about the number of rows and row content of the AutoStatus table, see Collecting AutoStatus Snapshot Rowsets (see page 147).

Starting Allocations

The following are the suggested initial allocations:

Name	Job Code	allocation
IXX1018	highlevl.AUTOSTAT.IXX1018	135 Tracks

Name	Job Code	allocation
STA1018	highlevl.AUTOSTAT.STA1018	465 Tracks

For recommended starting allocations, see the *CA Datacom Installation and Maintenance Guide.*

Chapter 11: Collecting AutoStatus Snapshot Rowsets

This chapter covers the collection of Snapshot rowsets in the AutoStatus database table, S_MUF_ACTIVE_TASKS (STA1018).

Using the AUTOSTAT Command to Create Status Snapshots

Use the AUTOSTAT function when you want to create a Snapshot of selected MUFs with their active task information so that you can process the information and analyze the results. The data is saved in the AutoStatus database.

Command

AUTOSTAT

Specifies to create a Snapshot of the specified MUFs with their active tasks.

Selection Parameters

MUFNAME=

Identifies the MUF from which the task data is retrieved. You can repeat this parameter to include more than one MUF. Alternately, you can use a wildcard character (*) to specify to select all MUFs whose names begin with the characters you specify.

Repeat/Frequency parameters

REPEATS=

Specifies whether the AutoStatus function should remain active and perform periodic Snapshots (until stopped by the user). A value of zero indicates that the function should run only once and then end. A numeric value between 3 and 64800 indicates that the function should remain active and the interval between the Snapshots is set equal to the selected value in seconds. The interval is measured as the time from the end of one Snapshot and the beginning of the next Snapshot. You may want to set the parameter to a sufficient value, such as 600 (10 minutes), to keep the function from generating large numbers of rows in the AutoStatus database. If you are setting a repeats value, set-up the appropriate job parameters to keep the system from canceling the job due to long execution times.

Output parameters

OUTTAG=

Specifies a 1- to 16-character string that is used to fill in the USERTAG column on each row in the Snapshot table. You can use OUTTAG in various ways. For example, if you are collecting status information from multiple LPARs, you can indicate the LPAR where a given status was collected. In another case, you can process the status rows with your own program and, once a row has been read, you could place a value in the USERTAG column to indicate it has been processed.

Exclusion/Output reduction parameters

SNAPEXCL=

Specifies the items to exclude.

DURATION0

Suppresses the creation of any detail task rows that have a DURATION of blanks. During the task inquiry, any task that has not had its current command in process for at least one second has a duration of blanks. This parameter is used to reduce the amount of detail task rows that are written to the AUTOSTAT database. With this option in place, only tasks with a command in process for a duration of one second (:01) or greater are written to the SNAPSHOT database.

HEADER

Suppresses the writing of a new page header as each SYSPRINT page fills. When you want to keep the SYSPRINT to a minimum, you can use this option.

NOTACTIVE

Suppresses the creation of any detail task rows that have a CURRENT_STATUS of NOTACTIVE. This parameter is used to reduce the amount of detail task rows that are written to the AUTOSTAT database.

REPCONNECT

Suppresses the open and close of the AUTOSTAT database URT for each Snapshot (see REPEATS). Specifying SNAPEXCL=REPCONNECT tells the utility to open the URT at the beginning of the AUTOSTAT function and to leave it open until the function is ended. In this mode, the AUTOSTAT function issues a COMMIT at the end of each Snapshot process, but not close the URT.

Without the SNAPEXCL=REPCONNECT, the AUTOSTAT function disconnects from the repository MUF after each status Snapshot. Then, just prior to the next Snapshot, the function reopens the URT and reestablishes communication to the repository MUF. This helps ensure that the URT does not remain open for long periods of time while the function is in a "wait" between Snapshots. When you want to limit the amount of opens and closes, and the associated messages in the repository MUF region, specify the SNAPEXCL=REPCONNECT option.

Note: With the SNAPEXCL=REPCONNECT option specified, the user must terminate AUTOSTAT before issuing the EOJ to the repository MUF.

