5.4

# *IBM OMEGAMON for Db2 Performance Expert on z/OS*
# *Buffer Pool Analyzer User's Guide*

IBM

**SH12-7075**

**2023-10-26 edition**

This edition applies to Version 5 Release 4 of IBM® OMEGAMON for DB2® Performance Expert on z/OS (product number 5655-W37) and to all subsequent releases and modifications until otherwise indicated in new editions.

# Contents

# About this information

IBM OMEGAMON for Db2 Performance Expert on z/OS (also referred to as OMEGAMON for Db2 Performance Expert) is a performance analysis, monitoring, and tuning tool for Db2 on z/OS® environments.

The document is part of the OMEGAMON for Db2 Performance Expert documentation library which provides instructions for installing, configuring, and using OMEGAMON for Db2 Performance Expert and is designed to help database administrators, system programmers, application programmers, and system operators perform these tasks:

- Plan for the installation of OMEGAMON for Db2 Performance Expert
- Install and operate OMEGAMON for Db2 Performance Expert
- Customize your OMEGAMON for Db2 Performance Expert environment
- Diagnose and recover from OMEGAMON for Db2 Performance Expert problems
- Design and write applications for OMEGAMON for Db2 Performance Expert
- Use OMEGAMON for Db2 Performance Expert with other DB2 products

# Chapter 1. Overview

IBM OMEGAMON for Db2 Performance Expert on z/OS (OMEGAMON for Db2 Performance Expert) enables you to monitor, analyze, and tune the performance of your Db2 subsystems and Db2 applications.

## Service updates and support information

Service updates and support information for this product, including software fix packs, PTFs, frequently asked questions (FAQs), technical notes, troubleshooting information, and downloads, are available from the web.

To find service updates and support information, see the following website:

https://www.ibm.com/support/pages/omegamon-xe-db2-pepm-web-based-delivery-and-updates-windows-and-unix-based-components

## How to read syntax diagrams

The rules in this section apply to the syntax diagrams that are used in this publication.

**Arrow symbols**

Read the syntax diagrams from left to right, from top to bottom, following the path of the line.

►►——

Two right arrows followed by a line indicate the beginning of a statement.

——►

One right arrow at the end of a line indicates that the statement syntax is continued on the next line.

►——

One right arrow followed by a line indicates that a statement is continued from the previous line.

——►◄

A line followed by a right arrow and a left error indicates the end of a statement.

**Conventions**

- SQL commands appear in uppercase.
- Variables appear in italics (for example, *column-name*). They represent user-defined parameters or suboptions.
- When entering commands, separate parameters and keywords by at least one blank if there is no intervening punctuation.
- Enter punctuation marks (slashes, commas, periods, parentheses, quotation marks, equal signs) and numbers exactly as given.
- Footnotes are shown by a number in parentheses, for example, (1).

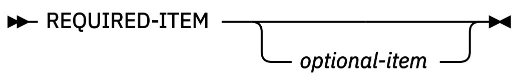
**Required items**

Required items appear on the horizontal line (the main path).

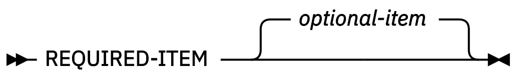
```
►►─ REQUIRED-ITEM ─►◄
```

**Optional items**

Optional items appear below the main path.

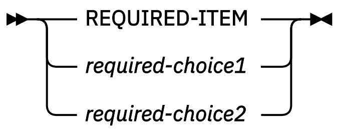


If an optional item appears above the main path, that item has no effect on the execution of the statement and is used only for readability.

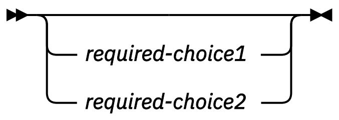


**Multiple required or optional items**

If you can choose from two or more items, they appear vertically in a stack. If you *must* choose one of the items, one item of the stack appears on the stack main path.

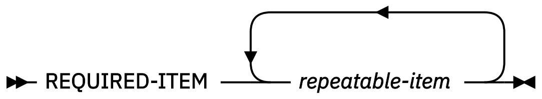


If choosing one of the items is optional, the entire stack appears below the main path.

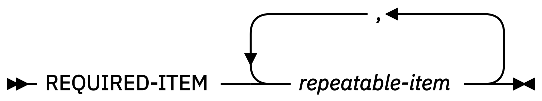


**Repeatable items**

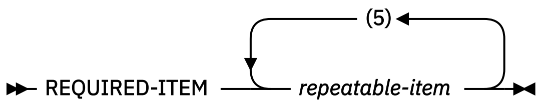
An arrow returning to the left above the main line indicates that an item can be repeated.



If the repeat arrow contains a comma, you must separate repeated items with a comma.



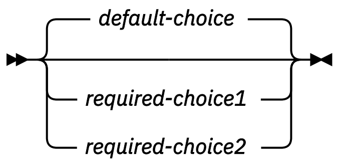
If the repeat arrow contains a number in parenthesis, the number represents the maximum number of times that the item can be repeated.



A repeat arrow above a stack indicates that you can specify more than one of the choices in the stack.

**Default keywords**

IBM-supplied default keywords appear above the main path, and the remaining choices are shown below the main path. In the parameter list following the syntax diagram, the default choices are underlined.



## Conventions

These conventions are used throughout the documentation.

### Symbols

The following symbols might appear in command syntax:

| Symbol | Usage |
|---|---|
| \| | The **or** symbol is used to denote a choice. You can use the argument on the left or the argument on the right. For example: `YES \| NO` In this example, you can specify YES or NO. |
| ( ) | Denotes optional arguments. Arguments that are not enclosed in square brackets are required. For example: `APPLDEST DEST (ALTDEST)` In this example, DEST is a required argument and ALTDEST is optional. |
| { } | Some documents use braces to denote mandatory arguments, or to group arguments for clarity. For example: `COMPARE {workload} - REPORT={SUMMARY \| HISTOGRAM}` In this example, the workload variable is mandatory. The REPORT keyword must be specified with a value of SUMMARY or HISTOGRAM. |
| _ | Default values are underscored. For example: `COPY infile outfile - [COMPRESS={YES \| NO}]` In this example, the COMPRESS keyword is optional. If specified, the only valid values are YES or NO. If omitted, the default is YES. |

### Notation conventions

The following conventions are used when referring to high-level qualifiers:

***hilev***

A high-level qualifier. The high-level qualifier is the first prefix or set of prefixes in the data set name. Site-specific high-level qualifiers are shown in italics.

For example:

- *thilev* refers to the high-level qualifier for your target data set.
- *rhilev* refers to the high-level qualifier for your runtime data set.

  For members in target libraries, the high-level qualifier is *thilev* rather than *rhilev*.
- *shilev* refers to the SMP/E library high-level qualifier.

## Terminology

The following table shows the products that are described in this publication and the short names with which they are referred to throughout this publication.

*Table 1. Product names and their short names*

| Product name | Short name |
|---|---|
| IBM OMEGAMON for Db2 Performance Expert on z/OS | OMEGAMON for Db2 Performance Expert |

## Accessibility features

Accessibility features help a user who has a physical disability, such as restricted mobility or limited vision, to use a software product successfully.

The major accessibility features in this product enable users to perform the following activities:

- Use assistive technologies such as screen readers and screen magnifier software. Consult the assistive technology documentation for specific information when using it to access z/OS interfaces.
- Customize display attributes such as color, contrast, and font size.
- Operate specific or equivalent features by using only the keyboard. Refer to the following publications for information about accessing ISPF interfaces:
  - *z/OS ISPF User's Guide, Volume 1*
  - *z/OS TSO/E Primer*
  - *z/OS TSO/E User's Guide*

  These guides describe how to use the ISPF interface, including the use of keyboard shortcuts or function keys (PF keys), include the default settings for the PF keys, and explain how to modify their functions.

# Chapter 2. Overview

Buffer Pool Analyzer provides a suite of tools that support in-depth analysis of the performance of DB2 buffer pools.

- Collection of buffer pool related performance data
- Host-based creation of reports about buffer pool performance and group buffer pool performance
- Conversion of performance data to formats suitable for client-based functions and for loading into DB2 tables
- Client-based graphical representation of buffer pool performance data
- Client-based optimization of major buffer pool attributes, like the optimum assignment of objects in buffer pools and optimum buffer pool sizes, based on actual performance data
- Client-based simulation of the effects of different buffer pool attributes, based on actual performance data
- Client-based long-term analysis of historical and current performance data

These tools help performance analysts and database administrators to monitor, analyze, and optimize DB2 buffer pools on different levels in an effective way.

## How Buffer Pool Analyzer is used

The suite of Buffer Pool Analyzer tools can be used to monitor, analyze, and tune buffer pools.

The tools' usage varies with the goal to be achieved.

- If frequent monitoring and performance observation with minimum effort is the goal, high level summary reports can be created on the host from collected performance data. This process can be automated by frequently running batch jobs. Data collection and report creation can be configured for individual needs. Optionally, data can be viewed on the Buffer Pool Analyzer client in attractive and intuitive graphical representations.
- If an analysis of buffer pool related problems is needed, several summary and detail reports can be created to quickly identify possible problem areas. Reports can be customized to provide timely and content-specific information.
- If optimization and tuning of buffer pool resources and usage is the goal, the object placement and buffer pool sizing tool and the buffer pool simulation tool are first choice. Based on real and representative buffer pool performance data, these tools ease the process of finding optimal use of buffer pool resources and simulating the effects of possible changes.
  - The object placement and initial buffer pool sizing tool uses predefined and modifiable expert rules and the objects actual access behavior to calculate optimized buffer pool arrangements. It recommends ready-to-use SQL ALTER statements and Db2 ALTER commands, with their parameters set to the recommended values. The tool can be used to balance buffer pool sizes, for example to separate sequentially from randomly accessed objects into different buffer pools, to optimize memory usage, and to improve application response times.
  - The simulation tool uses actual objects' access behavior and simulates different object placements and buffer pool size ranges. The simulation results provide a reliable prediction about the effects that different placements and sizes would have on a system. Simulation is used to perform what-if scenarios to balance buffer pool sizes and performance and to provide precise information about the prospective effects of different buffer pool scenarios.

  Both tools complement each other by performing the (often complex and iterative) task of optimization and tuning on a client, thereby still relying on actual performance data. The strength of object placement and initial buffer pool sizing is its rule-based algorithm and its ready-to-use recommendations. Simulation takes the surprises out of planned changes and minimizes the number of system disruptions.

The long-term analysis function adds another dimension to monitoring, analysis, and tuning: historical and current performance data can be combined and analysed as a whole to easily detect trends, hourly, daily, and weekly peaks, repetitive performance pattern, unbalanced resource usage, and much more. The client-based long-term analysis function provides an array of intuitive selections to focus on important performance indicators, buffer pools, and database objects.

## Benefits of using Buffer Pool Analyzer

Buffer Pool Analyzer offers performance analysts and database administrators tuning advice that is based on the analysis of DB2 trace data.

The benefits from analyzing buffer pool performance with Buffer Pool Analyzer are:

- Easy monitoring of the performance of buffer pools and group buffer pools to detect bottlenecks, trends, and unused resources
- Fast adaptation of buffer pool parameters to changing DB2 usage conditions
- Optimized use of buffer pools by aligning buffer pool size and object placement to available resources
- Non-disruptive simulation of buffer pool behavior to test the impact of changes before they are applied
- Long-term analysis of factual performance for improved prediction of future performance and resource needs

## The role and importance of Db2 buffer pools

Db2 buffer pools are the means of caching frequently used Db2 data in fast memory to prevent or at least reduce the number of slow input/output (I/O) operations.

Buffer pools are used to cache disk pages of databases. Buffer pool management algorithms handle prefetching of blocks of data before the pages are needed, maintain them in buffers for faster access by Db2 applications, and write them back to disk asynchronously, thus maximizing the performance of applications.

When Db2 is started, buffer pools are initiated with attributes that determine, for example, the sizes and thresholds of individual buffer pools. The activity in buffer pools and their efficiency is affected by these static attributes.

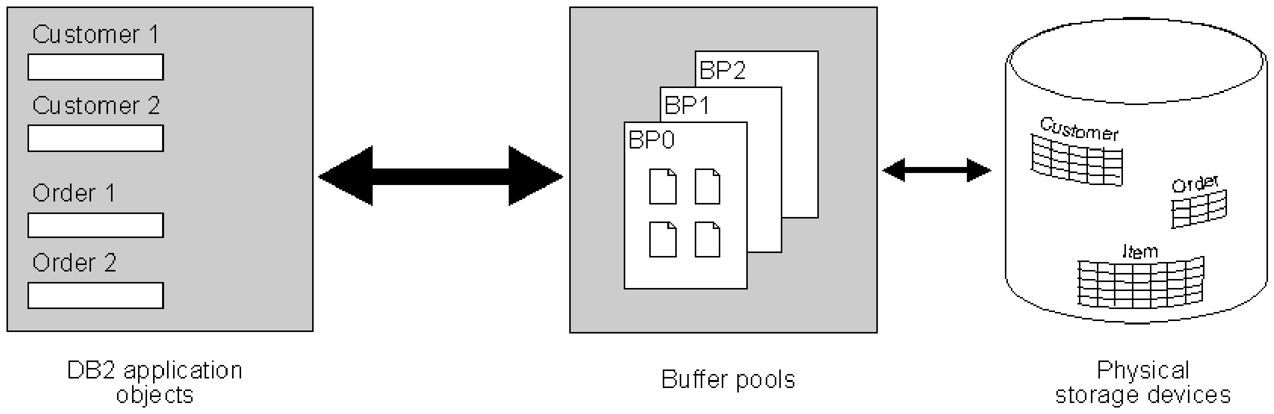


*Figure 1. The role of buffer pools in Db2 systems*

However, the activity in buffer pools is far more dependent on how the data in Db2 table spaces and index spaces is accessed. Table spaces and index spaces, commonly called *objects* in this information, hold Db2 tables and associated indexes. These spaces are divided into equal-sized pages, which are written to or read from disk in one operation. When the size of Db2 tables and indexes changes over time, or when the frequency and nature of accesses from Db2 changes, the initial buffer pool attributes might not be optimal after some time.

Db2 provides a set of commands and SQL statements to alter the size of buffer pools and the assignments of Db2 objects to buffer pools. These commands provide a means of altering buffer pool attributes that were set when Db2 was started, and they can be used to adapt the buffer pool characteristics to match the changing usage of Db2 data.

Despite the availability of commands, expert knowledge is required to optimally lay out a system and set up the parameters correctly. Important buffer pool attributes, like buffer pool sizes and assignments of objects to buffer pools, should not be seen as being final and might need to be varied over time to efficiently use the available buffer pools. Therefore, it is essential to frequently monitor the actual usage and performance of buffer pools to identify bottlenecks and to adjust the buffer pool attributes to their most efficient values. The performance of buffer pools strongly influences the data throughput of a Db2 system.

## The role and importance of Db2 group buffer pools in data sharing groups

In a parallel sysplex environment two or more Db2 subsystems can be grouped in a so-called *data sharing group* to share a single set of data while maintaining data integrity.

Each member, respectively Db2 subsystem, of a data sharing group continues to own its local set of buffer pools for the purposes and benefits described so far. However, to coordinate the flow of data between multiple subsystems and the shared set of physical I/Os, a common set of buffer pools is required as intermediary. This set is called *group buffer pool* because it serves all members of a data sharing group. Group buffer pools (GBPs) are located in a coupling facility (CF), the piece of hardware that provides a shared memory capability in a parallel sysplex environment.

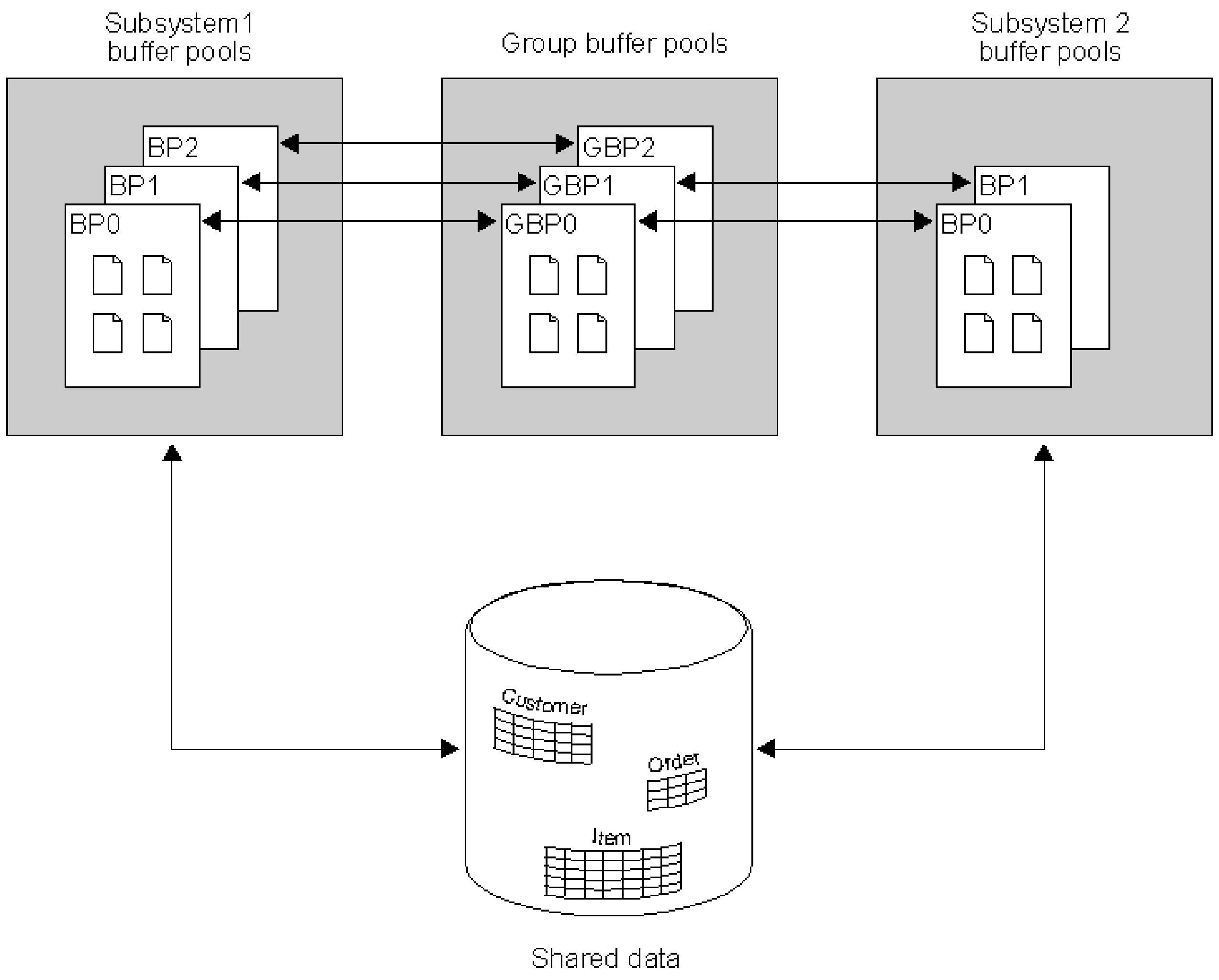


*Figure 2. The role of a group buffer pool in a Db2 data sharing group*

Db2 provides the necessary data sharing mechanisms for locking and caching of data to ensure data coherence among member and group buffer pools, and to ensure data consistency in the entire data sharing group. Further, Db2 provides commands to create, alter, and monitor group buffer pools, similar to commands for buffer pools of individual subsystems.

With regard to performance, it can easily be seen that group buffer pools are subject to similar criteria than the members' buffer pools: They prefetch data from disks in advance, cache it for use by the members' buffer pools, and cast out data to disks asynchronously. The important point is that the activity in group buffer pools is the cumulative activity of the members' buffer pools. Therefore, it is even more essential to frequently monitor and tune the performance of group buffer pools to maintain a high data throughput.

For a detailed introduction to Db2 data sharing and group buffer pools, see *Db2 11 Data Sharing: Planning and Administration*.

## How Buffer Pool Analyzer supports performance analyses

Buffer Pool Analyzer specializes in the analysis of buffer pool-related performance data. It belongs to a suite of DB2 tools and products that assist in the management of DB2 systems. It is also an integrated part of DB2 Performance Expert that specializes in the entire performance of DB2 systems.

At this time, it should be clear that buffer pool-related analysis, monitoring, and tuning are accompanying measures in the overall analysis, monitoring, and tuning processes of the operating system, the DB2 system, the SQL operations, and the applications. Buffer pool tuning is not an alternative to other tuning measures, although it might compensate some poorly tuned components.

The word *analysis* throughout this information means a detailed examination of buffer pool activities and buffer pool performance to discover the essential features of its operational behavior. The Buffer Pool Analyzer tools provide adequate information to you to perform the analysis, which enables you to assess the performance and to identify potential improvements. The outcome of an analysis is one or more *activity reports* at several levels of detail in text or graphical form.

Buffer Pool Analyzer supports the analysis by also providing two powerful analysis tools. The first is an *optimization* tool, called *object placement and initial sizing*, and generates recommendations for both the assignment of objects to buffer pools and for buffer pool sizes and parameters. The second is a *simulation* tool and permits the user to anticipate the effects of different settings before they are applied to a system. It recommends an ideal distribution of memory between buffer pools for the given setup.

The *long-term analysis* tool can use the repository of historical and current performance data and facilitates analyses and comparisons as preferred and needed by the user: a flexible selection of periods to analyze, an easy selection of performance indicators and buffer pool objects of interest, and a multitude of graphical representations and charts of the analysis results provides comprehensive insights into buffer pool performance.

*IBM DB2 11 for z/OS: An Introduction to DB2 for z/OS* dedicates a chapter to performance management on a DB2 level, which also correlates the role of buffer pool performance with overall DB2 performance. *DB2 11 Administration Guide* provides more details about performance monitoring and tuning on a DB2 level.

## Buffer pool analysis and tuning processes

Analyzing the performance of buffer pools and tuning a Db2 system for optimum performance is a process that can serve several purposes and that involves one or more tasks dependent on your requirements and motivations.

Different tasks take varying amounts of time, might require different levels of knowledge, and require different level of information or support. This topic outlines some typical database administrator goals and tasks. It is basically about *why*, *what*, and *when* something should be done.

## Observe performance of buffer pools

Administrators with many systems need a single method of monitoring those systems, especially when they are complex as in data sharing groups.

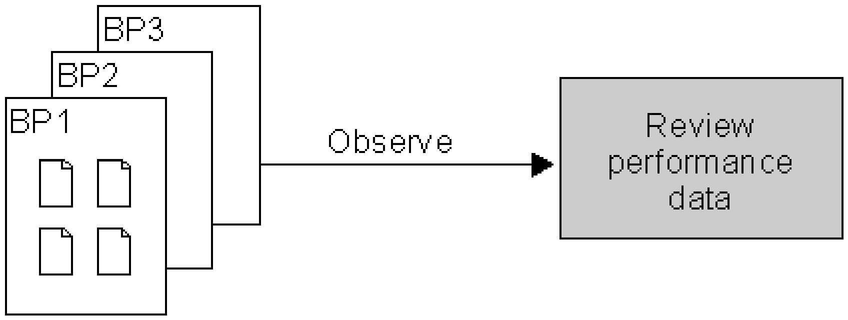


*Figure 3. Observing and reviewing buffer pool performance*

A single observation provides you with a snapshot of current system behavior. Repeated observation keeps systems healthy by regularly checking for deviations from an expected level of performance.

These tasks require high-level summary information about critical buffer pool components and activities. The information should reflect a representative workload, and it should ideally be comparable with previous information. It should be easy to obtain and cause no significant load on the monitored system.

Current summary information should be collected and analyzed regularly.

You can use this information to detect out-of-line situations, to develop reference points for future comparisons, and to detect performance trends. Administrators can automate the collection of data by setting up scheduled batch jobs that collect specified performance data and create appropriate activity reports about buffer pools and group buffer pools. This ensures that up-to-date reports are always available for analysis.

## React to out-of-line situations

Up-to-date performance data is used to quickly react to out-of-line situations.

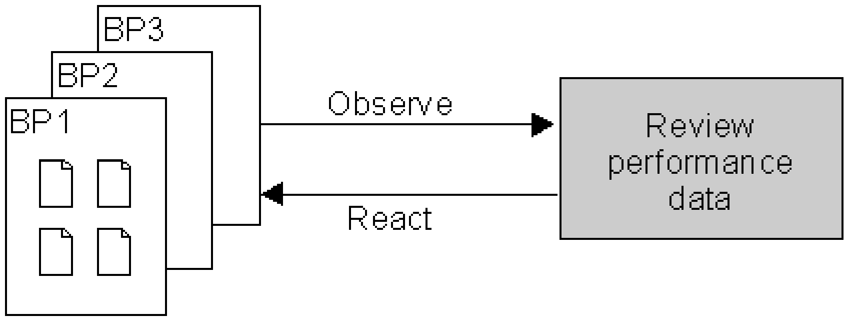


*Figure 4. Using up-to-date performance data to react quickly*

This task is often driven by user concerns about the performance of Db2 applications. It can also be driven by unexpected results observed in basic reports described previously. If such problems prove to be related to buffer pools or group buffer pools, they require in-depth analysis.

You need to be able to recognize these problems quickly and to react to them quickly. Your immediate needs are up-to-date performance information at several levels of detail, usually beginning with summary information, followed by more detailed information. This information must reflect the performance from the time that applications were observed to run slowly, or when summary report information shows unexpected behavior.

## Optimize buffer pool usage and sizes

A major task of database administrators is the tuning of complex systems.

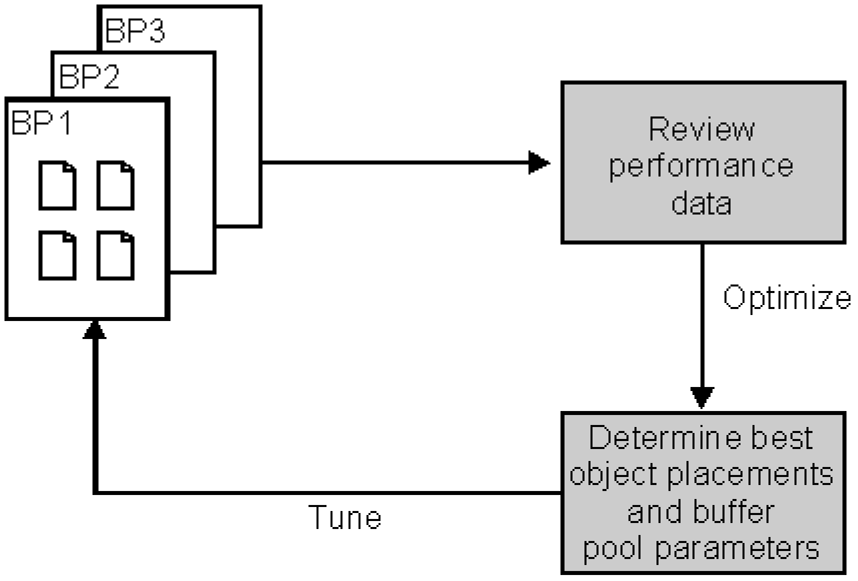


*Figure 5. Optimizing buffer pools and reviewing the success*

This task includes the optimization of buffer pools for the most effective use by Db2. It is the process of finding the most effective distribution of Db2 objects in the available buffer pools, based on the characteristics of the objects. And it includes determining efficient buffer pool parameters for these placements. An optimized buffer pool provides the best possible hit ratios, in terms of pages found in the buffer pools, related to the given sizes of the buffer pools.

The need for optimizing the usage of buffer pools can have several reasons:

- New Db2 applications might require new objects, and they might need to share existing buffer pool resources.
- Table spaces are resized, table columns are added, or page sizes are changed.
- The usage of Db2 objects changed over time and demands a new balancing of individual buffer pools.
- Regular review and analysis of buffer pool performance indicates inefficiencies or shows unused buffer pool resources.

Database administrators often perceive buffer pool performance in context with the design and usage of tables and indexes, and also in context with the available buffer pool memory. Database systems are often so complex that it is more or less impossible to tune them manually. Expert knowledge is often required to find a good balance among the many parameters and rules.

The task of optimizing the performance will usually take several iterations of analyzing the current state, finding better object placements and buffer pool sizes, tuning the system, and reviewing the effects.

## Predict the effect of changes

Database administrators need to predict the effect of changes to buffer pool parameters before the changes are applied to the system.

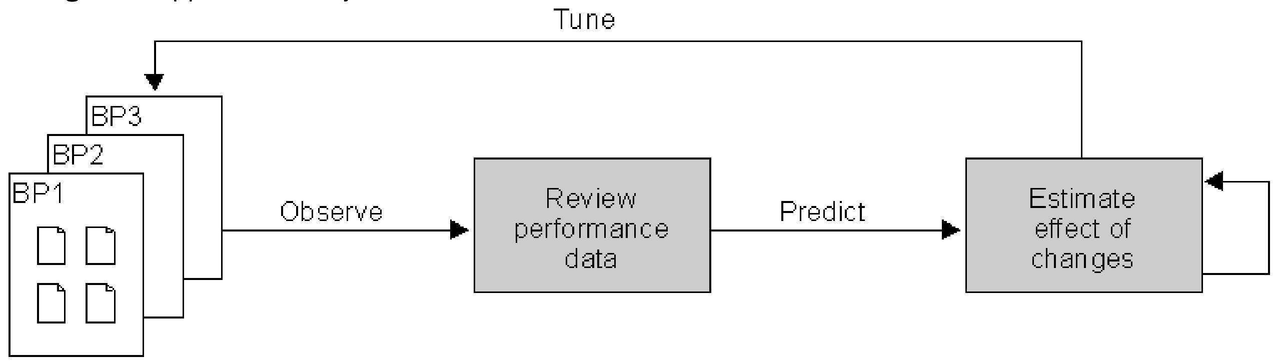


*Figure 6. Simulating planned changes based on actual performance*

Database administrators often want to evaluate tuning alternatives. Therefore, they need to estimate the effects of alternative object placements and buffer pool sizes and parameters to find an acceptable correlation between buffer pool effectiveness (in terms of hit ratios) and buffer pool size (cost). Most important, they need to be able to estimate the effect of changes to buffer pool parameters before these changes are applied to a system.

Considering the effect of different buffer pool sizes and parameters is an iterative and time-consuming task, especially under restricting conditions such as limited total buffer pool size and fixed object to buffer pool assignments. Expert knowledge is required to obtain reliable predictions and to minimize the number of tuning iterations.

Database administrators need to be able to verify their changes by comparing expected with actual performance data after they have tuned a system.

## Iterations of analysis and tuning

Analyzing the performance of buffer pools and tuning a Db2 system for optimum performance is a process that can serve several purposes and that involves one or more tasks dependent on your requirements and motivations.

Different tasks take varying amounts of time, might require different levels of knowledge, and require different level of information or support. This topic outlines some typical database administrator goals and tasks. It is basically about *why*, *what*, and *when* something should be done.

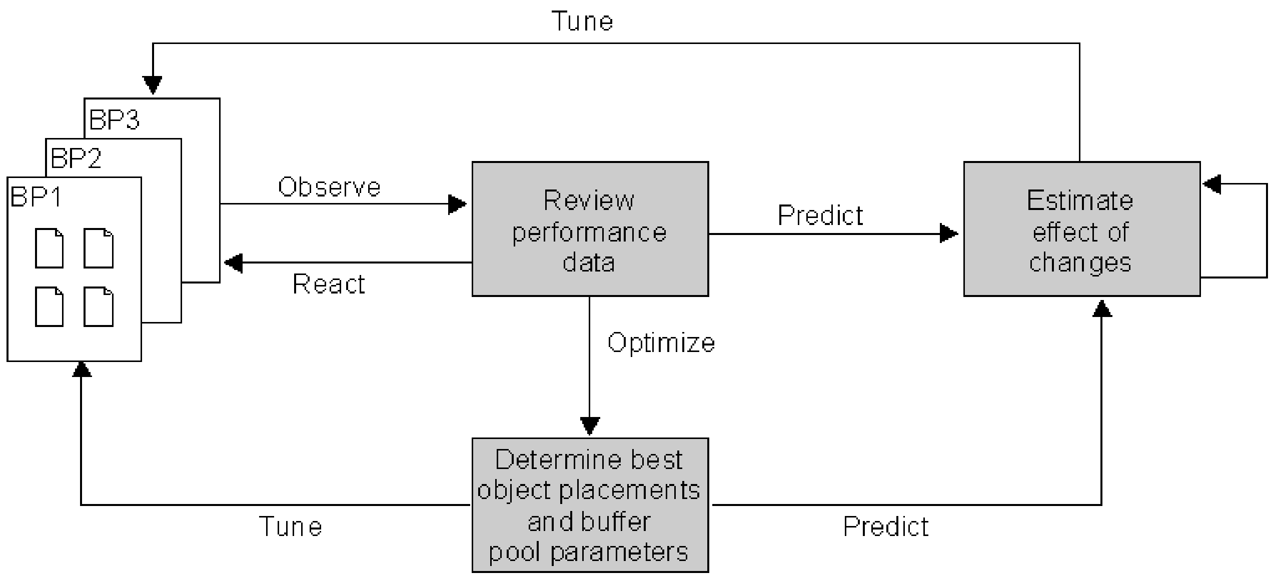

Tune
BP3
BP2
BP1
Observe
React
Review performance data
Predict
Estimate effect of changes
Optimize
Determine best object placements and buffer pool parameters
Tune
Predict

*Figure 7. Observation, optimization, simulation – an iterative process*

As described, buffer pool analysis and tuning is an iterative process that involves assessing the actual state, recognizing problems, identifying solutions in the form of changes, and applying changes to a system by means of Db2 ALTER BUFFERPOOL commands and SQL ALTER statements. The process is repeated by verifying the success of changes until further improvements are not visible or no longer economical. Buffer Pool Analyzer supports all the tasks described previously. The solutions it provides are described in the following topic.

Analysis and tuning can serve different purposes and can be performed at different levels. The previous figure outlines how you can combine several tasks for successful tuning over a longer period. You should have a work plan that clearly records your goals, the current state of buffer pool performance, the tasks you have performed, and the tuning actions you have applied. Consider also the following tips:

- Review and analysis of the actual performance should always be the first step. It should also be repeated as the last step to verify the success of tuning.
- Finding the optimal buffer pool usage and size can directly lead to a tuning action, or can be followed by an intermediate step to estimate the probable effects of changes.

## The functions of Buffer Pool Analyzer

This topic introduces the suite of Buffer Pool Analyzer tools that are available for z/OS, respectively the equivalent tools of Db2 Performance Expert for z/OS. This topic is relevant only for users of the Buffer Pool Analyzer stand-alone product or Db2 Performance Expert for z/OS.

- Before Buffer Pool Analyzer can provide any useful information about buffer pool activities and its performance, it needs to *collect data* from a Db2 subsystem. Buffer Pool Analyzer lets you collect buffer pool related Db2 trace data on the host and makes this data available in data sets on the host. Buffer Pool Analyzer uses this trace data for performance reports, object placement optimization, and simulation tasks.
- Buffer Pool Analyzer can *create activity reports* at different levels from collected trace data. Reports are provided in textual form in data sets on the host. You can use them to review the buffer pool performance or to perform an in-depth analysis of the buffer pool behavior.
- Buffer Pool Analyzer can *create aggregated buffer pool data files*, referred to as *bpd files*, from collected trace data. These bpd files contain data in a format that is suitable for most client-based functions of Buffer Pool Analyzer. The contents of bpd files can also be loaded into Db2 tables.
- In addition to the host-based text reports, you can *view performance data* on a client in graphical form. This function shows buffer pool comparison data and individual buffer pool information.

- Buffer Pool Analyzer can *optimize the object placement* in buffer pools. It analyzes bpd file data on the client and generates recommendations for the assignment of objects to buffer pools and for initial buffer pool sizes and parameters. This function provides a wizard that guides you through the optimization. You can manipulate the predefined expert rules and the parameters of an optimization. The function generates appropriate Db2 ALTER BUFFERPOOL commands and SQL ALTER statements. You can use these results to tune a system directly.
- Buffer Pool Analyzer can perform a *simulation* of the effects of different object placements and buffer pool sizes. It uses representative trace data on a client, lets you vary the parameters, and predicts the effectiveness of buffer pools in terms of hit ratios. You can use this function to assess the effects of planned changes before you actually apply them to a system.

The following topics provide the basic knowledge about each function and should enable you to perform the *how-to* instructions that are described in the remaining topics.

## Buffer Pool Analyzer functions and components

Buffer Pool Analyzer consists of several functions and components. They are identical for the Buffer Pool Analyzer stand-alone product and the integrated Buffer Pool Analysis functions of Db2 Performance Expert.

- Db2 performance data is collected on a z/OS or OS/390® host. ISPF and the Collect Report Data (CRD) function, or the batch JCL, is used to configure and control a collect task.
- Reports are created from collected data through batch JCL and the **`BPACTIVITY`** (Buffer Pool Activity) command, which provides options to customize reports for different needs. The same function is used for conversion of performance data.
- The client-based Buffer Pool Analyzer functions are combined in a Windows-based application. Through its graphical user interface you can view reports, use the optimization and simulation functions, perform long-term analyses, and get access to previously generated results.

## Collecting data

Db2 performance data is made accessible through the Db2 Instrumentation Facility as Db2 trace data. Db2 tools and products, including Buffer Pool Analyzer, can get access to this trace data through the Instrumentation Facility Interface (IFI), and the Db2 command START TRACE can be used to record trace data.

A few hundred different types of trace records exist for different purposes; each is identified by an Instrumentation Facility Component ID (IFCID). Buffer Pool Analyzer collects only buffer pool related IFCIDs.

Different data collection options result in a varying performance overhead to a Db2 subsystem. Further, not all Buffer Pool Analyzer functions require the same set of trace data. Therefore, when you use Buffer Pool Analyzer to collect trace data, you can specify *what* trace data you want Buffer Pool Analyzer to collect, *when* to collect it, and *how* to collect it. This keeps any overhead minimized for a given task.

Trace data is collected on the host. You can use ISPF and the Collect Report Data (CRD) function of Buffer Pool Analyzer, or you can use a batch job to collect data. The CRD function provides a menu-driven interface to interactively configure and control this task. In a batch job you can use JCL to prepare and configure this task. Both methods provide the same results. The CRD function provides instant feedback about the collection process and status, but requires that you are logged on to TSO/E. The batch JCL lets you prepare the job offline and run it unattended, but requires that you verify the success of the job. The CRD function might be more appropriate for shorter, single tasks. Batch jobs are better for longer, repetitive, or scheduled tasks.

When a task is started by one or the other method, the Db2 command START TRACE is used to actually collect the data. However, you do not need to care about the trace command and its parameters. The command is performed invisible.

## Determining what to collect

By making certain specifications, you can determine what data should be collected.

When you collect data with ISPF, or prepare a batch job to collect data, you can determine what to collect by specifying:

- A record format, which determines whether `Standard` or `Short` header information from each IFCID record is collected.

  `Standard` includes all IFCID record header information, which allows you to create more sophisticated reports from the collected data (inclusion of associated information, better aggregation and presentation, and better sorting).

  `Short` includes only part of the IFCID record header information, which minimizes the amount of collected data and is appropriate when collecting large amounts of data.
- A data type, which determines whether `Summary` or `Detail` data is collected.

  The data type affects the content. `Summary` and `Detail` are the base for the corresponding summary and detail reports.

  Technically, `Summary` collects buffer pool statistics (IFCID 2), data set statistics (IFCID 199), and buffer pool characteristics (IFCID 202) data. `Detail` additionally collects buffer pool activity data (IFCIDs 6, 7, 8, 9, 10, and 198). Note that especially IFCID 198 can cause noticeable overhead to a system during the collection of trace data. (It records the page requests `Getpage`, `Set write intend`, and `Release page` being sent to the Db2 Buffer Manager.)

  Beginning with Buffer Pool Analyzer Version 2, the group buffer pool related IFCIDs 230, 251, and 254 are collected in addition to a subsystem's buffer pool related IFCIDs. If the Db2 subsystem from which performance data is collected is a member of a data sharing group, summary reports contain several additional topics with group buffer pool specific performance information. The collection of group buffer pool specific trace data and its inclusion in activity reports is performed automatically and remains hidden to you. Chapter 5, "Interpreting activity reports," on page 47 describes also the group buffer pool specific details, including the IFCIDs from which this data is derived.

  Besides the technical aspect of what is collected, `Summary` and `Detail` data require further distinction regarding dynamic availability of current data. Both types of data are provided and recorded by Db2. Detail data is recorded by Db2 at the time an activity occurs. This means that the activity counts of the associated IFCIDs are current. However, summary data is recorded by Db2 at so-called statistics intervals. The interval value is a DB2 subsystem parameter, with a default setting of 1 minute (or the value specified as STATIME in DSNZPARM). This means that statistics records are to be written at the end of this interval. Beginning with DB2 10, the STATIME subsystem parameter applies only to IFCIDs 0105, 0106, 0199, and 0365. IFCIDs 0001, 0002, 0202, 0217, 0225, and 0230 are no longer controlled by STATIME, and the corresponding trace records are written at fixed, one-minute intervals. In addition, another Db2 subsystem parameter (specified as SYNCVAL in DSNZPARM) can be set to determine whether the recording and update is synchronized with some part of the hour, for example, 15, 30, 45 minutes past the hour (no synchronization is the default). The consequence for collecting summary data is that you need to consider also for how long you collect data. As a rule of thumb, assuming that you do not know the STATIME and SYNCVAL parameter settings, the time should span two default statistics intervals. Usually, one hour is a reliable choice to obtain meaningful summary reports from collected data.
- A continuity, which determines for how long trace data is collected and whether it is collected continuously or in regular intervals (for example, every 30 minutes for 40 seconds). The basic rules are:
  - Continuous collection of data simplifies matters and is recommended when the overhead to a system is negligible (for example, when you collect summary data).
  - Collection in regular intervals is recommended to minimize the overhead to a system or to minimize the amount of data being collected (for example, when you collect `Detail` data on a heavily used system).

Your specifications for record format, data type, and continuity are highly dependent on the intended usage of collected data, as outlined in Table 2 on page 15 and the following topics.

*Table 2. Intended usage of collected data and recommended specifications of record format, data type, and continuity*

| **Intended usage** | **Record format** | **Data type** | **Continuity** |
|---|---|---|---|
| On the host:<br>• To create summary reports | `Short` or `Standard` | `Summary` | Continuously to analyze activity during a certain time (for example, between 10:00 a.m. and 12:00 a.m.), or in intervals to analyze the performance of longer periods (for example, every 60 minutes for 60 seconds for all day long). |
| On the host:<br>• To create detail reports | `Short` or `Standard`<br>If sophisticated reports are not required, use `Short` for overhead reasons. | `Detail` | As required, but should be limited to reduce overhead (IFCID 198). |
| On the host:<br>• To create buffer pool data files for use on the client | Format, type, and continuity is determined by the client-based functions that require bpd files as input. See further entries. | | |
| On the client:<br>• To view performance data<br>This function uses bpd files as input. | `Short` is sufficient. | `Detail` | Continuously, or in intervals to analyze the performance of longer periods. |
| On the client:<br>• To optimize object placements and buffer pool sizes<br>This function uses bpd files as input. | `Short` is sufficient. | `Detail` is recommended. | Continuously or in intervals, depending on the goal of the optimization. |
| On the client:<br>• To perform simulations<br>This function uses raw trace data as input. | `Short` is required.<br>Simulation does not use extended IFCID header information. Further, this minimizes the amount of collected data and the system overhead. | `Detail` | Continuous collection is required, for approximately 20 minutes (subject to system load and the amount of collected data). |

*Table 2. Intended usage of collected data and recommended specifications of record format, data type, and continuity (continued)*

| Intended usage | Record format | Data type | Continuity |
|---|---|---|---|
| On the client:<br>• To analyze the long-term performance of buffer pools<br>This function uses bpd files as input. | `Short` is sufficient. | `Detail` is recommended. | As available, because this function usually uses existing bpd files as input. |

## Determining when and how long to collect

When you use ISPF or a batch job to collect data, you can specify whether data should be collected immediately or at a specific time, and you can specify whether a collect task should stop after an elapsed time or after a number of records are collected.

The decision when to start and stop a collect task depends on the system load, the intended usage of the data, and the purpose of the analysis. The following scenarios provide some ideas:

- To collect summary information (data type `Summary`) of the buffer pool performance over a day, you can start collecting data at midnight (start time) and run the job for 24 hours (elapsed time), but only take samples of 10 seconds every 60 minutes (continuity). You can also combine data from multiple collect tasks and use the accumulated data as input when you create reports. This might be helpful to detect trends over a longer period or regular peaks.

  Note that Db2 updates data that is used for summary information at so-called *statistics intervals*. Also, most statistics counters are incremental counters. To report a valid counter value for a given period, the difference between the latest and earliest counter value is computed. This means that a collect task should cover at least a statistics interval to produce meaningful summary information. The sampling duration, here 10 seconds, is long enough to capture any updates made by Db2.
- To collect detailed information (data type `Detail`) during peak times in your organization's business, which might be around 11:00 a.m. and 3:00 p.m., you can start two collect tasks through ISPF, one at 10:30 and one at 2:30 p.m. (start times). Each task can collect data for an hour (elapsed time) and take samples every 10 minutes for five seconds (continuity).