WARNMSG

Suppresses the creation of warning messages when specified MUFs are not found (not active) when the status Snapshot is executed.

The expected return code is 0 when executing the AUTOSTAT command. For details on the AUTOSTAT syntax, parameters, JCL and sample output report see the *CA Datacom/DB DBUTLTY Reference Guide*.

Communicating with AUTOSTAT Using Console Commands

When you specify a REPEATS= value, the AutoStatus process remains active and repeats the function as requested. In addition, AUTOSTAT attaches a console subtask to allow you to communicate to the AUTOSTAT while it is executing.

Any console command that changes an operational parameter except REPEATS, such as EXCLUDE DURATION0, goes into effect at the next processing cycle.

For example, if the task is running with REPEATS=600 (10 minutes), and you issue the console command EXCLUDE DURATION0, the change is made in the operational parameters, but it does not take effect until the next status Snapshot occurs according to the repeat value. Issuing the REPEATS console command changes the current operational value for the REPEATS parameter and also triggers an immediate status Snapshot to be taken.

When you have a long REPEATS value, the console interaction allows you to make the change and also force an immediate status Snapshot. In cases, where you want to force a status Snapshot without changing the overall operational parameters, you can do this by just issuing the REPEATS console command with a value equal to the current REPEATS value.

Issuing the AS_OPTIONS console command does not change any operational values or trigger a Snapshot. AS_OPTIONS only requests that the current operational values are printed to SYSPRINT for review purposes.

Console interaction is provided through the following /STOP, /CANCEL, and /MODIFY processes.

/STOP jobname

Terminates the AUTOSTAT task. The process is the same as the following process "/MODIFY jobname,EOJ".

/CANCEL jobname

Causes the AUTOSTAT task to cancel with a S222 abend.

/MODIFY jobname,command

Issues changes using a specific command format as documented in the following section. It allows you to pass information to the active AUTOSTAT task.

AutoStatus MODIFY Command Format

The MODIFY command has the following format:

```
/MODIFY — jobname,command
```

The jobname is the name of the AUTOSTAT job. Command is the action taken and value is the optional value supplied to the command. There must only be a comma separating the jobname and the command and there must be a space separating the command and the value, if specified.

Console commands are echoed to the SYSPRINT output.

Command

ADDMUF newmuf

Add the new MUFNAME to the list of MUFs that are being targeted for the status Snapshots. newmuf is 1 to 8 alphanumeric characters.

Note: There is a limit of 100 MUFNAMEs per AUTOSTAT execution.

AS_OPTIONS

Produces a listing of all AUTOSTAT options in effect to the SYSPRINT output.

COMM EOJ

Same as EOJ.

EOJ

Complete current Snapshot, set return code, and terminate the AUTOSTAT program.

EXCLUDE DURATION0

Excludes tasks that have a "blank" duration.

EXCLUDE HEADER

Tells the utility to stop producing heading pages to the SYSPRINT output file.

EXCLUDE NOTACTIVE

Excludes tasks that have a NOT ACTIVE status.

EXCLUDE REPCONNECT

Suppresses the URT open and close that occurs with the REPEATS= parameter.

Note: With the EXCLUDE REPCONNECT option specified, you must terminate AUTOSTAT before issuing the EOJ to the repository MUF.

EXCLUDE WARNMSG

Suppresses warning messages.

INCLUDE DURATION0

Includes tasks regardless of duration.

INCLUDE HEADER

Produces a page header at the top of each new page of the SYSPRINT output file. Each page is typically 60 lines.

INCLUDE NOTACTIVE

Includes tasks regardless of task status.

INCLUDE REPCONNECT

Activates the URT open and close that occurs with the REPEATS= parameter.

INCLUDE WARNMSG

Enables warning messages.

OUTTAG tag1

Changes the current OUTTAG value to a new value. Value must be from 1 through 16 alphanumeric characters.

REMOVEMUF oldmuf

Remove the existing MUFNAME from the list of MUFs that are being used as the source for the status Snapshots.