  Note that detail information in activity reports is based on actual counts of events, opposed to summary information. You can keep a collect task as short as required without loosing accurateness. More details are described in "Preliminary remarks about the accuracy of summary and detail reports" on page 48.
- If you must analyze the cause of a current performance problem, you can start a collect task immediately (start time) and run it for 45 minutes (elapsed time) to collect summary data. This step might be followed by a second step that collects detailed data for a shorter time.
- If you need to collect data to perform an optimization or simulation (both are described in detail in the following topics), for example to tune a subsystem or peak load periods, you need to identify the peak load periods and collect a representative mean of trace data from this time.
- The collection of data can also serve multiple purposes. For example, if you need to perform an optimization of object placements *and* a simulation (before you apply any changes to a system), you collect data continuously for approximately 20 minutes during a defined system load period. Simulation requires the record format `Short`, the data type `Detail`, and a continuous collection of data, but these specifications serve as well the requirements for an optimization.

The important point is that the collection of data (what, when, and how) must always be performed with regard to the intended usage of this data. Buffer Pool Analyzer can analyze and report performance data, but inadequate selection of trace data might lead to wrong conclusions. For successful tuning, you should understand that the results of an analysis are always based on workload at the time trace data was collected. Other workloads, which run at other times, may have different results. If you plan to optimize the buffer pool usage, you should carefully determine which workload you consider representative for your optimization.

### Using the collected data

The collected data can be used in different ways.

When trace data is actually collected, it is written to an output data set. The name of the data set is specified by you, either during an interactive ISPF dialog or in a batch job. Nevertheless, some conventions are recommended in “File and data set naming conventions” on page 18.

Briefly, the trace data in an output data set can be used as follows. More details are described in the following topics.

- The data can directly be used to create summary or detail activity reports on the host.
- To view the data in graphical form on the client, a bpd file must be created from the trace data and downloaded to the client.
- To perform optimizations of object placements and buffer pool sizes on the client, a bpd file must be created from the trace data and downloaded to the client.
- To perform simulations on the client, the trace data must be downloaded to the client. The creation of a bpd file is not required because Buffer Pool Analyzer uses the raw, binary data for this purpose.
- To perform object placements *and* simulations, the trace data *and* the bpd file must be downloaded.
- To perform long-term analyses of performance data, one or more historical or current bpd files must be available on the client.

For the matter of completeness, note that data from several collect tasks can also be accumulated in a single data set, which then can be used to create reports and bpd files. Also, multiple input data sets can be used as combined input to create reports and bpd files. However, these uses are subject to some restrictions, which are described at the appropriate places in this information.

Finally, note that the functions that create activity reports and bpd files provide options to select subsets from collected and accumulated data to limit the scope of the output in reports or bpd files. All details are described in Chapter 4, “Creating activity reports and bpd files,” on page 37.

## Creating activity reports

Buffer Pool Analyzer provides the **BPACTIVITY** command (for "Buffer Pool Activity") and its subcommand **REPORT** to create activity reports from collected trace data.

The command is used in batch jobs, together with data definition (DD) statements that specify, among others, one or more input data sets with collected trace data. The command creates one or more reports in table form and stores them in a sequential data set. The reports can be viewed or printed by means of appropriate system utilities.

Command options are available to specify the type of report (summary or detail), to selectively use the input data for reports (for example, to limit the time frame or to filter out uninteresting information), and to specify the aggregation and sorting of the reported data.

The reports provide you with comprehensive information about how the buffer pools and the objects are used, for example:

- System and application hit ratios, buffer pool activity counts, I/O activity counts
- Information sorted according to different identifiers, such as buffer pool, plan name, object, or primary authorization ID
- Data ranked by the type of buffer pool operation, for example, by the number of Getpage requests, Sequential prefetches, or Synchronous reads
- Changeable thresholds to show only the most active objects in reports
- Information filtered to include, or to exclude, only specific buffer pools, plans, or time frames

- If the trace data in the input data sets is from a member of a data sharing group, where multiple Db2 subsystems share a group buffer pool, activity reports additionally contain detailed performance information about the group buffer pool.

Chapter 4, “Creating activity reports and bpd files,” on page 37 describes the details of using the BPACTIVITY command and its options in batch jobs to create summary and detail reports.

Chapter 5, “Interpreting activity reports,” on page 47 shows examples of summary reports and detail reports and explains how to interpret them.

## Creating buffer pool data (bpd) files

Buffer Pool Analyzer provides the **BPACTIVITY** command (for "Buffer Pool Activity") and its subcommand **FILE** to create buffer pool data (bpd) files from collected trace data.

The command is used in batch jobs, together with DD statements that specify, among others, one or more input data sets with collected trace data. It creates bpd files that contain the data in a format that is required by all the client-based functions of Buffer Pool Analyzer, except simulation.

Command options are available to specify the type of data to be included (`Summary` or `Detail`) and to selectively use the input data for bpd files (for example, to include only data about specific identifiers).

The **FILE** subcommand is similar to the **REPORT** subcommand. Both use the same type of input data (Db2 trace data). Both can extract `Summary` or `Detail` data from the input and selectively use input data. However, only the **REPORT** subcommand has options to manipulate the aggregation and sorting of data. These options are not needed for bpd files. The **FILE** subcommand has an option to exclude data from inactive objects, which creates smaller, better manageable bpd files.

Chapter 4, “Creating activity reports and bpd files,” on page 37 describes the details of using the **BPACTIVITY** command and its options in batch jobs to create bpd files.

The data in bpd files is not intended for direct interpretation. It is in Db2 load format and can be loaded into Db2 tables for additional analysis by SQL queries. Chapter 13, “Loading a bpd file into a Db2 table,” on page 155 provides some information how the contents of bpd files can be loaded into Db2 tables. However, the further use is outside the scope of this information.

## File and data set naming conventions

Adhering to the following naming conventions eases data set and file handling on the host and client, and you do not need to rename files after they are downloaded to the client.

The Buffer Pool Analyzer functions introduced so far are performed on the host; the functions introduced in the following topics are performed on the client. To summarize, the input data required by the client-based functions is either in the format of bpd files (for all functions except simulation) or raw trace data (for simulation).

On the client these files require the file name extension bpd for buffer pool data files, respectively `trace` for raw trace data (and `terse`, if the trace data is compressed). You can apply these file name extensions already on the host when you collect data or create a bpd file. Use TRACE as low-level qualifier in the name of the output data set that holds collected data (and TRACE.TERSE as low-level qualifiers if the trace data is compressed). Use BPD as low-level qualifier when you create a bpd file from the trace data (the `trace` data set).

## Viewing performance data on the client

This function lets you view buffer pool performance data in graphical form and as diagrams on the client. It uses buffer pool data (bpd) files that are created on the host system. The bpd files need to be downloaded to the client before they can be viewed.

The client-based graphical user interface (GUI) of Buffer Pool Analyzer provides a convenient environment to select a bpd file and to view system and buffer pool information. For example, you can view:

- General system information, such as:

  - Db2 location, group, and member information
  - Start and end timestamps of trace data contained in a bpd file
  - System and application hit ratios, and the number of accessed buffer pools and objects
  - Counter information, such as Getpage, Read request, Write page
- Buffer pool comparison data, such as a comparison by Read request or Write request
- Individual buffer pool characteristics and counters

Chapter 6, “Viewing performance data on the client,” on page 89 describes how to use this function and shows examples of how the information is presented as diagrams, pie charts, and graphs.

## Optimizing object placements and buffer pool sizes

This function determines the optimal placements of table spaces and index spaces in buffer pools, the optimal buffer pool sizes, and the optimal values for some buffer pool thresholds. It uses predefined and modifiable expert rules and the object's access behavior to determine the optimum. The function is performed on the client and uses buffer pool performance data (from a bpd file) as input. These bpd files need to be downloaded to the client before they can be used.

This function analyzes the data, finds the optimum placements, sizes, and thresholds, and generates recommendations as ready-to-use SQL statements and Db2 commands:

- The SQL ALTER statements contain parameters that determine which object (table space or index spaces) should be assigned to which buffer pool.
- The Db2 ALTER BUFFERPOOL commands contain parameters that determine the recommended size and thresholds of each buffer pool.

The trace data must be a representative snapshot of the buffer pool performance, as described in “Determining when and how long to collect” on page 16. This function uses the objects' access behavior to determine the optimal object placements, and it uses many factors to determine the buffer pool sizes based on the placements. Approximations are used by the algorithms for data that cannot be retrieved from the input data.

You use the object placement tool on the client to select an appropriate bpd file and perform the optimization. The GUI provides an easy-to-use wizard that guides you through a few steps to determine the optimal object placements and buffer pool sizes. The wizard uses defaults based on information from the bpd file, but you can influence the optimization in several ways:

- Buffer Pool Analyzer determines the available memory for buffer pools from the bpd file and uses this value as the default for the optimization.

  You can adjust the total buffer pool size, if you want this function to use a different size.
- Buffer Pool Analyzer uses one of several predefined *pattern files* to determine the object placements. Pattern files contain expert rules that define which objects should be placed in which buffer pool according to each object's characteristics. The rules define criteria that must be met to assign an object to a buffer pool. The sequence of rules defines in which order the rules are applied to the objects.

  Buffer Pool Analyzer preselects a pattern file based on the total buffer pool size of the Db2 subsystem.

  You can choose a different pattern file. You can also edit a pattern file to adjust the object placement rules according to your needs. Modified pattern files can be saved and will automatically be preselected whenever a bpd file from the same subsystem is opened.

  **Note:** You can save a pattern file only if the file has at least one rule.
- Buffer Pool Analyzer calculates the optimized assignments of objects to buffer pools, based on the available memory for buffer pools and the placement rules, and it calculates the optimum size of each buffer pool.

  You can adjust the assignments and the sizes of individual buffer pools, if required.

- When Buffer Pool Analyzer has generated its recommendations for object placements and buffer pool sizes, you can adjust them according to specific needs. Your adjustments are reflected in the generated SQL ALTER statements and Db2 ALTER BUFFERPOOL commands.

The results from optimizations are lists of SQL ALTER statements and Db2 ALTER BUFFERPOOL commands that have their parameters set to the recommended values. Your adjustments and changes to an optimization are reflected in the results.

Results from optimizations are kept on the client. You can select them from the Buffer Pool Analyzer main window and view them in a web browser to assess them. To apply the statements and commands to a subsystem, you must upload them to the host and run them as usual. Right-click an object placement result to directly start a simulation using this placement.

You can also work with different performance scenarios by using different bpd files. You can compare the results and assess the variations on the client before you apply the recommendations to a Db2 subsystem.

Chapter 7, "Optimizing object placements and initial buffer pool sizes," on page 97 describes how to use this function and explains how to work with object placement rules in pattern files.

## Simulating buffer pool behavior

This function simulates different object placements and buffer pool size ranges in a representative performance snapshot and lets you see and interpret the effects. You can use it to perform what-if scenarios to balance buffer pool sizes and performance. It is performed on the client and uses buffer pool performance data (as raw Db2 trace data) as input. The file containing the trace data (in `short` format) needs to be downloaded to the client before it can be used.

If you use this function to test the recommendations from the object placement tool, the trace data file downloaded should be ideally the file from which the bpd file for object placement was created. To directly start the simulation function, right click the object placement result, then select **Simulate...**.

The trace data that is used for simulations must be a good representation of the activity of the buffer pools that you want to optimize, as described in "Determining when and how long to collect" on page 16. This function uses recorded detailed activity of each object to accurately determine the effects of changes to the system.

You use the simulation tool on the client to select an appropriate trace data file and to perform simulations. The GUI provides an easy-to-use wizard that guides you through the few steps to adjust the simulation parameters and to assign objects to buffer pools. The wizard uses defaults that are based on information from the trace data file, but you can adjust these defaults interactively to see the effects of changes.

- You can vary simulation parameters, such as:
  - The buffer pools to be included in a simulation.
  - The minimum and maximum buffer pool sizes to be simulated, and the increments by which the sizes are varied during a simulation.
  - The sequential steal thresholds to be simulated.
- You can vary the assignment of Db2 table spaces and index spaces to buffer pools

Results from simulations are kept on the client. You can select them from the Buffer Pool Analyzer main window and view them in a web browser. The results show the recommended distribution of memory between buffer pools, and detailed information about misses and hit ratios as functions of buffer pool size.

You can also work with different performance scenarios by using different trace data files, and you can apply different simulation parameters to these simulations. By comparing the results, your predictions about the effects of changes become more reliable.

Chapter 8, "Simulating buffer pool behavior," on page 111 describes how to perform simulations and shows examples of simulation results.

### Analyzing long-term buffer pool performance

This function analyzes performance data from several bpd files according to your needs. The function is performed on the client, which means that the bpd files need to be downloaded before they can be used as input for the long-term analysis.

The client-based graphical user interface (GUI) provides the environment to select the bpd files to use, to specify the type of analysis to be performed, to specify counters and buffer pool objects of interest, and to view and save the result of an analysis.

Long-term analysis can mean anything, but ready-made results might not fit your needs and interests. The long-term analysis function of Buffer Pool Analyzer provides a flexible way of specifying your needs and provides instant results. The following procedure introduces the functionality and your interaction with this function:

- You select the bpd files to be included in the analysis. Any number of bpd files existing on the client can be selected. You do not need to care from which subsystem they were created.
- The long-term analysis function identifies the subsystems from which these bpd files were created. If multiple subsystems are involved, you need to choose one (which causes the function to ignore all bpd files from the other subsystems).
- You select the *type of analysis* to be performed, which can be, for example, a weekly view by day, a daily view by hour, a view of a period of time, bar charts, or pie charts. Every type has its strengths and is instantly explained when you use this function.
- You select the counters and buffer pool objects you are interested in and the relationships between them. Counters can be single counters, groups of counters, even ratios. Objects can be all objects, single objects, some or all in a buffer pool, a mixture from several pools, and more. You can also restrict the time frame to consider for the analysis. The long-term analysis function takes care that only those data from the bpd files is used for the analysis that matches your selections.
- The long-term analysis function shows the result instantly in the main window. The result is kept on the client for future use and comparison with other results.

Chapter 9, "Analyzing long-term buffer pool performance," on page 121 describes how to use this function and also shows and explains several examples of the various analysis types.

## Summary of user tasks

This topic illustrates user tasks and their relationships.

Figure 8 on page 22 illustrates and summarizes the previously introduced user tasks and their relationships. It shows the order of tasks that must be performed to create reports on the host or to perform one or more of the client-based functions. For example, to perform a simulation of buffer pool behavior based on actual trace data, first you need to collect data in a trace data file and download this file to the client. Note that the descriptive text of the tasks complies with the topic titles in this information.

The illustration also shows the flow of data between the components. Collected trace data can be used on the host to create reports, or downloaded and used for simulations. Just as well, collected data can be aggregated in bpd files and loaded into Db2 tables, or downloaded to the client and viewed or used for an optimization.

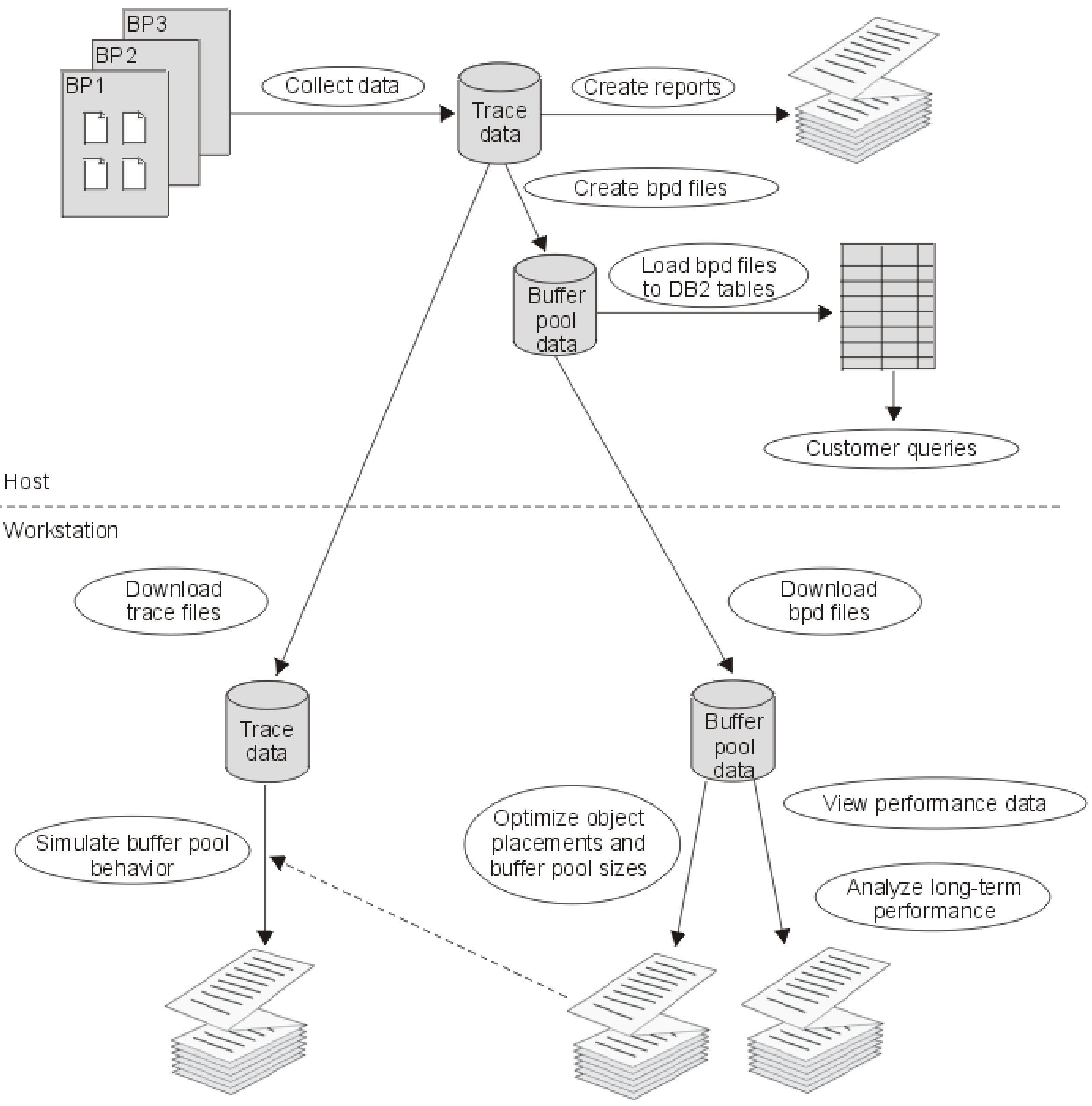


*Figure 8. Summary of user tasks*

Note that Buffer Pool Analyzer provides sample performance data as bpd files and trace data files on the client. You can use them to acquaint yourself with Buffer Pool Analyzer. The individual topics describe the locations of these files and how to select, open, and use them.

## Where to start

This topic shows possible workflows and the needed steps to perform them.

So far, the introduction explained the buffer pool analysis and tuning tasks and what Buffer Pool Analyzer offers to solve these tasks. Table 3 on page 23 summarizes possible workflows that guide you from a potential buffer pool related task through a sequence of activities to achieve your aim. For a more problem-oriented approach see also “A generalized approach to performing analyses” on page 24.

*Table 3. Possible workflows for Buffer Pool Analyzer*

| If you want to | Perform the following steps |
|---|---|
| Create several activity reports on the host. | 1. Collect performance data (or use data from a previous data collection). Refer to Chapter 3, “Collecting data,” on page 27 and also to “Determining what to collect” on page 14. Decide about the type of report (summary or detail).<br>2. Write a batch job that creates a report. See Chapter 4, “Creating activity reports and bpd files,” on page 37. Use the trace data file as input. Note that you can use the **BPACTIVITY REPORT** in the batch job without any options.<br>3. Study the activity report in detail.<br>4. Modify your batch job and use some of the **BPACTIVITY** options. Compare the effects in the reports.<br>5. Use different input data (created with different collection parameters) and different **BPACTIVITY** options in your batch job and see how this effects the reports.<br><br>You will notice how data collection parameters (format, type, continuity, duration) relate to **BPACTIVITY** command options; above all, that data can only be reported if it was previously collected. |
| Learn about what kind of buffer pool information can be viewed on the client. | 1. Start reporting and select and open one of the sample bpd files. See Chapter 6, “Viewing performance data on the client,” on page 89.<br>2. Expand the tree in the **Reporting** folder and study the contents in the different subfolders.<br>3. View how comparison data is presented in graphical forms. |
| Get a first glance at the object placement or simulation capabilities on the client. | 1. Select and open a sample file. See Chapter 7, “Optimizing object placements and initial buffer pool sizes,” on page 97, respectively Chapter 8, “Simulating buffer pool behavior,” on page 111.<br>You will notice when bpd files and trace data files are used.<br>2. Start and follow the wizard. Accept the default input values.<br>3. Note the purpose of each step, and the output from these functions. |

*Table 3. Possible workflows for Buffer Pool Analyzer (continued)*

| If you want to | Perform the following steps |
|---|---|
| Perform a simulation on the client with real performance data. | 1. Collect performance data through ISPF. Choose `Short` format, `Detail` data type, and collect data for approximately 10 minutes. See Chapter 3, "Collecting data," on page 27 and also to "Determining what to collect" on page 14.<br>2. Note the options you have chosen in specifying data collection parameters. Note how many trace records were collected during the specified time.<br>3. Download the trace data file to the client. See Chapter 11, "Downloading files from the host to the client," on page 151.<br>4. Start the simulation function on the client and select and open the trace data file you have created.<br>5. Optional: To directly simulate the results of an object placement, right-click the object placement result file, then select **Simulate...**<br>6. Repeat the simulations with different simulation parameters, object placements or both. Note how the parameters influence the execution times of simulations.<br>7. Study the simulation results in more detail. |
| View performance data or perform an optimization of object placements and buffer pool sizings on the client with real performance data. | 1. Collect performance data (or use data from a previous data collection). Choose `Short` format, `Detail` data type, and collect data for a few minutes. See Chapter 3, "Collecting data," on page 27 and "Determining what to collect" on page 14.<br>2. Write a batch job that creates a bpd file. See Chapter 4, "Creating activity reports and bpd files," on page 37. Use the trace data file as input. Note that you can use the **`BPACTIVITY FILE`** command in the batch job without any options for default behavior.<br>3. Download the bpd file to the client. See Chapter 11, "Downloading files from the host to the client," on page 151.<br>4. Start the view function or object placement function on the client and select and open the bpd file you have created.<br>5. Study the optimization results in more detail. Note the recommendations. You might want to compare them with your actual system settings.<br><br>Consider creating a trace data file *and* a bpd file from every data collection, and keeping them together on the client. This lets you iteratively perform optimizations and simulations on the same snapshot of performance data. |
| Study a detailed use case. | Read Chapter 10, "Example of a use case," on page 137. |

## A generalized approach to performing analyses

This topic outlines a problem-oriented approach of how to analyze buffer pool performance and related problems.

The proposed approach is a combination of using the suite of Buffer Pool Analyzer tools in reasonable sequence and applying reasoning and experience to a well-defined task. As each system setup and behavior is different, the following approach is one possible example. It is not meant as a step-by-step instruction.

- On the host, use a trace data file that contains data from a representative time and create a summary activity report, a few detail activity reports (for example, a TOP(25) report, sorted by Getpage, Readreq, or Readpage request), and a bpd file. See Chapter 4, "Creating activity reports and bpd files," on page 37 for details.
- Inspect the summary activity report to determine how the system is set up (the number and the sizes of buffer pools, the threshold values for certain buffer pool operations, and others). See "The Buffer Pool Characteristics section" on page 50 for details.

  Also pay special attention to the "Buffer Pool Statistics Highlights" section of the summary report. It highlights certain counter values with an asterisk (*). For example, nonzero values for most "threshold reached" values will be flagged. It is still up to you to decide whether these values are acceptable for that system setup and workload. See "The Buffer Pool Statistics Highlights section" on page 50 for details.
- Download the bpd file to the client and load it into the graphical report utility. See Chapter 6, "Viewing performance data on the client," on page 89 for details. This function is excellent when used in parallel to the host reports, to get a feeling for the distribution of work and the type of access behavior of each buffer pool.

  For example, when you navigate to **Buffer Pools —► Buffer Pool Comparison —► Getpages**, the buffer pools with the most Getpage operations are immediately identified. All other buffer pools can usually be ignored until a simulation is performed later. The other graphical buffer pool comparisons, by ReadRequest, ReadPage, WriteRequest, and Writepage operations, are also important indicators for directing further analysis.
- Inspect the detail activity reports to determine whether a small number of objects are dominating the overall system or single buffer pool activity. Investigate these objects in detail. See "The Detail Activity section" on page 77 for details.

  For example:

  – `Read Request - Delay (msec) - Synchronous`

    If some of these high-activity objects have a relatively high value, compared to the average for this counter, you may be experiencing disk problems and should consider moving such objects to a faster disk.

  – `BP Hit Ratio (%) - System / Application`

    A large discrepancy among system and application hit ratios may indicate a conflict, such as a wrongly set Virtual Sequential Threshold, or the object is mainly accessed sequentially but the buffer pool also contains many objects with high synchronous access.
- Use the graphical report utility on the client to further check for "unusual" objects:

  – For each of the high-access buffer pools navigate to **'BPx —► Object Comparison**. The graphical representation makes it immediately apparent if some high-activity objects do not "fit" in this buffer pool, as their access type is not typical of the rest of the buffer pool objects. A reorganization of the buffer pools, using the object placement function, might be appropriate.

  – Be especially critical of high-activity objects with an unexpectedly large amount of RID-List activities. This may indicate an overdue REORG, a missing index, or an application programming style issue.
- Use the simulation function to determine if memory distribution between buffer pools is correct. See Chapter 8, "Simulating buffer pool behavior," on page 111 for details. Perform the simulation with "Minimum / Maximum buffer simulation sizes" initially set to approximately 50 percent, respectively 200 percent of the current buffer pool sizes. Initially, only check the recommended memory distribution in the "list of recommended buffer pool sizes" (the second table in the simulation results) for the value nearest to the current total buffer pool size. If these values differ significantly from the current memory distribution, you should question whether the activity data in the trace data file is typical for the workload you wish to optimize, especially regarding buffer pools that the simulation recommends to

make smaller. You may need to collect traces from other times and also perform the same simulation on them, determining a best middle value from the combined reports.

- In a following step, the simulation results should also make it immediately clear whether an increase in overall memory would result in a significant performance increase. If a recommended size for a buffer pool is the smallest or largest size that was simulated for that buffer pool, consider performing a further simulation with a larger range of values.
- Consider simulating what happens if you move any "unusual" objects found earlier into other or even new buffer pools. Compare the results with the original simulation with no object movement.
- Finally, even if it was not determined earlier that object placement is required, consider using it, followed by simulation to determine whether a different distribution of objects can also result in a performance improvement. This is a time-consuming exercise, but can produce significant performance improvements.

# Chapter 3. Collecting data

This topic describes how to collect the performance data that is used by Buffer Pool Analyzer. It describes two methods to collect buffer pool trace data. The first method uses ISPF and the Collect Report Data (CRD) function to configure and control a collect task, the second method uses a batch job that contains equivalent specifications for a collect task.

## About this task

For the sake of completeness, note that the Generalized Trace Facility (GTF) and the System Management Facility (SMF) can also collect buffer pool related trace data. The data is recorded in appropriate GTF and SMF data sets, which can be used as alternative or additional input for the creation of activity reports and bpd files. In Chapter 4, "Creating activity reports and bpd files," on page 37, the description of the INPUTDD statement provides more details about specifying alternative or multiple input data sets. However, the important point is that GTF or SMF must be set up in `SYS1.PROCLIB` to collect, besides others, also buffer pool related data (as specified in "Determining what to collect" on page 14).

Related tasks:

- Collecting data is always the first task before you can perform any of the other Buffer Pool Analyzer functions.
- After you have performed this task, the trace data is available in named data sets and can be used for the tasks described in:
  - Chapter 4, "Creating activity reports and bpd files," on page 37

    Activity reports are created on the host. Bpd files need to be downloaded to the client before they can be used for the tasks described in:

    - Chapter 6, "Viewing performance data on the client," on page 89
    - Chapter 7, "Optimizing object placements and initial buffer pool sizes," on page 97
    - Chapter 9, "Analyzing long-term buffer pool performance," on page 121
  - Chapter 8, "Simulating buffer pool behavior," on page 111

    Db2 trace data files need to be downloaded before this task can be performed on the client.

**General remarks:**

1. Ensure that your output data sets are large enough. The amount of data that is being collected depends largely on the activity in the buffer pools. If you are going to collect `detail` data, remember that each activity produces at least one trace record. On a busy system you can rapidly generate several million records. Limit the data collection time, or the number of records to be collected, until you have a feeling about the amount of trace data being produced on your system.
2. If you are going to collect data for optimizing the object placements, ensure that the Db2 catalog statistics are up to date. Among other factors, Buffer Pool Analyzer considers the size of page sets and might otherwise produce inaccurate results. Run the RUNSTATS utility, if required.
3. If you are going to collect data for simulation:
   - Ensure that you collect `detail` data, in `short` format, for approximately 20 minutes continuously, which generally gives a good representation of a particular workload. If the workload varies significantly, collect a slightly smaller trace for each workload type.
   - For large amounts of data you can optionally create an additional output data set that contains the collected data in compressed format. The size and the download time of such data sets are roughly 25 percent of the equivalent uncompressed data sets. The simulation function can handle both types. See Chapter 14, "The TRSMAIN terse utility," on page 159 for more details.

     Note that an uncompressed data set is always created. Therefore, if you choose to create the additional compressed data set, you should have approximately 1.25 times the required disk space available. However, if the data is exclusively used for simulations, you can erase the uncompressed data set after both data sets are created.

- Avoid collecting more than 2 GB of data. The simulation function on the client can handle trace data files of up to 2 GB (no matter whether the data is compressed or uncompressed). If you realize that the size of a trace data file on the client is too big, create and download a smaller file (less than 2 GB on the client), compare the actual sizes, and estimate the approximate maximum size of the host data set as follows:

```
Size_on_host_actual       Size_on_host_max
---------------------  ≈  ----------------
Size_on_client_actual           2 GB
```

  If necessary, collect a smaller trace to keep the trace data file below its maximal size.

4. If you are going to collect data for object placement *and* simulation, ensure that all requirements in remarks 2 and 3 are met. Furthermore, it is essential that you keep the trace data file and the bpd file together. (The bpd file must be created as described in Chapter 4, “Creating activity reports and bpd files,” on page 37.)

The following topics provide additional information:

- “Collecting data by using ISPF” on page 28
- “Collecting data by using the batch JCL” on page 35

## Collecting data by using ISPF

This section explains how to use the Collect Report Data (CRD) function of Buffer Pool Analyzer, respectively the equivalent function of Db2 Performance Expert.

### About this task

To collect data by using ISPF, perform the following steps:

### Procedure

1. Start ISPF from your TSO/E session.
2. Start the FPEJINIT exec.

   The exec automatically determines whether Db2 Performance Expert or the Buffer Pool Analyzer stand-alone product is installed on your system. If Db2 Performance Expert is installed on your system, the exec starts Db2 Performance Expert and you need to perform steps “3” on page 29 to “4” on page 29 to reach the Collect Report Data function. Otherwise, the exec directly starts the Collect Report Data function. If you see the Collect Report Data panel (shown in step “4” on page 29), continue with step “5” on page 30.

   IBM OMEGAMON for Db2 Performance Expert panel is displayed:

```
                IBM OMEGAMON for DB2 Performance Expert
Command ===>  ________________________________________________________________

Select one of the following.

__ 1. Create and execute reporting commands
   2. View online DB2 activity - Classic Interface
   3. View online DB2 activity - PE ISPF OLM
   4. Maintain parameter data sets
   5. Customize report and trace layouts
   6. Exception profiling










F1=Help    F2=Split    F3=Exit    F9=Swap    F12=Cancel
```

3. Select the following options to specify the Db2 subsystem from which you want to collect data, as follows:
   a) Select option 3 (View online Db2 activity - PE ISPF OLM). The Online Monitor Main Menu panel is displayed.
   b) Select option 4 (Options). The Options panel is displayed.
   c) Select option 1 (Db2 Subsystem). The Db2 Subsystem subpanel is displayed.
   d) Specify the Db2 subsystem and return to the Online Monitor Main Menu panel:

```
                      Online Monitor Main Menu
Command ===>  ________________________________________________________________

Select one of the following.

__   1. Display Thread Activity
     2. Display Statistics
     3. Display System Parameters
     4. Options
     5. Control Exception Processing
     6a. Collect Report Data - General
     6b. Collect Report Data - For Buffer Pool Analysis
     7.  Create and execute reporting commands
     8.  Maintain parameter data sets
     9.  Explain






F1=Help    F2=Split    F3=Exit    F9=Swap    F12=Cancel    F16=Look
F17=Collect
```

4. On the Online Monitor Main Menu panel select option 6b, Collect Report Data - For Buffer Pool Analysis.

   The Collect Report Data for Buffer Pool Analysis panel is displayed:

```
 06/23/13 10:16    Collect Report Data - For Buffer Pool Analysis
 Command ===>  ________________________________________________________________


PM01DC11              DC11


 Select one of the following.

 __ 1. Configure task
    2. Activate task
    3. Display task status
    4. Stop task

    Task Description                            Status
    collect_data_for_buffer_pool_activity___    Data available






 F1=Help    F2=Split    F3=Exit    F9=Swap    F12=Cancel    F16=Look
```

Above the options menu, you see the name of the Db2 subsystem that you specified in step 3d.

You use this panel to configure and control a collect task. You can work with one task at the same time. If you leave Buffer Pool Analyzer (the stand-alone product) or Db2 Performance Expert before a task has finished, the task is stopped (no configuration settings are saved, no data is available). A task has finished when it is configured, activated, and all data has been collected.

5. Select one of the following options:

   - **Configure task**

     This step is mandatory and must be performed before you can use any other options. See "Configuring a collect task" on page 31, which describes how to specify details of the collect task.

   - **Activate task**

     Use this option to activate the collect task that you have configured.

   - **Display task status**

     Use this option if you want to see status details of an activated collect task. These details are described in "Interpreting trace status summary and trace messages" on page 33.

     The overall status of a collect task is shown in the **Status** field. Depending on how a collect task was configured, one of the following status descriptors is shown:

     – `Not yet activated`: A collect task was configured but not yet activated.
     – `Activate issued`: A configured task was activated and started.
     – `Waiting for start`: A scheduled task was configured and activated, but the start time is not yet reached.
     – `Collecting data`: A task is active and is collecting data.
     – `Waiting for next interval`: A task was configured to collect data in intervals. The task is active, but waiting for the next interval to take place.
     – `Data available`: An activated task has stopped and written its data to the output data set.
     – `Stopped`: An activated task has finished, either because it was stopped by you or because a configured stop condition was reached.

   - **Stop task**

     Use this option to stop an activated collect task. If a Db2 trace is currently collecting data, this option also stops the Db2 trace.

6. In the **Task description** field, type a description for the collect task.

## Configuring a collect task

This section explains how to specify details of a collect task, such as the type of data to collect, and the start and stop conditions.

### About this task

If you have selected the **Configure task** option on the Collect Report Data panel, the Collect Report Data for Buffer Pool Analysis panel is displayed:

```
 06/23/13 14:16    Collect Report Data for Buffer Pool Analysis

                               Trace Configuration

   Command ===> ______________________________________________________________

   Task description . . . . : Collect data for buffer pool activity

   Output DS name . . . . . : 'NKA.COLLECT.TRACE'
   Disposition  . . . . . . : 3  1=Append
                                 2=Overwrite
                                 3=New

   Record format  . . . . . : 1  1=Standard
                                 2=Short

   Data type  . . . . . . . : 2  1=Detail
                                 2=Summary
                                 3=Catalog only

   OP buffer size . . . . . : 512   kB

   Start the DB2 trace  . . : 1  1=Immediately
                                 2=At 15 : 14 : 0  (hh:mm:ss)

   Trace and collect data . : 1  1=Continuously
                                 2=Every 5    minutes for 60   seconds

   Stop the DB2 traces when any of the following conditions occur:
   (Select at least one condition.)
   >  Elapsed time  . . . . . . . : 0       seconds
   >  Number of records collected : 0

    F1=Help     F2=Split    F3=Exit     F7=Up       F8=Down     F9=Swap
   F12=Cancel
```

The **Task description** field shows the description that you have entered in the previous step.

To configure a collect task, perform the following steps:

### Procedure

1. In the field **Output DS name** specify the data set name to which the data is to be written.

   Data sets used for this purpose should have a variable record format (RECFM) and a record length (LRECL) of at least 6000. The data set name should have the low-level qualifier TRACE, for example `NKA.COLLECT.TRACE`, for the following reasons:

   - If this data set contains trace data that will be used for simulations on the client, the downloaded file on the client must have a file name extension of `trace`. Keeping the extension on both sides also eases the download procedure.
   - Avoid BPD as low-level qualifier because this qualifier is recommended for bpd files that are created from trace data. The reason is that bpd files must have a file name extension of bpd on the client.

2. In the field **Disposition** specify how the data is to be written to the specified data set:

   **1=Append**

   Data collected during this task is appended to any previously collected data in the named data set.

   The disadvantage of appending data to existing data is that a copy of the catalog data is appended every time. If you want to use the data on the client, the bpd file becomes unnecessarily large and difficult to handle. Furthermore, if the system setup changes between two collect tasks, some report values can become undefined. If possible, use one of the following options.

If you collect data for a simulation on the client, never use this option.

**2=Overwrite**
Data collected during this task overwrites any previously collected data in the named data set.

**3=New**
The data set is allocated dynamically with RECFM=VBS, LRECL=32756, and BLKSIZE=6233 before data is written to it.

3. In the field **Record format** specify which IFCID record header information you want to be included in the collected data:

   **1=Standard**
   Includes the complete IFCID record header.

   Select this option if you want to create comprehensive activity reports. `Standard` provides the information that is used by the `INCLUDE`, `EXCLUDE`, `ORDER`, and `SORT` options of the **`BPACTIVITY REPORT`** command.

   Do not use this option if you want to use the data for simulation.

   **2=Short**
   Includes only part of the IFCID record header.

   Select this option if you want to use the data for the client-based functions. This option is mandatory for simulation.

   For most activity reports `Short` is also sufficient.

4. In the field **Data type** specify which Db2 trace data you want to collect:

   **1=Detail**
   Collects buffer pool statistics, catalog data, and buffer pool activity data.

   Select this option if you want to create detail activity reports, or bpd files or trace data files for use on the client. This option is mandatory for simulation.

   **2=Summary**
   Collects buffer pool statistics and catalog data.

   Select this option if you want to create summary activity reports.

   **Note:** Db2 updates summary data at statistics intervals. When you specify the duration of the collect task, permit for sufficient time to "capture" several intervals. If required, see“Determining when and how long to collect” on page 16 and “Preliminary remarks about the accuracy of summary and detail reports” on page 48 for more details.

   **3=Catalog only**
   Collects only catalog data. For possible uses see Chapter 12, “Concatenating trace data for activity reports and bpd files,” on page 153

5. In the field **OP Buffer size** specify a value from 8 KB to 1024 KB. The Online Performance (OP) buffer is used by Db2 to pass the trace data to Buffer Pool Analyzer (and other monitor programs).

   Specify a large buffer size to prevent a buffer overflow.

   Specify a moderate buffer size if you are constrained on virtual storage in DB2's database services address space (DBM1).

6. In the field **Start the DB2 trace** specify one of the following start conditions:

   **1=Immediately**
   Starts the Db2 trace immediately.

   **2=At (*hh*:*mm*:*ss*)**
   Starts the Db2 trace at a specified time, whereby *hh* is the hour, *mm* is the minute, and *ss* is the second. If the specified time is less than the current time, the trace starts the next day at the

specified time. Note that a trace is stopped if you leave Db2 Performance Expert or Buffer Pool Analyzer. See also step “4” on page 29.

7. In the field **Trace and collect data** specify how the data is to be traced:

   **1=Continuously**
   Runs the Db2 trace for the entire tracing period. Select this option if you want to use the trace data for simulation.

   **2=Every *x* minutes for *y* seconds**
   Runs the Db2 trace every *x* minutes for *y* seconds, whereby *x* denotes the specified minutes, and *y* denotes the specified seconds.

8. Specify one or two stop conditions for the Db2 trace. The trace stops when one of the conditions becomes true.

   - Activate one or both stop conditions by typing a slash (/) in the field preceding the condition. A greater-than (>) symbol in this field indicates that this condition was previously selected.
   - Specify one or both conditions as follows:
     - In the field **Elapsed time** specify the number of seconds the trace should run.
     - In the field **Number of records collected** specify a maximum number of records to be collected.

       Examples:

       - `10000`
       - `100K` (for `100000`)
       - `1M` (for `1000000`)

       If you want to use the trace data for simulation, ensure that you collect trace data for approximately 20 minutes, respectively 1200 seconds. In the field **Elapsed time** specify an appropriate value. Either deactivate the **Number of records collected** condition, or set it to 13000000 (approximately 2 GB of data), to ensure that the trace does not stop earlier.

## Interpreting trace status summary and trace messages

This topic shows examples of the Trace Status Summary panel and the Trace Messages panel. You can use this information to assess the progress and success of an activated or completed collect task.

### About this task

If you have selected the **Display task status** option on the Collect Report Data panel, the Trace Status Summary panel is displayed. The following is an example:

```
06/22/13 10:16    Collect Report Data for Buffer Pool Analysis

                              Trace Status Summary
  Command ===> ______________________________________________________________

  _  Display messages

  Task description . . . . : Collect data for buffer pool activity
  Data set name  . . . . . : 'NKA.COLLECT.TRACE'
  Data set status  . . . . : Closed
  Record format  . . . . . : Standard
  Data type  . . . . . . . : Summary
  Start trace  . . . . . . : Immediately
  Trace type . . . . . . . : Continuously
  Task activated . . . . . : 06/22/13 11:41:14.46
  Task stopped . . . . . . : 06/22/13 11:41:18.80
  DB2 Trace data started . : 06/22/13 11:41:15.71
  Last collected . . . . . : 06/22/13 11:41:15.71
  Records read . . . . . . : 1580
  Buffer overflow  . . . . : 0
  Records lost . . . . . . : 0

   F1=Help     F2=Split    F3=Exit     F9=Swap      F12=Cancel
```

This panel shows details of the collect task, for example, when the task was started and stopped.

## Procedure

1. Assess this information carefully with regard to the intended usage. For example:
   - The `Records read` count should show a reasonable number of collected trace records.
   - The `Buffer overflow` count gives an indication whether you should configure a larger **OP Buffer size** for this collect task.
   - The `Records lost` count should be less than 2% of the total number of records read (especially if the collected data is used for simulation).

     During data collection, trace records can get lost if the CRD task cannot keep up with reading the records at the speed that Db2 writes them. For most of the Buffer Pool Analyzer functions, this is not a grave situation. The only function that is sensitive to trace record loss is simulation.

     If the loss rate is too high, ensure that the Performance Expert address spaces are dispatched with a priority equal or higher than the Db2 database services address space (DBM1).
2. Select **Display messages**.

   The Trace Messages panel is displayed. The following is an example:

```
06/22/13 10:16    Collect Report Data for Buffer Pool Analysis

                              Trace Status Summary

                  IBM DB2 Buffer Pool Analyzer for z/OS   Row 1 to 18 of 19
  Command ===> ______________________________________________________________


                          Trace Messages
  FPEM0800I Task started at 06/22/13 11:40:41.915 for DB2 subsystem D721
  FPEM0819I Task description...Collect data for buffer pool activity
  FPEM0811I Task trigger...Immediate start
  FPEM0518E Nonzero IFI return code. RC 8, REASON X'00E60820'
  DSNW135I   -D721 P TRACE ALREADY ACTIVE, TRACE NUMBER 04
  DSN9023I   -D721 DSNWVCM1 '-START TRACE' ABNORMAL COMPLETION
  FPEM0818E An error occurred starting the DB2 traces above
  FPEM0801I Task stopped at 06/22/13 11:40:43.090
  FPEM0800I Task started at 06/22/13 11:41:14.464 for DB2 subsystem D721
  FPEM0819I Task description...Collect data for buffer pool activity
  FPEM0811I Task trigger...Immediate start
  DSNW130I   -D721 P TRACE STARTED, ASSIGNED TRACE NUMBER 04
  DSN9022I   -D721 DSNWVCM1 '-START TRACE' NORMAL COMPLETION
  FPEM0813I DB2 traces to OP1  started at 06/22/13 11:41:15.710
  DSNW131I   -D721 STOP TRACE SUCCESSFUL FOR TRACE NUMBER(S) 04
  DSN9022I   -D721 DSNWVCM1 '-STOP TRACE' NORMAL COMPLETION
  FPEM0814I DB2 traces to OP1  stopped at 06/22/13 11:41:15.713
  FPEM0815I 9 records written to 'NKA.COLLECT.TRACE'
```

   This panel shows messages about the progress and success of the collect task.

   In this example, the message FPEM0518E indicates that Db2 returned with a bad return code, `RC 8`, and with reason code `REASON X'00E60820'`. The Db2 trace command `STOP TRACE (P) TNO (4)` was issued, thereafter the collect task completed successfully.

   If you have requested that the collected data is to be compressed into a separate data set, and if errors occur during the compression, you also see messages preceded by message identifier TERSEMVS. These messages are from the TRSMAIN utility, which is used to perform the compression. For these errors, use the system code and the following information to diagnose the error.

   For more information, see *IBM DB2 10 for z/OS: Messages and Codes* or *IBM DB2 11 for z/OS: Messages and Codes*.

## Collecting data by using the batch JCL

This topic shows an example of a batch job that performs a so-called Collect Report Data task.

### About this task

The meaning of the parameters in the batch job is identical to those described in "Configuring a collect task" on page 31. Regarding the parameter syntax, note that keywords are succeeded by values in parentheses (), comment lines start with an asterisk (*), and blanks are allowed between keywords and values.

Note that you can use and modify the JCL sample provided in data set member *prefix*.TKO2SAMP(BPOMACRD). It is usually more current than the example in this information.