Note: There must be at least one valid MUFNAME per AUTOSTAT execution. If you attempt to remove the last MUFNAME, the command is ignored and a message issued.

REPEATS nnnnn

Changes the current REPEATS= parameter timer to this new value. Value must be 0 or from 3 through 64800. Changing the value to 0 causes the utility to do one last Snapshot and end. Changing the value to other than 0 forces an immediate Snapshot and changes the wait interval between further Snapshots and the utility continues to execute in the loop with the new REPEATS value.

Reporting Status Rowsets

As needed, a status rowset report can be created by executing DBSQLPR, DQRY, or a user program. For more information about the DBSQLPR program, see the *CA Datacom/DB SQL User Guide.*

Deleting Status Rowsets

As needed, status rowsets can be deleted using DBSQLPR, DQRY or a user program. For more information about the DBSQLPR program, see the *CA Datacom/DB SQL User Guide*.

Chapter 12: Suggested AutoStatus Implementation

The following is a suggested implementation if you are a first-time user of AutoCollect.

First, perform the following research and testing at your site.

1. Determine which MUFs are selected for AutoStatus processing.
2. Determine which MUFs are Source MUFs

   Verify that the Source MUF(s) have the Dynamic System Tables activated.
3. Determine which MUF(s) are the Repository MUF

   Note: If you are a first-time user, we suggest that you select one MUF and use it as both the Source and Repository MUF.
4. Verify that the repository MUF(s) have the AutoStatus Snapshot database installed in the CXX (DBID 1018). A simple CXX report should be able to validate their presence and also provide the current allocation sizes.
5. If the database is not present, check the Version 12 installation process for the steps to install the database. If you cannot determine why the database was not installed, contact CA Support.
6. If the dataset sizes are not large enough or if you have not allocated and initialized the two datasets for the database, follow the instructions earlier in this guide to allocate and initialize the AutoStatus database.
7. Before starting data collection, run a DBUTLTY LOAD FORMAT=NONE to verify that the database is empty and ready to receive AutoStatus data.

We strongly recommend that the user implement the AutoStatus databases as 1018 (Snapshot). If the user chooses to use another DBID for this database, the various AutoStatus functions of DBUTLTY will need to include the DBID on every AutoStatus function execution.

The user must next create a set of DBUTLTY jobs as follows to execute and test the various AutoStatus functions.

1. Execute OPTION=SNAPSHOT several times.
2. (For sites with SQL) Follow the examples for using DBSQLPR to print a report of the data in a Snapshot rowset.
3. Execute OPTION=SNAPSHOT utilizing the various exclude commands.
4. Execute OPTION=SNAPSHOT utilizing the ability to gather statistics from multiple MUFs.
5. Execute OPTION=SNAPSHOT utilizing the repeat option to create a long-running job.
6. Use the console commands to alter the Snapshot options
7. Use the AS_OPTIONS console commands to verify the changes
8. (For sites with SQL) Follow the examples for using DBSQLPR to print a report of the data in a Snapshot rowset.

Once the initial testing phase has been completed, the user should then begin to plan the AutoStatus snapshot collection process. Key to this process will be the repeat time values for how often a status Snapshot should be taken while the Source MUF is normally executing.

As we have seen in the various examples in Collecting AutoStatus Snapshot Rowsets chapter, the timing of the MUF startup and shutdown will affect when we do over status Snapshot collection points.

The process below provides a simple implementation for collection status Snapshots every 10 minutes.

- Select a set of MUFs that you would like to monitor active task status.
- Determine a reasonable interval for the status snapshots:
  - For typical implementations this could be every 600 seconds.
  - If the repeat value is smaller than one minute you may create significant amount of data rows.
  - If the repeat value is too high you may not get the level of granularity that you need.