If you want to compress collected data into a separate data set, equivalent to the Collect Report Data (CRD) function, use a batch job similar to that shown in Chapter 14, "The TRSMAIN terse utility," on page 159.

You must assign a job class with a high priority to data collection batch jobs. This priority must be at least as high as that of Db2. Otherwise, trace records are not collected fast enough, resulting in lost records, or the trace might be started and stopped immediately after the catalog data is collected.

When the BPA CRD program is started from the APF-authorized steplib, it attempts to establish itself as an independent enclave that has a Workload Manager (WLM) subsystem type of Db2. It is possible that your WLM policy causes the generic WLM subsystem type of Db2 to get low dispatching priority. In this case, you can start the BPA CRD job from the non-APF copy of RKANMOD data set so the collection process will use the dispatching priority of the batch job.

Example:

```
//*******************************************************************//
//*                                                                 *//
//*   MODULE NAME : BPOMACRD                                        *//
//*                                                                 *//
//*   DESCRIPTION : DB2 BPA Collect Report Data Batch Job           *//
//*                                                                 *//
//*   COPYRIGHT   : IBM DB2 Buffer Pool Analyzer for z/OS  V5R4M0   *//
//*                 Licensed Materials - Property of IBM            *//
//*                 5655-W35 (C) Copyright IBM Corp. 2001, 2016     *//
//*                                                                 *//
//*   STATUS      : Version 5.4.0                                   *//
//*                                                                 *//
//*   FUNCTION    : Collect Report Data in Batch Mode               *//
//*                                                                 *//
//*                                                                 *//
//*   Notes =                                                       *//
//*     1.  Add a valid job card                                    *//
//*     2.  Change the prefix of the DB2 BPA load library db2bpa    *//
//*     3.  Change the prefix of the DB2 load library db2load       *//
//*     4.  Change the DPCOLLDD DD statement for the trace data     *//
//*     5.  Change the collect parameters in the SYSIN data set     *//
//*                                                                 *//
//*End of Specifications********************************************//
//*
//DB2BPA   EXEC PGM=DB2BPCRD
//STEPLIB  DD  DSN=db2bpa.SDSNLOAD,DISP=SHR
//         DD  DSN=db2load.RKANMOD,DISP=SHR
//SYSPRINT DD  SYSOUT=*
//SYSOUT   DD  SYSOUT=*
//* Protocol of the CRD parameters and status information
//DPMLOG   DD  SYSOUT=*
//* Report of the data collection job
//JOBSUMDD DD  SYSOUT=*
//*PCOLLDD DD  DISP=OLD,DSN=your.db2trace.dsname
//DPCOLLDD DD  DISP=(NEW,CATLG),DSN=your.db2trace.dsname,
//             DCB=(RECFM=VBS,BLKSIZE=9076,LRECL=32756),
//             SPACE=(TRK,(500,100)),VOL=SER=xxxxxx,UNIT=3390
//SYSIN    DD  *
* Mandatory parameters
  DB2SSID       (ssid)       * DB2 subsystem id
  PLANNAME      (planname)   * DB2 BPA planname
* Optional parameters, for fixed values enter either the characters
* specified in capital letters or the full word
```

```
  RECORD_FORMAT (short)      * STandard or SHort(default)
  DATATYPE      (detail)     * DEtail(default), SUmmary, or CAtalog
  STARTTIME     (im)         * IMmediately(default) or hh:mm:ss,
                             * where hh:mm:ss gives the time within the
                             * next 24 hours when the trace is to start
* Instead of specifying a start time you can use your batch scheduling
  DURATION      (nnnu)       * Maximum job duration, where nnn specifies
                             * time units and u = s for seconds
                             *                    m for minutes or
                             *                    h for hours
                             * Default is 30m.
  MAX_RECORDS   (rrrrr)      * Maximum number of records to be
                             * collected, optionally in K (=1000)
                             * or M (= 1000000). An example is 25000 which
                             * is the same as 25K.
  SAMPLING      (mmm,sss)    * Indicates that tracing is done in
                             * sampling mode. mmm denotes the
                             * time interval between 2 collection
                             * periods in minutes. sss denotes
                             * the time in seconds when DB2 trace is
                             * active during a sampling interval.
                             * If SAMPLING is omitted (default), DB2
                             * trace data are collected continuously.
  BUFSIZE       (nnnn)       * Specifies the op buffer size in the
                             * DB2 Start Trace command. nnnn indicates
                             * the number of KB and ranges from 8 to
                             * 16384, default is 2048 for 2048 KB.
  TRIGGGER      (nn)         * Specifies the threshold at which DB2 posts
                             * the monitoring program to read from the OP
                             * buffer. nn ranges from 1 to 99 indicating
                             * the fill percentage of the OP buffer.
                             * The default setting is 20.
```

The batch job generates the following files:

**DPCOLLDD**
: Contains the trace data.

**DPMLOG**
: Contains information about the parameter stream.

**JOBSUMDD**
: Contains a protocol about the process and trace messages from Buffer Pool Analyzer and Db2. The messages are identical to those shown in the Trace Messages panel, which is described in "Interpreting trace status summary and trace messages" on page 33.

# Chapter 4. Creating activity reports and bpd files

This topic describes how to create activity reports and buffer pool data (bpd) files from collected trace data. It describes an example of a batch job and the use of the **BPACTIVITY** command.

### About this task

Related tasks:

- Before you can perform these tasks, you must have performed the task described in Chapter 3, "Collecting data," on page 27. Note that the collected trace data must be available in uncompressed format.
- After you have performed this task, the reports and bpd files are available in named data sets and can be used for the tasks described in:
  - Chapter 5, "Interpreting activity reports," on page 47
  - Chapter 6, "Viewing performance data on the client," on page 89
  - Chapter 7, "Optimizing object placements and initial buffer pool sizes," on page 97
  - Chapter 9, "Analyzing long-term buffer pool performance," on page 121

Activity reports and buffer pool data files are created on the host by means of a batch job. In a batch job you specify:

1. The source of your input data (the data set containing the collected trace data).
2. The data you want to extract from the input data set and include in the activity reports or buffer pool data files.
3. For activity reports, the appearance of the reported data.
4. The output data set where the reports or the bpd files are to be stored.

Step 2 and step 3 use the **BPACTIVITY** command and its **REPORT** (for activity reports) and **FILE** (for bpd files) subcommands. Both subcommands have options to specify the content and the level of detail to be included in the output. The content can be manipulated, for example, by selecting only a specified time frame, or by including or excluding specific identifiers. The level of detail defines that either summary information or detail information is taken from the input and included in the output. The **REPORT** subcommand has additional options to specify the aggregation and sorting of the reported data. The **FILE** subcommand has an additional option to exclude information about inactive objects from the bpd file, which results in a smaller file size.

The following topics provide additional information:

- "Preliminary remarks about the content and filtering of input data " on page 37
- "Specifying a JCL command stream" on page 38
- "Specifying reports and bpd files with BPACTIVITY" on page 40

## Preliminary remarks about the content and filtering of input data

The **BPACTIVITY** command and its subcommands can only create information in reports or bpd files from information that is contained in the input data sets that are specified in the batch job.

### About this task

For example, if you collected trace data of data type `Summary`, you should not expect detail information in reports even if you specify this with the **BPACTIVITY** command options. However, if you collected trace data of data type `Detail`, you can specify that only summary information is reported. `Detail` data always includes `summary` data, as described in "Determining what to collect" on page 14.

The same considerations pertain to time frames of collected data in contrast to reported data. For example, if you collected trace data between 9:00 a.m. and 10:00 a.m., but specified 10:00 a.m. to 11:00 a.m. with the **BPACTIVITY** command options **FROM** and **TO**, your report or bpd file remains empty.

If you create bpd files (with the **FILE** subcommand) for use with the object placement function on the client, note that this function has options whether to include inactive objects into the assignment of objects to buffer pools. If you use the **FILE** subcommand with its **ACTIVEOBJECTS** option to exclude information about inactive objects from a bpd file, the object placement function consequentially treats only active objects. Plan ahead how you want to treat inactive objects with object placement and create the bpd file accordingly.

In your batch jobs you can use the **GLOBAL** command to preprocess input data before this data is processed by the **BPACTIVITY** command. The **GLOBAL** command has similar options to filter input data, and additional options to set default values for subcommands and to define processing options (such as DD statements for different data sets or time zone adjustments). The use of the **GLOBAL** command can improve the performance of your batch job, for example, if the **BPACTIVITY** command is used to produce multiple reports with a single invocation, or if the amount of input data is a multiple of the required output data.

Related reading: The **GLOBAL** command is available with Buffer Pool Analyzer and other Db2 performance tools and is described in the *Report Command Reference*. This book also provides more details about how to create reports and traces.

## Specifying a JCL command stream

This topic describes a typical batch job that creates a detail report and a bpd file. The JCL command stream and the DD statements are described to enable you to write your customized batch job.

### About this task

Note that you can use and modify the JCL sample provided in data set member *prefix*.TKO2SAMP(BPOQBTCH). It is usually more current than the example in this information.

In the following example the Buffer Pool Analyzer is installed under the high-level qualifier db2bpa. The batch job creates three different activity reports (using the **REPORT** subcommand with different options) and a trace data file (using the **FILE** subcommand without any option).

Example:

```
//*******************************************************************//
//*                                                                 *//
//*   MODULE NAME : BPOQBTCH                                        *//
//*                                                                 *//
//*   DESCRIPTION : DB2 BPA Batch Sample Job                        *//
//*                                                                 *//
//*   COPYRIGHT   : IBM DB2 Buffer Pool Analyzer for z/OS  V5R4M0   *//
//*                 Licensed Materials - Property of IBM            *//
//*                 5655-W35 (C) Copyright IBM Corp. 2001, 2016     *//
//*                                                                 *//
//*   STATUS      : Version 5.4.0                                   *//
//*                                                                 *//
//*   FUNCTION    : Create Batch Reports and a File                 *//
//*                                                                 *//
//*                                                                 *//
//*   Notes =                                                       *//
//*     1.  ADD A VALID JOB CARD                                    *//
//*     2.  Change the INPUTDD and BPFILDD1 DD statements           *//
//*     The commands in the SYSIN DD file can be changed            *//
//*     as described in the BPA User's Guide                        *//
//*End of Specifications*********************************************//
//*
//DB2BP   EXEC PGM=DB2BP
//*   CHANGE THE PREFIX OF THE LIBRARY db2bpa
//STEPLIB  DD  DSN=db2bpa.RKANMOD,DISP=SHR
//*   DD statement for trace data set
//INPUTDD  DD  DISP=SHR,DSN=bpa.trace.dataset
//SYSPRINT DD  SYSOUT=*
//SYSOUT   DD  SYSOUT=*
//*  DD statement for messages referring to the trace data set
```

```
//JOBSUMDD DD  SYSOUT=*
//*  DD statement for messages referring to execution of the job phases
//DPMLOG   DD  SYSOUT=*
//*  DD statement for second report
//BPAREP2  DD  SYSOUT=*
//*  DD statement for File, change parameters according to your needs
//BPFILDD1 DD  DISP=(NEW,CATLG),DSN=file.name,
//             DCB=(RECFM=VBS,BLKSIZE=9076,LRECL=32756),UNIT=SYSDA,
//             SPACE=(9096,(1000,500),RLSE)
//*  DD statement for BPA commands and parameters
//SYSIN    DD  *
*    Global command to adjust reported GMT to local time
GLOBAL
      timezone (-1:00)

*    Command with subcommands to create buffer pool activity Reports
*    and File with data to be loaded into a DB2 table or used on the
*    client for graphical display or for expert analysis
BPACTIVITY

*    Default report, output goes to default DD name BPREPDD
   REPORT

*    Following report has data summarized by plan, buffer pool id, and
*    page set. The data are sorted by plan and number of getpages.
*    Only the first 5 combinations of buffer pool id and page set
*    per plan are reported and the remainder.
   REPORT
         level(detail)
         order(planname-bpid-qpageset
               sortby(planname,getpage) top(5))
*    Output goes to DD name BPAREP2
         ddname(bparep2)

*    The next is a summary report where only the 11 most
*    active page sets in terms of asynchronous page activity is
*    produced. Buffer pool activity is always there.
   REPORT
         level(summary)
         order( sortby(asyncpage) top(11))
*    Output goes to the same DD name as the previous report.
         ddname(bparep2)

*    File, output goes to default DD name BPFILDD1
   FILE
*    EXEC command terminates reading of command input and starts
*    processing of trace input.
EXEC
//
```

**INPUTDD**

Lists one or more input data sets that contain trace data to be used to create activity reports and buffer pool data files. Usually, this is the data that you collected with the Collect Report Data (CRD) function of Buffer Pool Analyzer. The name of the input data set usually has a low-level qualifier of TRACE, as explained in "Configuring a collect task" on page 31.

The default ddname for the input data set is **INPUTDD**. If you specify a different ddname, use the **INPUTDD** option of the **GLOBAL** command. In this case, ensure that your JCL includes a valid DD statement for the new name.

You can also use Db2 trace data that is created by other means, such as data in GTF or SMF data sets, and it is generally possible to process multiple input data sets. See Chapter 12, "Concatenating trace data for activity reports and bpd files," on page 153 for more details.

**SYSOUT**

Contains messages from DFSORT. If SYSOUT is not specified, it is dynamically allocated to the SYSOUT message class of the job. The format of SYSOUT is RECFM=FBA, LRECL=133, BLKSIZE=6251.

**JOBSUMDD**

If specified, it contains the job summary log and the IFCID frequency distribution log. The format of JOBSUMDD is RECFM=FBA, LRECL=133, BLKSIZE= 6251.

**DPMLOG**

Contains messages from the Buffer Pool Analyzer command processor. If DPMLOG is not specified, it is dynamically allocated to the SYSOUT message class of the job. The format of DPMLOG is RECFM=FBA, LRECL=133, BLKSIZE=6251.

**BPRPTDD**

Contains the output from the **BPACTIVITY REPORT** subcommand. If BPRPTDD is not specified, it is dynamically allocated to the SYSOUT message class of the job. The format of BPRPTDD is RECFM=FBA, LRECL=81, BLKSIZE=8100.

If multiple **REPORT** subcommands are used, the resulting activity reports are written to BPRPTDD in corresponding sequence.

If you specify a different ddname with the **DDNAME** option of the **REPORT** subcommand, ensure that your JCL includes a valid DD statement for the new name.

**BPFILDD1**

Contains the output from the **BPACTIVITY FILE** subcommand. This is the bpd file that can be used on the client for viewing performance data and optimizing object placements. Also, its content can be loaded into Db2 tables. Ensure that your JCL contains a valid DD statement for this ddname. The format of BPFILDD1 is RECFM=VB, LRECL=9072, BLKSIZE=9076. The DD statement should specify a data set name with a low-level qualifier of BPD (for buffer pool data). After this data set is downloaded to the client, it must have a file name extension of bpd.

If you specify a different ddname with the **DDNAME** option of the **FILE** subcommand, ensure that your JCL includes a valid DD statement for the new name.

**BPWORK**

If specified, it determines where Buffer Pool Analyzer stores its temporary data, which can be up to 68 MB. Usually, this data set is created on the MVS-defined work volumes and deleted by Buffer Pool Analyzer. Use BPWORK if you want to control placement or size of the data set, or if you receive a B37 abend. The format of BPWORK is RECFM=VBS, LRECL=32756, BLKSIZE=6233.

Do not specify DUMMY or DISP=MOD for this data set.

**SYSIN**

This DD statement is mandatory. It contains the commands to be run by the job stream.

The format of SYSIN is RECFM=FB, LRECL=80, BLKSIZE=6160.

## Specifying reports and bpd files with BPACTIVITY

### Context

Use the **BPACTIVITY** command within a JCL command stream to specify the contents of reports and bpd files.

### Purpose

The **BPACTIVITY** command and its **REPORT** subcommand is used to create reports from collected data. Subcommand options specify the contents and appearances of reports. The **REPORT** subcommand can be used up to five times within a batch job. This means that you can create up to five different reports by using different options.

The **BPACTIVITY** command and its **FILE** subcommand is used to create bpd files from collected data. Subcommand options specify the contents of bpd files. The **FILE** subcommand can be used only once within a batch job.

### Usage

Some options are identical for both subcommands: **FROM** and **TO** limit the time frame, **INCLUDE** and **EXCLUDE** explicitly include or exclude specific contents, **LEVEL** specifies whether to include detail data or only summary data in a report or bpd (buffer pool data) file, and **DDNAME** overwrites the default output ddname.

The **REPORT** subcommand has one unique option: **ORDER** arranges the appearance of the reported data.

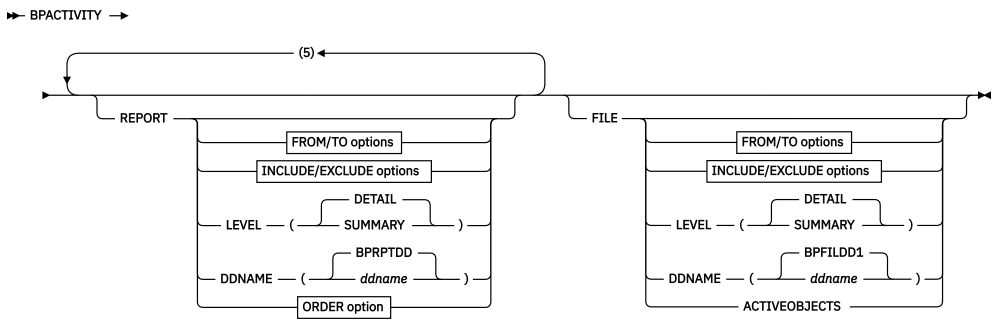
The **FILE** subcommand has one unique option: **ACTIVEOBJECTS** writes only information about active objects to the output ddname.

## Defaults

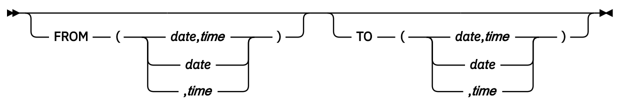
All options provide default values, if none are specified. Thus, the simplest command usage is `BPACTIVITY REPORT` or `BPACTIVITY FILE` to produce usable results.

By default, the reports are written to BPRPTDD, and the bpd file is written to BPFILDD1. These defaults can be changed by means of the **DDNAME** subcommand option.

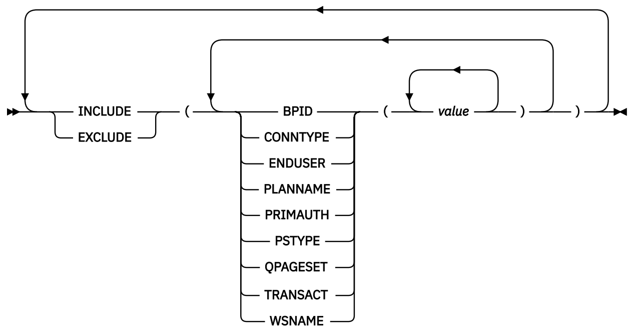
## Syntax



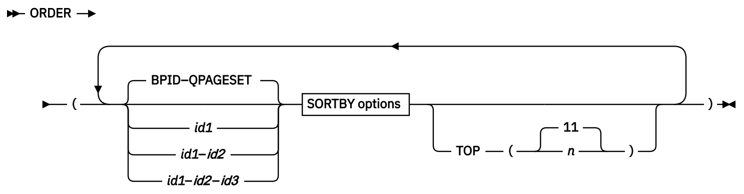
### FROM/TO options



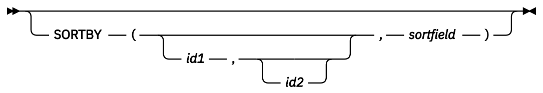
### INCLUDE/EXCLUDE options



### ORDER option



### SORTBY options



## Subcommand options and keywords

**FROM**
**TO**

Use these options with the **REPORT** or **FILE** subcommand if you want to selectively use trace data from input data sets for activity reports or bpd files. You can specify the selection by a start date and time (FROM) or an end date and time (TO), or any meaningful combination of both. If these options are used, trace records with timestamps equal or greater, respectively less or equal, to the criteria are included.

Specify dates as `mm/dd/yy`, whereby `mm` is the month, dd the day, and `yy` the year. Specify times as `hh/mm/ss.th`, whereby hh is the hour, `mm` the minute, `ss` the second, and `th` the thousands of a second (two digits each).

The use of theses options, aligned with appropriate data collection times and intervals, facilitates sophisticated reporting methods. For example, consider a collect task that accumulates summary data of your business' daily peak hours. Using this data as input, you can specify multiple **REPORT** subcommands in your batch job with different FROM and TO values to create separate summary reports for a comparison by day.

Usually, the use of this option is not required with the **FILE** subcommand. However, limiting the time scope of the bpd file content can make sense depending on the intended use on the client.

- For viewing performance data (described in Chapter 6, “Viewing performance data on the client,” on page 89) or object placement optimization (described in Chapter 7, “Optimizing object placements and initial buffer pool sizes,” on page 97) you can limit the time scope, for example, to isolate a known peak load period from the collected data. You can also segregate multiple periods from a single input data set to create several bpd files, each covering a different period, for example, to perform separate object placement optimizations. This of cause requires multiple runs of your batch job because the **FILE** subcommand can only be used once in a batch job. Note that both client functions do not provide means to limit the time scope; they use the entire content of the bpd file.
- For long-term analyses, described in Chapter 9, “Analyzing long-term buffer pool performance,” on page 121, you should ignore the FROM and TO options with the **FILE** subcommand. This function usually uses several bpd files as input and provides its own means to limit the time scope of data to be included in the analysis.

**INCLUDE**
**EXCLUDE**

Use these options with the **REPORT** or **FILE** subcommand if you want to selectively use trace data from input data sets for activity reports or bpd files. You can specify the inclusion (in the meaning of "include only these") or exclusion (in the meaning of "use all but these"), or any meaningful combination. If these options are used, data associated with the specified *identifier* and *value* combination is included, respectively excluded.

Note that, although these options can be used with the **FILE** subcommand, they are of limited use and should therefore be avoided. If you are anyway using these options, for whatever reason, be aware that the content of the resulting bpd file is no longer a reliable input to the object placement and initial buffer pool sizing function.

You can use the *identifiers* listed in Table 4 on page 43. However, two side effects should be noted:

- The absence of some identifiers affects the contents of summary reports and bpd files. To prevent missing information, ensure that these identifiers are not excluded.
- If the input data was collected with a short record format, some record header information is not present in the trace records. If you specify an identifier for which no header information is available, it will have no effect in reports or bpd files. For example, if you want to explicitly exclude records with a specific end user workstation name (identifier `WSNAME`), and short record format was used, those records cannot be identified because the `WSNAME` information is missing in the record header. Nevertheless, the presence of those identifiers do not harm.

*Table 4. Possible identifiers for the INCLUDE and EXCLUDE options of the* ***BPACTIVITY*** *command*

| Identifier | Meaning | Affects summaries | No effect with short record format |
|---|---|---|---|
| BPID | Buffer pool ID | X | |
| CONNTYPE | Connection type | | X |
| ENDUSER | End user ID | | X |
| PLANNAME | Plan name | | |
| PRIMAUTH | Primary authorization ID | | |
| PSTYPE | Type of page set for table space (T) or index space (I) | | |
| QPAGESET | Combination of database and page set | X | |
| TRANSACT | End user transaction name | | X |
| WSNAME | End user workstation name | | X |

**LEVEL**

Use this option with the **REPORT** or **FILE** subcommand if you want to create a summary report (where detail information is not required) or to avoid detail data in a bpd (buffer pool data) file. By default, if **LEVEL** is not specified, detail information is processed, which results in a detail report, respectively detail data in a bpd file. (Provided that a data type of `Detail` was specified when the data was collected.)

If used with the **REPORT** subcommand, **LEVEL(SUMMARY)** simply determines that a summary report is to be created. **LEVEL(DETAIL)** is the default and creates a detail report.

If used with the **FILE** subcommand, **LEVEL(SUMMARY)** can considerably reduce the processing time and the file size of the resulting bpd file by eliminating detail data. However, most client-based functions that use bpd files as input require detail data. See “Determining what to collect” on page 14 for further details and about which IFCIDs are included in detail and summary data. If processing time and file size are not critical, do not specify **LEVEL** and accept the default **LEVEL(DETAIL)**.

**Note:** If the **FILE** and **REPORT** subcommands are used together in the same batch job, the same **LEVEL** specification is required for both subcommands to ensure that either summary or detail data is processed with one invocation of the **BPACTIVITY** command.

The following examples are valid:

```
//* Valid invocation. Both subcommands assume LEVEL(DETAIL) as default
BPACTIVITY
   FILE
   REPORT
```

```
//* Valid invocation, both subcommands use only summary data
BPACTIVITY
   FILE LEVEL(SUMMARY)
   REPORT LEVEL(SUMMARY)
```

```
//* Also a valid invocation because the second REPORT processes
//* a subset of DETAIL data.
BPACTIVITY
   FILE
   REPORT LEVEL(DETAIL)
   REPORT LEVEL(SUMMARY)
```

The following examples are not valid:

```
//* Not a valid invocation because FILE excludes detail data.
BPACTIVITY
   REPORT LEVEL(DETAIL)
   FILE LEVEL(SUMMARY)
```

```
//* Not a valid invocation because FILE excludes detail data.
BPACTIVITY
   FILE LEVEL(SUMMARY)
   REPORT LEVEL(DETAIL)
   REPORT LEVEL(SUMMARY)
```

**DDNAME**

Use this option with the **REPORT** or **FILE** subcommand if you want to specify a ddname other than the default for the output reports or bpd files. By default, reports are written to BPRPTDD, and bpd files are written to BPFILDD1. If you specify a ddname, ensure that your JCL contains a valid DD statement for this ddname. If a DD statement is not specified, it is dynamically allocated to the SYSOUT message class of the job. See "Specifying a JCL command stream" on page 38 for more information.

**ORDER**

Use this option with the **REPORT** subcommand if you want to manipulate the aggregation level and sequence of reported statistics, the sorting of the aggregation, and the threshold for a "top-*n*" list of an activity report.

Aggregation summarizes the trace records by specific identifiers. By default report entries are aggregated by buffer pool ID (BPID) and by a combination of database and page set (QPAGESET). Further, the topmost 11 entries are included in the activity report.

- If you want to modify the aggregation level and sequence, you can specify up to three identifiers (*id1* to *id3*) of those listed in Table 5 on page 44. The number of identifiers determines the aggregation level, and the sequence of identifiers determines the aggregation sequence. Multiple identifiers are separated by a dash (–). Note that some identifiers have no effect if the input data was collected with a short record format (where extensive IFCID header information is omitted).

*Table 5. Possible aggregation identifiers for the **ORDER** option of the **BPACTIVITY** command*

| Identifier | Meaning | No effect with short record format |
|---|---|---|
| BPID | Buffer pool ID | |
| CONNTYPE | Connection type | X |
| ENDUSER | End user ID | X |
| PLANNAME | Plan name | |
| PRIMAUTH | Primary authorization ID | |
| PSTYPE | Type of page set for table space (T), or index space (I), or undetermined. | |
| QPAGESET | Combination of database and page set | |
| TRANSACT | End user transaction name | X |
| WSNAME | End user workstation name | X |

- If you want to modify the sort sequence of the (default or specified) aggregation, use the **SORTBY** option. The sort sequence can be specified by up to two optional identifiers (of those used for the aggregation, listed in Table 5 on page 44) and one mandatory sort field from those listed in Table 6 on page 45. Note that the identifiers correspond to record header fields and sort fields to record data fields (containing activity counts).

Table 6. Possible sort fields for the ***SORTBY*** option of the ***BPACTIVITY*** command

| Sort field | Meaning | Valid for detail reports | Valid for summary reports |
|---|---|---|---|
| GETPAGE | Getpage - total | X | |
| READREQ | Read request - total | X | |
| READSEQ | Read request - Synchronous | X | |
| READSYNC | Read request - Seq prefetch | X | |
| MISSRAND | Getpage Miss Random | X | |
| MISSASYN | Getpage Miss Asynch | X | |
| READPAGE | Read page total | X | |
| WRITEPAGE | Write page total | X | |
| WRITEREQ | Write request total | X | |
| SYNCREQ | Synchronous request | | X |
| ASYNCREAD | Asynchronous request | | X |
| ASYNCPAGE | Asynchronous page | | X |

For example, if you specify `ORDER (BPID-CONNTYPE SORTBY (BPID,GETPAGE))`, the statistics records are aggregated by BPID and CONNTYPE, but sorted by BPID and the number of Getpage operations.

- If you want to modify the default threshold (11) for the inclusion of statistics records in the activity report, use the **TOP** option and specify any number other than 11. The **TOP** option skips reporting for objects that have low usage rates. A value of 0 (or a very high value) includes all records in the activity report. See "The report header" on page 49 for an example of how this setting affects the report.

**ACTIVEOBJECTS**

Specify this option with the **FILE** subcommand if you want only information about active objects (those with buffer pool activities) in the output data set (the bpd file). By default, if this option is not used, also information about inactive objects (those without activities during the collection time) is included in the output.

The purpose of this subcommand option is to reduce the size of the bpd file if information about inactive objects is not needed. In large installations, the percentage of objects that show no activity during the time data is collected can be high, thus creating large bpd files. Smaller files are faster to download to the client and cause shorter preprocessing times.

Note that the object placement function on the client *can* make use of information about inactive objects. See "Preliminary remarks about the content and filtering of input data " on page 37 and "Step 3: Assigning objects to buffer pools" on page 103 (the description about the Used column and the **Assign objects not accessed during data collection** check box).

# Chapter 5. Interpreting activity reports

This topic shows examples of a summary report and a detail report and describes the layout and the elements of these reports. This topic helps to understand the host reports created by Buffer Pool Analyzer.

Related tasks:

- Before you can interpret reports, you must have performed the tasks described in:
  - Chapter 3, "Collecting data," on page 27
  - Chapter 4, "Creating activity reports and bpd files," on page 37

**General remarks:**

1. The reported data reflects the performance for the interval for which trace data was collected and for the time frame that was specified with the **GLOBAL** and the **BPACTIVITY** command.
2. Activity reports are composed of several sections in a fixed sequence. If data for a specific section is not available, a "no data to report" statement or similar is shown. For example, if group buffer pool specific information cannot be reported because the data was collected from a Db2 subsystem that is not a member of a data sharing group, several sections in the summary report have no data to report.
3. If a counter value or specific information in reports, in windows, or on panels is not shown, the following notation is used to indicate the reason:

   **N/A**
   Not applicable is shown if Db2 never produces a counter value in a specific context. Examples are:

   - A counter is not available in one Db2 version.
   - Counters are mutually exclusive.

   **N/C**
   Not calculated is shown for a derived field where the value cannot be calculated or is useless. Examples are:

   - A divide by zero (percentages, ratios).
   - Suppression of negative elapsed time values.
   - Required counter values for calculation marked as N/A or N/P.
   - Insufficient data or small counter values to allow significant statements (meaningless or misleading averages).

   **N/P**
   Not present is shown for a field where Db2 can present values, but does not in this instance. Examples are:

   - When counter values are not generated because of operational conditions (a trace class is not active).
   - An application does not provide a value because it is optional.

Summary reports and detail reports show several elements that can be changed by means of the Db2 ALTER BUFFERPOOL command. For details on how to change them see the *IBM DB2 11 for z/OS: Command Reference*.

The following topics provide additional information:

- "Preliminary remarks about the accuracy of summary and detail reports" on page 48
- "Summary reports" on page 49
- "Detail reports" on page 75

## Preliminary remarks about the accuracy of summary and detail reports

Summary and detail reports show in parts identical information, for example, the number of Getpage requests during the time data was collected.

If you work with summary and detail reports that cover identical time frames, you expect that identical counters report identical numbers, but might encounter that these numbers sometimes are not equal. This topic discloses the technical causes and helps to understand the accuracies of both reports.

As described in "Collecting data" on page 13, different data types, identified by IFCIDs, are used for summary and detail reports. Summary reports use buffer pool and data set *statistics* data from Db2, whereas detail reports use *activity* data from Db2. Both data types are continuously provided by Db2. Tools like Buffer Pool Analyzer collect this data for specified time frames that you want to analyze.

Activity data is purely event based. Db2 keeps a record of every single event. When Buffer Pool Analyzer collects activity data from Db2 for a duration or time frame specified by you, it obtains precise information about every single activity during that time frame. This information and the known data collection start time and end time can be used for precise totaling and calculations and results in exact numbers in detail reports.

However, statistics data is recorded by Db2 at intervals, and this interval can vary dependent on the initial system settings. When Buffer Pool Analyzer collects statistics data from Db2 for a duration or time frame specified by you, it gets hold of a number of interval recordings during that time frame. Worse, the collection start time and end time rarely perfectly match the system's interval recordings. As a result, any calculations and the numbers in summary reports are based on the time frame between the first and last interval recording that is covered by the specified start and end times. Partial intervals at the beginning and ending of the collection time remain uncovered.

Example:

```
 |------|------|------|------|------|------|------|------|-->  Time line
t0     t1     t2     t3     t4     t5     t6     t7     t8

 |  4   |  10  |  3   |  7   |  4   |  8   |   2  |   5  |     Activities between time slots


 |-------------|-------------|-------------|-------------|-->  Statistics intervals

              14            24            36            43     Statistics counts at intervals
                                                               (accumulative)

        |------------------------------------------|           Data collection time

        |  10  |  3   |  7   |  4   |  8   |   2  |           34 actual activities

        |     14            24            36      |           36 - 14 = 22 calculated
activities
```

*Figure 9. Example of how the statistics interval influences the accuracy*

This example uses fictitious numbers and an unrealistically short data collection time to emphasize the cause of inaccuracies. The data collection starts at t1 and ends at t7. A detail report would show precisely 34 activities during this time, simply by counting the events that DB2 has recorded for every single activity. Opposed to this, summary reports rely on the statistics counter, which is updated (incremented in this case) at times t2, t4, t6, t8, and so on. Here, only the counter values at times t2, t4, and t6 are covered by the data collection time. The number of activities (22) is calculated by determining the difference between the smallest (14) and greatest (36) counter value. No attempt is made to estimate how the smallest value (14) developed between t1 and t2. Also, the time between t6 and t7 remains unconsidered.

In practice these inaccuracies are marginal, if at all visible. They do not degrade the expressiveness of summary reports. Keep in mind that two different methods are used, and do not try to match any counter values down to a single digit in both reports.

If you want to know more about Db2 statistics data, different counter types (watermarks, snapshots, accumulative counters), different processing modes (regular, interval, delta processing), see, for example, *Monitoring Performance from the Workstation*.

## Summary reports

This topic shows and describes the elements of a summary report.

A summary report is created as a single entity; however, to facilitate reading it is shown here in separate sections, as follows:

- "The report header" on page 49
- "The Buffer Pool Statistics Highlights section" on page 50
- "The Group Buffer Pools Activity Data Highlights section" on page 50
- "The Buffer Pool Characteristics section" on page 50
- "The Buffer Pool Statistics section" on page 53
- "The Data Set Statistics section" on page 61
- "The Group Buffer Pools Activity Data section" on page 63
- "The Group Buffer Pool Attributes section" on page 70
- "The Buffer Manager PSET/Part P-lock Request section" on page 71
- "The CF Cache Structure Statistics section" on page 73

Note that the first two sections of a summary report are highlights sections. They show selected elements from "The Buffer Pool Statistics section" on page 53 and "The Group Buffer Pools Activity Data section" on page 63 of statistical significance, for example, when certain counter values in these sections deserve closer attention. Start your analysis and interpretation with these highlight sections.

### The report header

The report header is shown at the top of every report page and identifies the report and the command options that were used to create the report.

The following is an example of a report header:

```
1     OMEGAMON XE FOR DB2 PE (V5R4M0) - BUFFER POOL ACTIVITY REPORT    PAGE: 1-1
                            ORDER: BPID-QPAGESET
                  SORTBY: BPID,ASYNCPAGE  TOP: 17  LEVEL: SUMMARY
 GROUP:     DSNJ        LOCATION:       PMODSNJ             DB2 VERSION: V11
 MEMBER:    SGJ1        REQUESTED FROM: NOT SPECIFIED       TO: NOT SPECIFIED
 SUBSYSTEM: SGJ1        INTERVAL FROM:  01/24/13 07:21:46   TO: 01/24/13 09:27:13
⋮
```

- LEVEL specifies the type of report, here, a summary report.
- ORDER specifies the aggregation, here, by buffer pool ID (BPID) and a combination of database and page set (QPAGESET).
- SORTBY specifies the sorting of the aggregated information, here, by buffer pool ID (BPID) and asynchronous page (ASYNCPAGE).
- TOP specifies that the 17 topmost aggregations are reported. If the trace data contains more than 17 objects, they are aggregated under the label `Others` in the report.

This example was created with the following command:

```
BPACTIVITY REPORT LEVEL(SUMMARY)
                  ORDER(BPID-QPAGESET SORTBY(BPID, ASYNCPAGE) TOP(17))
```

Note that the ORDER, SORTBY, and TOP options affect only the information in "The Data Set Statistics section" on page 61.

## The Buffer Pool Statistics Highlights section

This report section shows selected elements from "The Buffer Pool Statistics section" on page 53 that deserve further investigation. Note especially those values that are marked with asterisks; they mark counter values that should typically be zero. The content and length of this report section varies. Several elements are always present, others only if the underlying counter values demand highlighting. If you need a detailed description of these elements, see "The Buffer Pool Statistics section" on page 53. The elements are displayed in equal order in both sections. The following example shows a hypothetical section, with some of the elements that can show up in your actual report.

```
1     OMEGAMON XE FOR DB2 PE (V5R4M0) - BUFFER POOL ACTIVITY REPORT    PAGE: 1-2
                               ORDER: BPID-QPAGESET
                 SORTBY: BPID,ASYNCPAGE  TOP: 17  LEVEL: SUMMARY
 GROUP:     DSNJ        LOCATION:       PMODSNJ             DB2 VERSION: V11
 MEMBER:    SGJ1        REQUESTED FROM: NOT SPECIFIED       TO: NOT SPECIFIED
 SUBSYSTEM: SGJ1        INTERVAL FROM:  01/24/13 07:21:46   TO: 01/24/13 09:27:13

                   =======  Buffer Pool Statistics Highlights  ==========
 BUFFER  POOL ID                  BP0        BP2
 ------------------------- ---------- ----------
 Buffers allocated               1484       2642
 System hit ratio               99.34        n/c
 Application hit ratio          99.34        n/c
 Getpage request                 5921          0
 Synchron.read sequential           4*        16*
 Parallel query request
  Reduced                           5*         8*
 ========================= ========== ==========
⋮
```

## The Group Buffer Pools Activity Data Highlights section

This report section shows selected elements from "The Group Buffer Pools Activity Data section" on page 63 that deserve further investigation. Note especially those values that are marked with asterisks; they mark counter values that should typically be zero. The content and length of this report section varies. Several elements are always present; others only if the underlying counter values demand highlighting. If you need a detailed description of these elements, see "The Group Buffer Pools Activity Data section" on page 63. The elements are displayed in equal order in both sections. The following example shows a hypothetical section, with some of the elements that can show up in your actual report.

```
1     OMEGAMON XE FOR DB2 PE (V5R4M0) - BUFFER POOL ACTIVITY REPORT    PAGE: 1-3
                               ORDER: BPID-QPAGESET
                 SORTBY: BPID,ASYNCPAGE  TOP: 17  LEVEL: SUMMARY
 GROUP:     DSNJ        LOCATION:       PMODSNJ             DB2 VERSION: V11
 MEMBER:    SGJ1        REQUESTED FROM: NOT SPECIFIED       TO: NOT SPECIFIED
 SUBSYSTEM: SGJ1        INTERVAL FROM:  01/24/13 07:21:46   TO: 01/24/13 09:27:13

              =======     Group Buffer Pools Activity Data Highlights    =======
 Group Buffer Pool                        GBP0
 ------------------------------------ ----------
 Group BP Hit Ratio (%)                  20.00
 Write failed-no storage                     8*
 Write to secondary GBP failed              16*
 ==================================== ==========
⋮
```

## The Buffer Pool Characteristics section

Buffer pool characteristics are attributes of individual buffer pools, such as sizes and thresholds.

They are defined during installation in the Db2 bootstrap data set. When Db2 is started, these settings apply by default. If these attributes are changed (by means of the ALTER BUFFERPOOL command), they are stored and used until they are changed again. The buffer pool characteristics values are retrieved from IFCID 202.

```
    OMEGAMON XE FOR DB2 PE (V5R4M0) - BUFFER POOL ACTIVITY REPORT    PAGE: 1-2
                               ORDER: BPID-QPAGESET
                               TOP: 11  LEVEL: SUMMARY
 GROUP:     N/P         LOCATION:       PMODB11             DB2 VERSION: V11
 MEMBER:    N/P         REQUESTED FROM: NOT SPECIFIED       TO: NOT SPECIFIED
 SUBSYSTEM: DB11        INTERVAL FROM:  12/03/14 13:19:39   TO: 12/03/14 14:32:00

                     =========   Buffer Pool Characteristics   =========
 BPID                             BP0      BP1      BP2      BP3      BP4      BP5
 -------------------------- -------- -------- -------- -------- -------- --------
 General
  Virtual pool size              5000    10000    20000     5000     1000      100
  Pages fixed in real stor         No       No       No       No       No       No
  Page steal method               LRU      LRU      LRU      LRU      LRU      LRU
  Autosize attribute               No       No       No       No       No       No
  VPool size min                    0        0        0        0        0        0
  VPool size max                    0        0        0        0        0        0
  Frame size                       4K       4K       4K       4K       4K       1M
 Thresholds
  Virtual sequential               80       80       80       80       80       80
  Deferred write                   30       30       30       30       30       30
  Vert deferred write(buff)         0        0        0        0        0        0
  Vert deferred write   (%)         5        5        5        5        5        5
  Parallel sequential              50       50       50       50       50       50
  Assisting parallel seq            0        0        0        0        0        0
 Simulated BP
  Simulated BP size               550      550      550        0        0    10000
  Simulated BP seq thresh          60       60       60        0        0       80
 ========================== ======== ======== ======== ======== ======== ========

 BPID                             BP6      BP7      BP8      BP9     BP10     BP11
 -------------------------- -------- -------- -------- -------- -------- --------
 General
  Virtual pool size                50       50     1000     2000     1000     1000
  Pages fixed in real stor         No      Yes       No       No       No       No
  Page steal method               LRU      LRU      LRU      LRU      LRU      LRU
  Autosize attribute              Yes       No       No       No       No       No
  VPool size min                   50        0        0        0        0        0
  VPool size max                  500        0        0        0        0        0
  Frame size                       4K       1M       4K       4K       4K       4K
 Thresholds
  Virtual sequential               80       99       80       80       80       80
  Deferred write                   30       30       30       30       30       30
  Vert deferred write(buff)         0        0        0        0        0        0
  Vert deferred write   (%)         5        5        5        5        5        5
  Parallel sequential              50       50       50       50       50       50
  Assisting parallel seq            0        0        0        0        0        0
 Simulated BP
  Simulated BP size              5000    50000        0        0        0        0
  Simulated BP seq thresh          80       50        0        0        0        0
 ========================== ======== ======== ======== ======== ======== ========
⋮
```

**General — Virtual pool size**
Size of the virtual buffer pool.

**General — Page steal method**
Page stealing algorithm (PGSTEAL). When Db2 removes a page in the buffer pool to make room for a newer page, this action is called stealing the page from the buffer pool. By default, Db2 uses a Least-Recently-Used (LRU) algorithm for managing pages in storage. This means that it takes away pages that are not used so that more recently used pages can remain in the virtual buffer pool.

You can determine that Db2 uses a first-in-first-out (FIFO) algorithm. In this case, Db2 does not check how often a page is referenced. The oldest pages are always moved out, no matter how frequently they are referenced. This results in a small decrease in the cost of a Getpage operation. It can reduce internal Db2 latch contention in environments that require very high concurrency.

**VPool size min**
The minimum size of the auto-sized virtual pool. The field name is QDBPVPMI.

**VPool size max**
The maximum size of the auto-sized virtual pool. The field name is QDBPVPMA.

**Frame size**
The z/OS page frame size to back up the virtual pool buffers. The field name is QDBPFRAM.

**Thresholds — Virtual sequential**
Virtual pool sequential steal threshold (VPSEQT). This threshold is a percentage of the virtual buffer pool that might be occupied by sequentially accessed pages. The pages can be in the state `updated`,

`in use`, or `available`. Therefore, each page might count regarding exceeding any other buffer pool threshold.

The default value for VPSEQT is 80%. You can change this value to a value from 0% to 100%.

VPSEQT is checked before stealing a buffer for a sequentially accessed page instead of accessing the page in the virtual buffer pool. If the threshold is exceeded, Db2 tries to steal a buffer that holds a sequentially accessed page rather than one that holds a randomly accessed page.

If you set VPSEQT to 0%, sequential pages cannot occupy space in the virtual buffer pool. In this case, prefetch is disabled, and sequentially accessed pages are discarded when they are released.

If you set VPSEQT to 100%, sequential pages can monopolize the entire virtual buffer pool.

**Thresholds — Deferred write**

Deferred write threshold (DWQT). This threshold is a percentage of the virtual buffer pool that might be occupied by unavailable pages, including updated pages and pages in use.

The default value for DWQT is 50%. You can change this value to any value from 0% to 90%.

Db2 checks DWQT when an update to a page is complete. If the percentage of unavailable pages in the virtual buffer pool exceeds DWQT, write operations are scheduled for up to 128 pages per data set to decrease the number of unavailable buffers to 10% below DWQT. For example, if DWQT is 50%, the number of unavailable buffers is reduced to 40%.