- Collect 24 hours worth of status snapshots.
- Use DBSQLPR to do various searches on the data.
    - Look for any tasks that have a long duration:
        - Typically for RAAT systems this would be DURATION of 2 seconds or greater.
        - Typically for SAAT or SQL systems this would be DURATION of 5 seconds or greater.
        - Use the S_MUF_DATETIME value from the “selected row” to re-query the status snapshots to print all task rows from the selected status snapshots. Review the information to determine what caused the request to be delayed.
    - Look for any tasks that have a high I/O consumption:
        - Typically for RAAT systems this would be PHYSICAL_EXCPS of 2 I/Os or greater.
        - Typically for RAAT systems this would be PHYSICAL_EXCPS of 3 I/Os or greater.
        - Use the S_MUF_DATETIME value from the “selected row” to re-query the status snapshots to print all task rows from the selected status snapshots. Review the information to determine what caused the high IO.
    - Look for any tasks that have a high buffer utilization:
        - Typically for RAAT systems this would be BUFFER_REFERENCES of 50 or more.
        - Typically for RAAT systems this would be BUFFER_REFERENCES of 150 or greater.
        - Use the S_MUF_DATETIME value from the “selected row” to re-query the status snapshots to print all task rows from the selected status snapshots. Review the information to determine what caused the high buffer references.
- Save the status snapshots for a reasonable period of time. The information can be utilized if a user reports a MUF disturbance. If a time is provided, you can review the status information from around the problem period to see if there was problem with MUF.
- Delete AutoStatus rows as needed. You can use DBSRPPR to delete older status rowsets.

Chapter 13: AutoStatus Table Definitions

AutoStatus Column Definitions

The columns found in the AutoStatus table are the same definitions as the columns in the Dynamic Systems Table MUF_ACTIVE_TASKS table. For information about the DST tables, see the *CA Datacom/DB System Tables Reference Guide*.

For additional details about the values for task status and other text fields, see the DBUTLTY COMM STATUS section of the *DBUTLTY Reference Guide for z/OS*.

In addition to the DST columns, the AutoStatus status table includes the S_MUF_DATETIME column which is the time of the status snapshot and the S_USERTAG column which provides you a way to tag specific status snapshots for later reference.

S_MUF_ACTIVE_TASKS (MFQ)

SQL NAME	SQL TYPE	CLA	TYP	SGN	LNG	NUL	REP	RDF	DIS
S_MUF_DATETIME	SQL-STMP	S	B	N	10	N	1	N	0
MUF_NAME	CHAR	S	C	N	8	N	1	N	10
TASK_NUMBER	SMALLINT	S	B	N	2	N	1	N	18
JOB_NAME	CHAR	S	C	N	8	N	1	N	20
RUN_UNIT	INTEGER	S	B	N	4	N	1	N	28
OPTIONAL_ID	CHAR	S	C	N	16	N	1	N	32
CURRENT_STATUS	CHAR	S	C	N	15	N	1	N	48
OWNER_TASK	SMALLINT	S	B	N	2	N	1	N	63
DB_COMMAND	CHAR	S	C	N	5	N	1	N	65
REQUEST_SEQ_NO	INTEGER	S	B	N	4	N	1	N	70
DURATION	CHAR	S	C	N	6	N	1	N	74
PHYSICAL_EXCPS	INTEGER	S	B	N	4	N	1	N	80
BUFFER_REFERENCES	INTEGER	S	B	N	4	N	1	N	84
DBID	SMALLINT	S	B	N	2	N	1	N	88
TABLE_NAME	CHAR	S	C	N	3	N	1	N	90

SQL NAME	SQL TYPE	CLA	TYP	SGN	LNG	NUL	REP	RDF	DIS
USER_RQ_DATA	CHAR	S	C	N	32	N	1	N	93
TRN_SEQ_NO	CHAR	S	C	N	8	N	1	N	125
MUFPLEX_OWNER	CHAR	S	C	N	2	N	1	N	133
LOCK_VALUE	CHAR	S	C	N	64	N	1	N	135
TSN_DURATION	CHAR	S	C	N	6	N	1	N	199
RUN_TIME	CHAR	S	C	N	6	N	1	N	205
CPU_TIME	CHAR	S	C	N	6	Y	1	N	212
WAIT_DURATION	CHAR	S	C	N	6	Y	1	N	219
WAIT_TIME	CHAR	S	C	N	6	N	1	N	225
USER_PATH	CHAR	S	C	N	5	N	1	N	231
USER_SYSTEM	CHAR	S	C	N	12	N	1	N	236
USER_JOBID	CHAR	S	C	N	12	N	1	N	244
EOJ_OK	CHAR	S	C	N	1	N	1	N	252
S_USERTAG	CHAR	S	C	N	16	N	1	N	253