When the limit of DWQT is reached, the data sets containing the oldest updated pages are written asynchronously. Db2 continues to write pages until the ratio goes below the DWQT.

The number of pages is determined by the buffer pool page size as shown in Table 7 on page 52.

*Table 7. Number of changed pages based on buffer pool size*

| Buffer pool size for pages | Number of changed pages |
|---|---|
| 4 KB | 40 |
| 8 KB | 24 |
| 16 KB | 16 |
| 32 KB | 12 |

**Thresholds — Vert deferred write**

Vertical deferred write threshold (VDWQT). This threshold is similar to the deferred write threshold (DWQT), but it applies to the number of updated pages for one single page set in the buffer pool. If the percentage or number of updated pages for the data set exceeds the threshold, writes up to 128 pages are scheduled for that data set.

You can specify VDWQT in one of the following ways:

- As a percentage of the virtual buffer pool that might be occupied by updated pages from one single page set. The default value for this threshold is 10%. You can change the percentage to any value from 0% to 90%.
- As the total number of buffers in the virtual buffer pool that might be occupied by updated pages from one single page set. You can specify the number of buffers from 0 to 9999. If you want to use the number of buffers as your threshold, you must set the percentage threshold to 0.

Because any buffers that count for VDWQT also count for DWQT, setting the VDWQT percentage higher than DWQT has no effect: DWQT is reached first, write operations are scheduled, and VDWQT is never reached. Therefore, the ALTER BUFFERPOOL command does not allow you to set the VDWQT percentage to a value greater than DWQT. You can specify a number of buffers for VDWQT that is higher than DWQT, but this specification has no effect. The threshold is overridden by specific Db2 utilities that use a constant limit of 64 pages rather than a percentage of the virtual buffer pool size. LOAD, REORG, and RECOVER use a constant limit of 128 pages.

If you set VDWQT to zero for the percentage and number of buffers, the minimum number of pages written is the same as for DWQT.

**Thresholds — Parallel sequential**
Virtual buffer pool parallel sequential threshold (VPPSEQT). This threshold is a part of the virtual buffer pool that might support parallel operations. It is measured as a percentage of the sequential steal threshold (VPSEQT). Setting VPPSEQT to zero disables parallel operation.

The default value for this threshold is 50% of the sequential steal threshold (VPSEQT). You can change the default value to any value from 0% to 100%.

**Thresholds — Assisting parallel seq**
Virtual buffer pool assisting parallel sequential threshold (VPXPSEQT). This threshold is a part of the virtual buffer pool that might support parallel operations initiated from another Db2 in the data sharing group. It is measured as a percentage of VPPSEQT.

Setting VPXPSEQT to zero prevents Db2 from supporting Sysplex query parallelism at run time for queries that use this buffer pool.

The default value for this threshold is 0% of the parallel sequential threshold (VPPSEQT). You can change the default value to any value from 0% to 100%.

**Simulated BP size**
Simulated Buffer pool size.

**Simulated BP seq thresh**
The sequential steal threshold (1-100) for the simulated buffer pool, expressed as a percentage of the total simulated buffer pool size.

## The Buffer Pool Statistics section

The buffer pool statistics values are retrieved from IFCID 2.

```
1   OMEGAMON XE FOR DB2 PE (V5R4M0) - BUFFER POOL ACTIVITY REPORT    PAGE: 1-6
                                   ORDER: BPID-QPAGESET
                SORTBY: BPID,ASYNCPAGE  TOP: 17  LEVEL: SUMMARY
 GROUP:     N/P         LOCATION:       OMPDC61            DB2 VERSION: V12
 MEMBER:    N/P         REQUESTED FROM: NOT SPECIFIED       TO: NOT SPECIFIED
 SUBSYSTEM: DC61        INTERVAL FROM:  04/05/16 13:40:56   TO: 04/05/16 14:47:00

                  =======     Buffer Pool Statistics     =======
 BUFFER  POOL ID               BP32K     BP32K4      BP8K0      BP8K4     BP16K0
 ------------------------- ---------- ---------- ---------- ---------- ----------
 Buffers allocated              1000        250       2000       1000       2000
 Reached threshold
  Deferred write                   0          0          0          0          0
  Vertical deferred write         16         14          0         12          0
  Data manager                     0          0          0          0          0
  SLRU length equ. VPSEQT          0          0          0          0          0
 Current active buffer           126        202          2        947          0
 Buffer pool full                  0          0          0          0          0
 Data set opens                    0          0          0          0          0
 Migrated data set                 0          0          0          0          0
 Recall timeout                    0          0          0          0          0
 Expansion or contraction          0          0          0          0          0
 Expansion failure                 0          0          0          0          0
 Concurrent prefetch I/O           0          0         20          0          0
 Prefetch I/O reduction            0          0          0          0          0
 Parallel query request            0          0          0          0          0
  Reduced                          0          0          0          0          0
 Pref quantity reduced
  Reduced to 1/2                   0          0          0          0          0
  Reduced to 1/4                   0          0          0          0          0
 Min buffers on SLRU              89         48       1579         53         10
 Max buffers on SLRU              89         48       1579         53         10
 Random getpage SLRU hits          1          0          0          0          0
 Pages added to LPL                0          0          0          0          0

 System hit ratio              99.28      70.83     100.00      75.76        n/c
 Application hit ratio         99.52      99.57     100.00      99.32        n/c
 Getpage request                5661        696        935       2780          0
  Sequential                    2277        514        826       2076          0
   Overflow sequential             0        212          0        746          0
  Random                        3384        182        109        704          0
   Overflow random                 0        172          0        594          0
  Unsuccessful getpages            0          0          0          0          0
  Unsucc seq getpages              0          0          0          0          0
 Read
  Synchronous read                27          3          0         19          0
   Sequential                      1          2          0         18          0
    Overflow sequential            0          2          0         18          0
   Random                         26          1          0          1          0
    Overflow random                0          1          0          1          0
  Sequential prefetch
   Request                         0         52          0         46          0
   Read                            0         52          0         43          0
   Pages read                      0        200          0        655          0
   Pages read/read               n/c       3.85        n/c      15.23        n/c
  List prefetch
   Request                         0          0          0          0          0
   Read                            0          0          0          0          0
   Pages read                      0          0          0          0          0
   Pages read/read               n/c        n/c        n/c        n/c        n/c

                                                              1   OMEGAMON XE FOR DB2 PE
(V5R4M0) - BUFFER POOL ACTIVITY REPORT    PAGE: 1-7
                               ORDER: BPID-QPAGESET
                 SORTBY: BPID,ASYNCPAGE  TOP: 17  LEVEL: SUMMARY
 GROUP:     N/P         LOCATION:       OMPDC61             DB2 VERSION: V12
 MEMBER:    N/P         REQUESTED FROM: NOT SPECIFIED       TO: NOT SPECIFIED
 SUBSYSTEM: DC61        INTERVAL FROM:  04/05/16 13:40:56   TO: 04/05/16 14:47:00

 BUFFER  POOL ID              BP32K     BP32K4      BP8K0      BP8K4     BP16K0
           (continue)
 ------------------------- ---------- ---------- ---------- ---------- ----------
  Dynamic prefetch
   Request                         7          0          5          0          0
   Read                            6          0          0          0          0
   Pages read                     14          0          0          0          0
   Pages read/read              2.33        n/c        n/c        n/c        n/c
  Prefetch disabled
   No buffer                       0          0          0          1          0
   No read engine                  0          0          0          0          0
  Page-ins required               41          0          0          0          0
 Write
  Buffer updates                5973      25948         67      27010          0
  Page write                    1027        192          9       1153          0
  Updates/page write            5.82     135.15       7.44      23.43        n/c
  Synchronous write                0          0          3        410          0
  Asynchronous write             271         61          6         72          0
  Pages/write req
  Page-ins required                0          0          0          0          0
 Sort/merge
 Merge
  Pass requested                   0          0          0          0          0
  Pass degraded low buffer         0          0          0          0          0
 Workfile
  Max concurrent used              2          0          0          0          0
  Req rejected low buffer          0          0          0          0          0
  Req all merge passes             0          0          0          0          0
  Not created no buffer            0          0          0          0          0
  Prefetch not scheduled           0          0          0          0          0
  Pages to destruct                0          0          0          0          0
  Pages not written                0          0          0          0          0
 Unlock castout
  I/O operations                   0          0          0          0          0
  Pages written                    0          0          0          0          0

 Simulated BP Activity
  Current pages in use           n/p        n/p        n/p        n/p        n/p
  Max pages in use               n/p        n/p        n/p        n/p        n/p
  Current seq pages in use       n/p        n/p        n/p        n/p        n/p
  Max seq pages in use           n/p        n/p        n/p        n/p        n/p
  Avoidable read I/O
   Sync  read I/O (R)            n/p        n/p        n/p        n/p        n/p
   Sync  read I/O (S)            n/p        n/p        n/p        n/p        n/p
   Async read I/O                n/p        n/p        n/p        n/p        n/p
   Sync  GBP reads (R)           n/p        n/p        n/p        n/p        n/p
   Sync  GBP reads (S)           n/p        n/p        n/p        n/p        n/p
   Async GBP reads               n/p        n/p        n/p        n/p        n/p
```

Asterisks (*) beside elements denote those elements that can show up in "The Buffer Pool Statistics Highlights section" on page 50.

**Buffers allocated ***
The number of buffers that are allocated to a virtual buffer pool.

The number of buffers within each pool is always less than or equal to the corresponding value specified at installation time or when using the ALTER BUFFERPOOL command.

**Reached threshold — Deferred write ***
The number of times the deferred write threshold (DWTH) was reached.

This threshold is a percentage of the virtual buffer pool that unavailable pages might occupy, including both updated pages and pages in use. Db2 checks this threshold when an update to a page is completed. If the percentage of unavailable pages in the virtual buffer pool exceeds the threshold, write operations are scheduled for enough data sets (at up to 128 pages per data set) to decrease the number of unavailable buffers to 10% below the threshold.

**Reached threshold — Vertical deferred write ***
The number of times the vertical deferred write threshold was reached. This threshold is expressed as a percentage of the virtual buffer pool that may be occupied by updated pages from one single data set. This threshold is checked whenever an update to a page is completed. If the percentage of updated pages for the data set exceeds the threshold, writes are scheduled for that data set.

**Reached threshold — Data manager ***
The number of times the data manager critical threshold (DMTH-95%) was reached.

This field shows how many times a page was immediately released because the data management threshold was reached. If the threshold is constantly reached, you need to identify the objects that are monopolizing the buffer pool.

The threshold is checked before a page is read or updated. If the threshold has not been exceeded, Db2 accesses the page in the virtual buffer pool once for each page, no matter how many rows are retrieved or updated in that page. If the threshold has been exceeded, Getpage and Release requests apply to rows instead of pages. That is, when more than one row is retrieved or updated in a page, more than one Getpage and Release request is performed on that page.

The data manager threshold (DMTH) is fixed threshold, set to 95% of the virtual pool size. Reaching this threshold has a significant impact on a system's performance. Reaching this threshold for one pool can cause Db2 not to release pages in other pools as well.

**Reached threshold — SLRU length equ. VPSEQT**
The total number of times when length of SLRU = VPSEQT.

**Current active buffer ***
The total number of currently active (nonstealable) buffers. This field is an instantaneous sample of the number of buffers in the buffer pool that were updated or in use at the time this monitor data was requested. Because this field gives a snapshot value at statistics collection time, it only shows a problem if it happens at this time.

**Buffer pool full ***
The number of times a usable buffer cannot be located in the virtual buffer pool because the virtual buffer pool was full.

**Data set opens**
The number of data sets physically opened successfully. This value is cumulative from the start of the Db2 statistics interval.

**Migrated data set**
The number of times migrated data sets were encountered.

**Recall timeout ***
The number of recall timeouts.

**Expansion or contraction**
The number of successful virtual buffer pool expansions or contractions due to the ALTER BUFFERPOOL command. An increase in this counter indicates that buffer-pool-related system parameters have been changed.

**Expansion failure**
The total number of virtual buffer pool expansion failures due to the lack of virtual storage space.

**Concurrent prefetch I/O**
The highest number of concurrent prefetch I/O streams allocated to support a parallel I/O or CP query in this buffer pool. It reflects prefetch activities for non-work-file page sets. This number only applies to query I/O and CP parallelism.

**Prefetch I/O reduction ***
The total number of requested prefetch I/O streams that were denied because of a lack of buffer pool storage space.

It only applies to query I/O and CP parallelism. For example, if 100 prefetch I/O streams are requested and only 80 are granted, then 20 is added to the number in this field.

**Parallel query request**
The total number of requests made for parallel query support in this buffer pool. This field only applies to non-work-file page sets in query I/O and CP parallelism.

**Parallel query request — Reduced ***
The number of times that Db2 cannot allocate the requested number of buffer pages to allow a parallel group to run as planned. This field only applies to non-work-file page sets in query I/O and CP parallelism.

**Pref quantity reduced — Reduced to 1/2 ***
The total number of times prefetch quantity is reduced from normal to 50% of normal. The normal size depends on the page size of the buffer pool. This field only applies to query I/O and CP parallelism.

**Pref quantity reduced — Reduced to 1/4 ***
The total number of times prefetch quantity is reduced from 50% to 25% of normal. The normal size depends on the page size of the buffer pool. This field only applies to query I/O and CP parallelism.

**Min buffers on SLRU**
The minimum number of buffers on SLRU, low water mark within an interval.

**Max buffers on SLRU**
The maximum number of buffers on SLRU, high water mark within an interval.

**Random getpage SLRU hits**
The total number of times that the random Getpage request has a buffer hit and the buffer is on the least-recently-used (SLRU) chain.

**Pages added to LPL**
The number of times that one or more pages were added to the logical page list (LPL). The field name is QBSTLPL.

**System hit ratio ***
The number of Getpage requests by Db2 and satisfied by the buffer pool, expressed as a percentage of all Getpage requests.

This shows the percentage of pages that are found in the buffer pool without doing any type of I/O.

The system hit ratio is affected by prefetch I/O. The value is usually lower in an application that causes mostly sequential accesses, respectively higher if a series of similar operations are performed on the same data.

A negative system hit ratio indicates that the number of prefetched pages is greater than the number of Getpages. This happens if prefetch operations are bringing in pages that are not subsequently referenced. The reason for this is that the query stops before it reaches the end of the prefetched pages, or that the prefetched pages are stolen by Db2 for reuse before the query can access them. Consider increasing the sequential steal threshold (VPSEQT), increasing the buffer pool size, or revising the assignments of page sets to buffer pools.

Compare the value in this field with the application hit ratio to determine the efficiency of prefetch operations.

**Application hit ratio ***

The number of Getpage requests issued by applications and satisfied by the buffer pool, expressed as a percentage of all Getpage requests issued by applications.

A low hit ratio indicates the level of synchronous I/O because prefetched pages that are already in the buffer pool count as hits. The value is a relative value depending on the type of application. For example, an application that browses large amounts of noncontinuous data might have a buffer pool hit ratio of 0. Check those cases in which the hit ratio drops significantly for the same application.

**Getpage request ***

This counter is incremented for:

- Each successful or unsuccessful page request, where the query is not processed in parallel.
- Each successful page request, where the query is processed in parallel.

Unsuccessful page requests for queries processed in parallel are reported in the Unsuccessful Page Requests field.

**Getpage request — Sequential**

The number of Getpage requests issued by sequential access requesters.

**Getpage request — Overflow sequential**

The number of sequential GETPAGE requests using overflowed buffers (QBSTASGE).

**Getpage request — Random**

The number of random Getpage requests.

**Getpage request — Overflow random**

The number of non-sequential GETPAGE requests using overflowed buffers (QBSTAGET).

**Getpage request — Unsuccessful**

The number of times a conditional GETPAGE request could not be satisfied for this buffer pool during the specified time interval. This counter is used only when queries are processed in parallel. If the value is close to zero, most pages are already prefetched into the buffer pool and wait time for synchronous I/O is small. The field name is QBSTNGT.

**Getpage request — Unsucc seq getpages**

The total number of sequential getpage requests, which failed because the page was not in the buffer pool.

**Read — Synchronous read**

The number of synchronous read I/O operations performed by Db2 for applications and utilities.

**Read — Synchronous read — Sequential ***

The number of synchronous read I/O requests issued by sequential access requesters.

**Read — Synchronous read — Overflow sequential**

The number of synchronous read I/O operations for sequential GETPAGE requests using overflowed buffers (QBSTASSE).

**Read — Synchronous read — Random**

The number of random synchronous read I/O requests.

**Read — Synchronous read — Overflow random**

The number of synchronous read I/O operations for non-sequential GETPAGE requests using overflowed buffers (QBSTASYN).

**Read — Sequential prefetch — Request**

The number of sequential prefetch requests. This counter is incremented for each prefetch request (which can result in an I/O read). If it results in an I/O read, up to 32 pages may be read for SQL, and up to 64 pages for utilities. A request does not result in an I/O read if all pages to be prefetched are already in the buffer pool.

Sequential detection is not included in this counter but is separately recorded in the Dynamic Prefetch - Requested field.

**Read — Sequential prefetch — Read**
The number of asynchronous read I/O operations due to normal sequential prefetch (applications and utilities).

**Read — Sequential prefetch — Pages read**
The total number of pages read due to a normal sequential prefetch. A sequential prefetch request does not result in a read I/O if all the requiredd pages are found in the buffer pool.

**Read — Sequential prefetch — Pages read/read**
The number of sequential prefetch pages read per sequential prefetch read I/O operation.

**Read — List Prefetch — Request**
The number of list sequential prefetch requests.

List sequential prefetch allows Db2 to access data pages efficiently even when the required data pages are not contiguous. It allows CP and I/O operations to be overlapped.

**Read — List Prefetch — Read**
The number of asynchronous read I/O operations caused by the list sequential prefetch.

The number of pages read is recorded in the List Prefetch Pages Read field.

**Read — List Prefetch — Pages read**
The number of pages read due to a list prefetch. A list sequential prefetch request does not result in a read I/O if all the requiredd pages are found in the buffer pool.

**Read — List Prefetch — Pages read/read**
The number of list prefetch pages read per list prefetch read I/O.

**Read — Dynamic Prefetch — Request**
The number of dynamic prefetch requests. Dynamic prefetch is the process that is triggered because of sequential detection. If the prefetch request results in an I/O read, up to 32 advancing pages may be read at a time.

**Read — Dynamic Prefetch — Read**
The number of asynchronous read I/Os because of dynamic prefetch. The number of pages read is recorded in the Dynamic Prefetch Pages Read field.

**Read — Dynamic Prefetch — Pages read**
The number of pages read because of dynamic prefetch. Dynamic prefetch is the process that is triggered because of sequential detection.

**Read — Dynamic Prefetch — Pages read/read**
The number of dynamic prefetch pages read per dynamic prefetch read I/O.

**Read — Prefetch disabled — No buffer ***
The total number of times sequential prefetch was disabled or canceled because buffers were not available. This is the number of times the sequential prefetch threshold (SPTH) is reached. Ideally, this value should be 0. If the threshold is constantly reached, you need to identify the objects that are monopolizing the buffer pool.

The sequential prefetch threshold (SPTH) is a fixed threshold, set to 90% of the virtual pool size, that is compared prior to a sequential prefetch. If the threshold is reached, the prefetch is disabled.

**Read — Prefetch disabled — No read engine ***
The total number of times a sequential prefetch is disabled because of an unavailable read engine.

**Read — Page-ins required ***
The number of page-ins required for a read I/O.

If the number of Page-ins required is roughly approximately 5% of the total number of Getpage requests, the paging activity is at an acceptable rate. A rate near zero might indicate that the buffer pool is oversized.

**Write — Buffer updates**
The number of times buffer updates were requested against pages in the buffer pool.

**Write — Page write**
The number of pages in the buffer pool written to a hard disk drive.

**Write — Updates/page write**

The number of buffer update requests, divided by the number of pages written from the buffer pool to a hard disk drive.

The ratio of BUFFER UPDATES to PAGES WRITTEN suggests a high level of efficiency as the ratio increases, because more updates are being externalized per physical write. For example, if there are 10 updates on the same page before it is externalized, then the ratio is 10:1 or 10. If all 10 updates are on 10 distinct pages, then the ratio is 10:10 or 1.

**Write — Synchronous write**

The total number of immediate writes.

Immediate writes occur when:

- An immediate write threshold is reached
- No deferred write engines are available
- More than two checkpoints pass without a page being written

Immediate writes are a type of synchronous write, but not the only one. Sometimes Db2 uses synchronous writes even when the immediate write threshold (IWTH) is not exceeded, for example when more than two checkpoints pass without a page being written. This type of situation does not indicate a buffer shortage.

The immediate write threshold (IWTH) is a fixed threshold, set to 97.5% of the virtual pool size, that is checked whenever a page needs to be updated. If the threshold is reached, writes are synchronous. Then, the application cannot proceed until the write operation has completed.

**Write — Asynchronous write**

The number of asynchronous write I/O operations performed by media manager to a direct access storage device.

**Write — Pages/write req**

The number of pages written from the buffer pool to a hard disk drive per synchronous or asynchronous write I/O. This count does not include preformatting I/O, such as I/O needed to prepare a data set for use.

**Write — Page-ins required ***

The number of page-ins required for a write I/O.

This counter is incremented each time the media manager does not find a page in central storage. This counter does not differentiate between expanded storage and page data sets.

**Merge — Pass requested**

The total number of merge passes for Db2 sort activities. This value reflects how many merge passes were requested for Db2 to determine the number of work files permitted to support each merge pass.

**Merge — Pass degraded low buffer ***

The number of times that a merge pass was not efficiently performed due to a shortage of space in the buffer pool. The number in this field is incremented for each merge pass where the maximum number of work files allowed is less than the number of work files requested.

**Workfile — Max concurrent used**

The maximum number of work files concurrently used during merge processing within this statistics period.

Ideally, each work file needs 16 buffers to allow Db2 to perform a sequential prefetch for work files.

**Workfile — Req rejected low buffer ***

The total number of work files that were rejected during all merge passes because of insufficient buffer resources.

**Workfile — Req all merge passes**

The total number of work files requested for all merge passes.

This field and the merge passes requested field can be used to determine the average number of work files requested in one single merge pass.

For Db2 to perform an efficient prefetch for work files, each work file should have at least 16 dedicated buffers. Work files used during sort phase processing or other non-sort-related processing are not included in this number.

**Workfile — Not created no buffer ***
Only applicable if Db2 is running under MVS/XA. The number of times a work file cannot be created due to insufficient buffer resources. It indicates that a sort is in progress and limited in regard to the number of work files it can use.

**Workfile — Prefetch not scheduled ***
The number of times a sequential prefetch was not scheduled for a work file because the dynamic prefetch quantity is zero.

**Workfile — Pages to destruct**
The number of pages for which a destructive read was requested.

**Workfile — Pages not written**
The number of pages removed from the data set deferred write queue for destructive Read requests.

**Unlock castout**
The number of times DB2 issued an unlock request to the coupling facility for completed castout I/Os.

**Unlock castout — I/O operations**

**Unlock castout — Pages written**

**Simulated BP Activity — Current pages in use**
The total number of pages currently in the simulated buffer pool.

**Simulated BP Activity — Max pages in use**
The highest number of pages in the simulated buffer pool.

**Simulated BP Activity — Current seq pages in use**
The total number of sequential pages in the simulated buffer pool.

**Simulated BP Activity — Max seq pages in use**
The highest number of sequential pages in the simulated buffer pool.

**Simulated BP Activity — Avoidable read I/O**

**Sync read I/O (R)**
The total number of pages found in the simulated buffer pool for a random request that resulted in a synchronous read I/O.

**Sync read I/O (S)**
The total number of pages found in the simulated buffer pool for a sequential request that resulted in a synchronous read I/O.

**Async read I/O**
The total number of pages found in the simulated buffer pool for a prefetch request that resulted in an asynchronous read I/O.

**Sync GBP reads (R)**
The total number of pages found in the simulated buffer pool for a random request that resulted in a synchronous GBP read.

**Sync GBP reads (S)**
The total number of pages found in the simulated buffer pool for a sequential request that resulted in a synchronous GBP read.

**ASync GBP reads**
The total number of pages found in the simulated buffer pool for a prefetch request that resulted in an asynchronous GBP read.

**Pages moved into sim BP**
The total number of pages logically moved into the simulated buffer pool.

**Total avoidable - sync I/O delay (msec)**
The total time in milliseconds waiting for synchronous read I/O for pages found in the simulated buffer pool.

## The Data Set Statistics section

The data set statistics values are retrieved from IFCID 199.

```
1   OMEGAMON XE FOR DB2 PE (V5R4M0) - BUFFER POOL ACTIVITY REPORT    PAGE: 1-10
                             ORDER: BPID-QPAGESET
                SORTBY: BPID,ASYNCPAGE  TOP: 17  LEVEL: SUMMARY
 GROUP:     N/P         LOCATION:       OMPDC61            DB2 VERSION: V12
 MEMBER:    N/P         REQUESTED FROM: NOT SPECIFIED       TO: NOT SPECIFIED
 SUBSYSTEM: DC61        INTERVAL FROM:  04/05/16 13:40:56   TO: 04/05/16 14:47:00

                    =======     Dataset Statistics      =======
 BPID                  BP0        BP0        BP0        BP0        BP0
 PSTYPE                  T          I          T          T          I
 QPAGESET         DSNDB01    DSNDB01    DSNDB06    DSNDB06    DSNDB06
                 SYSSPUXB   DSNSPEXA   SYSTSTSP   SYSTSSFB   DSNDCX06
 PARTITION               1          1          1          1          1
 -------------- ---------- ---------- ---------- ---------- ----------
 Synchronous
  Request              202        134         62         61         60
  Avg delay ms       0.969      0.866      0.763      0.789      2.074
  Max delay ms      21.957     13.411      8.566      9.661     35.771
  Tot delay sec      0.196      0.116      0.047      0.048      0.124
 Asynchronous
  Request                4          5         61         48         23
  Page                  70        132        400        452         27
  Avg delay ms       0.050      0.111      0.524      0.445     11.201
  Max delay ms       0.376      0.329     39.198     17.518    171.133
  Tot delay sec      0.000      0.001      0.032      0.021      0.258
 VP current
  Avg pages            201        134         15         59          4
  Max pages            201        134         15         59          4
 VP changed
  Avg pages              0          0          1          2          1
  Max pages              0          0          1          2          1
 GBP-dependent          No         No         No         No         No
 Shadow DS              No         No         No         No         No
 ============== ========== ========== ========== ========== ==========

 BPID                BP32K
 PSTYPE                  T
 QPAGESET          DSNDB01
                     SPT01
 PARTITION               1
 -------------- ----------
 Synchronous
  Request               78
  Avg delay ms       4.413
  Max delay ms      33.024
  Tot delay sec      0.344
 Asynchronous
  Request               52
  Page                 139
  Avg delay ms       0.743
  Max delay ms       9.071
  Tot delay sec      0.039
 VP current
  Avg pages            132
  Max pages            132
 VP changed
  Avg pages              0
  Max pages              0
 GBP-dependent          No
 Shadow DS              No
 ============== ==========

1   OMEGAMON XE FOR DB2 PE (V5R4M0) - BUFFER POOL ACTIVITY REPORT    PAGE: 1-11
                             ORDER: BPID-QPAGESET
                SORTBY: BPID,ASYNCPAGE  TOP: 17  LEVEL: SUMMARY
 GROUP:     N/P         LOCATION:       OMPDC61            DB2 VERSION: V12
 MEMBER:    N/P         REQUESTED FROM: NOT SPECIFIED       TO: NOT SPECIFIED
 SUBSYSTEM: DC61        INTERVAL FROM:  04/05/16 13:40:56   TO: 04/05/16 14:47:00

                    =======     Dataset Statistics      =======
 BPID                 BP8K4
 PSTYPE                  T
 QPAGESET           EDVADB
                   EDVATSA
 PARTITION               1
 -------------- ----------
 Synchronous
  Request             429
  Avg delay ms      0.327
  Max delay ms      2.724
  Tot delay sec     0.140
 Asynchronous
  Request              83
  Page                948
  Avg delay ms      0.068
  Max delay ms      0.364
  Tot delay sec     0.006
 VP current
  Avg pages           450
  Max pages           450
 VP changed
  Avg pages             0
  Max pages             0
 GBP-dependent         No
 Shadow DS             No
 ============== ==========

 BPID              BP16K4
 PSTYPE                 T
 QPAGESET          EDVADB
                  EDVATSA
 PARTITION              2
 -------------- ----------
 Synchronous
  Request             525
  Avg delay ms      0.337
  Max delay ms      2.227
  Tot delay sec     0.177
 Asynchronous
  Request             136
  Page                910
  Avg delay ms      0.120
  Max delay ms      1.066
  Tot delay sec     0.016
 VP current
  Avg pages           505
  Max pages           505
 VP changed
  Avg pages            45
  Max pages            45
 GBP-dependent         No
 Shadow DS             No
 ============== ==========

1   OMEGAMON XE FOR DB2 PE (V5R4M0) - BUFFER POOL ACTIVITY REPORT    PAGE: 1-12
                              ORDER: BPID-QPAGESET
                 SORTBY: BPID,ASYNCPAGE  TOP: 17  LEVEL: SUMMARY
 GROUP:     N/P         LOCATION:       OMPDC61             DB2 VERSION: V12
 MEMBER:    N/P         REQUESTED FROM: NOT SPECIFIED       TO: NOT SPECIFIED
 SUBSYSTEM: DC61        INTERVAL FROM:  04/05/16 13:40:56   TO: 04/05/16 14:47:00

                         **********  TOTAL  **********
 BPID                 BP0      BP32K      BP8K4     BP16K4
 -------------- ---------- ---------- ---------- ----------
 Synchronous
  Request             519         78        429        525
  Avg delay ms      1.024      4.413      0.327      0.337
  Max delay ms     35.771     33.024      2.724      2.227
  Tot delay sec     0.532      0.344      0.140      0.177
 Asynchronous
  Request             141         52         83        136
  Page               1081        139        948        910
  Avg delay ms      2.211      0.743      0.068      0.120
  Max delay ms    171.133      9.071      0.364      1.066
  Tot delay sec     0.312      0.039      0.006      0.016
 VP current
  Avg pages            83        132        450        505
  Max pages           201        132        450        505
 VP changed
  Avg pages             1          0          0         45
  Max pages             2          0          0         45
 GBP-dependent         No         No         No         No
 Shadow DS             No         No         No         No
 ============== ========== ========== ========== ==========

⋮
```

**Synchronous — Request**
Number of synchronous I/Os in the reported interval for the page set.

**Synchronous — Avg delay ms**
Average synchronous I/O delay for pages in the page set, in milliseconds.

**Synchronous — Max delay ms**
Maximum synchronous I/O delay for pages in the page set, in milliseconds.

**Synchronous — Tot delay sec**
Total accumulated synchronous I/O delay for pages in the page set, in seconds.

**Asynchronous — Request**
Number of asynchronous I/Os for the page set in the reported interval.

**Asynchronous — Page**
Number of page set pages read or written asynchronously in the reported interval.

**Asynchronous — Avg delay ms**
Average asynchronous I/O delay for pages in the page set, in milliseconds.

**Asynchronous — Max delay ms**
Maximum asynchronous I/O delay for pages in the page set, in milliseconds.

**Asynchronous — Tot delay ms**
Total accumulated asynchronous I/O delay for pages in the page set, in seconds.

**VP current**
Number of page set pages in the virtual buffer pool.

**VP current— Avg pages**
Average number of page set pages in the virtual buffer pool per IFCID 199 interval.

**VP current— Max pages**
Maximum number of page set pages in the virtual buffer pool per IFCID 199 interval.

**VP changed**
Number of changed page set pages in the virtual buffer pool.

**VP changed— Avg pages**
Average number of changed page set pages in the virtual buffer pool per IFCID 199 interval.

**VP changed — Max pages**
Maximum number of changed page set pages in the virtual buffer pool per IFCID 199 interval.

## The Group Buffer Pools Activity Data section

This report section shows group buffer pool activity data, which is retrieved from IFCID 2 (Db2 statistics — Group Buffer Pool Activity data).

```
1     OMEGAMON XE FOR DB2 PE (V5R4M0) - BUFFER POOL ACTIVITY REPORT    PAGE: 1-9
                                ORDER: BPID-QPAGESET
                  SORTBY: BPID,ASYNCPAGE  TOP: 17  LEVEL: SUMMARY
 GROUP:     DSNJ        LOCATION:       PMODSNJ            DB2 VERSION: V11
 MEMBER:    SGJ1        REQUESTED FROM: NOT SPECIFIED       TO: NOT SPECIFIED
 SUBSYSTEM: SGJ1        INTERVAL FROM:  01/24/13 07:21:46   TO: 01/24/13 09:27:13

              =======     Group Buffer Pools Activity Data    =======
 Group Buffer Pool                         GBP0       GBP2     GBP8K0
 ---------------------------------- ---------- ---------- ----------
 Group BP Hit Ratio (%)                   28.57       0.00       0.00
 GBP-Dependent Getpages                       8          2          0
 Syn.Read(XI)-Data returned                   2          0          0
 Syn.Read(XI)-No data return                  5          1          0
 Syn.Read(NF)-Data returned                   0          0          0
 Syn.Read(NF)-No data return                  0          0          0
 Pages written using write-around             0          0          0
 Clean pages sync. written                    0          0          0
 Changes pages sync. written                 10          3          0
 Clean pages async. written                   0          0          0
 Changes pages async. written                 0          0          0
 Reg.Page List(RPL) request                   0          0          0
 Clean pages after RPL                      n/p        n/p        n/p
 Pages retrieved from GBP after RPL           0          0          0
```

```
  Async. read-No data return                n/p        n/p        n/p
  Async. read-Data returned                 n/p        n/p        n/p

  Castout class threshold                     0          0          0
  Group BP castout threshold                  0          0          0

  Pages castout                               7          1          0
  Unlock castout                              7          1          0
  Read castout class                          6          2          0
  Read castout statistics                    16         16         16
  RFCOM requests                              0          0          0
  RFCO requests                               7          1          0
  Read directory info                         0          0          0
  Read storage statistics                   224        224        224
  WAR requests                               10          1          0
  WARM requests                               0          1          0
  Pages written via WARM                      0          2          0
  Register page                               6          2          0
  Unregister Page                             0          0          0
  Delete name                                 2          1          0
  Asynch. GBP requests                       16         17         16
  Explicit X-invalidations                    0          0          0
  GBP checkpoints triggered                   8          8          8

  Participation GBP rebuild                 n/p        n/p        n/p
  Castout engine not available              n/p        n/p        n/p
  Write engine not available                n/p        n/p        n/p
  Read failed-no storage                    n/p        n/p        n/p
  Write failed-no storage                     0          0          0

  Write to secondary GBP                    n/p        n/p        n/p
  Write to secondary GBP failed               0          0          0
  Delete name list secondary GBP              0          0          0
  Delete name from secondary GBP              0          0          0
  Read castout stat. secondary GBP            0          0          0
  Asynch. secondary GBP requests              0          0          0
  =================================== ========== ========== ==========
 ⋮
```

Asterisks (*) beside elements denote those elements that can show up in "The Group Buffer Pools Activity Data Highlights section" on page 50.

**Group BP Hit Ratio (%) ***
The group buffer pool hit ratio, expressed as a percentage. This is the percentage of pages successfully retrieved from the group buffer pool to those retrieved from a hard disk drive. Derived from the Db2 field SGBPHITP.

Calculated as:

```
(qbglxd + qbglmd + qbglad + qbglay + qbglaz)
-------------------------------------------- * 100
         (qbglxr + qbglmr + qbglar)
```

**GBP-Dependent Getpages**
The number of Getpage requests made for GBP-dependent objects. Derived from the Db2 field QBGLGG.

**Syn.Read(XI)-Data returned**
The number of requests made to read a page from the group buffer pool because the page was invalidated in the member's buffer pool. The member found the required page in the group buffer pool. Derived from the Db2 field QBGLXD.

When you increase the size of the group buffer pool, the number of pages returned from the GBP can increase. Conversely, decreasing the size of the GBP can cause Db2 to return fewer pages because the GBP cannot hold pages long enough to allow them to be retrieved again.

**Syn.Read(XI)-No data return**
The number of requests to read a page from the group buffer pool that were required because the page was invalidated in the member's buffer pool. The member did not find the data in the group buffer pool and had to retrieve the page from a hard disk drive. Derived from the Db2 field SBGLXR.

Normally, when the page in a member's buffer is cross-invalidated, the buffer is refreshed from the group buffer pool. In this instance, the requested page was not found in the group buffer pool though the page set is still GBP-dependent. The page has been removed from the group buffer pool for one of two reasons:

- Shortage of data pages and consequent reclamation of this page
- Shortage of directory entries and consequent removal of the page together with cross-invalidation (XI) of that page in the local buffer pools of all members using that page

If the value in this field is high, you may want to tune the group buffer pool (GBP). Depending on the reason, increase the number of GBP data pages, increase the size of the directory entry space, or increase both the number of GBP data pages and the space for directory entries. Oversizing the group buffer pool can cause unnecessary GBP checkpoint overhead.

**Syn.Read(NF)-Data returned**

The number of requests made to read a page from the group buffer pool because the page was not in the member's buffer pool (NF = page not found). The member found the page in the group buffer pool. Derived from the Db2 field QBGLMD.

The requesting member needs a page from a table space or index space that is GBP-dependent or has GBPCACHE ALL defined. To get that page, the group buffer pool is checked before the page set on a hard disk drive.

If the group buffer pool is used to cache both clean and changed pages (GBPCACHE ALL is used for all data), you can try to get more pages returned from the group buffer pool by increasing the size of the group buffer pool. Do not tune the GBP based on this counter if it is used for caching changed pages only (GBPCACHE CHANGED).

**Syn.Read(NF)-No data return**

The number of requests made to read a page from the group buffer pool because the page was not in the member's buffer pool (NF = page not found). The member did not find the required data in the group buffer pool and had to retrieve the page from a hard disk drive. Derived from the Db2 field SBGLMR.

The requesting member needs a page from a table space or index space that is GBP-dependent or has GBPCACHE ALL defined. To get that page, the group buffer pool is checked before the page set on a hard disk drive.

You can compare the value in this counter with the number of pages that were returned from the group buffer pool, see "Sync.Read (Not Found) - Data Returned". If the group buffer pool is used to cache both clean and changed pages (GBPCACHE ALL is used for all data), you can try to get more pages returned from the group buffer pool by increasing the size of the group buffer pool. Do not tune the GBP based on this counter if it is used for caching changed pages only (GBPCACHE CHANGED).

**Pages written using write-around**

The total number of pages in write around. This means the pages written by DB2 to DASD directly from the local buffer pools thus eliminating page placement to GBP and associated overhead. Derived from the DB2 field QBGLWA.

**Clean pages sync. written**

The number of clean pages that were synchronously written to the group buffer pool from the virtual pool. Derived from the Db2 field QBGLWC.

Only GBPCACHE ALL causes clean (unchanged) pages to be written to the coupling facility. The pages are written to the coupling facility even when the page set is not GBP-dependent. When group buffer pool caching works effectively for prefetch, the value in this field should be much smaller than the value in "Synchronous Read (Not Found) - Data Returned".

**Changed pgs.sync. written**

The number of changed pages written synchronously to the group buffer pool. Pages are written with Write and Register (WAR) requests or Write and Register Multiple (WARM) requests. At commit time changed pages are forced from the member's virtual buffer pool to the coupling facility. For duplexed GBPs the counter values reflect writes to both primary and secondary group buffer pools. Derived from the Db2 field QBGLSW.

In data sharing, changed pages must have been written to the group buffer pool by the time a transaction commits. The pages are written either synchronously (force at commit) or asynchronously, for example, when a local buffer pool threshold is reached or at a member's checkpoint. The number of pages that have to be forced out synchronously (in "burst mode") at commit time can be reduced if asynchronous writes are triggered more frequently.

You can use the vertical deferred write threshold (VDWQT) to reduce the number of pages that have to be forced out synchronously and to increase the number of pages that are asynchronously written before the transaction commits. For GBP-dependent page sets, writes triggered by the vertical deferred write threshold go to the coupling facility. You can cause changed pages to be written out quicker and in smaller increments, by reducing the vertical deferred write threshold (VDWQT).

**Clean pages async. written**

Not applicable for versions later than Db2 Version 7. The number of clean pages that were asynchronously written to the group buffer pool from the virtual pool. Derived from the Db2 field QBGLAC.

Only GBPCACHE ALL causes clean (unchanged) pages to be written to the group coupling facility. In this instance pages are written even when the page set is not GBP-dependent. Asynchronous write is done under prefetch processing.

When group buffer pool caching works effectively for prefetch, the value in this field should be much smaller than the combined values in

- "Synchronous Read (Not Found) - Data Returned"
- "Asynchronous Reads - Data Returned" and
- "Clean pages - Read after Register Page List (RPL)"

**Changed pages async. written**

The number of changed pages written asynchronously to the group buffer pool. Pages are written in response to Write and Register (WAR) and Write and Register Multiple (WARM) requests. Changed pages can be written from the member's virtual buffer pool to the group coupling facility before the application commits. This happens when, for example, a local buffer pool threshold is reached, or when P-lock negotiation forces the pages on the vertical deferred write queue to be written to the group buffer pool. For duplexed GBPs the counter values reflect writes to both primary and secondary group buffer pools. Derived from the Db2 field QBGLAW.

In data sharing, changed pages must have been written to the group buffer pool before a transaction commits. The pages are written either synchronously during commit processing or asynchronously before the transaction commits when, for example, a local buffer pool threshold is reached or at a member's checkpoint. See Changed Pages - Written Synchronously for the number of changed pages synchronously written to the group buffer pool.

The vertical deferred write threshold (VDWQT) can be used to reduce the number of pages that have to be forced out synchronously and to increase the number of pages that are asynchronously written before the transaction commits. For GBP-dependent page sets, writes triggered by the vertical deferred write threshold go to the coupling facility. If you want changed pages to be written out quicker and in smaller increments, you can lower the vertical deferred write threshold (VDWQT).

**Reg.Page List (RPL) request**

The number of Register Page List (RPL) requests made by prefetch. The group buffer pool must be allocated in a group coupling facility with CFLEVEL=2 or higher. Derived from the Db2 field QBGLAX.

Performance might be improved by enabling RPL.

**Clean pages read after RPL**

Not applicable for versions later than Db2 Version 7. The number of coupling facility reads performed by prefetch to retrieve a clean page from the group buffer pool. Derived from the Db2 field QBGLAZ.

**Castout class threshold**

The number of times group buffer pool castout was initiated because the group buffer pool class castout threshold was detected. Derived from the Db2 field QBGLCT.

The class castout threshold is one of two group buffer pool thresholds. In most cases the default value for the class threshold (10 percent) is a good choice. Depending on your workload, altering this value can reduce hard disk drive contention during castout.

**Group BP castout threshold**

The number of times a group buffer pool castout was initiated because the group buffer pool castout threshold was detected. Derived from the Db2 field QBGLGT.

The GBP castout threshold, together with the GBP class castout threshold and the length of the GBP checkpoint interval determine the castout characteristics of the group buffer pool.

You can consider this threshold a safety margin to protect the group buffer pool from being accidentally flooded by overactive applications.

In most situations, the default value for the group buffer pool castout threshold of 50 percent is a good choice. Use the ALTER GROUPBUFFERPOOL command to tune the group buffer pool thresholds.

**Pages castout**

The number of data pages that were cast out of the member's group buffer pool. Castout to a page set or partition is done by the castout owner of the page set or partition. This is normally the Db2 subsystem that had the first update intent on the page set or partition. Derived from the Db2 field QBGLRC.

The number of pages written per I/O is normally close to the value of this field divided by the value in "Unlock castout". For example, if an average of four pages is written per castout write I/O, the number of pages cast out should be four times the number in this field.

Because Db2 usually includes more than one page in the request to write pages to a hard disk drive, the number in this field should always be significantly more than "Unlock castout". If it is not (for example, when "Unlock castout" is more than half of "Pages castout"), the castout write I/O is inefficient; probably because you have random update patterns on the Db2 data or a low castout threshold.

**Unlock castout**

The number of times Db2 issued an unlock request to the coupling facility for completed castout I/Os. When pages are cast out to a hard disk drive, they are locked for castout in the coupling facility. This castout lock is not an IRLM lock; it is to ensure that only one system can cast out a given page at a time. Derived from the Db2 field QBGLUN.

The number of pages written per I/O is normally close to the value of "Pages castout" divided by the value of this field. For example, if an average of four pages is written per castout write I/O, the number of pages cast out should be four times the value in this field.

Because Db2 usually includes more than one page in a write request, the number in this field should always be significantly less than "Pages castout". If it is not (for example, when "Unlock castout" is more than half of "Pages castout"), the castout write I/O is inefficient; possibly because you have random update patterns on the Db2 data or a low castout threshold.

**Read castout class**

The number of requests made to the group buffer pool to determine which pages, from a particular page set or partition, must be cast out because they are cached as changed pages. Derived from the Db2 field QBGLCC.

This request is issued either by the page set or partition castout owner, or, when the group buffer pool castout threshold is reached, by the group buffer pool structure owner.

**Read castout statistics**

The number of requests issued by the group buffer pool structure owner to determine which castout classes have changed pages. Derived from the Db2 field QBGLCS.

This request is made by the group buffer pool structure owner when the group buffer pool threshold is reached. Normally, you would expect only one or two requests each time the group buffer pool threshold is reached.

**RFCOM requests**
The number of Read For Castout Multiple (RFCOM) requests. Derived from the Db2 field QBGLCM.

**RFCO requests**
The number of Read For Castout (RFCO) requests. One page read per request. Derived from the Db2 field QBGLCR.

**Read directory info**
The number of requests issued by the group buffer pool structure owner to read the directory entries of all changed pages in the group buffer pool. This request is issued at group buffer pool checkpoints to record the oldest recovery log record sequence number (LRSN). It is used as a basis for recovery if the group buffer pool fails. Such requests might have to be issued several times for each group buffer pool checkpoint to read the directory entries for all changed pages. Derived from the Db2 field QBGLRD.

If the value of this counter seems to be abnormally high, consider upgrading the coupling facility to CFLEVEL=2 or higher to raise the number of directory entries that can be read with one request. You can also increase the group buffer pool checkpoint interval, but this can lengthen the recovery for the group buffer pool.

**Read storage statistics**
The number of times DB2 requested statistics information from the group buffer pool. It is issued by the group buffer pool structure owner at timed intervals to determine whether the group buffer pool castout threshold (GBPOOLT) has been reached. Derived from the Db2 field QBGLOS.

**WAR requests**
The number of Write and Register (WAR) requests. Derived from the Db2 field QBGLWS.

**WARM requests**
The number of Write and Register Multiple (WARM) requests. Derived from the Db2 field QBGLWM.

**Pages written via WARM**
Not applicable to Db2 Version 7 and earlier. (With Db2 Version 8, the group coupling facility allows multiple pages to be written and registered with a single write request.) The number of pages written using Write and Register Multiple (WARM) requests. Derived from the Db2 field QBGLWP.

**Register page**
The number of times DB2 registered interest in a single page. These are "register-only" requests, which means that Db2 is not requesting any data back from the request. This request is made only to create a directory entry for the page to be used for cross-invalidation when the page set or partition P-lock is downgraded from S to IS mode, or from SIX to IX mode. Derived from the Db2 field QBGLRG.

**Unregister Page**
The number of times Db2 unregistered interest for a single page. This happens when Db2 steals pages from the member's buffer pool that belong to GBP-dependent page sets or partitions. Derived from the Db2 field QBGLDG.

A large value here indicates that the local buffer pool contains a mixture of GBP-dependent data and non-GBP-dependent data.

The page stolen from the local buffer pool is replaced by a new one. This counter makes a distinction on whether the new page depends on the group buffer pool or not.

Usually a page of a GBP-dependent page set or partition is replaced by a page that is also GBP-dependent. In this instance, the unregister request for the page being stolen is combined with the read and register request for the new page. These combined requests do not contribute to this counter.

If, however, a page of a GBP-dependent page set or partition is replaced by a page that is not GBP-dependent, then only an unregister request is sent to the coupling facility. These separate requests are counted here.

**Delete name**
The number of requests made by Db2 to delete directory and data entries associated with a particular page set or partition from the group buffer pool. Db2 issues this request when it changes a page set or partition from GBP-dependent to non-GBP-dependent. Db2 also issues this request for objects

that are defined with GBPCACHE ALL when those objects are first opened. Derived from the Db2 field QBGLDN.

This counter is a measure of how often page sets or partitions change between being and not being dependent on the group buffer pool. You can prevent Db2 going in and out of GBP-dependency too often by tuning the following subsystem parameters that affect when data sets are switched to a different state:

**PCLOSEN**
Pseudoclose frequency. The number of checkpoints required before a data set that was not updated can be a pseudoclose candidate.

If the PCLOSEN condition is met, the page set or partition is converted from read-write to read-only state. Depending on other concurrent users, this could raise the chance for the page set or partition to go out of GBP-dependency.

**PCLOSET**
Pseudoclose time. The amount of time (in minutes) that must elapse before a data set can be a pseudoclose candidate.

If the PCLOSEN or PCLOSET condition is met, the page set or partition is converted from read-write to read-only state. Depending on other concurrent users, this could raise the chance for the page set or partition to go out of GBP-dependency.

**LOGLOAD**
The number of log records that Db2 writes between successive checkpoints.

These parameters are specified in the CHECKPOINT FREQ field in panel DSNTIPN.

**Asynch. GBP requests**
The number of IXLCACHE invocations for the primary group buffer pool. Derived from the Db2 field QBGLHS.

**Explicit X-invalidations**
The number of times an explicit coupling facility cross-invalidation request was issued. Derived from the Db2 field QBGLEX.

**GBP checkpoints triggered**
The number of group buffer pool checkpoints triggered by this member. Derived from the Db2 field QBGLCK.

The value of this counter depends on the length of the group buffer pool checkpoint interval.

**Write failed-no storage ***
The number of coupling facility write requests that could not complete due to a lack of coupling facility storage resources. Derived from the Db2 field QBGLWF.

A value greater than zero indicates that the data page resources of the coupling facility are being consumed faster than the Db2 castout processes can free them.

On write failure, the affected Db2 member initiates castout and retries several times, and finally, if it is a changed page, it will be added to the logical page list (LPL) requiring recovery.

On write failure, the affected DB2 member initiates castout and retries several times, and finally, if it is a changed page, it will be added to the logical page list (LPL) requiring recovery. If the problem is not simply due to a momentary surge in activity, you need either to decrease the group buffer pool castout thresholds, or to increase the number of data entries in the group buffer pool. To increase the number of data entries, you can:

- Increase the total size of the group buffer pool
- Adjust the ratio of directory entries to data entries in favor of data entries

**Write to secondary GBP failed ***
The number of coupling facility requests to write changed pages to the secondary group buffer pool for duplexing that failed because of a lack of storage in the coupling facility. Derived from the Db2 field QBGL2F.

**Delete name list secondary GBP**
The number of DELETE NAME LIST requests to delete pages from the secondary group buffer pool that have just been cast out from the primary. Derived from the Db2 field QBGL2D.

**Delete name from secondary GBP**
The number of group buffer pool requests to delete a page from the secondary group buffer pool. These requests are issued by the group buffer pool structure owner to delete orphaned data entries in the secondary GBP as part of the garbage collection logic. Derived from the Db2 field QBGL2N.

**Read castout statistics secondary GBP**
The number of coupling facility requests to read the castout statistics for the secondary group buffer pool. These requests are issued by the group buffer pool structure owner to check for orphaned data entries in the secondary group buffer pool. Derived from the Db2 field QBGL2R.

**Asynch. secondary GBP requests**
The number of asynchronous IXLCACHE invocations for the secondary group buffer pool. Derived from the Db2 field QBGL2H.

## The Group Buffer Pool Attributes section

This report section shows group buffer pool attributes, which are retrieved from IFCID 230.

```
             =======     Group Buffer Pool Attributes   =======
 Group Buffer Pool                  GBP0       GBP2     GBP8K0
 --------------------------- ---------- ---------- ----------
 Allocated GBP size (4K)            768        768        768
 Current dir.to data ratio            5          5          5
 Class castout thresh. (%)            5          5          5
 Class castout thresh. (#)            0          0          0
 Actual nbr. of dir. entrs         1450       1450        621
 Pending dir. to data ratio           5          5          5
 GBP castout thresh. (%)             30         30         30
 Actual nbr. of data entrs          287        287        124
 Checkpoint interval (min)            4          4          4
 Autorec                            yes        yes        yes
 Directory-entry-reclaim              0          0        318
 Data-entry-reclaim                8154        867       1574
 GBP cache                          yes        yes        yes
 Total-changed                        3          2          0
 XI-dir.-entry-reclaim                0          0        321
 Mode                           SIMPLEX    SIMPLEX    SIMPLEX
 Secondary-GBP
  Alloc GBP size (4K)               n/a        n/a        n/a
  Directories entries               n/a        n/a        n/a
  Data entries                      n/a        n/a        n/a
 =========================== ========== ========== ==========
⋮
```

**Allocated GBP size (4K)**
The allocated size of the group buffer pool in 4 KB blocks. Derived from the Db2 field QBGBGSZ.

**Current dir. to data ratio**
The current directory entry per data entry ratio. Derived from the Db2 field QBGBGR1.

**Class castout thresh. (%)**
The threshold at which class castout is to be initiated. It is expressed as a percentage of the group buffer pool size. Derived from the Db2 field QBGBGCT.

If the GBP castout thresholds are not set correctly, castout processing is not kept in pace with the changed pages written to the CF (coupling facility). You can either use the ALTER GROUPBUFFERPOOL command to reduce the castout thresholds, or increase the number of GBP data pages by increasing the GBP size or by reducing the GBP RATIO.

**Class castout thresh. (#)**
Class level castout threshold (buffer num). Derived from the Db2 field QBGBGCTN.

**Actual nbr. of dir. entrs**
The actual number of allocated directory entries. Derived from the Db2 field QBGBGDR.

**Pending dir. to data ratio**
The pending directory entry per data entry ratio. Derived from the Db2 field QBGBGR2.

**GBP castout thresh. (%)**
The threshold at which castout is to be initiated for the group buffer pool. It is expressed as a percentage of the size of the group buffer pool. Derived from the Db2 field QBGBGGT.

**Actual nbr. of data entrs**
The actual number of allocated data entries. Derived from the Db2 field QBGBGDT.

**Checkpoint interval (min)**
The time interval (in minutes) between successive group buffer pool checkpoints. Derived from the Db2 field QBGBGCK.

**Autorec**
A flag indicating how the AUTOREC option of the ALTER GROUPBUFFERPOOL command has been set. It specifies whether automatic recovery takes place in case of a structure failure or loss of connectivity of all members of the group buffer pool. Derived from the Db2 field QBGBGAS.

**Directory-entry-reclaim**
The number of times that a page name assignment required that a coupling facility directory entry be reclaimed (stolen). Derived from the Db2 field QBGBDRR.

**Data-entry-reclaim**
The number of times that a page name assignment required that a coupling facility data entry be reclaimed (stolen). Derived from the Db2 field QBGBDTR.

**GBP cache**
Caching attribute. Possible values are:

- YES: The GBP is used for both data caching and cross-invalidation.
- NO: The GBP is used only for cross-invalidation.

**Total changed**
The number of allocated data entries that are currently in "changed" state. This is a snapshot value and is not cumulative. Derived from the Db2 field QBGBTCC.

**XI-dir.-entry-reclaim**
The number of times that a directory entry was stolen and one or more XI signals had to be sent because the page in the directory was cached in one or more Db2 buffer pools. Derived from the Db2 field QBGBRXI.

**Mode**
Possible values are:

- DUPLEX
- SIMPLEX

Derived from the Db2 field QBGBDUP.

**Secondary-GBP — Alloc. GBP size (4K)**
When MODE is DUPLEX, the allocated size of the secondary group buffer pool. Derived from the Db2 field QBGBGSZ2.

**Secondary-GBP — Directories entries**
When MODE is DUPLEX, the number of allocated directory entries in the secondary group buffer pool. Derived from the Db2 field QBGBGDR2.

**Secondary-GBP — Data entries**
When MODE is DUPLEX, the number of allocated data entries in the secondary group buffer pool. Derived from the Db2 field QBGBGDT2.

## The Buffer Manager PSET/Part P-lock Request section

This section of the report provides information about physical lock (P-lock) activities in group buffer pools.

The Db2 Buffer Manager uses the P-lock mechanism to manage, negotiate, and resolve inter-DB2 R/W interests on a page set or partition level, that is, when programs running on different data sharing group members request incompatible locks on the same resource. The activity counter values are retrieved from IFCID 251.

```
   OMEGAMON XE FOR DB2 PE (V5R4M0) - BUFFER POOL ACTIVITY REPORT    PAGE: 1-91
                              ORDER: BPID-QPAGESET
                SORTBY: BPID,ASYNCPAGE  TOP: 17  LEVEL: SUMMARY
GROUP:     DBE1        LOCATION:       PMODBE1            DB2 VERSION: V11
MEMBER:    SE11        REQUESTED FROM: NOT SPECIFIED      TO: NOT SPECIFIED
SUBSYSTEM: SE11        INTERVAL FROM:  12/04/15 13:32:25  TO: 12/04/15 13:44:27

             =======     Buffer Manager PSET/Part P-lock Request    =======
BPID                 GBP0       GBP0       GBP0       GBP0       GBP0       GBP0
PSTYPE                  T          T          I          I          I          I
QPAGESET           DSNDB06    DSNDB04    DSNDB06    DSNDB06    DSNDB06    DSNDB06
                  SYSTSISS   BPTYUNZO   DSNADH01   DSNDDH01   DSNDDX02   DSNSFX01
PARTITION               1          0          0          0          0          0
-------------- ---------- ---------- ---------- ---------- ---------- ----------
IRLM func mode
 Lock                   2          2          2          2          2          2
 Unlock                 0          0          0          0          0          0
 Change                 1          0          0          0          0          0
 Change P-lock          0          0          0          0          0          0
New held state
  IS                    0          0          0          0          0          0
  IX                    0          0          0          0          0          0
  S                     2          1          1          1          1          1
  SIX                   1          0          1          1          1          1
  NSU                   0          0          0          0          0          0
  X                     0          1          0          0          0          0
  denied                0          0          0          0          0          0
Confl. member           0          0          0          0          0          0
New cach.state
  IS                    0          0          0          0          0          0
  IX                    0          0          0          0          0          0
  S                     2          1          1          1          1          1
  SIX                   1          0          1          1          1          1
  NSU                   0          0          0          0          0          0
  X                     0          1          0          0          0          0
  denied                0          0          0          0          0          0
Request type
 Condition              0          0          0          0          0          0
 Restart                0          0          0          0          0          0
 Modify                 0          0          0          0          0          0
============== ========== ========== ========== ========== ========== ==========
```

**IRLM func mode**

The number of requests to the data sharing group member's Internal Resource Lock Manager (IRLM) by lock type for an object. Note that an object is identified by BPID (buffer pool ID), PSTYPE (object type), QPAGESET (combination of database and page set), and partition number. Derived from the Db2 field QW0251IF.

- `Lock` requests
- `Unlock` requests
- `Change` requests by a member holding the lock because of a change of interest
- `Change P-lock` (short for `Change from P-lock Exit`) requests by another member causing the P-lock exit of this member

**New held state**

The number of new held P-lock states (the inter-DB2 interest level) determined for an object. Derived from the Db2 field QW0251NS.

- `IS` (Intent Shared): This Db2 has R/O interest in the object and one or more other Db2 members have R/W interest.
- `IX` (Intent Exclusive): This Db2 has R/W interest in the object and one or more other Db2 members have R/W interest.
- S (Shared): This Db2 has R/O interest in the object and no other Db2 member has R/W interest.
- `SIX` (Shared Intent Exclusive): This Db2 has R/W interest in the object and one or more other Db2 members have R/O interest.
- NSU (Non-Shared Update): Acts like an X lock, but is only used during P-lock negotiation from an X to an SIX.

- X (eXclusive): This Db2 has R/W interest in the object. No other Db2 member has declared interest.
- `denied`: A request was denied; the object had `Change P-lock` active.

Normally, Db2 holds the P-lock in the cached state (see later in this list). In some special or abnormal cases, the P-lock will not be held in the cached state. It is the actual held state of the P-lock that determines whether the object is GBP-depended. If the held state is S or X, the object is not GBP-depended. Otherwise, the object is GBP-depended.

**Confl. member**

The number of conflicts determined by the IRLM for an object. (IRLM function code of `Change From P-lock Exit` active). Derived from the Db2 field QW0251DB.

**New cach. state**

The number of cached P-lock states (the inter-DB2 interest level) determined for an object. Derived from the Db2 field QW0251NC. See also New held state.

- `IS`
- `IX`
- `S`
- `SIX`
- `NSU`
- `X`
- `denied`

**Request type**

The number of P-lock requests by request type for an object. Derived from the Db2 field QW0251F1.

- `Condition`: Conditional request.
- `Restart`: Restart lock request. Locks retained by a Db2 system are changed from retained to active.
- `Modify`: Modify lock request.

## The CF Cache Structure Statistics section

This report section shows coupling facility cache structure statistics, which are retrieved from IFCID 254.

The statistics break out the major activity details of the cache structure. A cache structure is a storage area that is used as group buffer pool for a Db2 data sharing group.

```
    OMEGAMON XE FOR DB2 PE (V5R4M0) - BUFFER POOL ACTIVITY REPORT    PAGE: 1-90
                              ORDER: BPID-QPAGESET
                SORTBY: BPID,ASYNCPAGE  TOP: 17  LEVEL: SUMMARY
GROUP:     DBE1        LOCATION:       PMODBE1              DB2 VERSION: V11
MEMBER:    SE11        REQUESTED FROM: NOT SPECIFIED       TO: NOT SPECIFIED
SUBSYSTEM: SE11        INTERVAL FROM:  12/04/15 13:32:25   TO: 12/04/15 13:44:27

                 =======     CF Cache Structure Statistics       =======
Group Buffer Pool              GBP0     GBP32K     GBP8K0
------------------------- ---------- ---------- ----------
Explicit XI counter               0          0          0
Read Hit                       3290          0          0
Read miss directory Hit        7298          0          0
Read miss assign.suppres.      3224        604        676
Read miss name assigned       37514        117        161
Read miss cache full              0          0          0
Changed page write Hit       105145        117       1872
Clean   page write Hit            0          0          0
Write miss cache full             0          0          0
Total changed                     1          0          0
Directory entry                 349          0         14
Data entry                        1          0         10
Directory entry reclaim           0        221          0
Date entry reclaim            54540          0        193
XI dir. entry reclaim             0        208          0
Castout                       59185        117        512
Secondary-GBP
  Directory entry                 0          0          0
  Data entry                      0          0          0
```

```
   Changed page write Hit           0         0         0
   Total changed                    0         0         0
   Write miss cache full            0         0         0
 ======================== ========== ========== ==========
```

**Explicit XI counter**
The number of times a request was made to the group coupling facility to explicitly cross-invalidate a page and a number of XI signals were sent because the page was cached in one or more Db2 buffer pools. Derived from the Db2 field QW0254CI.

**Read hit**
The number of coupling facility read requests in which data was returned. Derived from the Db2 field QW0254RH.

**Read miss directory hit**
The number of coupling facility read requests for a page in which data was not returned but the page name was already assigned in the coupling facility directory (SES did not have to assign a directory entry for the page). Derived from the Db2 field QW0254RD.

**Read miss assign. suppres.**
The number of times that a coupling facility read request specified a page for which no directory entry exists and no directory entry is created. Db2 does not create a directory entry if it does not need to register the page to the coupling facility for cross-invalidation (XI); that is when no other Db2 member in the group has R/W interest in the page set/partition. Derived from the Db2 field QW0254RS.

**Read miss name assigned**
The number of times that a coupling facility read request specified a page for which a directory entry was created. Derived from the Db2 field QW0254RN.

**Read miss cache full**
The number of times that a coupling facility read request specified a page for which no directory entry exists and no directory entry is created due to the lack of storage in the group buffer pool. A nonzero value in this field indicates that the backing coupling facility cache structure size might be too small to support the current workload. Derived from the Db2 field QW0254RF.

**Clean page write hit**
The number of facility write requests for clean pages successfully completed. Derived from the Db2 field QW0254WC.

**Changed page write hit**
The number of coupling facility write requests for changed pages that have successfully completed. Derived from the Db2 field QW0254WH.

**Write miss cache full**
The number of coupling facility write requests that could not complete due to a lack of coupling facility storage resources. Derived from the Db2 field QW0254WF.

Ideally, this value should be zero. These GBP write fails occur when a changed page must be written to the GBP and no GBP data entries are available. A data entry is unavailable if it contains a changed page that has not yet been externalized to a hard disk drive (casted out). A nonzero value indicates that castout processing cannot keep pace with the rate at which changed pages are being written to the group buffer pool. Usually, the best solution is to enlarge the GBP. Changing the castout threshold could lead to higher processor utilization for the mainframes handling castout processing.

**XI dir. entry reclaim**
The number of times that a directory entry was stolen and XI signals had to be sent because the page for the directory entry was cached in one or more Db2 buffer pools. Derived from the Db2 field QW0254XR.

**Directory entry reclaim**
The number of times that a page name assignment required a coupling facility directory entry to be reclaimed (stolen). Derived from the Db2 field QW0254DR.

Directory entry reclaims occur when a data or index page must be registered in the GPB but all the directory entries are in use. Then, an in-use entry will be reclaimed. When this happens, the copies of the page associated with the reclaimed directory entry are invalidated, even if they have not been changed. Invalidation causes extra reads from a hard disk drive, which can reduce system throughput. Consequently, larger group buffer pools reduce or eliminate directory entry reclaims because they can hold more entries. See also "Data entry reclaim".

**Data entry reclaim**
The number of times that a page name assignment required a coupling facility data entry to be reclaimed (stolen). Derived from the Db2 field QW0254TR.

See also "Directory entry reclaim". Besides the size of the group buffer pool, reclaims are also influenced by the ratio between directory entries and data entries. Nonzero values do not necessarily indicate a performance bottleneck; however, they should be further investigated.

**Total changed**
The snapshot value of the current number of changed pages. Derived from the Db2 field QW0254TC.

**Castout**
The number of castout operations performed. Derived from the Db2 field QW0254CC.

**Directory entry**
The number of allocated directory entries (not cumulative). Derived from the Db2 field QW0254DE.

**Data entry**
The number of allocated data entries (not cumulative). Derived from the Db2 field QW0254TE.

**Secondary-GBP — Directory entry**
The number of allocated directory entries. This is a snapshot value. Derived from the Db2 field QW02542D.

**Secondary-GBP — Data entry**
The number of allocated data entries. This is a snapshot value. Derived from the Db2 field QW02542T.

**Secondary-GBP — Changed page write hit**
The number of successful coupling facility write requests for changed pages. Derived from the Db2 field QW02542W.

**Secondary-GBP — Total changed**
The number of allocated data entries that are currently in "changed" state. This is a snapshot value. Derived from the Db2 field QW02542C.

**Secondary-GBP — Write miss cache full**
The number of unsuccessful coupling facility write requests because of insufficient coupling facility storage resources. Derived from the Db2 field QW02542F.

## Detail reports

This topic shows and describes the elements of a detail report.

A detail report is created as a single entity; however, to facilitate reading it is shown here in separate sections.

## The report header

The report header is shown at the top of every report page and identifies the report and the command options that were used to create the report.

The following is an example of a report header:

```
1    DB2 BUFFER POOL ANALYZER (V5R4M0) - BUFFER POOL ACTIVITY REPORT   PAGE: 1-1

                              ORDER: BPID-QPAGESET
                              TOP: 11  LEVEL: DETAIL
 GROUP:     N/P         LOCATION:       PMO2D721          DB2 VERSION: V11
 MEMBER:    N/P         REQUESTED FROM: NOT SPECIFIED     TO: NOT SPECIFIED
 SUBSYSTEM: D721        INTERVAL FROM:  12/06/13 16:08:30   TO: 12/06/13 16:09:22
⋮
```

- LEVEL specifies the type of report, here, a detail report.

- ORDER specifies the aggregation, here, by buffer pool ID (BPID) and a combination of database and page set (QPAGESET).
- SORTBY is not used for this example.
- TOP is not used; therefore, the default 11 applies. This means that the 11 topmost aggregations are reported. If the trace data contains more than 11 objects, they are aggregated under the label `Others` in the report.

This example was created with the following command:

```
BPACTIVITY REPORT LEVEL(DETAIL)
                  ORDER(BPID-QPAGESET)
```

Note that the ORDER, SORTBY, and TOP options affect only the information in "The Detail Activity section" on page 77.

## The Buffer Pool Characteristics section

The buffer pool characteristics values are retrieved from IFCID 202.

This section is the same as in summary reports. See "The Buffer Pool Characteristics section" on page 50 for a description of the elements.

```
                     =========   Buffer Pool Characteristics   =========
 BPID                             BP0      BP1      BP2      BP3      BP4      BP5
 -------------------------- -------- -------- -------- -------- -------- --------
 General
  Virtual pool size              2000     1000     1000     1000     1000     1000
  Pages fixed in real stor         No       No       No       No       No       No
  Page steal method               LRU      LRU      LRU      LRU      LRU      LRU
  Autosize attribute               No       No       No       No       No      Yes
 Thresholds
  Virtual sequential               80       80       80       80       80       80
  Deferred write                   50       50       50       50       50       50
  Vert deferred write(buff)         0        0        0        0        0        0
  Vert deferred write   (%)        10       10       10       10       10       10
  Parallel sequential              50       50       50       50       50       50
  Assisting parallel seq            0        0        0        0        0        0
 ========================== ======== ======== ======== ======== ======== ========

 BPID                             BP6      BP7      BP8      BP9    BP32K   BP32K1
 -------------------------- -------- -------- -------- -------- -------- --------
 General
  Virtual pool size              1000     1000     1000     1000       24       24
  Pages fixed in real stor         No       No       No       No       No       No
  Page steal method               LRU      LRU      LRU      LRU      LRU      LRU
  Autosize attribute               No       No       No       No       No      Yes
 Thresholds
  Virtual sequential               80       80       80       80       80       80
  Deferred write                   50       50       50       50       50       50
  Vert deferred write(buff)         0        0        0        0        0        0
  Vert deferred write   (%)        10       10       10       10       10       10
  Parallel sequential              50       50       50       50       50       50
  Assisting parallel seq            0        0        0        0        0        0
 ========================== ======== ======== ======== ======== ======== ========

 BPID                          BP32K2    BP8K0    BP8K1    BP8K2   BP16K0   BP16K1
 -------------------------- -------- -------- -------- -------- -------- --------
 General
  Virtual pool size                24      100      100      100       50       50
  Pages fixed in real stor         No       No       No       No       No       No
  Page steal method               LRU      LRU      LRU      LRU      LRU      LRU
  Autosize attribute               No       No       No       No       No      Yes
 Thresholds
  Virtual sequential               80       80       80       80       80       80
  Deferred write                   50       50       50       50       50       50
  Vert deferred write(buff)         0        0        0        0        0        0
  Vert deferred write   (%)        10       10       10       10       10       10
  Parallel sequential              50       50       50       50       50       50
  Assisting parallel seq            0        0        0        0        0        0
 ========================== ======== ======== ======== ======== ======== ========

1    DB2 BUFFER POOL ANALYZER (V5R2M0) - BUFFER POOL ACTIVITY REPORT   PAGE: 1-2
                              ORDER: BPID-QPAGESET
                              TOP: 11  LEVEL: DETAIL
 GROUP:     N/P         LOCATION:       PMO2D721           DB2 VERSION: V11
 MEMBER:    N/P         REQUESTED FROM: NOT SPECIFIED      TO: NOT SPECIFIED
 SUBSYSTEM: D721        INTERVAL FROM:  12/06/13 16:08:30   TO: 12/06/13 16:09:22

                     =========   Buffer Pool Characteristics   =========
 BPID                        BP16K2
 ------------------------- --------
 General
  Virtual pool size              50
  Pages fixed in real stor       No
  Page steal method             LRU
  Autosize attribute             No
 Thresholds
  Virtual sequential             80
  Deferred write                 50
  Vert deferred write(buff)       0
  Vert deferred write   (%)      10
  Parallel sequential            50
  Assisting parallel seq          0
 ========================= ========
```

## The Detail Activity section

The detail activity counter values about buffer pool operations are retrieved from IFCIDs 6, 7, 8, 9, 10, and 198.

```
1    DB2 BUFFER POOL ANALYZER (V5R4M0) - BUFFER POOL ACTIVITY REPORT   PAGE: 1-3

                              ORDER: BPID-QPAGESET
                              TOP: 11  LEVEL: DETAIL
 GROUP:     N/P         LOCATION:       PMO2D721           DB2 VERSION: V11
 MEMBER:    N/P         REQUESTED FROM: NOT SPECIFIED      TO: NOT SPECIFIED
 SUBSYSTEM: D721        INTERVAL FROM:  12/06/13 16:08:30   TO: 12/06/13 16:09:22

                      =======     Detail Activity     =======
 BPID                     BP0        BP0        BP0        BP0        BP0        BP0
 QPAGESET             DAADB01    DAADB01    DAADB01    DAADB06    DAADB06    DAADB06
                     DAALLX01   DAALLX02   SYSLGRNX   DAAAUH01   DAASSX01    SYSUSER
 -------------- ---------- ---------- ---------- ---------- ---------- ----------
 BP Hit ratio(%)
  System             88.9       77.8      100.0      100.0      100.0      100.0
  Application        88.9       77.8      100.0      100.0      100.0      100.0
  Read I/O           80.0       77.8      100.0      100.0      100.0      100.0
 Getpage                9          9          9         32         44          4
  Sequential            0          0          0          0          0          0
  Random                9          9          9         32         44          4
  Ridlist               0          0          0          0          0          0
  Hit                   8          7          9         32         44          4
  Miss random           1          2          0          0          0          0
  Miss asynch           0          0          0          0          0          0
  Noread                0          0          0          0          0          0

 Read request           1          2          0          0          0          0
  Synchronous           1          2          0          0          0          0
  Seq prefetch          0          0          0          0          0          0
  List pref             0          0          0          0          0          0
  Dyn prefetch          0          0          0          0          0          0
  Delay(msec)
   Synchronous        1.6        2.0        n/c        n/c        n/c        n/c
   Seq pref           n/c        n/c        n/c        n/c        n/c        n/c
   List pref          n/c        n/c        n/c        n/c        n/c        n/c
   Dyn pref           n/c        n/c        n/c        n/c        n/c        n/c

 Read page              1          2          0          0          0          0
  Synchronous           1          2          0          0          0          0
  Seq prefetch          0          0          0          0          0          0
  List pref             0          0          0          0          0          0
  Dyn prefetch          0          0          0          0          0          0

 Upd/wrt page         0.7        0.7        0.5        n/c        n/c        n/c
 Page/wrt req         1.5        1.5        1.0        n/c        n/c        n/c

 Buf Update             2          2          1          0          0          0
 Write request          2          2          2          0          0          0
  Synchronous           1          1          1          0          0          0
  Asynchr               1          1          1          0          0          0
  Delay(msec)
   Synchr             1.9        2.2        2.6        n/c        n/c        n/c
```

```
    Asynchr          199.5       93.2      129.3        n/c        n/c        n/c
  Write page             3          3          2          0          0          0
   Synchronous           1          1          1          0          0          0
   Asynchr               2          2          1          0          0          0
  ============== ========== ========== ========== ========== ========== ==========

 1    DB2 BUFFER POOL ANALYZER (V5R4M0) - BUFFER POOL ACTIVITY REPORT   PAGE: 1-4
                              ORDER: BPID-QPAGESET
                              TOP: 11  LEVEL: DETAIL
  GROUP:     N/P         LOCATION:       PMO2D721            DB2 VERSION: V11
  MEMBER:    N/P         REQUESTED FROM: NOT SPECIFIED       TO: NOT SPECIFIED
  SUBSYSTEM: D721        INTERVAL FROM:  12/06/13 16:08:30   TO: 12/06/13 16:09:22

                       =======     Detail Activity     =======
  BPID                     BP1        BP1        BP1        BP1        BP1        BP1
  QPAGESET            FIJ1DB01   FIJ1DB01   FIJ1DB01   FIJ1DB01   FIJ1DB01   FIJ1DB01
                      FIJCACIT   FIJCAMD1   FIJCAMHD   FIJCAMSG   FIJCAMTX   FIJCCONT
  -------------- ---------- ---------- ---------- ---------- ---------- ----------
  BP Hit ratio(%)
   System             98.9       66.7       93.3       95.1       93.0       90.9
   Application        98.9       66.7       93.3       95.1       93.0       90.9
   Read I/O           50.0       67.8      100.0      100.0      100.0      100.0
  Getpage              350          6         30         61         43         22
   Sequential            0          0          0          0          0          0
   Random              350          6         30         61         43         22
   Ridlist               0          0          0          0          0          0
   Hit                 346          4         28         58         40         20
   Miss random           4          2          2          3          3          2
   Miss asynch           0          0          0          0          0          0
   Noread                0          0          0          0          0          0

  Read request           4          2          2          3          3          2
   Synchronous           4          2          2          3          3          2
   Seq prefetch          0          0          0          0          0          0
   List pref             0          0          0          0          0          0
   Dyn prefetch          0          0          0          0          0          0
   Delay(msec)
    Synchronous        5.6        8.3        8.7        5.3       20.6        9.6
    Seq pref           n/c        n/c        n/c        n/c        n/c        n/c
    List pref          n/c        n/c        n/c        n/c        n/c        n/c
    Dyn pref           n/c        n/c        n/c        n/c        n/c        n/c

  Read page              4          2          2          3          3          2
   Synchronous           4          2          2          3          3          2
   Seq prefetch          0          0          0          0          0          0
   List pref             0          0          0          0          0          0
   Dyn prefetch          0          0          0          0          0          0

  Upd/wrt page        22.3        n/c        n/c        5.0        9.5        n/c
  Page/wrt req         2.0        n/c        n/c        1.0        1.0        n/c
  Buf Update            89          0          0         10         19          0
  Write request          2          0          0          2          2          0
   Synchronous           1          0          0          1          1          0
   Asynchr               1          0          0          1          1          0
   Delay(msec)
    Synchr             2.0        n/c        n/c        1.9        2.0        n/c
    Asynchr          146.3        n/c        n/c      146.8      165.1        n/c
  Write page             4          0          0          2          2          0
   Synchronous           1          0          0          1          1          0
   Asynchr               3          0          0          1          1          0
  ============== ========== ========== ========== ========== ========== ==========

 1    DB2 BUFFER POOL ANALYZER (V5R4M0) - BUFFER POOL ACTIVITY REPORT   PAGE: 1-5
                              ORDER: BPID-QPAGESET
                              TOP: 11  LEVEL: DETAIL
  GROUP:     N/P         LOCATION:       PMO2D721            DB2 VERSION: V11
  MEMBER:    N/P         REQUESTED FROM: NOT SPECIFIED       TO: NOT SPECIFIED
  SUBSYSTEM: D721        INTERVAL FROM:  12/06/13 16:08:30   TO: 12/06/13 16:09:22

                       =======     Detail Activity     =======
  BPID                     BP1        BP1        BP1        BP1        BP1        BP1
  QPAGESET            FIJ1DB01   FIJ1DB01   FIJ1DB01   FIJ1DB01   FIJ1DB01     Others
                      FIJCENGC   FIJCGROS   FIJCGRPG   FIJCINPG   FIJCINVD         29
  -------------- ---------- ---------- ---------- ---------- ---------- ----------
  BP Hit ratio(%)
   System              0.0       99.1       99.9       83.3       83.0       88.2
   Application         0.0       99.2       99.9       83.3       90.6       96.6
   Read I/O           90.0       97.8      100.0      100.0      100.0      100.0
  Getpage                2       6445       3662         12         53      38605
   Sequential            0          0          0          0          0       4860
   Random                2       6445       3662         12         53      33493
   Ridlist               0          0          0          0          0        252
   Hit                  0       6396       3660         10         48      37286

   Miss random          2         49          2          2          5        261
   Miss asynch          0          0          0          0          0       1040
   Noread               0          0          0          0          0         18

  Read request          2         52          2          2          6       1426
   Synchronous          2         49          2          2          5       1299
   Seq prefetch         0          0          0          0          0         97
   List pref            0          0          0          0          0          8
   Dyn prefetch         0          3          0          0          1         22
   Delay(msec)
    Synchronous       9.2        3.7       11.0       23.5        5.3       11.7
    Seq pref          n/c        n/c        n/c        n/c        n/c       38.1
    List pref         n/c        n/c        n/c        n/c        n/c       21.1
    Dyn pref          n/c       11.9        n/c        n/c        3.9       22.1

  Read page             2         59          2          2          9       4572
   Synchronous          2         49          2          2          5       1299
   Seq prefetch         0          0          0          0          0       2857
   List pref            0          0          0          0          0         83
   Dyn prefetch         0         10          0          0          4        333

  Upd/wrt page        n/c       37.7        n/c        n/c        n/c       25.1
  Page/wrt req        n/c       16.3        n/c        n/c        n/c        6.4
  Buf Update            0       1848          0          0          0       9097
  Write request         0          3          0          0          0         57
   Synchronous          0          1          0          0          0         31
   Asynchr              0          2          0          0          0         26
   Delay(msec)
    Synchr            n/c        3.4        n/c        n/c        n/c        7.2
    Asynchr           n/c      121.0        n/c        n/c        n/c       99.1
  Write page            0         49          0          0          0        363
   Synchronous          0          1          0          0          0         31
   Asynchr              0         48          0          0          0        332
  ============== ========== ========== ========== ========== ========== ==========

 1    DB2 BUFFER POOL ANALYZER (V5R4M0) - BUFFER POOL ACTIVITY REPORT   PAGE: 1-6
                               ORDER: BPID-QPAGESET
                               TOP: 11  LEVEL: DETAIL
  GROUP:     N/P         LOCATION:       PMO2D721            DB2 VERSION: V11
  MEMBER:    N/P         REQUESTED FROM: NOT SPECIFIED       TO: NOT SPECIFIED
  SUBSYSTEM: D721        INTERVAL FROM:  12/06/13 16:08:30   TO: 12/06/13 16:09:22

                       =======     Detail Activity     =======
  BPID                    BP2        BP2        BP2
  QPAGESET            WTNTEST    WTNTEST    WTNTEST
                    WTNMTS2    WTNMUS4    WTNMUS5
  -------------- ---------- ---------- ----------
  BP Hit ratio(%)
   System             0.0       -8.2       -2.2
   Application        0.0       99.6      100.0
   Read I/O          30.0       77.8      100.0
  Getpage               1        502       2002
   Sequential           0        501       2001
   Random               1          1          1
   Ridlist              0          0          0
   Hit                  0        500       2001
   Miss random          1          1          0
   Miss asynch          0          1          1
   Noread               0          0          0

  Read request          1         19         65
   Synchronous          1          2          1
   Seq prefetch         0         17         64
   List pref            0          0          0
   Dyn prefetch         0          0          0
   Delay(msec)
    Synchronous       1.8       10.7       45.0
    Seq pref          n/c       26.4       25.6
    List pref         n/c        n/c        n/c
    Dyn pref          n/c        n/c        n/c

  Read page             1        543       2046
   Synchronous          1          2          1
   Seq prefetch         0        541       2045
   List pref            0          0          0
   Dyn prefetch         0          0          0

  Upd/wrt page        0.3        n/c        n/c
  Page/wrt req        1.5        n/c        n/c
  Buf Update            1          0          0
  Write request           2          0          0
   Synchronous            1          0          0
   Asynchr                1          0          0
   Delay(msec)
    Synchr              2.1        n/c        n/c
    Asynchr           104.4        n/c        n/c
  Write page              3          0          0
   Synchronous            1          0          0
   Asynchr                2          0          0
  ============== ========== ========== ==========


 1    DB2 BUFFER POOL ANALYZER (V5R4M0) - BUFFER POOL ACTIVITY REPORT   PAGE: 1-7
                               ORDER: BPID-QPAGESET
                               TOP: 11  LEVEL: DETAIL
  GROUP:     N/P         LOCATION:       PMO2D721            DB2 VERSION: V11
  MEMBER:    N/P         REQUESTED FROM: NOT SPECIFIED       TO: NOT SPECIFIED
  SUBSYSTEM: D721        INTERVAL FROM:  12/06/13 16:08:30   TO: 12/06/13 16:09:22

                       =======     Detail Activity     =======
  BPID                  BP5
  QPAGESET         PARLDABA
                     TAB1TS
  -------------- ----------
  BP Hit ratio(%)
   System               1.9
   Application        100.0
   Read I/O            80.0
  Getpage             23812
   Sequential         23812
   Random                 0
   Ridlist                0
   Hit                23808
   Miss random            0
   Miss asynch            4
   Noread                 0

  Read request          738
   Synchronous            4
   Seq prefetch         734
   List pref              0
   Dyn prefetch           0
   Delay(msec)
    Synchronous        28.7
    Seq pref           38.6
    List pref           n/c
    Dyn pref            n/c

  Read page           23360
   Synchronous            4
   Seq prefetch       23356
   List pref              0
   Dyn prefetch           0

  Upd/wrt page          n/c
  Page/wrt req          n/c
  Buf Update              0
  Write request           0
   Synchronous            0
   Asynchr                0
   Delay(msec)
    Synchr              n/c
    Asynchr             n/c
  Write page              0
   Synchronous            0
   Asynchr                0
  ============== ==========

 1    DB2 BUFFER POOL ANALYZER (V5R4M0) - BUFFER POOL ACTIVITY REPORT   PAGE: 1-8
                               ORDER: BPID-QPAGESET
                               TOP: 11  LEVEL: DETAIL
  GROUP:     N/P         LOCATION:       PMO2D721            DB2 VERSION: V11
  MEMBER:    N/P         REQUESTED FROM: NOT SPECIFIED       TO: NOT SPECIFIED
  SUBSYSTEM: D721        INTERVAL FROM:  12/06/13 16:08:30   TO: 12/06/13 16:09:22

                       =======     Detail Activity     =======
  BPID                  BP7
  QPAGESET          DAADB07
                    DAA4K01
  -------------- ----------
  BP Hit ratio(%)
   System              79.5
   Application         99.9
   Read I/O            80.0
  Getpage              2873
   Sequential          2838
   Random                35
   Ridlist                0
   Hit                  640
   Miss random            0
   Miss asynch            3
   Noread              2230

  Read request          152
   Synchronous            3
   Seq prefetch         149
   List pref              0
   Dyn prefetch           0
   Delay(msec)
    Synchronous        35.8
    Seq pref           12.4
    List pref           n/c
    Dyn pref            n/c

  Read page             590
   Synchronous            3
   Seq prefetch         587
   List pref              0
   Dyn prefetch           0

  Upd/wrt page          2.1
  Page/wrt req         20.4
  Buf Update           4573
  Write request         108
   Synchronous            0
   Asynchr              108
   Delay(msec)
    Synchr              n/c
    Asynchr            39.5
  Write page           2200
   Synchronous            0
   Asynchr             2200
  ============== ==========


 1    DB2 BUFFER POOL ANALYZER (V5R4M0) - BUFFER POOL ACTIVITY REPORT   PAGE: 1-9
                               ORDER: BPID-QPAGESET
                               TOP: 11  LEVEL: DETAIL
  GROUP:     N/P         LOCATION:       PMO2D721            DB2 VERSION: V11
  MEMBER:    N/P         REQUESTED FROM: NOT SPECIFIED       TO: NOT SPECIFIED
  SUBSYSTEM: D721        INTERVAL FROM:  12/06/13 16:08:30   TO: 12/06/13 16:09:22

                       =======     Detail Activity     =======
  BPID               BP32K
  QPAGESET         FIJ1DB32
                   FIJS0001
  -------------- ----------
  BP Hit ratio(%)
   System             100.0
   Application        100.0
   Read I/O            80.0
  Getpage                 8
   Sequential             0
   Random                 8
   Ridlist                0
   Hit                    8
   Miss random            0
   Miss asynch            0
   Noread                 0

  Read request            0
   Synchronous            0
   Seq prefetch           0
   List pref              0
   Dyn prefetch           0
   Delay(msec)
    Synchronous         n/c
    Seq pref            n/c
    List pref           n/c
    Dyn pref            n/c

  Read page               0
   Synchronous            0
   Seq prefetch           0
   List pref            0
   Dyn prefetch         0

  Upd/wrt page        n/c
  Page/wrt req        n/c
  Buf Update            0
  Write request         0
   Synchronous          0
   Asynchr              0
   Delay(msec)
    Synchr            n/c
    Asynchr           n/c
  Write page            0
   Synchronous          0
   Asynchr              0
  ============== ==========

 1    DB2 BUFFER POOL ANALYZER (V5R4M0) - BUFFER POOL ACTIVITY REPORT   PAGE: 1-10
                               ORDER: BPID-QPAGESET
                               TOP: 11  LEVEL: DETAIL
  GROUP:     N/P         LOCATION:       PMO2D721            DB2 VERSION: V11
  MEMBER:    N/P         REQUESTED FROM: NOT SPECIFIED       TO: NOT SPECIFIED
  SUBSYSTEM: D721        INTERVAL FROM:  12/06/13 16:08:30   TO: 12/06/13 16:09:22

                       =======     Detail Activity     =======
                            **********  TOTAL  **********
  BPID                  BP0        BP1        BP2        BP5        BP7      BP32K
  -------------- ---------- ---------- ---------- ---------- ---------- ----------
  BP Hit ratio(%)
   System              97.2       90.5       -3.4        1.9       79.5      100.0
   Application         97.2       97.2       99.8      100.0       99.9      100.0
   Read I/O            80.0       77.8      100.0      100.0      100.0      100.0
  Getpage               107      49291       2505      23812       2873          8
   Sequential             0       4860       2502      23812       2838          0
   Random               107      44179          3          0         35          8
   Ridlist                0        252          0          0          0          0
   Hit                  104      47896       2501      23808        640          8
   Miss random            3        337          2          0          0          0
   Miss asynch            0       1040          2          4          3          0
   Noread                 0         18          0          0       2230          0

  Read request            3       1506         85        738        152          0
   Synchronous            3       1375          4          4          3          0
   Seq prefetch           0         97         81        734        149          0
   List pref              0          8          0          0          0          0
   Dyn prefetch           0         26          0          0          0          0
   Delay(msec)
    Synchronous         1.8       11.4       17.1       28.7       35.8        n/c
    Seq pref            n/c       38.1       25.8       38.6       12.4        n/c
    List pref           n/c       21.1        n/c        n/c        n/c        n/c
    Dyn pref            n/c       20.2        n/c        n/c        n/c        n/c

  Read page               3       4662       2590      23360        590          0
   Synchronous            3       1375          4          4          3          0
   Seq prefetch           0       2857       2586      23356        587          0
   List pref              0         83          0          0          0          0
   Dyn prefetch           0        347          0          0          0          0

  Upd/wrt page          0.6       26.3        0.3        n/c        2.1        n/c
  Page/wrt req          1.3        6.4        1.5        n/c       20.4        n/c
  Buf Update              5      11063          1          0       4573          0
  Write request           6         66          2          0        108          0
   Synchronous            3         35          1          0          0          0
   Asynchr                3         31          1          0        108          0
   Delay(msec)
    Synchr              2.2        6.7        2.1        n/c        n/c        n/c
    Asynchr           140.7      105.7      104.4        n/c       39.5        n/c
  Write page              8        420          3          0       2200          0
   Synchronous            3         35          1          0          0          0
   Asynchr                5        385          2          0       2200          0
  ============== ========== ========== ========== ========== ========== ==========

 1    DB2 BUFFER POOL ANALYZER (V5R4M0) - BUFFER POOL ACTIVITY REPORT   PAGE: 1-11
                               ORDER: BPID-QPAGESET
                               TOP: 11  LEVEL: DETAIL
  GROUP:     N/P         LOCATION:       PMO2D721            DB2 VERSION: V11
  MEMBER:    N/P         REQUESTED FROM: NOT SPECIFIED       TO: NOT SPECIFIED
  SUBSYSTEM: D721        INTERVAL FROM:  12/06/13 16:08:30   TO: 12/06/13 16:09:22

                       =======     Detail Activity     =======
                            **********  TOTAL  **********
  BPID           |    GRAND
```

```
                 |    TOTAL
 -------------- ----------
 BP Hit ratio(%)
  System         |     60.3
  Application    |     98.2
  Read I/O             80.0
 Getpage         |    78596
  Sequential     |    34012
  Random         |    44332
  Ridlist        |      252
  Hit            |    74957
  Miss random    |      342
  Miss asynch    |     1049
  Noread         |     2248

 Read request    |     2484
  Synchronous    |     1389
  Seq prefetch   |     1061
  List pref      |        8
  Dyn prefetch   |       26
  Delay(msec)
   Synchronous   |     11.5
   Seq pref      |     33.9
   List pref     |     21.1
   Dyn pref      |     20.2

 Read page       |    31205
  Synchronous    |     1389
  Seq prefetch   |    29386
  List pref      |       83
  Dyn prefetch   |      347

 Upd/wrt page    |      5.9
 Page/wrt req    |     14.5
 Buf Update      |    15642
 Write request   |      182
  Synchronous    |       39
  Asynchr        |      143
  Delay(msec)
   Synchr        |      6.2
   Asynchr       |     56.4
 Write page      |     2631
  Synchronous    |       39

  Asynchr        |     2592
 ============== ==========

0Buffer pool activity report complete
```

**Buffer pool hit ratios**

Primary indicator of buffer pool efficiency showing the number of pages contained in the buffer pool compared to the number of pages requested.