Note: EOJ_OK has the following possible values:

- Y - Task will be removed after EOJ
- N - Task will not be removed after EOJ or the source MUF is Version 12.0 where no such field exists

Primary KEY columns are as follows:

- MUF_NAME
- TASK_NUMBER
- S_MUF_DATETIME

Secondary KEY columns are as follows:

- S_USERTAG
- MUF_NAME
- TASK_NUMBER
- S_MUF_DATETIME

Chapter 14: AutoInfo Process

AutoInfo is like AutoCollect and AutoStatus in that it uses similar data collection facilities to gather DST information. In addition to DST information, AutoInfo also collects environmental information from the MUF address space and other memory areas associated with the MUF.

The AutoInfo process is for users that have limited database administrative resources or have limited experience.

During problem periods, the AutoInfo process can be used to create background documentation to assist in the problem determination.

In non-problem times, the AutoInfo process can be used to document the MUF environment and provide basic (single snapshot) performance information.

Create the AutoInfo report

The AutoInfo process provides a standardized report for each MUF at the client site. The report can be used to document current environmental information and problem documentation.

In addition to the report created at the user site, the AutoInfo function can generate a sequential file with the same information in DSV format. This file is easily included in an FTP file sent to support. When working with CA Support on an issue, the user site can use AutoInfo to easily and quickly document the MUF environment.

By providing this information, the user is free from spending additional time collecting the background information and CA Support is able to assist in quicker problem resolution.

Set up the AutoInfo Environment

This section discusses the setup of the AutoInfo environment. The AutoInfo process utilizes information in the dynamic system tables. It also collects information found in the MUF memory and other system memory areas.

The only required set-up for full functionality of AutoInfo in the implementation of the Dynamic System tables in the target MUF.

Determine the Location of the Source MUFs

AutoInfo provides you with the ability to collect environmental and performance data from a MUF, known as the source MUF.

Source MUF communication uses the standard navigation process

The AUTOINFO function uses the standard DBSIDPR navigation process to communicate to the repository MUF. This process allows communication to MUFs through SVC (same LPAR), XCF (different LPAR, but same Sysplex), or CCI (different Sysplex). Since information is also being collected from system memory, the AutoInfo process must be executed on the same LPAR at the target MUF.

Select a MUF as a Source for AutoInfo Collection

You can only select one MUF per AutoInfo collection.

The DBSIDPR module is used to target a MUF environment as the source for data collection.

Execute the AUTOINFO Function

The AUTOINFO function must be executed on the same LPAR as the source MUF. When executing the function, the STEPLIB concatenation must include the following:

- Library concatenation that provides a DBSIDPR module that points to the source MUF.
- Complete set of CA Datacom execution libraries.

The AUTOINFO function performs the following:

- Uses a standard URT to connect to the source MUF through DBSIDPR.
- Uses the dynamic navigation process to connect to the source MUF memory.
- Issues the appropriate commands to retrieve the DST rows from the source MUF.
- Formats and stores the DST rows as report output and as an optional sequential dataset.
- Produces a report providing as much information as is available.
  - Active MUFs provide a full report.
  - A MUF that has been active since the last IPL, but is not currently active provides some basic memory information.
  - A MUF which has not been started since the IPL does not provide any information.

The expected return code is 0 when executing the AUTOINFO function. For details on the AUTOINFO syntax, parameters, and sample output report see the *CA Datacom/DB DBUTLTY Reference Guide*.

Index