The highest possible hit ratio is 100%. This means that every page requested is always in the buffer pool. A low ratio indicates high read I/O.

To increase the buffer pool hit ratio, you can do one of the following:

- Run the REORG utility for table spaces or index spaces associated with the virtual buffer pool.
- Decrease the virtual pool sequential steal threshold (VPSEQT) to reserve more pages for random I/O.
- Increase the buffer pool size but be aware that the cost of paging can outweigh the benefit of I/O avoidance.
- Establish more separate buffer pools, for example, to isolate table spaces and index spaces according to their access characteristics.

**Buffer pool hit ratio (%) — System**

The number of Getpage requests by Db2 and satisfied by the buffer pool, expressed as a percentage of all Getpage requests.

This shows the percentage of pages in a prefetch Getpage request that are found in the buffer pool.

Usually, this value is low when prefetch is used. A high value indicates that applications perform a series of similar operations on the same data.

A negative value indicates that prefetched pages are not subsequently referenced. The reason for this is that the query stops before it reaches the end of the prefetched pages, or that the prefetched pages are stolen by Db2 for reuse before the query can access them.

Compare the value in this field with the application hit ratio to determine the efficiency of prefetch operations.

**Buffer pool hit ratio (%) — Application**
The number of Getpage requests issued by applications and satisfied by the buffer pool, expressed as a percentage of all Getpage requests issued by applications.

The hit ratio indicates the level of synchronous I/O because prefetched pages that are already in the buffer pool count as hits. The value is a relative value depending on the type of application. For example, an application that browses large amounts of noncontinuous data might have a buffer pool hit ratio of 0. Check those cases in which the hit ratio drops significantly for the same application.

**Buffer pool hit ratio (%) — Read I/O**
The number of read I/O requests without physical I/O activity (satisfied by the buffer pool), expressed as a percentage of all read I/O requests (with and without physical I/O activity).

The hit ratio indicates the percentage of read I/O requests that were satisfied by the buffer pool without requiring I/O activities to a hard disk drive.

**Getpage**
The total number of Getpage requests. Getpage requests are divided as follows:

**Getpage — Sequential**
The number of sequential Getpage requests because of prefetch operations.

**Getpage — Random**
The number of random Getpage requests, usually issued by applications.

**Getpage — Ridlist**
The number of record identifier (RID) list pages referenced.

**Getpage — Hit**
The number of Getpage requests for which pages are found in the buffer pool.

**Getpage — Miss random**
The number of random Getpage requests for which the page is not found in the buffer pool.

**Getpage — Miss asynch**
The number of Getpage Asynchronous requests for which the page is not found in the buffer pool.

**Getpage — Noread**
The number of Getpage requests for which the page is not found in the buffer pool and for which the request did not result in a read I/O operation. The page is a new page for INSERT.

**Read request**
The total number of read I/O requests (synchronous, sequential prefetch, list sequential prefetch, dynamic prefetch) with at least one page read from hard disk drive per request (content of Db2 field QW0007NP - number of pages read - greater zero). If no page is read from hard disk drive (all pages are found in the buffer pool), the counter is not incremented. The total number of read I/O requests is the result of the following types of read I/O requests:

**Read request — Synchronous**
The number of random, synchronous read I/O requests.

**Read request — Seq prefetch**
The number of sequential prefetch read I/O requests with at least one page read from hard disk drive per request (content of Db2 field QW0007NP - number of pages read - greater zero). If the prefetch results in an I/O read, up to 32 pages can be read for SQL, up to 64 pages can be read for

utilities. If all pages to be prefetched are already in the buffer pool, a request does not result in an I/O read.

Sequential prefetch reads a sequential set of pages. It allows CP and I/O operations to overlap. Db2 determines at BIND time whether sequential prefetch is used.

Sequential prefetch is generally used for a table space scan. It can also be used to read index pages in an index scan. For an index scan that accesses eight or more consecutive data pages, Db2 requests sequential prefetch at bind time. The index must have a cluster ratio of 80% or higher. You can use REORG and RUNSTATS, and rebind relevant SQL if you do not know whether the target was met previously.

**Read request — List pref**

The number of list sequential prefetch read I/O requests with at least one page read from hard disk drive per request (content of Db2 field QW0007NP - number of pages read - greater zero).

List prefetch allows Db2 to access data pages efficiently even if the required data pages are not contiguous. It allows CP and I/O operations to overlap.

Db2 uses list prefetch to do the following tasks:

- Always to access data by multiple index access.
- Always to access data from the inner table during a hybrid join.
- Usually with one single index that has a cluster ratio lower than 80%.
- Sometimes on one single index with a high cluster ratio. This increases the efficiency of sequential prefetch if the estimated amount of data to be accessed is too small.
- Never when the estimated number of RIDs to be processed would take more than 50% of the RID pool.

During execution time, list prefetch processing stops if more than 25% of the rows with a minimum of 4075 in the table must be accessed.

**Read request — Dyn prefetch**

The number of dynamic prefetch requests with at least one page read from hard disk drive per request (content of Db2 field QW0007NP - number of pages read - greater zero). If the prefetch request results in an I/O read, up to 32 advancing pages can be read at a time.

Dynamic prefetch reads a sequential set of pages. It allows CP and I/O operations to overlap.

If Db2 does not choose prefetch at bind time, it can sometimes use it at execution time through sequential detection.

For information on when sequential detection is used and when dynamic prefetch is triggered, see the *DB2 9 Administration Guide*.

**Read request — Delay (msec)**

The average elapsed time between start and completion of:

**Read request — Delay (msec) — Synchronous**

A synchronous read I/O

**Read request — Delay (msec) — Sequential pref**

A sequential prefetch read request

**Read request — Delay — List pref**

A list prefetch read request

**Read request — Delay — Dynamic prefetch**

A dynamic prefetch read request

**Read page**

The total number of pages read from a hard disk drive.

**Read page — Synchronous**

The number of pages read from a hard disk drive for applications and utilities.

**Read page — Sequential prefetch**
The number of pages read from a hard disk drive for sequential prefetch requests.

**Read page — List pref**
The number of pages read from a hard disk drive for list prefetch requests.

**Read page — Dyn prefetch**
The number of pages read from a hard disk drive for dynamic prefetch requests.

**Upd/wrt page**
The number of buffer updates per page written from the buffer pool to a hard disk drive.

A high value indicates a high level of efficiency because more updates are externalized per physical write cycle.

Buffer updates per pages written depends on the type of application. For example, a batch program that processes a table in skip sequential mode with a high row update frequency in a dedicated environment can achieve a high update efficiency. Usually, update efficiency is lower for transaction processing applications because these applications use more random page access.

The following factors can influence the number of updates per page:

- The number of rows per page: A small PCTFREE value gathers more rows on the same page. This might, however, impact concurrency.
- The buffer pool size and the deferred write thresholds: Increase the size of the buffer pool or the deferred write thresholds DWQT and VDWQT. This lets Db2 accumulate page updates in the buffer pool. Db2 might thus capture more updates per page. The effect depends on the type of transaction. It is less significant if the buffer pool is used concurrently by multiple transactions that access random pages.

**Page/wrt req**
The number of pages written from the buffer pool to a hard disk drive per synchronous or asynchronous write I/O. This count does not include preformatting I/O, such as I/O that is required to prepare a data set for use.

Use this field and the Upd/wrt page field to determine the efficiency of the buffer pool for write operations. The following factors impact the ratio of pages written per write I/O:

- The checkpoint frequency: At checkpoint time, I/Os write all updated pages on the deferred write queue to a hard disk drive. If this occurs too often, the deferred write queue does not grow large enough to achieve a high ratio of pages written per write I/O. The checkpoint frequency depends on the number of logs that are written between two consecutive checkpoints. This number is set at installation time.
- The frequency of active log switches: Db2 takes a system checkpoint when the active log is switched. Frequent active log switches cause a higher checkpoint frequency. This prevents the deferred write queue to grow to an optimum size.
- The buffer pool size and deferred write thresholds: The deferred write thresholds (VDWQT and DWQT) are a function of buffer pool size. If the buffer pool size decreases, these thresholds are reached more frequently and cause I/Os to write some of the pages on the deferred write queue to a hard disk drive more often. This prevents the deferred write queue from growing large enough to achieve a high ratio of pages written per write I/O.
- The number of data sets, and the spread of updated pages across them: The efficiency of write I/O also depends on the number of data sets associated with the buffer pool and spread of updated pages across them. Due to the way batch processing works, the ratio of pages written to write I/Os is expected to be higher than for transaction type workloads.

**Buf Update**
The number of times updates are requested against pages in the buffer pool.

**Write request**
The total number of write I/O operations that are made to a hard disk drive. The total number of write I/O requests is the result of the following types of write I/O requests:

**Write request — Synchronous**
The number of synchronous write I/O operations that are made to a hard disk drive. Synchronous or immediate writes occur if one of the following conditions apply:

- An immediate write threshold is reached.
- No deferred write engines are available.
- More than two checkpoints pass without a page being written. Note that this does not indicate a buffer shortage.

You should keep this value as small as possible. Synchronous writes occur if there are too many checkpoints, if the buffer pool is too small, or both.

**Write request — Asynchr**
The number of asynchronous write I/O operations made to a hard disk drive.

**Write request — Delay (msec)**
The average elapsed time between start and completion of:

**Write request — Delay (msec) — Synchronous**
a synchronous write I/O request.

**Write request — Delay (msec) — Asynchr**
an asynchronous write I/O request.

**Write page**
The total number of pages written to a hard disk drive.

**Write page — Synchronous**
The number of pages written synchronously to a hard disk drive.

**Write page — Asynchr**
The number of pages written to a hard disk drive by asynchronous write requests.

# Chapter 6. Viewing performance data on the client

This topic describes how to view detailed buffer pool performance data on the client.

### About this task

Related tasks:

- Before you can use this function, you must have performed the tasks described in:
  - Chapter 3, "Collecting data," on page 27
  - Chapter 4, "Creating activity reports and bpd files," on page 37. Here, only bpd files are of interest.
  - Chapter 11, "Downloading files from the host to the client," on page 151

  A buffer pool data file must be available on the client (the file with the recommended file name extension bpd).

**General remarks:**

- Your client should have approximately 40 MB of *available physical memory* (random access memory) for this function. You can check this in the Windows™ Task Manager. Close other applications, if necessary.
- The data used for this function reflects the performance for the interval for which trace data was collected and for the time frame that was specified with the **GLOBAL** and the **BPACTIVITY** command when the bpd file was created.
- If you want to print any of the graphical information from the following windows, place the cursor on the displayed graphical information and click **View —► Open dataview in browser**. When the browser window is displayed, use your browser's printing capabilities to print the information. For colored printouts, check the browser settings.

  Example: In the Internet Explorer, click **Tools —► Internet Options —► Advanced**. Under **Printing**, select Print background colors and images.

## Starting the view function

This section explains how to start the view function.

### About this task

To start the view function, perform the following steps:

### Procedure

1. Start the IBM Db2 Buffer Pool Analyzer by double-clicking the icon on your Windows desktop, or use the **Start** push button on the taskbar to start this client application.

   If you are using Db2 Performance Expert, click **Tools** > **Buffer Pool Analysis for z/OS** on the menu bar, or click the **Opens the Buffer Pool Analysis for z/OS window** toolbar button.

   The Db2 Buffer Pool Analyzer - z/OS main window is displayed.
2. On the menu bar, click **File** > **Open Report**. Alternatively, click the **Opens report file** toolbar button.

   The Open dialog box is displayed. You use this dialog box to select and open a bpd file that contains the performance data you want to view.

   Several sample buffer pool data files (`*.`bpd) are delivered with Buffer Pool Analyzer in the … `\samples\reporting` folder. You can use them to become familiar with this function.
3. Continue with "Selecting and opening a buffer pool data file" on page 90.

## Selecting and opening a buffer pool data file

This section explains how to select and open a buffer pool data file.

### About this task

### Procedure

1. Select a folder from the **Look in** list where the bpd file is located.

   The **File of type** field shows the file name extension of buffer pool data files (`*.`bpd). If your bpd file does not have the recommended file name extension bpd, select `All files (*.*)` from the **Files of type** list to see the appropriate files.
2. Click the bpd file to be opened.

   The **File name** field shows the name of the selected bpd file.
3. Click **Open**

   The selected bpd file is opened, and the following window is displayed:

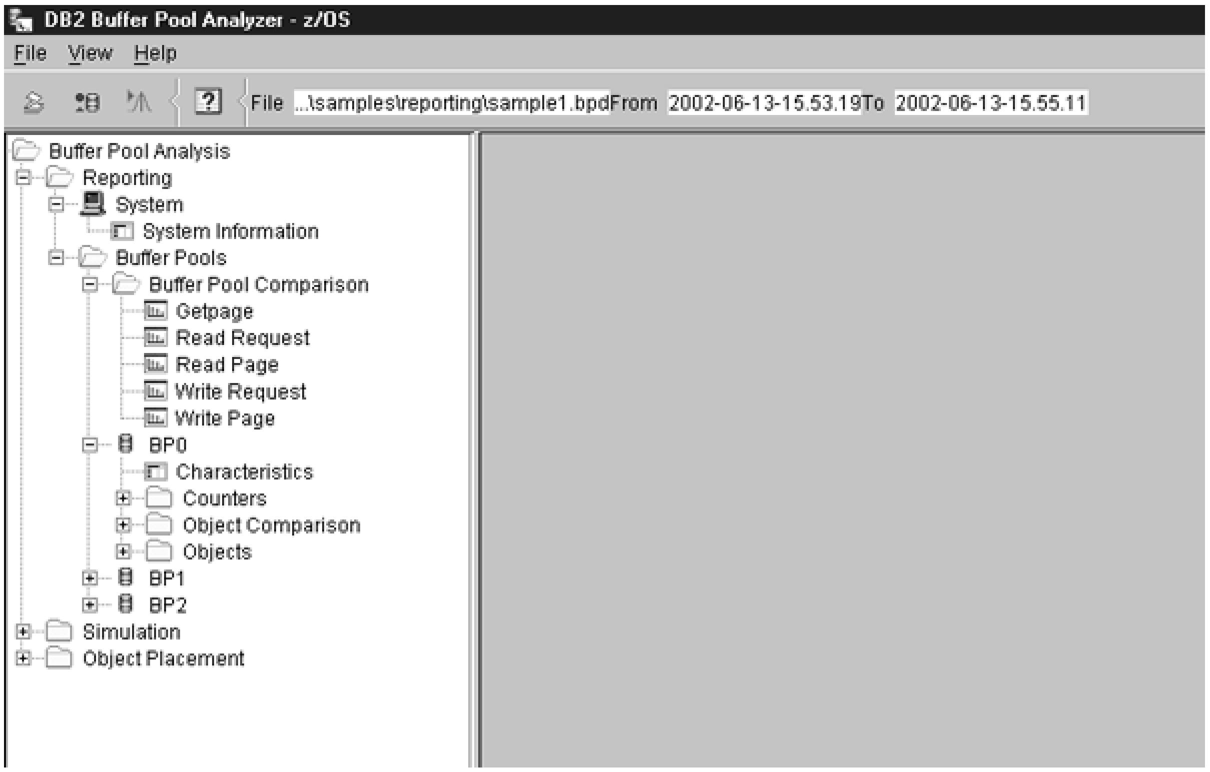


*Figure 10. Viewing performance data – The Open dialog window*

The menu bar of the window shows information about the opened bpd file:

- The **File** field shows the full path and name of the bpd file.
- The **From** and **To** fields show the start and end timestamps of data contained in the bpd file. This is the start and end of the data collection, respectively the corresponding values of the **From** and **To** options of the **`BPACTIVITY`** or **`GLOBAL`** command, if they were used to limit the time frame.

The contents pane on the left side of the window gives you access to Buffer Pool Analyzer data and results from other functions. You can expand or collapse the tree items by clicking the plus sign (+), respectively the minus sign (-), or by double-clicking the corresponding tree item.

Buffer pool performance data from the opened bpd file is in the **Reporting** folder and its subfolders.

On the following pages only a few examples of the available windows are shown. Use the contents pane to navigate through the information until you are familiar with its presentation.

4. Continue with one of the following:
   - "Getting system information" on page 91
   - "Getting information on buffer pools" on page 92

## Getting system information

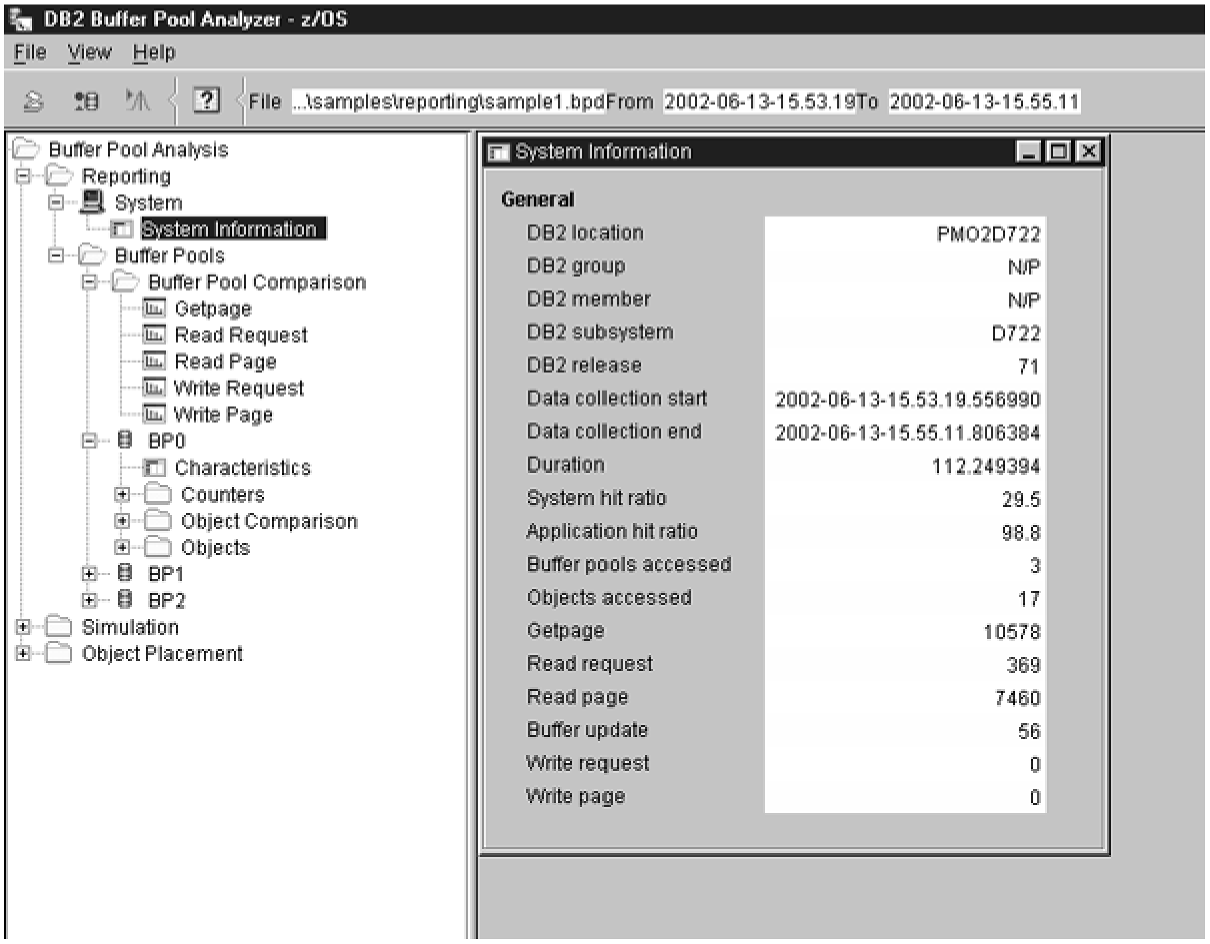
You can view system information of the DB2 subsystem from which performance data was collected.

### About this task

When you have opened a bpd file, as described in "Selecting and opening a buffer pool data file" on page 90, you can view system information as follows:

1. In the **Reporting** tree of the contents pane, double-click **System**, then double-click **System information**.

   The following window is displayed:



**General**

| | |
|---|---|
| DB2 location | PMO2D722 |
| DB2 group | N/P |
| DB2 member | N/P |
| DB2 subsystem | D722 |
| DB2 release | 71 |
| Data collection start | 2002-06-13-15.53.19.556990 |
| Data collection end | 2002-06-13-15.55.11.806384 |
| Duration | 112.249394 |
| System hit ratio | 29.5 |
| Application hit ratio | 98.8 |
| Buffer pools accessed | 3 |
| Objects accessed | 17 |
| Getpage | 10578 |
| Read request | 369 |
| Read page | 7460 |
| Buffer update | 56 |
| Write request | 0 |
| Write page | 0 |

*Figure 11. Viewing performance data – The System Information window*

The right pane shows general information about the Db2 subsystem from which performance data was collected, for example, Db2 location, Db2 group, and Db2 member.

## Getting information on buffer pools

This section lists the different ways how you can get information on buffer pools.

### About this task

When you have opened a bpd file, as described in "Selecting and opening a buffer pool data file" on page 90, you can view buffer pool information as follows:

1. In the **Reporting** folder of the contents pane, double-click **Buffer Pools**.
2. If you want to compare buffer pool data, continue with "Getting buffer pool comparison information" on page 92.
3. If you want to see detailed information about individual buffer pools, continue with "Getting individual buffer pool information" on page 93.

## Getting buffer pool comparison information

This section explains how to view and compare performance data of different buffer pools.

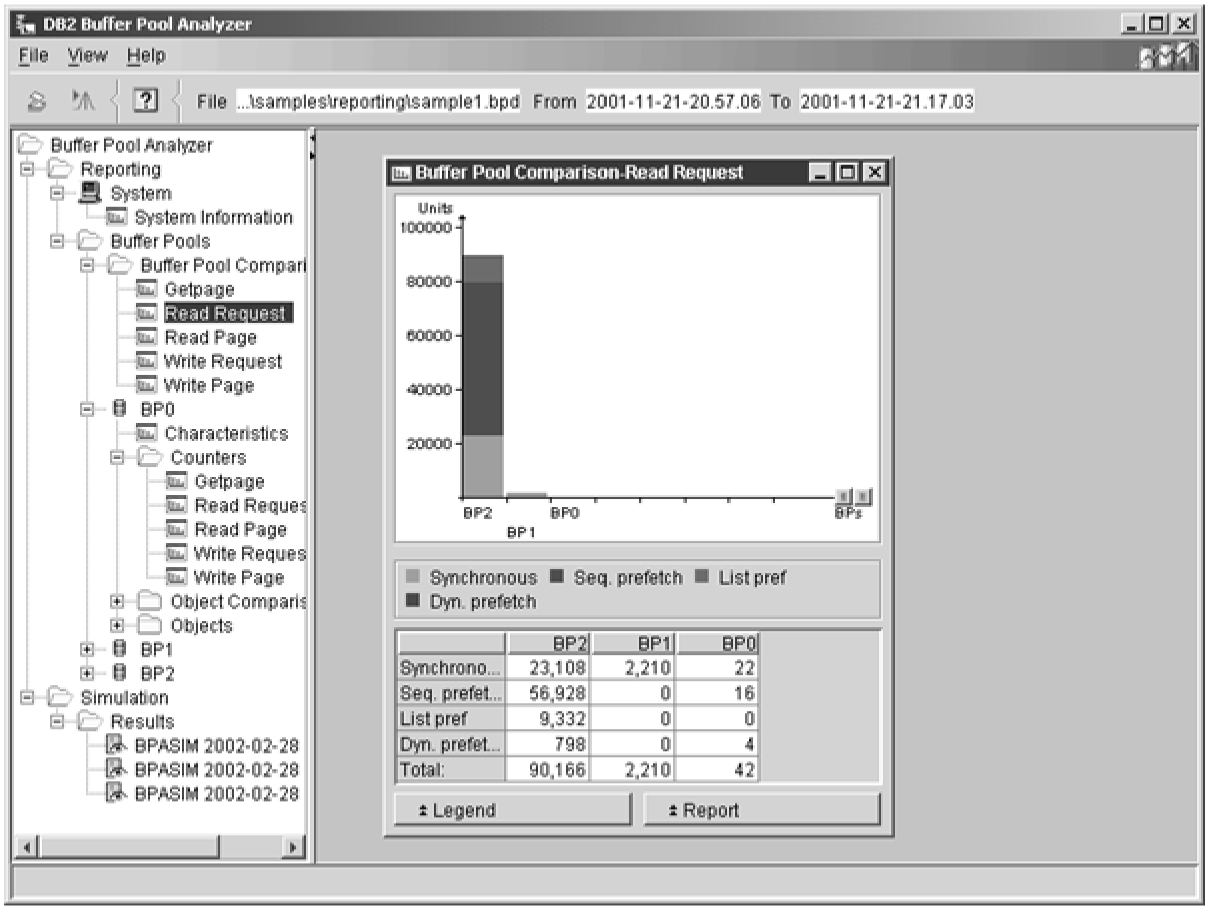
### Procedure

1. In the **Buffer Pools** folder of the contents pane, click **Buffer Pool Comparison**.

   The different buffer pool counters, such as **Getpage**, **Read Request**, **Write Request**, and **Write Page** are displayed.

   **Note:** The content pane displays only the active counters. Counters that have no activity are not displayed.
2. Double-click a counter, for example **Read Request**.

   The following window is displayed:



| | BP2 | BP1 | BP0 |
|---|---|---|---|
| Synchrono... | 23,108 | 2,210 | 22 |
| Seq. prefet... | 56,928 | 0 | 16 |
| List pref | 9,332 | 0 | 0 |
| Dyn. prefet... | 798 | 0 | 4 |
| Total: | 90,166 | 2,210 | 42 |

*Figure 12. Viewing performance data – The Buffer Pool Comparison window*

The right pane shows a comparison of buffer pools in your system regarding the selected counter in the form of a bar chart.

## Getting individual buffer pool information

You can view individual buffer pool characteristics and individual buffer pool counters of a selected buffer pool.

### Procedure

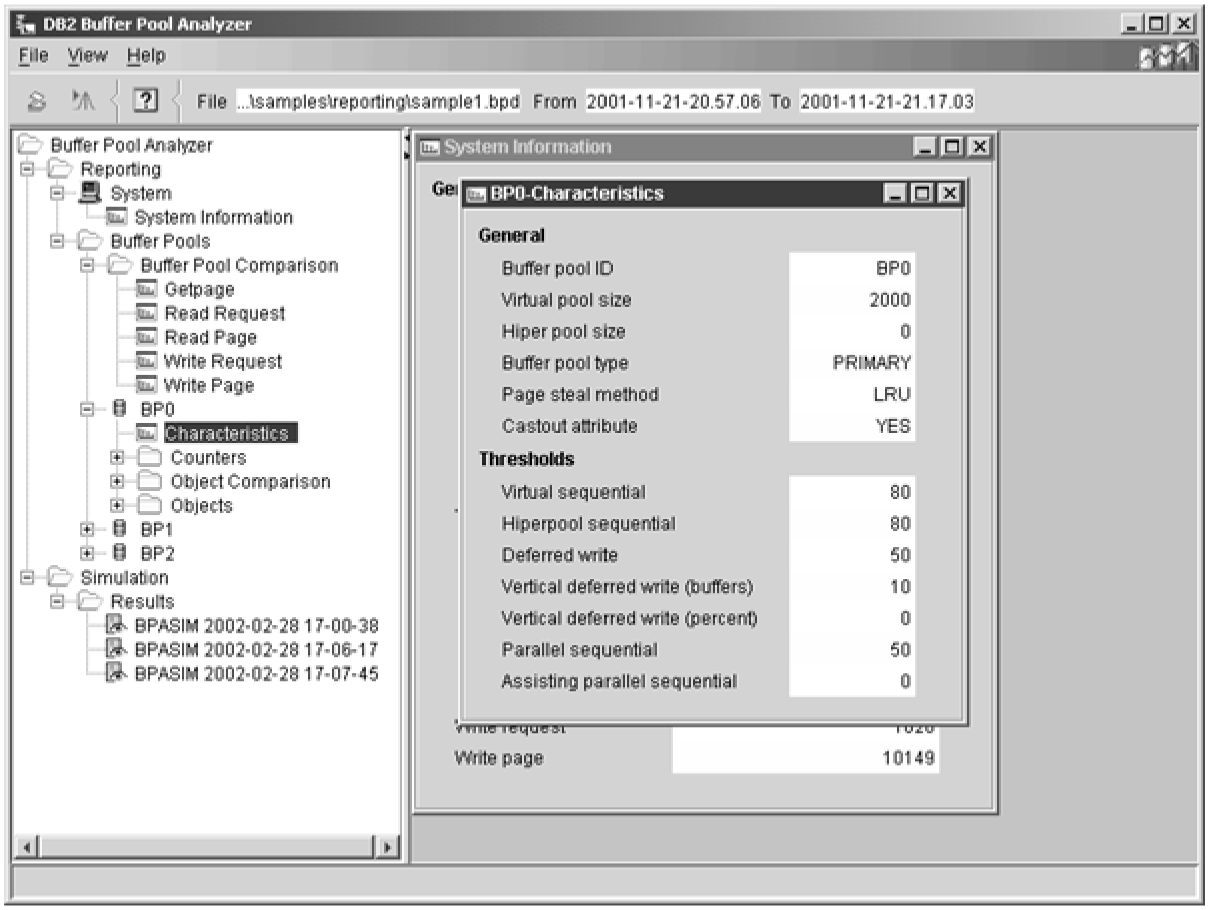
1. In the **Buffer Pools** folder of the contents pane, click one of the icons representing individual buffer pools, for example **BP0**.

   The following items are displayed for the selected buffer pool:

   - Characteristics
   - Counters
   - Object Comparison
   - Objects

   Each item contains additional information about the selected buffer pool.
2. If you want to see general information and thresholds of the selected buffer pool, double-click **Characteristics**.

   The following window is displayed:



**General**

| | |
|---|---|
| Buffer pool ID | BP0 |
| Virtual pool size | 2000 |
| Hiper pool size | 0 |
| Buffer pool type | PRIMARY |
| Page steal method | LRU |
| Castout attribute | YES |

**Thresholds**

| | |
|---|---|
| Virtual sequential | 80 |
| Hiperpool sequential | 80 |
| Deferred write | 50 |
| Vertical deferred write (buffers) | 10 |
| Vertical deferred write (percent) | 0 |
| Parallel sequential | 50 |
| Assisting parallel sequential | 0 |

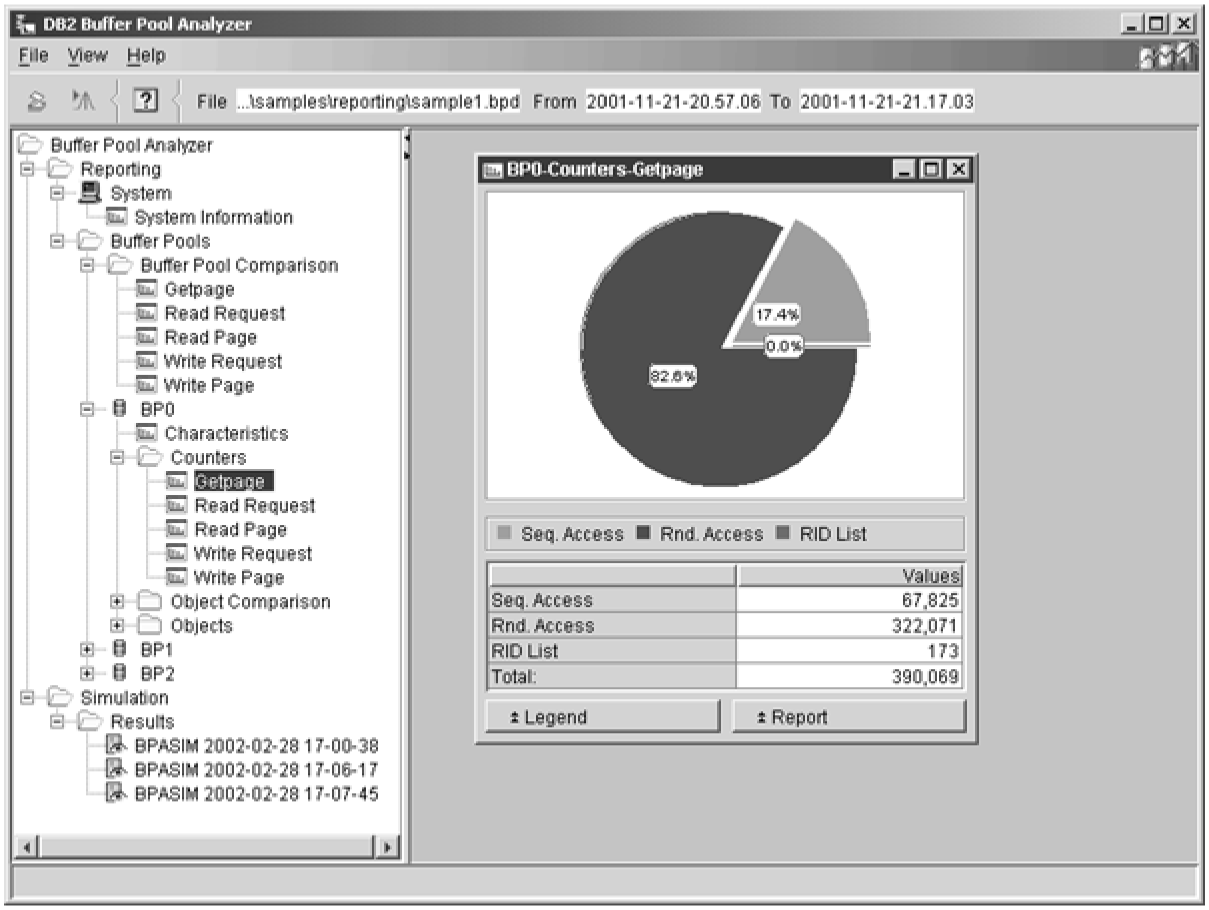
Write page 10149

*Figure 13. Viewing performance data – The Individual Buffer Pool Characteristics window*

The right pane shows general information, such as the buffer pool identifier and its virtual pool size, and thresholds for several types of buffer pool operations.

3. If you want to see buffer pool activity counters of the selected buffer pool, double-click **Counters**, then double-click one of the different counters, for example **Getpage**

   The following window is displayed:



File ...\samples\reporting\sample1.bpd From 2001-11-21-20.57.06 To 2001-11-21-21.17.03

| | Values |
|---|---|
| Seq. Access | 67,825 |
| Rnd. Access | 322,071 |
| RID List | 173 |
| Total: | 390,069 |

*Figure 14. Viewing performance data – The Individual Buffer Pool Counters window*

**Note:** The content pane displays only the active counters. Counters that have no activity are not displayed.

The right pane shows the types for the selected counter **Getpage** in the form of a pie chart. The types are **Sequential Access**, **Random Access**, and **RID List**.

4. If you want to compare object activities of the selected buffer pool, double-click **Object Comparison**. Then double-click one of the different counters, for example **Getpage**.

   A similar window is displayed. The right pane shows the types for the selected counter **Getpage** in the form of a bar chart. The types are **Sequential Access**, **Random Access**, and **RID List**. They are sorted in descending order.

5. If you want to see all objects and their counters of the selected buffer pool, double-click **Objects**. Then double-click one of the different counters, for example **Write Request**.

   A similar window is displayed. The right pane shows the types for the selected counter **Write Request** in the form of a pie chart. The types are **Synchronous** and **Asynchronous**.

# Chapter 7. Optimizing object placements and initial buffer pool sizes

This topic describes how to optimize the object placements in buffer pools and buffer pool sizes on the client.

### About this task

Related tasks:

- Before you can use this function, you must have performed the tasks described in:
  - Chapter 3, "Collecting data," on page 27
  - Chapter 4, "Creating activity reports and bpd files," on page 37. Here, only bpd files are of interest.
  - Chapter 11, "Downloading files from the host to the client," on page 151

  A buffer pool data file must be available on the client (the file with the recommended file name extension bpd).

**General remarks:**

1. Your client should have at least 40 MB of *available physical memory* (random access memory) for this function. You can check this in the Windows Task Manager. Close other applications, if you receive an `Out of memory` message. Note that free memory requirements increase with the number of objects to be treated.

   Example: 1 000 objects require approximately 60 MB, 25 000 objects require approximately 90 MB, 100 000 objects require approximately 230 MB, and 200 000 objects require more than 500 MB.
2. The data used for this function reflects the buffer pool activity for the interval for which trace data was collected and for the time frame that was specified with the **`GLOBAL`** and the **`BPACTIVITY`** command when the bpd file was created.
3. The bpd file used for the object placement function usually contains information about active and inactive objects. With the object placement function, you can specify whether to include the inactive objects (also called *unused objects*) in rule processing and object placement considerations. However, if inactive objects were explicitly excluded from the bpd file (by means of the **`BPACTIVITY FILE ACTIVEOBJECTS`** command), your specifications have no effect on the object placement result.
4. If you want to print the object placement results shown in your web browser, use your browser's printing capabilities to print the information. For colored printouts, check the browser settings.

   Example: In the Internet Explorer, click **Tools —► Internet Options —► Advanced**. Under **Printing**, select **Print background colors and images**.
5. In Db2 Performance Expert for z/OS, you can configure a different web browser. If required, click **Monitor —► Configuration —► Preferences** and follow the instructions.

## Starting the optimization function

### About this task

To start the optimization function, perform the following steps:

### Procedure

1. Start the IBM Db2 Buffer Pool Analyzer by double-clicking the icon on your Windows desktop, or use the **Start** push button on the taskbar to start this client application.

If you are using Db2 Performance Expert, click **Tools** > **Buffer Pool Analysis for z/OS** on the menu bar, or click the **Opens the Buffer Pool Analysis for z/OS window** toolbar button.

The Db2 Buffer Pool Analyzer - z/OS main window is displayed.

2. On the menu bar, click **File** > **Start Object Placement**. Alternatively, click the **Starts object placement** toolbar button.

   The Open dialog box is displayed. You use this dialog box to select and open a bpd file that contains the buffer pool activity data you want to use for the optimization.

   Several sample buffer pool data files (`*.bpd`) are delivered with Buffer Pool Analyzer in the … `\samples\reporting` folder. You can use them to become familiar with this function.
3. Continue with "Selecting and opening a buffer pool data file" on page 98.

## Selecting and opening a buffer pool data file

### About this task

To select and open a buffer pool data file, complete the following steps:

### Procedure

1. Select a folder from the **Look in** list where the bpd file is located.

   The **File of type** field shows the file name extension of buffer pool data files (`*.bpd`). If your bpd file lacks the recommended file name extension bpd, select `All files (*.*)` from the **File of type** list to see the appropriate files.
2. Click the bpd file to be opened.

   The **File name** field shows the name of the selected bpd file.
3. Click **Open**.

   The selected bpd file is opened and its content is checked. If the bpd file contains a concatenation of multiple sections, with performance data from different data sharing groups, different members of a data sharing group, or separated data collection time frames, only data from one section can be used for the object placement optimization. You need to select one of the sections found in the bpd file.

   Multiple sections in a bpd file are the result of multiple input data sets with trace data being used as input to create bpd files. For more information, see the `INPUTDD` statement in Chapter 4, "Creating activity reports and bpd files," on page 37 and Chapter 12, "Concatenating trace data for activity reports and bpd files," on page 153.

   - If the Buffer Pool Data File Section Selection dialog is displayed, select one section from the list and click **OK** to continue. You can click **Cancel** to return to the Open dialog. Note that you can expand the list by clicking the down arrow on the right side.

   The Db2 Buffer Pool Analyzer - Object Placement window is displayed. You see the first page of the object placement wizard, which guides you through the following steps:

   a. "Step 1: Selecting a pattern file" on page 99
   b. "Step 2: Editing a pattern file" on page 100
   c. "Step 3: Assigning objects to buffer pools" on page 103
   d. "Step 4: Setting the initial buffer pool sizes and characteristics" on page 105

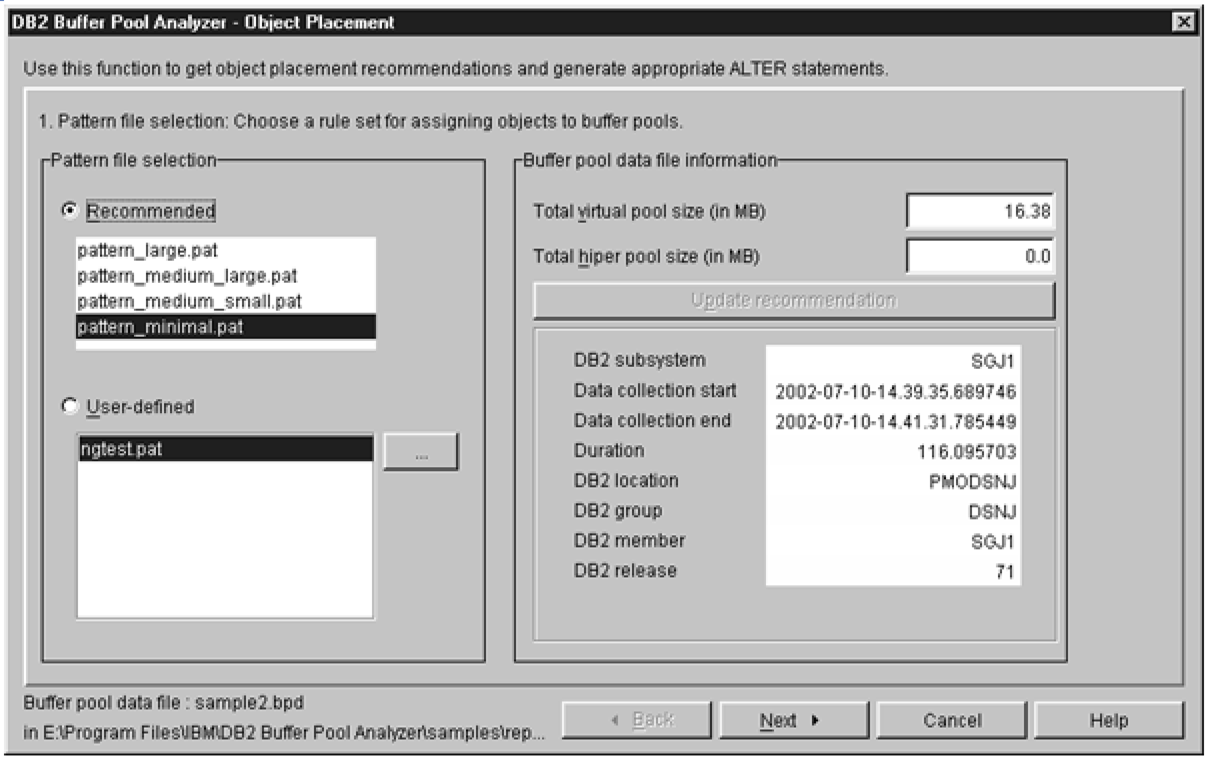
   When you have finished these steps, the object placement wizard closes, and you can continue with "Viewing the result of an optimization cycle" on page 107.

   You can navigate through the pages by clicking **Next** or **Back**. On all four pages, you can click **Cancel** to return to the main window, or **Help** to get help on the current page.

## Step 1: Selecting a pattern file

### About this task

When you have opened a bpd file, as described in "Selecting and opening a buffer pool data file" on page 98, the first page of the object placement wizard is displayed:



Use this function to get object placement recommendations and generate appropriate ALTER statements.

1. Pattern file selection: Choose a rule set for assigning objects to buffer pools.

pattern_large.pat
pattern_medium_large.pat
pattern_medium_small.pat
pattern_minimal.pat

ngtest.pat

| | |
|---|---|
| Total virtual pool size (in MB) | 16.38 |
| Total hiper pool size (in MB) | 0.0 |

| | |
|---|---|
| DB2 subsystem | SGJ1 |
| Data collection start | 2002-07-10-14.39.35.689746 |
| Data collection end | 2002-07-10-14.41.31.785449 |
| Duration | 116.095703 |
| DB2 location | PMODSNJ |
| DB2 group | DSNJ |
| DB2 member | SGJ1 |
| DB2 release | 71 |

Buffer pool data file : sample2.bpd
in E:\Program Files\IBM\DB2 Buffer Pool Analyzer\samples\rep...

*Figure 15. Object Placement – The Pattern File Selection window*

This page shows a list of pattern files, the available memory for buffer pools, and data about the Db2 subsystem from which data was collected.

- A pattern file contains a list of rules that determine which objects (table spaces and index spaces) should be placed in which buffer pool according to each object's characteristics.

  Initially, Buffer Pool Analyzer recommends one of four pattern files based on the total buffer pool size that is determined from the content of the bpd file. The rules in each pattern file are predefined. They do not vary with the content of a bpd file.

  If you edit a pattern file and save it under a user-defined name, as described in "Step 2: Editing a pattern file" on page 100, Buffer Pool Analyzer recommends this user-defined pattern file whenever a bpd file from the same Db2 subsystem is processed by the object placement wizard.
- The **Total virtual pool size (in MB)** value is determined from the content of the bpd file and reflect the values at the time the data was collected from the Db2 subsystem.
- Db2 subsystem and the data collection is also determined from the content of the bpd file.

You can use this page to select a different pattern file and to adjust the available memory for buffer pools. In "Step 4: Setting the initial buffer pool sizes and characteristics" on page 105, Buffer Pool Analyzer recommends a distribution of the available memory across the individual buffer pools. You can also accept the recommendations (the pattern file and the virtual pool) and proceed with the next step.

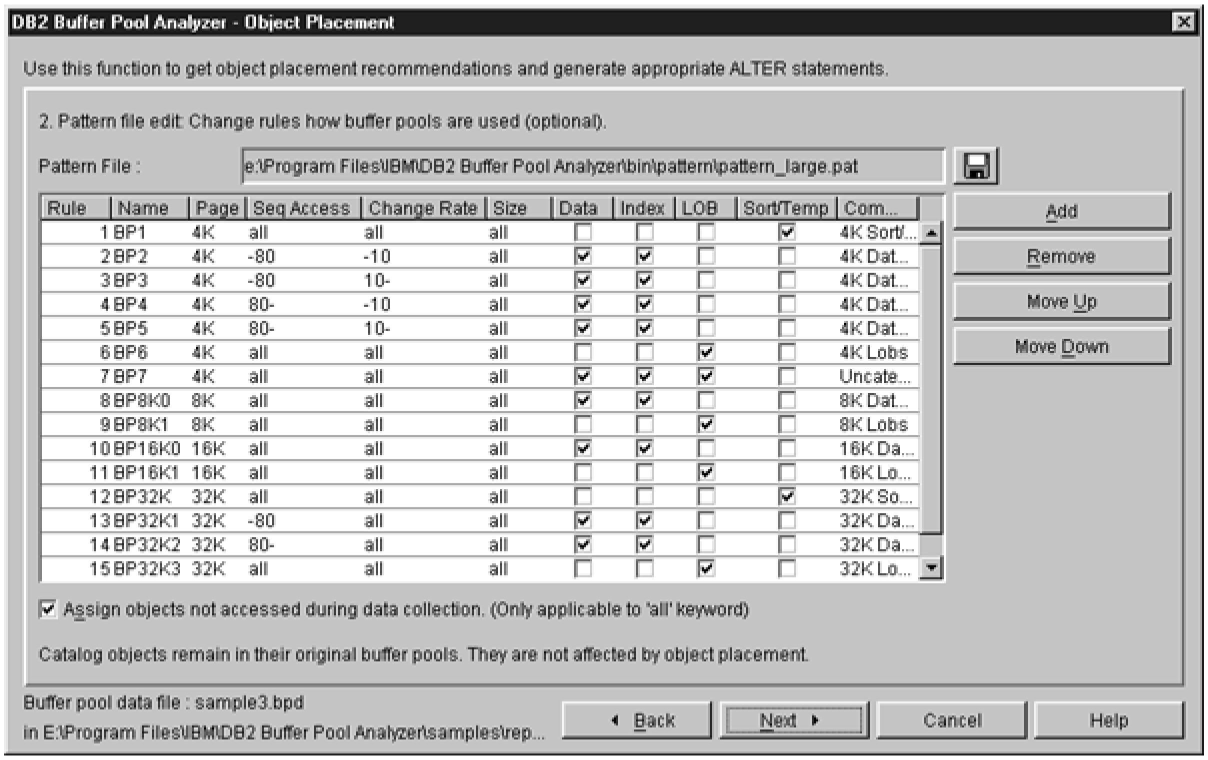
1. Under **`Pattern file selection`**, select **Recommended** or **User-defined**. Then click the pattern file that you want to use.
2. Under **Buffer pool data file information**, enter the value for **Total virtual pool size (in MB)** that you want the object placement wizard to use for the calculation of the initial buffer pool sizes.

3. Click **Next** and continue with "Step 2: Editing a pattern file" on page 100.

## Step 2: Editing a pattern file

### About this task

When you have performed "Step 1: Selecting a pattern file" on page 99, the second page of the object placement wizard is displayed:



Use this function to get object placement recommendations and generate appropriate ALTER statements.

2. Pattern file edit: Change rules how buffer pools are used (optional).

Pattern File : e:\Program Files\IBM\DB2 Buffer Pool Analyzer\bin\pattern\pattern_large.pat

| Rule | Name | Page | Seq Access | Change Rate | Size | Data | Index | LOB | Sort/Temp | Com... |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | BP1 | 4K | all | all | all | ☐ | ☐ | ☐ | ✓ | 4K Sort/... |
| 2 | BP2 | 4K | -80 | -10 | all | ✓ | ✓ | ☐ | ☐ | 4K Dat... |
| 3 | BP3 | 4K | -80 | 10- | all | ✓ | ✓ | ☐ | ☐ | 4K Dat... |
| 4 | BP4 | 4K | 80- | -10 | all | ✓ | ✓ | ☐ | ☐ | 4K Dat... |
| 5 | BP5 | 4K | 80- | 10- | all | ✓ | ✓ | ☐ | ☐ | 4K Dat... |
| 6 | BP6 | 4K | all | all | all | ☐ | ☐ | ✓ | ☐ | 4K Lobs |
| 7 | BP7 | 4K | all | all | all | ✓ | ✓ | ✓ | ☐ | Uncate... |
| 8 | BP8K0 | 8K | all | all | all | ✓ | ✓ | ☐ | ☐ | 8K Dat... |
| 9 | BP8K1 | 8K | all | all | all | ☐ | ☐ | ✓ | ☐ | 8K Lobs |
| 10 | BP16K0 | 16K | all | all | all | ✓ | ✓ | ☐ | ☐ | 16K Da... |
| 11 | BP16K1 | 16K | all | all | all | ☐ | ☐ | ✓ | ☐ | 16K Lo... |
| 12 | BP32K | 32K | all | all | all | ☐ | ☐ | ☐ | ✓ | 32K So... |
| 13 | BP32K1 | 32K | -80 | all | all | ✓ | ✓ | ☐ | ☐ | 32K Da... |
| 14 | BP32K2 | 32K | 80- | all | all | ✓ | ✓ | ☐ | ☐ | 32K Da... |
| 15 | BP32K3 | 32K | all | all | all | ☐ | ☐ | ✓ | ☐ | 32K Lo... |

[✓] Assign objects not accessed during data collection. (Only applicable to 'all' keyword)

Catalog objects remain in their original buffer pools. They are not affected by object placement.

Buffer pool data file : sample3.bpd
in E:\Program Files\IBM\DB2 Buffer Pool Analyzer\samples\rep...

*Figure 16. Object Placement – The Pattern File Editing window*

This page shows the object placement rules from the selected pattern file.

- The list of rules acts like a series of filters. The object placement wizard processes all objects (table spaces and index spaces) through these rules, starting at the top of the list. If the characteristics of an object match the criteria for a specific buffer pool, the object is assigned to this buffer pool.
- Each rule specifies a series of object characteristics as criteria for a buffer pool. If an object matches *all* criteria of a rule, the wizard recommends it for placement in the corresponding buffer pool. If an object does not match all criteria of a rule, it is passed to the next rule for evaluation.
- An object matches a rule if `Page Size`, `Seq Access`, `Change Rate` and `Size` match, and if an object is one of the *selected* data types `Data`, `Index`, `LOB`, or `Sort/Temp`. (*Selected* here means that a check box is marked. If all check boxes of a rule are cleared, an object does never match the rule.)
- In summary, the rules causes a 1:*n* mapping of buffer pools to objects. Objects with similar characteristics are assigned to the same buffer pool.

You can use this page to adjust the object placement rules for this session, and you can save the rules under a user-defined name for future use with performance data from the same Db2 subsystem.

- You can add more placement rules by clicking **Add**, and you can delete selected rules by clicking **Remove**.
- You can define more than one rule for a specific buffer pool, which permits a buffer pool to contain objects with different characteristics.
- You can also change the position of a placement rule in the list by clicking **Move Up** or **Move Down**.

- To change one of a rule's values, double-click the value and edit it. Then press Enter or select a different field.

If you adjust rules in pattern files:

- Ensure that all objects are covered by at least one rule. Otherwise, you cannot complete the next step.
- Place rules with restrictive criteria at the top of the list, and those with more general criteria at the bottom. Otherwise, the more specific rule might never become active.

  Example: Assume that you want objects with a `Change Rate` above 80 percent to be assigned to buffer pool BP3, all others to BP4. The recommended way is to specify the first rule for BP3 with a criteria range of `80-100` percent, followed by another rule for BP4 with a criteria range of `0-80` percent. Alternatively, the second rule could also have a criteria range of `0-100` percent, which matches everything that did not match the first rule.
- If criteria ranges are specified, as in `50-80` percent, the algorithms consider the lower bound as inclusive (≥), the upper bound as exclusive (<).

  Example: If two rules have specified criteria ranges of `50-80` and `80-90` percent, the first rule matches values equal or greater (≥) 50 and less than (<) 80, and the second rule matches values equal or greater (≥) 80 and less than (<) 90. The value 100 as an upper bound is an exception; it is interpreted as less or equal (≤) 100. Successive range specifications of, for example, `0-80` and `81-100` percent are likely to be erroneous.
- The specification of overlapping ranges in different rules for the same criterion is allowed. Nevertheless, the first rule has priority and consequently reduces the effect of any following rule to the non-overlapping part of the range.

  Example: Assume a single criterion where a rule specifies a criteria range of `10-70` percent, and a subsequent rule specifies a criteria range of `50-100` percent. Here, the overlapping range from 50 to 70 percent in the second rule is without effects because the first rule has priority. The second rule only gets objects for evaluation in the range above 70 percent.
- Note that Db2 catalog objects and directory objects always remain in their original buffer pool (which is BP0 for Db2 Version 7 and earlier). These objects are not taken into account by the object placement algorithms.

The following list describes the elements of a rule, as shown on the page:

**Rule**
Shows the numerical sequence in which the object placement rules are applied to each object.

You can change the sequence by selecting the rule, and clicking **Move Up** or **Move Down**.

**Name**
Shows a buffer pool name, such as BP0 or BP16K9.

**Page**
Shows the size of each buffer pool page. The size is implicitly extracted from the name of the buffer pool. Buffer pool BP32K, for example, always has a page size value of 32 KB.

**Seq Access**
Specify a percentage range as criteria for an object's sequential accesses. Objects with sequential accesses within the specified range, out of all accesses, are assigned to the specific buffer pool (if the other criteria are also met).

Example: A percentage range of `50-80` considers objects that are on average accessed sequentially 50 to less than 80 times out of 100 accesses.

You can also use the following notations to specify percentage ranges:

`-50` is the same as 0 to <50%
`50-` is the same as 50 to ≤100%
`all` is the same as 0 to ≤100%. If the **Assign objects not accessed during data collection** check box is selected, `all` also includes objects for which no sequential access characteristics could be

determined. (These objects are marked as `N/C` (not calculated), which means a formula could not be applied because of missing data.)

**Change Rate**
Specify a percentage range as criteria for an object's change rate. Objects with a change rate within the specified range, out of all accesses, are assigned to the specific buffer pool (if the other criteria are also met).

Example: A percentage of `50-80` considers objects that are changed 50 to less than 80 times out of 100 accesses.

You can also use the following notations to express percentage ranges:

`-50` is the same as 0 to <50%
`50-` is the same as 50 to ≤100%
`all` is the same as 0 to ≤100%. If the **Assign objects not accessed during data collection** check box is selected, `all` also includes objects for which no change rate characteristics could be determined. (These objects are marked as `N/C` (not calculated), which means a formula could not be applied because of missing data.)

**Size**
Specify a size range as criteria for an object's size. Objects with a size within the specified range are assigned to the specific buffer pool (if the other criteria are also met). Sizes are expressed as numbers of buffer pool pages. One buffer pool page can be 4, 8, 16, or 32 KB, depending on the page size of the buffer pool.

Example: A size range of `0-12` considers objects with a size of less than 12 buffer pool pages. For a buffer pool having a page size of 4 KB this affects objects having a size of less than 48 KB.

You can also use the following notations to express size ranges:

`-12` is the same as 0 to <12 pages
`50-` is the same as 50 to the maximum object size pages
`all` is the same as 0 to the maximum object size pages, and includes also objects for which the size could not be determined (marked as `?`).

**Data**
Select this check box if you want table space objects to be assigned to the specific buffer pool (if the other criteria are also met). Note that this field does not include LOB and Sort/Temp data. These must be explicitly selected if required.

**Index**
Select this check box if you want index space objects to be assigned to the specific buffer pool (if the other criteria are also met).

**LOB**
Select this check box if you want table space objects of data type LOB to be assigned to the specific buffer pool (if the other criteria are also met).

**Sort/Temp**
Select this check box if you want database objects of type Sort (work files) or Temp to be assigned to the specific buffer pool (if the other criteria are also met). This should only be selected for buffer pools having a page size of 4 KB or 32 KB.

**Comment**
Initially, this column shows a descriptive text for each rule. You can edit this information.

1. Review the rules on this page. Adjust them as required.
2. Use the **Assign objects not accessed during data collection** check box to indicate whether you want to include unused objects in the rule processing. Unused objects are table spaces and index spaces that are defined in the database catalog, but did not show any access or change information during the time trace data was collected. Note that unused (inactive) objects can be explicitly excluded from bpd files. If such bpd file was opened, the use of the check box has no effect.

If this check box is selected, unused objects are considered by the object placement rules. They match the `all` criteria for `Seq Access` and `Change Rate`. (Unused objects are marked as `N/C` (not calculated) in "Step 3: Assigning objects to buffer pools" on page 103, because they have no sequential access or change rate characteristics.)

If this check box is cleared, unused objects are left in their current buffer pools.

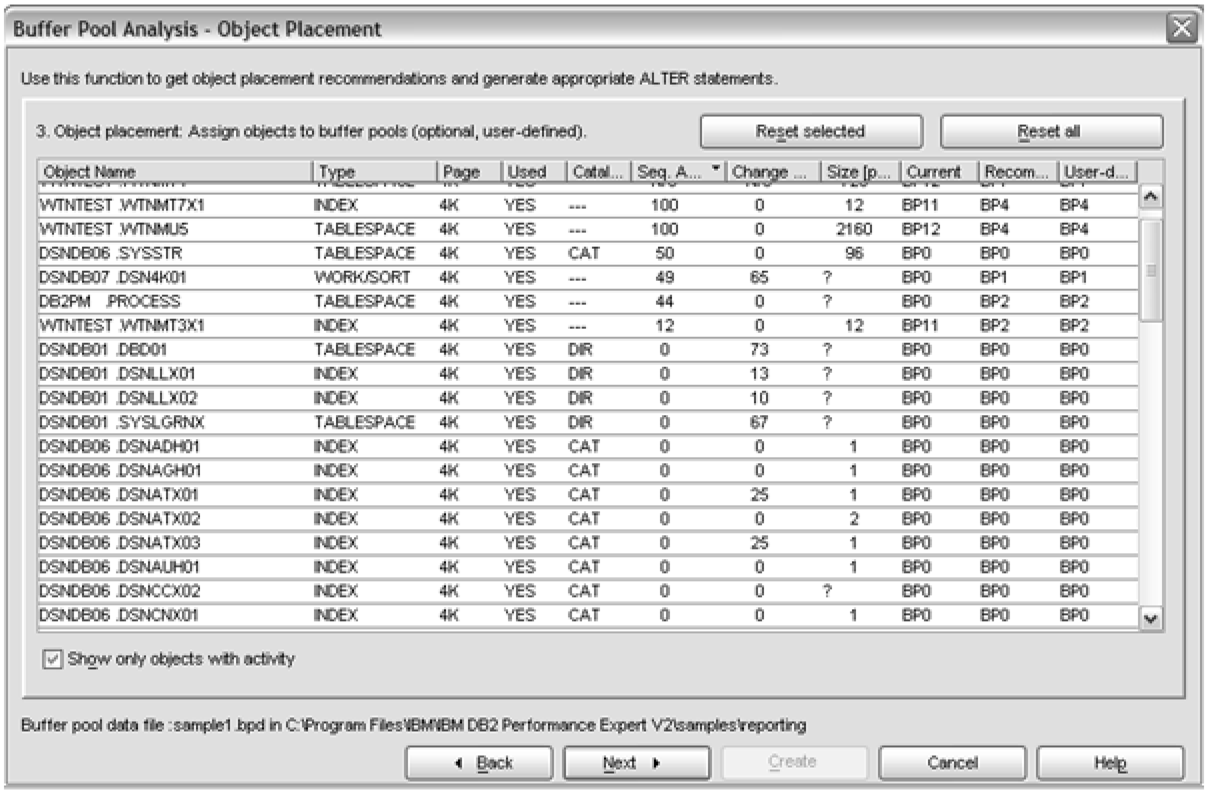
Note that any change that you make to this selection is retained for the next time you use the wizard.

3. If you want to save the pattern file, click the icon to the right of the current pattern file name and save it under a user-defined name. The file name extension should be `pat`.
4. Click **Next** and continue with "Step 3: Assigning objects to buffer pools" on page 103, or click **Back** to return to the previous page.

## Step 3: Assigning objects to buffer pools

### About this task

When you have performed "Step 2: Editing a pattern file" on page 100, the third page of the object placement wizard is displayed:



Use this function to get object placement recommendations and generate appropriate ALTER statements.

3. Object placement: Assign objects to buffer pools (optional, user-defined).

| Object Name | Type | Page | Used | Catal... | Seq. A... | Change ... | Size [p... | Current | Recom... | User-d... |
|---|---|---|---|---|---|---|---|---|---|---|
| WTNTEST .WTNMT7X1 | INDEX | 4K | YES | --- | 100 | 0 | 12 | BP11 | BP4 | BP4 |
| WTNTEST .WTNMU5 | TABLESPACE | 4K | YES | --- | 100 | 0 | 2160 | BP12 | BP4 | BP4 |
| DSNDB06 .SYSSTR | TABLESPACE | 4K | YES | CAT | 50 | 0 | 96 | BP0 | BP0 | BP0 |
| DSNDB07 .DSN4K01 | WORK/SORT | 4K | YES | --- | 49 | 65 | ? | BP0 | BP1 | BP1 |
| DB2PM .PROCESS | TABLESPACE | 4K | YES | --- | 44 | 0 | ? | BP0 | BP2 | BP2 |
| WTNTEST .WTNMT3X1 | INDEX | 4K | YES | --- | 12 | 0 | 12 | BP11 | BP2 | BP2 |
| DSNDB01 .DBD01 | TABLESPACE | 4K | YES | DIR | 0 | 73 | ? | BP0 | BP0 | BP0 |
| DSNDB01 .DSNLLX01 | INDEX | 4K | YES | DIR | 0 | 13 | ? | BP0 | BP0 | BP0 |
| DSNDB01 .DSNLLX02 | INDEX | 4K | YES | DIR | 0 | 10 | ? | BP0 | BP0 | BP0 |
| DSNDB01 .SYSLGRNX | TABLESPACE | 4K | YES | DIR | 0 | 67 | ? | BP0 | BP0 | BP0 |
| DSNDB06 .DSNADH01 | INDEX | 4K | YES | CAT | 0 | 0 | 1 | BP0 | BP0 | BP0 |
| DSNDB06 .DSNAGH01 | INDEX | 4K | YES | CAT | 0 | 0 | 1 | BP0 | BP0 | BP0 |
| DSNDB06 .DSNATX01 | INDEX | 4K | YES | CAT | 0 | 25 | 1 | BP0 | BP0 | BP0 |
| DSNDB06 .DSNATX02 | INDEX | 4K | YES | CAT | 0 | 0 | 2 | BP0 | BP0 | BP0 |
| DSNDB06 .DSNATX03 | INDEX | 4K | YES | CAT | 0 | 25 | 1 | BP0 | BP0 | BP0 |
| DSNDB06 .DSNAUH01 | INDEX | 4K | YES | CAT | 0 | 0 | 1 | BP0 | BP0 | BP0 |
| DSNDB06 .DSNCCX02 | INDEX | 4K | YES | CAT | 0 | 0 | ? | BP0 | BP0 | BP0 |
| DSNDB06 .DSNCNX01 | INDEX | 4K | YES | CAT | 0 | 0 | 1 | BP0 | BP0 | BP0 |

[✓] Show only objects with activity

Buffer pool data file :sample1.bpd in C:\Program Files\IBM\IBM DB2 Performance Expert V2\samples\reporting

*Figure 17. Object Placement – The Object Assignment window*

This page shows the object placements that Buffer Pool Analyzer recommends based on the previously specified placement rules.

- Buffer Pool Analyzer has scanned the bpd file and has analyzed the characteristics of each table space and index space.
- The `Current` column shows the placement of objects at the time the data was collected.
- The `Recommended` column shows the recommended object to buffer pool assignments, as calculated by Buffer Pool Analyzer. The calculations are based on the placement rules for each buffer pool and the object's characteristics.
- The `User-defined` column is initially identical with the `Recommended` column, and is intended to change assignments as described later in this topic.

You can use this page to change the object placements, which means, you can assign specific objects to buffer pools other than the ones determined by the placement rules. You might want to do this, for example, to assign certain objects to a particular large buffer pool to guarantee high performance for applications using these objects. These *user-defined* assignments overwrite the assignments recommended by Buffer Pool Analyzer.

- To change the assignment of an object, double-click the appropriate buffer pool name in the `User-defined` column and edit the buffer pool name. Then press Enter or select a different field.
- To reset a `User-defined` assignment of an object to the `Recommended` assignment, select the object and click **Reset selected**.

  You can reset multiple assignments by selecting multiple objects. To select a range of successive objects, click the first object, hold down the Shift key, then click the last object in the sequence. To select several separate objects, click the first object, hold down the Ctrl key, then click the other objects as required. To select all objects, press Ctrl+A.

The following list describes the elements of an assignment, as shown on the page. Note that you can sort the list by clicking a column header of choice one or more times. Small arrows indicate the sort order.

**Object Name**
Shows the name of an object, such as a table space or index space. If you selected the **Assign objects not accessed during data collection** check box on the previous page, this column also shows the names of unused objects. Note that unused (inactive) objects can be explicitly excluded from bpd files. If such bpd file was opened, the use of the check box has no effect.

**Type**
Shows the type of an object, as `TABLESPACE`, `INDEX`, `LOB`, `TEMP`, or `WORK/SORT`.

**Page**
Shows the page size of the buffer pool to which an object is currently assigned. If you assign an object to a different buffer pool, the new buffer pool must have the same page size.

**Used**
Shows whether an object was used during the time for which performance data was collected.

If an object was used (indicated as YES), Buffer Pool Analyzer assigned the object to a buffer pool according to the object's characteristics.

If an object was not used (indicated by `NO`), Buffer Pool Analyzer retains the original assignment, unless you checked the **Assign objects not accessed during data collection** check box on the previous page. In this case, Buffer Pool Analyzer assigned the object using only the object's `Page Size`, `Size`, and data type (`Data`, `Index`, `LOB`, or `Sort/Temp`).

**Catalog/Directory**
Shows whether this object is part of the database catalog (`CAT`) or database directory (`DIR`). No entry indicates that the object belongs to neither.

**Seq. Access [%]**
Shows how often the object was accessed sequentially, as a percentage of all accesses to this object.

`N/C` (not calculated) means that the percentage could not be computed because the object was not used.

**Change Rate [%]**
Shows how often the object was changed, as a percentage of all accesses to this object.

`N/C` (not calculated) means that the percentage could not be computed because the object was not used.

**Size [pages]**
Shows the size of an object, expressed as the number of buffer pool pages.

Example: For a buffer pool having a page size of 4 KB, a value of 12 corresponds to an object size of 48 KB.

**Current**
Shows the name of the buffer pool where an object is currently placed (at the time the data was collected).

**Recommended**
Shows the name of the buffer pool that Buffer Pool Analyzer recommends for the object.

**User-defined**
Initially, this column shows the same buffer pool names as in the `Recommended` column.

**Note:** If a field in this column is empty, an object was not assigned to a buffer pool because no matching rule was found. You must ensure that there are no empty user-defined fields before continuing. Return to “Step 2: Editing a pattern file” on page 100 and add rules that cover such objects.

If you want to assign objects to specific buffer pools (other than those shown in the corresponding fields), you can change the assignment in this column as described previously.

You can also reset your changes to the recommended values by clicking **Reset selected** or **Reset all**. The latter resets the assignments of all objects in the window, but not those that are hidden if the **Show only objects with activity** check box is selected. If you want to reset all used and unused objects, clear the check box (which shows used and unused objects in the window), click **Reset all**, then select the check box again.

Restriction: Note that catalog objects and directory objects cannot be assigned to different buffer pools.

1. Review the assignments on this page. Change them in the `User-defined` column, if required.
2. Use the **Show only objects with activity** check box to indicate whether you want to see only used objects, or whether you want to see also unused objects. See step “2” on page 102 for details about unused objects.

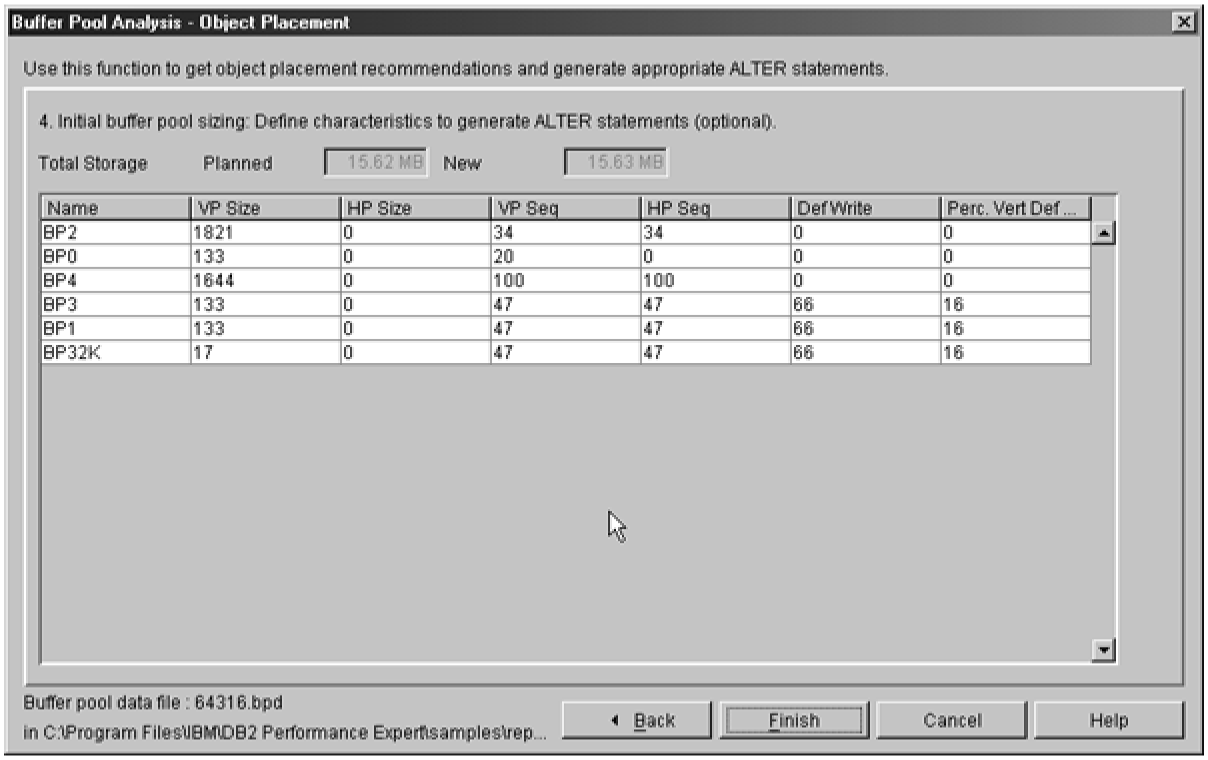
   If this check box is cleared, unused objects are included in the list of objects (their assignment to buffer pools is activated by the **Assign objects not accessed during data collection** check box on the previous page).

   Note that the state of the check box is retained for the next time you use the wizard.
3. Click **Next**. If any rows contain errors, they are highlighted in red. You must correct these errors before you can continue to “Step 4: Setting the initial buffer pool sizes and characteristics” on page 105. You can click **Back** to return to the previous page.

## Step 4: Setting the initial buffer pool sizes and characteristics

### About this task

When you have performed “Step 3: Assigning objects to buffer pools” on page 103, the fourth page of the object placement wizard is displayed:


Buffer Pool Analysis - Object Placement

Use this function to get object placement recommendations and generate appropriate ALTER statements.

4. Initial buffer pool sizing: Define characteristics to generate ALTER statements (optional).

Total Storage Planned 15.62 MB New 15.63 MB

| Name | VP Size | HP Size | VP Seq | HP Seq | Def Write | Perc. Vert Def ... |
|---|---|---|---|---|---|---|
| BP2 | 1821 | 0 | 34 | 34 | 0 | 0 |
| BP0 | 133 | 0 | 20 | 0 | 0 | 0 |
| BP4 | 1644 | 0 | 100 | 100 | 0 | 0 |
| BP3 | 133 | 0 | 47 | 47 | 66 | 16 |
| BP1 | 133 | 0 | 47 | 47 | 66 | 16 |
| BP32K | 17 | 0 | 47 | 47 | 66 | 16 |

Buffer pool data file : 64316.bpd
in C:\Program Files\IBM\DB2 Performance Expert\samples\rep...

*Figure 18. Object Placement – The Buffer Pool Size and Characteristics window*

This page shows the recommended initial buffer pool sizes and thresholds for the individual buffer pools, and shows the total storage required for these recommendations.

- The value in the **Planned** field, shown at the top of the page, is the sum of the **Total virtual pool size** and the **Total hiper pool size**. These are the pool sizes (in MB) that you have specified in "Step 1: Selecting a pattern file" on page 99.
- The object placement and sizing algorithm has used this total storage to calculate the initial virtual sizes for the individual buffer pools, based on the object placements specified in the `User-defined` column on the previous page.
- The sum of all buffer pool sizes (the recommended number of pages for a buffer pool, multiplied by the page size of the buffer pool in MB) yields the planned size in MB.
- By default, the sizing algorithm allocates a minimum buffer pool size of 20% of the average buffer pool size for buffer pools to which objects are assigned.

You can use this page to adjust the initial buffer pool sizes and thresholds of individual buffer pools. Buffer Pool Analyzer will use the adjusted values when it generates the Db2 ALTER BUFFERPOOL commands and SQL ALTER statements.

- If you change the virtual buffer pool size of one of the listed buffer pools, the `New` size, at the top of the page, reflects the new sum (in MB). Initially, **Planned** and **New** show the same value. (Small differences are possible because only integer pages are calculated, not fractions.)
- To change a value, double-click the value and edit it. Then, press Enter or select a different field.

Note that the size of buffer pools that no longer have objects assigned to them is set to zero. When you apply the recommendations from the object placement and buffer pool sizing, access to objects assigned to these buffer pools is disabled.

The following list describes the elements of the buffer pool sizings, as shown on the page. Note that you can edit these values, except the buffer pool names. For more detailed information, see also "The Buffer Pool Characteristics section" on page 50. Note that you can sort the list by clicking a column header of choice one or more times. Small arrows indicate the sort order. By default, the list is sorted by Name.

**Name**
Shows the name of the buffer pool.

**VP Size [pages]**
Shows the recommended virtual buffer pool size, in number of pages, for the affected buffer pool.

**VP Seq [%]**
Shows the recommended virtual pool sequential steal threshold (VPSEQT) for the affected buffer pool. This is the portion of the buffer pool that can be occupied by sequentially accessed pages. For more details, see `Thresholds – Virtual sequential` in "The Buffer Pool Characteristics section" on page 50.

**DefWrite [%]**
Shows the recommended deferred write threshold (DWQT) for the affected buffer pool. This is the percentage of the buffer pool that can be occupied by unavailable pages. For more details, see `Thresholds – Deferred write` in "The Buffer Pool Characteristics section" on page 50.

**VertDefWrite [%]**
Shows the recommended vertical deferred write threshold (VDWQT) for the affected buffer pool. This is the percentage of the buffer pool that can be occupied by updated pages of a single page set. For more details, see `Thresholds – Vert deferred write` in "The Buffer Pool Characteristics section" on page 50.

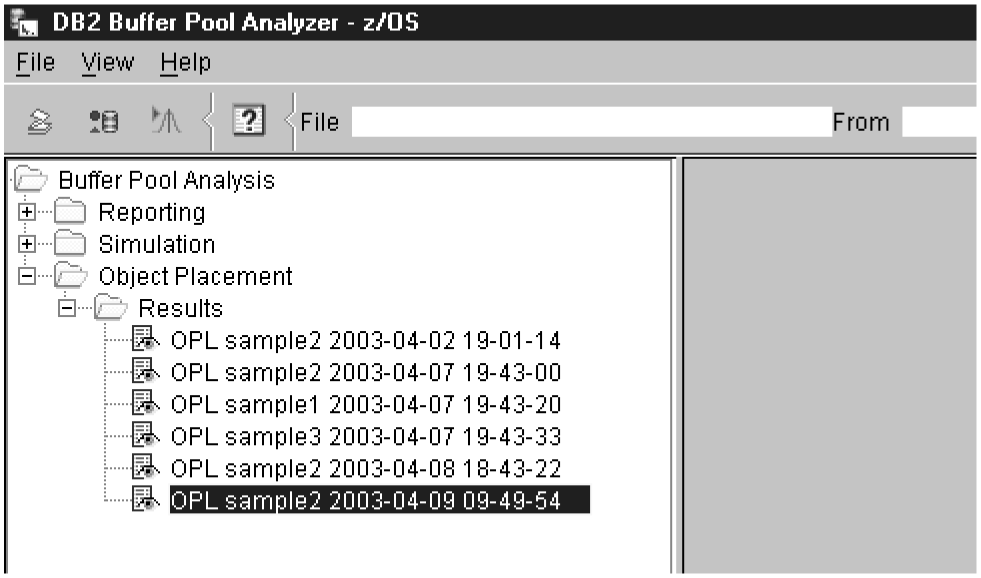
1. Review the recommended initial buffer pool sizes and thresholds. Adjust them, if required.
2. Click **Finish**, or **Back** to return to the previous page.

   When the object placement wizard finishes, the result is immediately shown in a new browser window. In addition, the result is saved in the **Results** subfolder of the **Object Placement** folder for later viewing.
3. Continue with "Viewing the result of an optimization cycle" on page 107, or close the browser window and return to the Buffer Pool Analyzer main window.

## Viewing the result of an optimization cycle

### About this task

When the object placement wizard finishes, the Buffer Pool Analyzer main window shows the result in the **Results** subfolder of the **Object Placement** folder. The subfolder can contain results from several optimization cycles. The result from the most recent optimization is highlighted.



Buffer Pool Analysis
Reporting
Simulation
Object Placement
Results
OPL sample2 2003-04-02 19-01-14
OPL sample2 2003-04-07 19-43-00
OPL sample1 2003-04-07 19-43-20
OPL sample3 2003-04-07 19-43-33
OPL sample2 2003-04-08 18-43-22
OPL sample2 2003-04-09 09-49-54

*Figure 19. Object Placement – The Results Selection window*

Results are named `OPL` *`<bpd_file> <date> <time>`*, whereby OPL stands for Object Placement, *`<bpd_file>`* for the name of the bpd file that was used for the optimization, *`<date>`* and *`<time>`* for the date and time when the optimization started.

1. If you want to delete results from the folder, select it by clicking it. Then press the Delete key. To delete all results, right-click **Results**. Then click **Delete all**. You are asked to confirm the deletion.

   Note that results remain on the hard disk drive and take up space until they are deleted. They are usually located in folder `C:\Documents and Settings \`*`<userid>`*` \db2pev`*`<version>`*` \object placement reports`. However, because of their special format, do not manipulate the folder contents manually.

2. To view the result of an object placement, double-click it, or select it and press Enter.

   The result is shown in your web browser and contains the following information:

   - A section showing details about the content of the bpd file that was used for this object placement optimization, such as the name of the Db2 subsystem from which data was collected, and the start and end times and the duration of the data collection. The bpd file is the one you selected when you performed the steps in "Selecting and opening a buffer pool data file" on page 98. The details are identical with the information that was shown in the Pattern File Selection window (see Figure 15 on page 99).
   - A section showing which pattern file and which object placement rules were used for this object placement optimization. This information corresponds to your specifications in "Step 1: Selecting a pattern file" on page 99 and "Step 2: Editing a pattern file" on page 100.
   - A section listing the other options that you specified for this object placement optimization.
   - A list of ALTER BUFFERPOOL commands for resizing and changing buffer pool characteristics.

     Example:

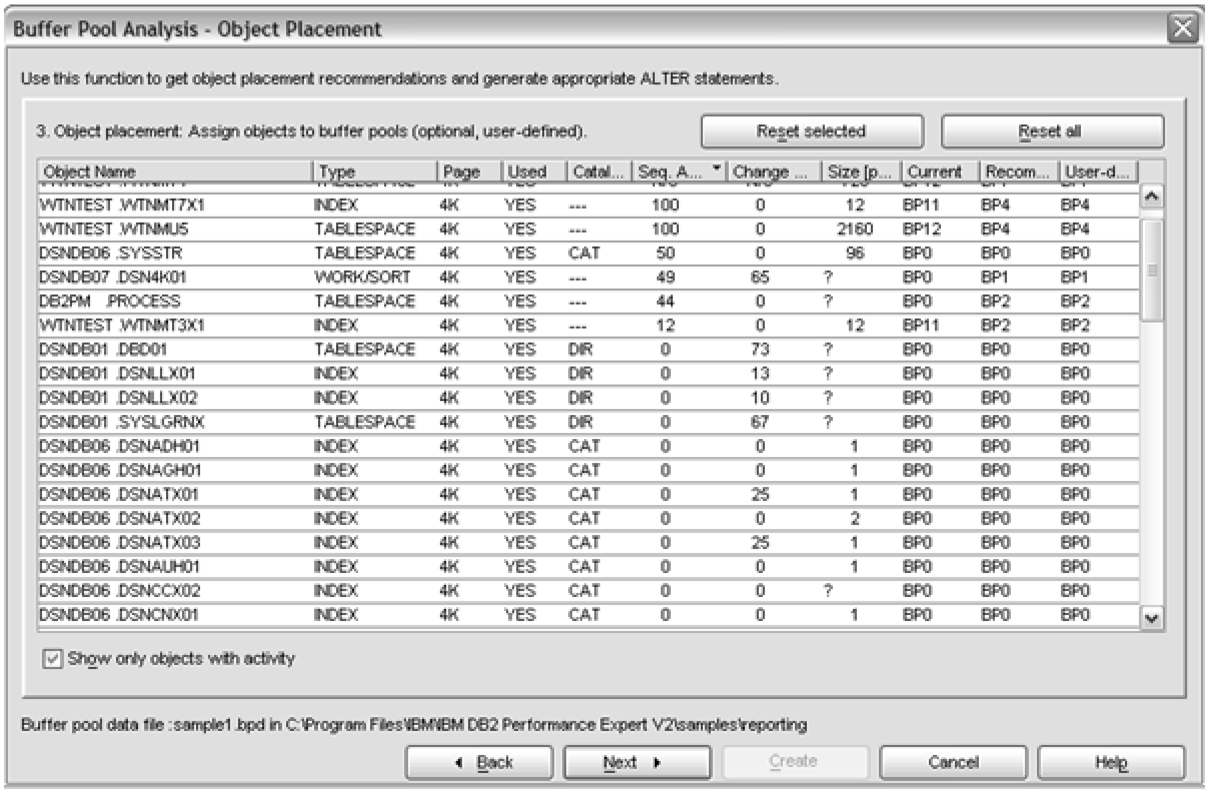
     ```
     ALTER BUFFERPOOL(BP0) VPSIZE(868) HPSIZE(0) VPSEQT(20) DWQT(0) VDWQT(0,0)
     ALTER BUFFERPOOL(BP7) VPSIZE(120) HPSIZE(0) VPSEQT(40) DWQT(10) VDWQT(3,0)
     ALTER BUFFERPOOL(BP32K) VPSIZE(15) HPSIZE(0) VPSEQT(100) DWQT(70) VDWQT(50,0)
     ALTER BUFFERPOOL(BP4) VPSIZE(1750) HPSIZE(0) VPSEQT(99) DWQT(0) VDWQT(0,0)
     ALTER BUFFERPOOL(BP1) VPSIZE(5) HPSIZE(0) VPSEQT(100) DWQT(70) VDWQT(50,0)
     ALTER BUFFERPOOL(BP2) VPSIZE(790) HPSIZE(0) VPSEQT(20) DWQT(0) VDWQT(0,0)
     ALTER BUFFERPOOL(BP3) VPSIZE(426) HPSIZE(0) VPSEQT(100) DWQT(70) VDWQT(50,0)
     ```

   - A list of STOP DATABASE commands. In data sharing environments, these commands *must* be performed before applying the following statements.
   - A list of ALTER INDEX and ALTER TABLESPACE statements for placing each reassigned object in its new buffer pool.

     Example:

     ```
     ALTER INDEX CC390.UTLEX01 BUFFERPOOL BP7;
     ALTER INDEX CC390.UTLSTX01 BUFFERPOOL BP7;
     ALTER INDEX CC390.UTPEBX01 BUFFERPOOL BP7;
     ALTER INDEX CC390.UTPETX01 BUFFERPOOL BP7;
     ALTER INDEX CC390.UTPEX01 BUFFERPOOL BP7;
     ALTER INDEX CC390.UTPRCX01 BUFFERPOOL BP7;
     ALTER TABLESPACE CC390.UTPROC BUFFERPOOL BP7;
     ALTER INDEX CC390.UTRESTART2X BUFFERPOOL BP7;
     ALTER INDEX CC390.UTRESTARTX BUFFERPOOL BP7;
     ALTER TABLESPACE CC390.UTRSTRT BUFFERPOOL BP7;
     ALTER TABLESPACE CC390.UTTEMPL BUFFERPOOL BP7;
     ```

   - A list of START DATABASE commands, corresponding to the preceding STOP DATABASE commands. Required in data sharing environments.
   - An object placement overview, which summarizes the information from the object placement task. Note that changed object placements are marked by a different color when the information is shown in the browser window. The following example shows the Object Placement window.


Buffer Pool Analysis - Object Placement

Use this function to get object placement recommendations and generate appropriate ALTER statements.

3. Object placement: Assign objects to buffer pools (optional, user-defined).

| Object Name | Type | Page | Used | Catal... | Seq. A... | Change ... | Size [p... | Current | Recom... | User-d... |
|---|---|---|---|---|---|---|---|---|---|---|
| WTNTEST .WTNMT7X1 | INDEX | 4K | YES | --- | 100 | 0 | 12 | BP11 | BP4 | BP4 |
| WTNTEST .WTNMU5 | TABLESPACE | 4K | YES | --- | 100 | 0 | 2160 | BP12 | BP4 | BP4 |
| DSNDB06 .SYSSTR | TABLESPACE | 4K | YES | CAT | 50 | 0 | 96 | BP0 | BP0 | BP0 |
| DSNDB07 .DSN4K01 | WORK/SORT | 4K | YES | --- | 49 | 65 | ? | BP0 | BP1 | BP1 |
| DB2PM .PROCESS | TABLESPACE | 4K | YES | --- | 44 | 0 | ? | BP0 | BP2 | BP2 |
| WTNTEST .WTNMT3X1 | INDEX | 4K | YES | --- | 12 | 0 | 12 | BP11 | BP2 | BP2 |
| DSNDB01 .DBD01 | TABLESPACE | 4K | YES | DIR | 0 | 73 | ? | BP0 | BP0 | BP0 |
| DSNDB01 .DSNLLX01 | INDEX | 4K | YES | DIR | 0 | 13 | ? | BP0 | BP0 | BP0 |
| DSNDB01 .DSNLLX02 | INDEX | 4K | YES | DIR | 0 | 10 | ? | BP0 | BP0 | BP0 |
| DSNDB01 .SYSLGRNX | TABLESPACE | 4K | YES | DIR | 0 | 67 | ? | BP0 | BP0 | BP0 |
| DSNDB06 .DSNADH01 | INDEX | 4K | YES | CAT | 0 | 0 | 1 | BP0 | BP0 | BP0 |
| DSNDB06 .DSNAGH01 | INDEX | 4K | YES | CAT | 0 | 0 | 1 | BP0 | BP0 | BP0 |
| DSNDB06 .DSNATX01 | INDEX | 4K | YES | CAT | 0 | 25 | 1 | BP0 | BP0 | BP0 |
| DSNDB06 .DSNATX02 | INDEX | 4K | YES | CAT | 0 | 0 | 2 | BP0 | BP0 | BP0 |
| DSNDB06 .DSNATX03 | INDEX | 4K | YES | CAT | 0 | 25 | 1 | BP0 | BP0 | BP0 |
| DSNDB06 .DSNAUH01 | INDEX | 4K | YES | CAT | 0 | 0 | 1 | BP0 | BP0 | BP0 |
| DSNDB06 .DSNCCX02 | INDEX | 4K | YES | CAT | 0 | 0 | ? | BP0 | BP0 | BP0 |
| DSNDB06 .DSNCNX01 | INDEX | 4K | YES | CAT | 0 | 0 | 1 | BP0 | BP0 | BP0 |

☑ Show only objects with activity

Buffer pool data file :sample1 .bpd in C:\Program Files\IBM\IBM DB2 Performance Expert V2\samples\reporting

*Figure 20. Example of optimization result, showing an object placement overview*

3. A section listing the original and recommended buffer pool characteristics (not shown here).
4. Use the hypertext links in the browser window to navigate through the report.

## Applying changes to a Db2 subsystem

### About this task

You can use the Db2 Commands feature of Db2 Performance Expert (for the ALTER BUFFERPOOL commands).

To apply the SQL statements, you can copy them to SPUFI, or you can run them via DB2 Connect. Additionally, consider the following tips when you are going to apply the proposed changes:

- If you see a buffer pool size set to zero in the optimization result, double-check that this buffer pool is no longer used. Consider the eventuality that this buffer pool did not show any activity during the data collection time, but an unused object can become active under certain circumstances.
- If possible, you should stop the database before applying the changes, especially in a data sharing environment. Also consider applying the changes during planned outages.
- If the virtual buffer pool size is limited, or the proposed size is less than the original size, you should first apply the changes that reduce the sizes of buffer pools, then apply the changes that increase the sizes of buffer pools. The opposite order can result in insufficient virtual storage. However, buffer pools are not always freed immediately. Db2 reduces the sizes of buffer pools by first identifying buffers that are to be deleted. That means, those buffers cannot be used again. Db2 releases buffers immediately or at a later moment, whatever is more appropriate. Therefore, it is good practice to ensure that buffers have actually been released before enlarging buffer pools.
- After applying the ALTER INDEX and ALTER TABLESPACE statements, the reassignments of objects to different buffer pools remain pending until Db2 happens to close and reopen the data sets of the changed page sets, which depends on several parameters. If immediate reassignments are required, you need to stop and start the database.

- For details about the Db2 command ALTER BUFFERPOOL see the *IBM DB2 11 for z/OS: Command Reference*.
- For details about the SQL statements ALTER and CREATE see the *IBM DB2 11 for z/OS: SQL Reference*.

## Performance-related tips

### About this task

The following tips might be useful when you perform an optimization:

- Generally, objects with similar access characteristics should be grouped and placed in the same buffer pool.

  Mixing page sets with a high sequential access characteristic with those that exhibit random access characteristics will be detrimental to both types of page sets. The pages belonging to the predominantly random page set could be stolen by the prefetch on the page set with sequential characteristic and it will increase the number of I/Os for the random page set.

  The more you separate sequential processing from random processing, the more benefit you get. To achieve this, you can adjust the sequential threshold of each buffer pool. See also the **Seq Access** column in Figure 16 on page 100.
- Assigning table spaces and index spaces into separate buffer pools that have sequential and random data access can have a more positive impact on overall buffer pool efficiency. It is generally better to put indexes in a separate buffer pool from the data because the access characteristics for indexes are usually very different from the data. See also the **User-defined** column in Figure 17 on page 103.

  Exceptions might be objects that are really small, such as lookup tables.
- Do not place any other objects in the buffer pool that is used by WORK/SORT data sets (DSNDB07), because of the intensive and special usage characteristics.
- BP0 should be used exclusively for Db2 catalog objects and directory objects. These objects are not taken into account by the object placement algorithms.
- A general recommendation is to distribute the objects in different pools based on the access intensity and the number of buffers the page set needs in the pool. The object placement function recommends the optimal size and thresholds for the buffer pools. The simulation function can be used to simulate if additional buffers to the pool will have an effect on reducing the buffer pool misses.
- Validate the recommended values for deferred write threshold (DWQT) and vertical deferred write threshold (VDWQT). If these values allow a particular page set to monopolize the buffer pool by filling up with too many changed pages, the amount of subsequent asynchronous writes can flood the I/O subsystem all at once, thus adversely affecting overall throughput. Usually, it is better to distribute these writes evenly over time rather than to have peaks. However, these thresholds depend on your workload and the type and size of data being cached.
- The first-in-first-out (FIFO) page steal algorithm is recommended, instead of the default Least-Recently-Used (LRU) algorithm, in either of the following situations:
  - For data and indexes residing entirely in the buffer pool
  - For objects with very low buffer pool hit ratio (< 1%)

  The FIFO algorithm, which can be specified by the ALTER BUFFERPOOL PGSTEAL command, will reduce processor cost under these conditions. See “The Buffer Pool Characteristics section” on page 50 for more information about the page steal methods.
- For buffer pools with high I/O rates (a high number of pages read or written) also consider the long-term page fix option for buffer pools, introduced with Db2 UDB for z/OS Version 8. This option fixes a buffer pool in real storage for an extended period of time and can help reduce the number of I/Os for I/O-intensive buffer pools. For more information, see the description of the ALTER BUFFERPOOL command and its PGFIX keyword in the *IBM DB2 11 for z/OS: Command Reference*.

# Chapter 8. Simulating buffer pool behavior

This topic describes how to simulate buffer pool behavior on the client.

### About this task

Related tasks:

- Before you can use this function, you must have performed the tasks described in:
  - Chapter 3, "Collecting data," on page 27. The collected data can be made available for the simulation function in uncompressed or compressed format.
  - Chapter 11, "Downloading files from the host to the client," on page 151

    A trace data file must be available on the client (the file with the recommended file name extension `trace`, or `terse` if compressed).

**General remarks:**

1. Your client should have approximately 40 MB of *available physical memory* (random access memory) for this function. You can check this in the Windows Task Manager. Close other applications, if you receive an `Out of memory` message.
2. The data used for this function reflects the buffer pool activity for the interval for which trace data was collected.
3. Data used for simulations must be collected with short record format, continuously, and should have a record lost rate of less than 2%. If required, see "Configuring a collect task" on page 31 and "Interpreting trace status summary and trace messages" on page 33 for more details.
4. The simulation function can handle trace data files of up to 2 GB.

   This size limit pertains to compressed and uncompressed trace data files. Uncompression requires additional time when a compressed trace data file is opened. Uncompression does not require additional disk space because uncompressed data is directly written into memory.
5. The time to preprocess the trace data file and the time to perform a simulation very largely depends on the number of active objects in the trace data file, the number of different buffer pools, and the buffer pool sizes to be simulated. Note that a compressed trace data file contains approximately four times as much data than an uncompressed file.

   Example: On a 2.4 GHz client, a 1 GB trace data file takes roughly 1.5 minutes to be preprocessed and approximately five minutes to simulate four buffer pools from 25 000 to 1 000 000 pages (40 sizes). Note that a simulation runs considerably slower if other tasks are using the processor at the same time.
6. If you want to print the simulation results shown in your web browser, use your browser's printing capabilities to print the information. For colored printouts, check the browser settings.

   Example: In the Internet Explorer, click **Tools —► Internet Options —► Advanced**. Under **Printing**, select **Print background colors and images**.
7. In Db2 Performance Expert for z/OS, you can configure a different web browser. If required, click **Monitor —► Configuration —► Preferences** and follow the instructions.

## Starting the simulation function

### About this task

To start the simulation function, perform the following steps:

### Procedure

1. Start the IBM Db2 Buffer Pool Analyzer by double-clicking the icon on your Windows desktop, or use the **Start** push button on the taskbar to start this client application.

   If you are using Db2 Performance Expert, click **Tools** > **Buffer Pool Analysis for z/OS** on the menu bar, or click the **Opens the Buffer Pool Analysis for z/OS window** toolbar button.

   The Db2 Buffer Pool Analyzer - z/OS main window is displayed.
2. On the menu bar, click **File** > **Start Simulation**. Alternatively, click the **Starts simulating buffer pools** toolbar button.

   To directly simulate Object Placement results right-click the object placement result file and select **Simulate...**.

   **Note: Simulate...** passes the object placement recommendations to the simulation function. If the files match, the object placement output is used, otherwise an error message is displayed.

   The Open dialog box is displayed. You use this dialog box to select and open a trace data file that contains the buffer pool activity data you want to use for the simulation.

   Several sample buffer pool trace data files (`*.trace`) are delivered with Buffer Pool Analyzer in the …`\samples\simulation` folder. You can use them to become familiar with this function.
3. Continue with "Selecting and opening a trace data file" on page 112.

## Selecting and opening a trace data file

### About this task

To select and open a trace data file, complete the following steps:

### Procedure

1. Select a folder from the **Look in** list where the trace data file is located

   The **File of type** field shows the file name extension of buffer pool trace data files (`*.trace` for uncompressed files and `*.terse` for compressed files). If your trace data file lacks the recommended file name extension `trace` or `terse`, select `All files (*.*)` from the **File of type** list to see the appropriate files.
2. Click the trace data file to be opened.

   The **File name** field shows the name of the selected trace data file.
3. Click **Open**.

   The selected trace data file is opened, uncompressed if necessary, and the trace data is preprocessed. These steps can take some time, depending on compression and the size and contents of the trace data file.

   The Db2 Buffer Pool Analyzer - Simulation window is displayed. Perform the following steps before you start the simulation:

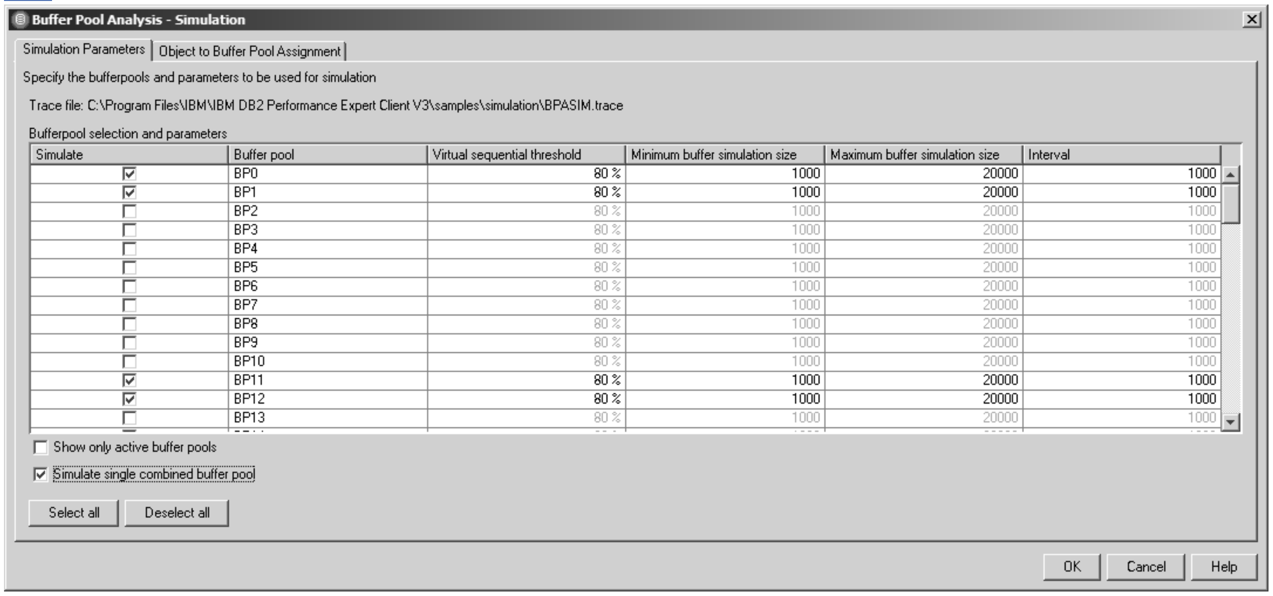
   a. "Step 1: Setting simulation parameters" on page 113
   b. "Step 2: Assigning objects to buffer pools" on page 114

   You can navigate through the pages by clicking the appropriate page tab. On both pages, you can click **Cancel** to return to the main window, **Help** to get help on the current page, **OK** to start the simulation. When the simulation ends, continue with "Viewing the result of a simulation cycle" on page 116.

## Step 1: Setting simulation parameters

### About this task

When you have opened a trace data file, as described in "Selecting and opening a trace data file" on page 112, and have clicked the Simulation Parameters tab, the following page is displayed:



Buffer Pool Analysis - Simulation
Simulation Parameters
Object to Buffer Pool Assignment

Specify the bufferpools and parameters to be used for simulation

Trace file: C:\Program Files\IBM\IBM DB2 Performance Expert Client V3\samples\simulation\BPASIM.trace

Bufferpool selection and parameters

| Simulate | Buffer pool | Virtual sequential threshold | Minimum buffer simulation size | Maximum buffer simulation size | Interval |
|---|---|---|---|---|---|
| ✓ | BP0 | 80 % | 1000 | 20000 | 1000 |
| ✓ | BP1 | 80 % | 1000 | 20000 | 1000 |
| ☐ | BP2 | 80 % | 1000 | 20000 | 1000 |
| ☐ | BP3 | 80 % | 1000 | 20000 | 1000 |
| ☐ | BP4 | 80 % | 1000 | 20000 | 1000 |
| ☐ | BP5 | 80 % | 1000 | 20000 | 1000 |
| ☐ | BP6 | 80 % | 1000 | 20000 | 1000 |
| ☐ | BP7 | 80 % | 1000 | 20000 | 1000 |
| ☐ | BP8 | 80 % | 1000 | 20000 | 1000 |
| ☐ | BP9 | 80 % | 1000 | 20000 | 1000 |
| ☐ | BP10 | 80 % | 1000 | 20000 | 1000 |
| ✓ | BP11 | 80 % | 1000 | 20000 | 1000 |
| ✓ | BP12 | 80 % | 1000 | 20000 | 1000 |
| ☐ | BP13 | 80 % | 1000 | 20000 | 1000 |

☐ Show only active buffer pools

✓ Simulate single combined buffer pool

*Figure 21. Simulation – The Simulation Parameters window*

This page shows a list of available buffer pools and their default parameters, and it provides controls to select buffer pools and change their parameters for the simulation.

- The top of the page shows the full path and name of the trace data file that you just opened.
- Each row of the list represents a single buffer pool and its simulation parameters. The list shows a set of 80 selectable buffer pools in the ranges of BP0 to BP49 (the 4 KB buffer pools), BP8K0 to BP8K9 (the 8 KB buffer pools), BP16K0 to BP16K9 (the 16 KB buffer pools), and BP32K0 to BP32K9 (the 32 KB buffer pools). Initially, active buffer pools (those for which activities were recorded during the time trace data was collected and stored in the trace data file) are preselected in the **Simulate** column.
- The **Virtual sequential threshold** field shows the percentage of the virtual buffer pool that might be occupied by sequentially accessed pages. The default value is 80%.
- The **Minimum buffer simulation size** field shows the minimum buffer pool size to be simulated. The size is shown in number of buffer pool pages (whereby a single page has a size of 4 KB, 8 KB, 16 KB, or 32 KB, depending on the buffer pool).
- The **Maximum buffer simulation size** field shows the maximum buffer pool size to be simulated. The size is shown in number of buffer pool pages.
- The **Interval** field shows the increment by which a buffer pool size is changed (between the minimum and maximum size) during the simulation. The increment is shown in number of buffer pool pages.

You use this page to determine which buffer pools you want to include in the simulation and to specify individual simulation parameters for each selected buffer pool.

- Select one or more buffer pools by selecting or clearing the respective check boxes. You can also use the **Select all** or **Deselect all** button to act on active buffer pools. Select the **Show only active buffer pools** check box to show only objects with buffer pool activity.
- You can adjust the initial simulation parameters as required. Note that the time to perform a simulation increases with the number of buffer pools and the different buffer pool sizes to be simulated. Select the **Simulate single combined buffer pool** check box to run the simulation for the preselected buffer pools as a single combined buffer pool.

Example: With the initial values shown on this page, Buffer Pool Analyzer will simulate buffer pool sizes of 1 000, 2 000, 3 000 pages, and so on, up to 20 000 pages, for a selected buffer pool.

For practical reasons, choose a minimum and maximum buffer pool size and an interval that does not result in more than 40 buffer pool sizes per selected buffer pool. Otherwise, the time to generate the simulation result increases excessively. In addition, the simulation result becomes complex and difficult to interpret.

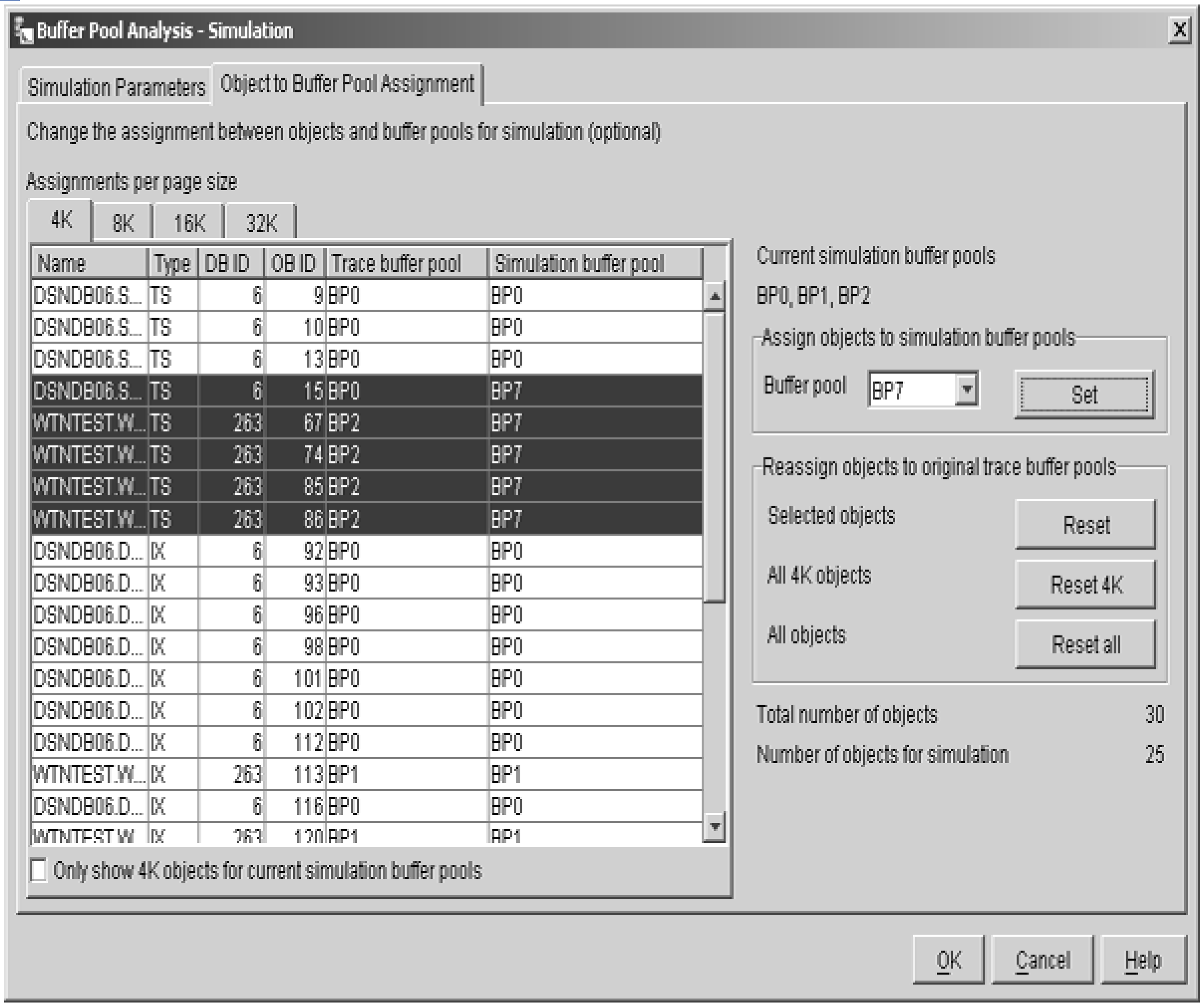
- Note that you can sort the list by clicking a column header of choice one or more times. Small arrows indicate the sort order.

1. Review your selections and the simulation parameters for each selected buffer pool. To modify a parameter, ensure that the buffer pool is selected. Then click the respective field and edit the shown value. (Internally, the minimum, maximum, and interval values are slightly rounded to avoid odd-numbered simulation cycles.)
2. If you have already performed "Step 2: Assigning objects to buffer pools" on page 114, click **OK** to start the simulation, otherwise continue with "Step 2: Assigning objects to buffer pools" on page 114.

## Step 2: Assigning objects to buffer pools

### About this task

When you have opened a trace data file, as described in "Selecting and opening a trace data file" on page 112, and have clicked the Object to Buffer Pool Assignment tab, the following page is displayed:



Buffer Pool Analysis - Simulation

Simulation Parameters | Object to Buffer Pool Assignment

Change the assignment between objects and buffer pools for simulation (optional)

Assignments per page size

4K | 8K | 16K | 32K

| Name | Type | DB ID | OB ID | Trace buffer pool | Simulation buffer pool |
|---|---|---|---|---|---|
| DSNDB06.S... | TS | 6 | 9 | BP0 | BP0 |
| DSNDB06.S... | TS | 6 | 10 | BP0 | BP0 |
| DSNDB06.S... | TS | 6 | 13 | BP0 | BP0 |
| DSNDB06.S... | TS | 6 | 15 | BP0 | BP7 |
| WTNTEST.W... | TS | 263 | 67 | BP2 | BP7 |
| WTNTEST.W... | TS | 263 | 74 | BP2 | BP7 |
| WTNTEST.W... | TS | 263 | 85 | BP2 | BP7 |
| WTNTEST.W... | TS | 263 | 86 | BP2 | BP7 |
| DSNDB06.D... | IX | 6 | 92 | BP0 | BP0 |
| DSNDB06.D... | IX | 6 | 93 | BP0 | BP0 |
| DSNDB06.D... | IX | 6 | 96 | BP0 | BP0 |
| DSNDB06.D... | IX | 6 | 98 | BP0 | BP0 |
| DSNDB06.D... | IX | 6 | 101 | BP0 | BP0 |
| DSNDB06.D... | IX | 6 | 102 | BP0 | BP0 |
| DSNDB06.D... | IX | 6 | 112 | BP0 | BP0 |
| WTNTEST.W... | IX | 263 | 113 | BP1 | BP1 |
| DSNDB06.D... | IX | 6 | 116 | BP0 | BP0 |
| WTNTEST.W... | IX | 263 | 120 | BP1 | BP1 |

☐ Only show 4K objects for current simulation buffer pools

Current simulation buffer pools

BP0, BP1, BP2

Assign objects to simulation buffer pools

Buffer pool: BP7 | Set

Reassign objects to original trace buffer pools

| | |
|---|---|
| Selected objects | Reset |
| All 4K objects | Reset 4K |
| All objects | Reset all |

| | |
|---|---|
| Total number of objects | 30 |
| Number of objects for simulation | 25 |

OK | Cancel | Help

*Figure 22. Simulation – The Object Assignment window*

This page shows objects and their assignments to buffer pools for this simulation, and provides controls to change these assignments.

- The list on the left side shows selectable objects, their original placements (in column **Trace buffer pool**), and their assignments (in column **Simulation buffer pool**) for this simulation.

  The objects are shown in different sub-pages according to their sizes. Click the **4K**, **8K**, **16K**, or **32K** tab to see the respective objects. Note that these selections also change several button labels dynamically.

  The **Name** field shows the name of the object. The **Type** field identifies the object as a table space (TS) or index space (IX). **DB ID** shows the corresponding database ID. **OB ID** shows the object ID.

  The **Trace buffer pool** field shows the name of the buffer pool where the object is originally placed (determined from the content of the trace data file). The **Simulation buffer pool** field shows the name of the buffer pool to which the object is assigned for this simulation. Initially, before you change an assignment, both names are identical.
- On the right side the previously selected buffer pools are shown, and controls are given to change the objects' assignments for this simulation. Further, status information about the assignments is shown.

  The **Current simulation buffer pools** field lists the names of the buffer pools to be simulated (those selected during the previous step). If the list becomes too long, which is indicated by trailing dots (...), move the mouse pointer above the list to display the complete list.

  The **Total number of objects** field shows the total number of active objects contained in the trace data file. Active objects are objects for which buffer pool activities were recorded during the time trace data was collected (unlike inactive objects and objects in the Db2 catalog).

  The **Number of objects for simulation** field shows the number of objects that are assigned to the current simulation buffer pools.

You use this page to change the assignment of objects to buffer pools for this simulation.

- Initially, the assignments are not changed. The buffer pool names in columns **Trace buffer pool** and **Simulation buffer pool** are identical for each object. If the assignments are not changed, the simulation is performed with the assignments that were active at the time trace data was collected.
- Objects can be assigned to buffer pools of matching page sizes (4 KB objects to 4 KB buffer pools, and so on). Use the **4K**, **8K**, **16K**, or **32K** tab to see and assign the respective objects.
- You can assign objects to any matching buffer pool, independently of whether a buffer pool is selected for this simulation. However, the simulation considers only those objects that are assigned to the buffer pools currently selected for simulation. Your assignments are saved and can be used for other simulations that use the same trace data file. The assignments are used if you select the corresponding buffer pools for simulation on the Simulation Parameters page.

  To view only objects in the list that are assigned to the buffer pools to be simulated, select the **Show only 4K objects for current simulation buffer pools** check box. (The objects' page size changes dynamically.) If this check box is cleared, all objects (of that page size) that are contained in the trace data file are shown.
- To select a single object for an assignment, click the name of an object. To select a range of successive objects, click the first object, hold down the Shift key, then click the last object in the sequence. To select several separate objects, click the first object, hold down the Ctrl key, then click the other objects as required.
- To assign one or more selected objects to a buffer pool, select a buffer pool from the **Buffer pool** list and click **Set**.
- To return one or more selected objects to their original placements, click **Reset**. To return all objects of the currently displayed page size, for example, 4 KB objects, click **Reset 4K** (the button label changes dynamically). To return all objects, independently of their page sizes, click **Reset All**.
- Note that you can sort the list by clicking a column header of choice one or more times. Small arrows indicate the sort order.

1. Review your assignments and adjust them as required.

2. If you have already performed "Step 1: Setting simulation parameters" on page 113, click **OK** to start the simulation. Otherwise complete "Step 1: Setting simulation parameters" on page 113 before you start the simulation.

   When the simulation starts, a progress indicator is shown. Note the estimated completion time. If you want to stop the simulation before completion, click **Cancel**. Control is returned to the Db2 Buffer Pool Analyzer - Simulation window.

   When the simulation finishes, the result is saved in the **Results** subfolder of the **Simulation** folder for later viewing. You are asked whether the simulation report should be opened now.

   - **Yes** immediately displays the result in a new browser window. The Db2 Buffer Pool Analyzer - Simulation window is closed.
   - **Cancel** returns control to the Db2 Buffer Pool Analyzer - Simulation window.

3. Continue with "Viewing the result of a simulation cycle" on page 116.

## Viewing the result of a simulation cycle

### About this task

When the simulation finishes, the Buffer Pool Analyzer main window shows the result in the **Results** subfolder of the **Simulation** folder. The subfolder can contain results from several simulation cycles. The result from the most recent simulation is highlighted.

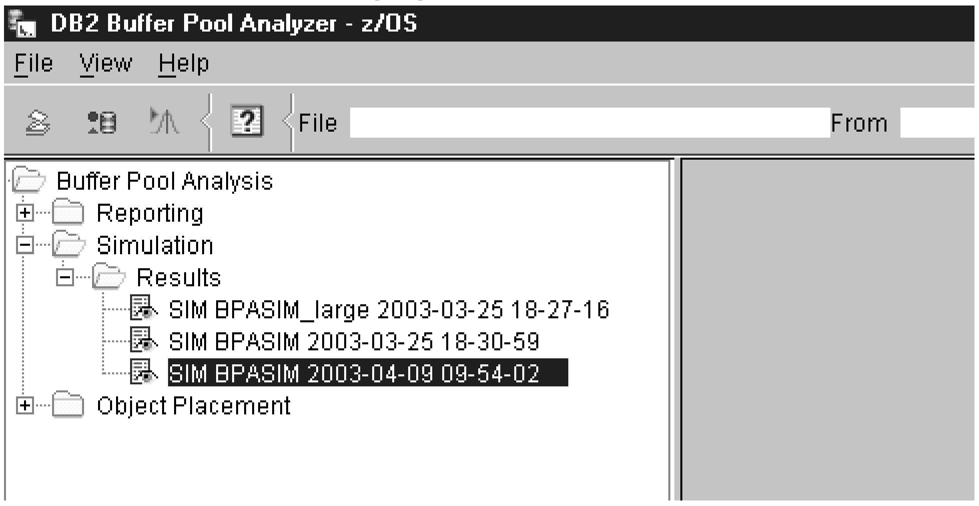

DB2 Buffer Pool Analyzer - z/OS
File View Help
File
From
Buffer Pool Analysis
Reporting
Simulation
Results
SIM BPASIM_large 2003-03-25 18-27-16
SIM BPASIM 2003-03-25 18-30-59
SIM BPASIM 2003-04-09 09-54-02
Object Placement

*Figure 23. Simulation – The Results Selection window*

Results are named `SIM` *`<trace_file>`* *`<date>`* *`<time>`*, whereby `SIM` stands for Simulation, *`<trace_file>`* for the name of the trace data file that was used for the simulation, *`<date>`* and *`<time>`* for the date and time when the simulation started.

1. If you want to delete results from the folder, select a specific result by clicking it. Then press the Delete key. To delete all results, right-click **Results**. Then click **Delete all**. You are asked to confirm the deletion.

   Note that results remain on the hard disk drive and take up space until they are deleted. They are usually located in folder `C:\Documents and Settings \`*`<userid>`*` \db2pev`*`<version>`*` \simulation reports`. However, because of their special format, do not manipulate the folder contents manually.

2. To view the result from a simulation, double-click it, or select it and press Enter.

   The result is shown in your web browser and contains the following information:

   - A comparison of buffer pool efficiency for separate buffer pools versus a single combined buffer pool, as function of the simulated total buffer pool sizes.

Table 8. Example of simulation result, showing a comparison of buffer pool efficiency

| | Separate Buffer Pools | | | Combined Buffer Pool | | |
|---|---|---|---|---|---|---|
| **Total Pages** | **Misses** | **Application Hit Ratio** | **Global Miss Ratio** | **Misses** | **Application Hit Ratio** | **Global Miss Ratio** |
| 200 | 15332 | 14.9 | 83.7 | 92401 | 48.7 | 50.4 |
| 250 | 141398 | 21.5 | 77.2 | 35845 | 80.1 | 19.6 |
| 300 | 133484 | 25.9 | 72.8 | 24977 | 86.1 | 13.6 |
| 350 | 93542 | 48.1 | 51.0 | 23049 | 87.2 | 12.6 |
| 400 | 43537 | 75.8 | 23.8 | 21780 | 87.9 | 11.9 |
| 450 | 31378 | 82.6 | 17.1 | 20891 | 88.4 | 11.4 |
| 500 | 23449 | 87.0 | 12.8 | 19977 | 88.9 | 10.9 |
| | | | | | | |

- A list of recommended buffer pool sizes for the simulated buffer pools, as function of the simulated total buffer pool sizes.

Table 9. Example of simulation result, showing a list of recommended buffer pool sizes

| Total Pages | BP0 pages | BP1 pages | BP2 pages |
|---|---|---|---|
| 150 | 50 | 50 | 50 |
| 200 | 100 | 50 | 50 |
| 250 | 50 | 50 | 150 |
| 300 | 50 | 50 | 200 |
| 350 | 100 | 50 | 200 |
| | | | |

- A detailed breakdown of efficiency by buffer pool operation for each simulated buffer pool, as function of the simulated total buffer pool sizes.

Table 10. Example of simulation result, showing a detailed breakdown of efficiency

| | All | | Random | | Sequential Prefetch | | List Prefetch | | Set write intent | |
|---|---|---|---|---|---|---|---|---|---|---|
| **Buffer Pool Pages** | **Total Misses** | **Application Hit Ratio** | **Misses** | **% of Total Misses** | **Misses** | **% of Total Misses** | **Misses** | **% of Total Misses** | **Misses** | **% of Total Misses** |
| 50 | 24022 | 45.2 | 6269 | 26.1 | 14286 | 59.5 | 44 | 0.2 | 3423 | 14.2 |
| 100 | 15670 | 64.3 | 1833 | 11.7 | 13795 | 88.0 | 42 | 0.3 | 0 | 0.0 |
| 150 | 14563 | 66.8 | 1409 | 9.7 | 13112 | 90.0 | 42 | 0.3 | 0 | 0.0 |
| 200 | 10195 | 76.7 | 1160 | 11.4 | 8993 | 88.2 | 42 | 0.4 | 0 | 0.0 |
| 250 | 4080 | 90.7 | 955 | 23.4 | 3083 | 75.6 | 42 | 1.0 | 0 | 0.0 |
| 300 | 3516 | 92.0 | 719 | 20.4 | 2755 | 78.4 | 42 | 1.2 | 0 | 0.0 |
| 350 | 3018 | 93.1 | 467 | 15.5 | 2509 | 83.1 | 42 | 1.4 | 0 | 0.0 |
| | | | | | | | | | | |

- Summaries of data collection details and simulation parameters, as they were used with this simulation. A summary of buffer pool activity counts, as found in the trace data file, and a summary of those counts resulting from a changed object to buffer pool assignment of this simulation.
- A table of object details. In this report section you can interactively select for which simulated buffer pool and buffer pool size you want to display the details of assigned objects. (This method avoids having to scroll through a long list of details.) For the objects assigned to the selected buffer pool the table shows object attributes, as found in the trace data file, and selective buffer pool activity counts resulting from the simulated buffer pool size.

  In addition, this report section provides facilities to interactively sort the information in the table and to randomly search for specified information. See the actual report section header for instructions.

*Table 11. Example of simulation result, showing a table of object details*

| Name | Type | DB ID | OB ID | Trace buffer pool | Trace entries | Getpages | Random misses | Sequential misses | List misses | Setwrite misses | Residency time |
|---|---|---|---|---|---|---|---|---|---|---|---|
| DSNDB06.SYSGROUP | TS | 6 | 12 | 0 | 2 | 2 | 1 | 0 | 0 | 0 | 00:00:33 |
| DSNDB06.SYSUSER | TS | 6 | 15 | 0 | 3 | 3 | 1 | 0 | 0 | 0 | 00:02:17 |
| DSNDB06.DSNAPH01 | IX | 6 | 101 | 0 | 3 | 3 | 2 | 0 | 0 | 0 | 00:01:31 |
| | | | | | | | | | | | |

**Name**
The name of the object.

**Type**
The type of the object (table space or index space).

**DB ID**
The corresponding database ID.

**OB ID**
The object ID.

**Trace buffer pool**
The original buffer pool to which the object was assigned at the time trace data was collected.

**Trace entries**
The number of buffer pool activity entries in the trace data file for this object.

**Getpages**
The total number of Getpage requests (Random, Sequential, List) for this object.

**Random misses**
The number of misses of Getpage - Random requests for this object resulting from the simulated size.

**Sequential misses**
The number of misses of Getpage - Sequential requests for this object resulting from the simulated size.

**List misses**
The number of misses of Getpage - List requests for this object resulting from the simulated size.

**Setwrite misses**
The number of misses of Setwrite Intent requests for this object resulting from the simulated size.

**Residency time**
The average duration, in hours, minutes, and seconds, this object would be kept in the simulated buffer pool and size.

3. Use the hypertext links to navigate through the report.

   You can open sections of the report by clicking **Open this report in a new browser window**.

   Note especially the **Click here to see more online help** links. They provide detailed information about how to interpret the report.

## Performance-related tips

### About this task

The following tips might be useful when you perform a simulation:

- The result from a simulation shows a comparison of buffer pool efficiency for separate buffer pools versus a single combined buffer pool (see step 2). Generally, a single buffer pool improves the total buffer pool hit ratio, requires less monitoring and tuning, treats applications equally, and offsets increasing workload of one application by decreasing workload in another. However, with a single buffer pool no preference to applications with different levels of importance can be given, and different access and usage pattern cannot be isolated.

Multiple buffer pools allow for performance preferences, grouping according to access pattern, different thresholds, separation of table spaces from index spaces to optimize the hit rate for indexes, and more. In most cases, multiple buffer pools are necessary to optimize among total buffer pool size and its cost, and overall transaction performance. However, too many buffer pools also increase the effort to monitor and administer the buffer pools.

- The total buffer pool size has a great effect on performance. However, if it is too large, and there is not enough memory to allocate them, then a minimum buffer pool for each page size will be allocated, and the performance will be sharply reduced. To calculate the maximum buffer pool size, all other storage utilization must be considered by Db2 as well as the operating system and any other applications. When the total available size is determined, this area can be divided into different buffer pools to improve utilization.
- Smaller buffer pools are more likely affected by fluctuating workload. Smaller buffer pools tend to show more performance peaks than larger buffer pools.
- The size of buffer pools that predominantly process write requests can be minimized. These buffer pools usually show a low hit ratio. Consecutive write operations fill up the buffer pool and require the data to be frequently written to disk. This applies typically to objects that log or journalize transaction data.

# Chapter 9. Analyzing long-term buffer pool performance

This topic describes how to view performance data from several bpd files on the client to perform a long-term analysis of buffer pool performance.

### About this task

Related tasks:

- Before you can use this function, you must have performed the tasks described in:
  - Chapter 3, "Collecting data," on page 27
  - Chapter 4, "Creating activity reports and bpd files," on page 37

    Here, only bpd files are of interest. Long-term analysis requires at least one bpd file as input; however, the intention is to analyze data from multiple bpd files to span a longer period.
  - Chapter 11, "Downloading files from the host to the client," on page 151

    One or more buffer pool data files must be available on the client (the files with the recommended file name extension bpd).

**General remarks:**

1. Your client should have approximately 40 MB of *available physical memory* (random access memory) for this function. You can check this in the Windows Task Manager. Close other applications, if necessary.
2. The data used for this function reflects the performance for the intervals for which trace data was collected and for the time frames that were specified with the **`GLOBAL`** and the **`BPACTIVITY`** command when the bpd files were created.
3. If you want to print any of the graphical information from the following windows, place the cursor on the displayed graphical information and click **View —► Open dataview in browser**. When the browser window is displayed, use your browser's printing capabilities to print the information. For colored printouts, check the browser settings.

   Example: In the Internet Explorer, click **Tools —► Internet Options —► Advanced**. Under **Printing**, select **Print background colors and images**.

## Starting the long-term analysis function

### About this task

### Procedure

1. Start the IBM Db2 Buffer Pool Analyzer by double-clicking the icon on your Windows desktop, or use the **Start** push button on the taskbar to start this client application.

   If you are using Db2 Performance Expert, click **Tools** > **Buffer Pool Analysis for z/OS** on the menu bar, or click the **Opens the Buffer Pool Analysis for z/OS window** toolbar button.

   The Db2 Buffer Pool Analyzer - z/OS main window is displayed.
2. On the menu bar, click **File** > **Long-Term Analysis**. Alternatively, click the **Starts buffer pool long-term analysis** toolbar button.

   The Buffer Pool Analysis - Long-Term Analysis window is displayed. You see the first page of the long-term analysis wizard, which guides you through the following steps:

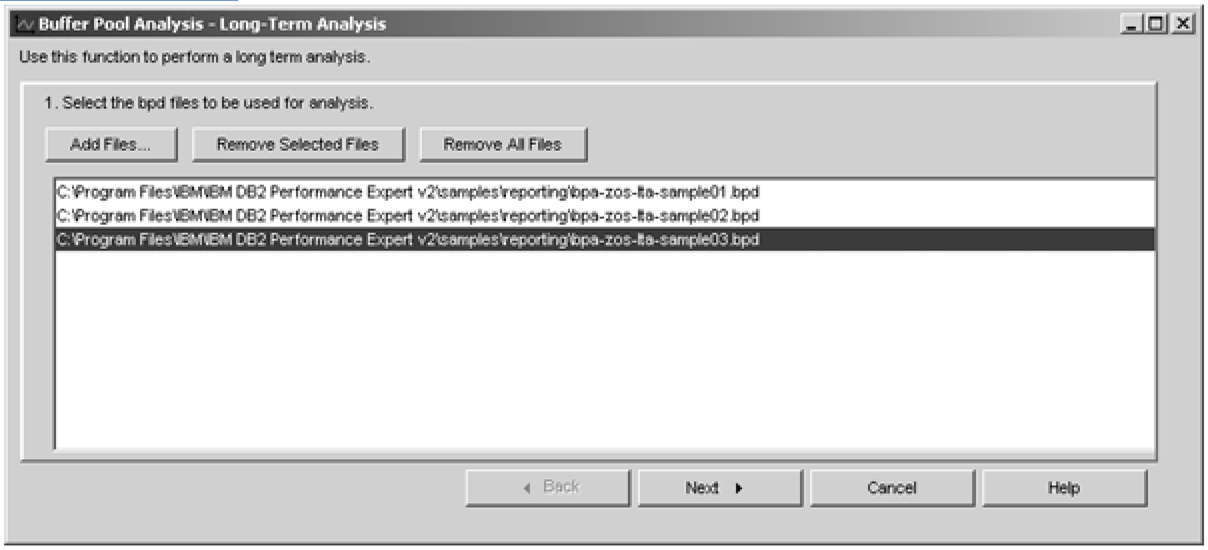
a. “Step 1: Selecting and opening buffer pool data files” on page 122
b. “Step 2: Choosing a subsystem and specifying an analysis type” on page 123
c. “Step 3: Specifying counters, objects, time frame, and output” on page 125

You can navigate through the pages by clicking **Next** or **Back**. On all three pages, you can click **Cancel** to return to the main window, or **Help** to get help on the current page. When you have finished these steps, you can continue with “Viewing the result of a long-term analysis” on page 128.

## Step 1: Selecting and opening buffer pool data files

### About this task

When you have started the long-term analysis function, as described in “Starting the long-term analysis function” on page 121, the first page of the long-term analysis wizard is displayed:



Use this function to perform a long term analysis.

1. Select the bpd files to be used for analysis.

C:\Program Files\IBM\IBM DB2 Performance Expert v2\samples\reporting\bpa-zos-lta-sample01.bpd
C:\Program Files\IBM\IBM DB2 Performance Expert v2\samples\reporting\bpa-zos-lta-sample02.bpd
C:\Program Files\IBM\IBM DB2 Performance Expert v2\samples\reporting\bpa-zos-lta-sample03.bpd

*Figure 24. Long-Term Analysis – The Files Selection window*

You use this page to select one or more bpd files that contain the performance data you want to analyze. Initially, this dialog box is empty. The example lists three bpd files that were already selected.

### Procedure

1. Select one or more bpd files to be opened, using the following steps in reasonable order:
   - To add one or more bpd files to the (initially empty) list of selected bpd files, click **Add Files**.

     The Open dialog box is displayed. You use this dialog box to select one or more bpd files that you want to add to the list of selected bpd files.

     Several sample buffer pool data files (`*.bpd`) are delivered with Buffer Pool Analyzer in the … `\samples\reporting` folder. The sample files are named `bpa-zos-lta-sample`*nn*`.bpd`. You can use them to become familiar with this function.

     a. Select a folder from the **Look in** list where the bpd files you want to use are located.

        The **File of type** field shows the file name extension of buffer pool data files (`*.bpd`). If your bpd files do not have the recommended file name extension bpd, select `All files (*.*)` from the **File of type** list to see the appropriate files.

     b. Click the bpd file to be added to the list of selected bpd files.

        You can add multiple files by selecting multiple files in the Open dialog box. To select a range of files, click the first file, hold down the Shift key, then click the last file in the sequence. To select several separate files, click the first file, hold down the Ctrl key, then click the other files as required. To select all files, press Ctrl+A.

The **File name** field shows the names of the selected bpd files.

c. Click **Open**.

The Open dialog box closes and the bpd files are added to the list of selected bpd files.

Restriction: bpd files are usually named at your own discretion, except for the recommended file name extension bpd. Thus, it is possible that bpd files exist with different names, but identical or partially identical performance data. If you accidentally select such files for the long-term analysis, the duplicated performance records falsify the result. Buffer Pool Analyzer does not filter out duplicates from the selected bpd files.

- To remove one or more bpd files from the list of selected bpd files, select one or more files in the list, then click **Remove Selected Files**.
- To remove all bpd files from the list of selected bpd files, click **Remove All Files**.

2. When you have completed your selection of bpd files, and the list of selected bpd files contains at least one file, click **Next**.

   The selected bpd files are opened, and the data is preprocessed. This step can take some time, depending on the sizes and contents of the bpd files. A progress indicator is displayed that shows the percentage of preprocessing that has been completed. You can click **Cancel** to cancel preprocessing and restore the file selection page.

   Preprocessing performs several activities on each bpd file, which eases your selections and specifications in the following steps:

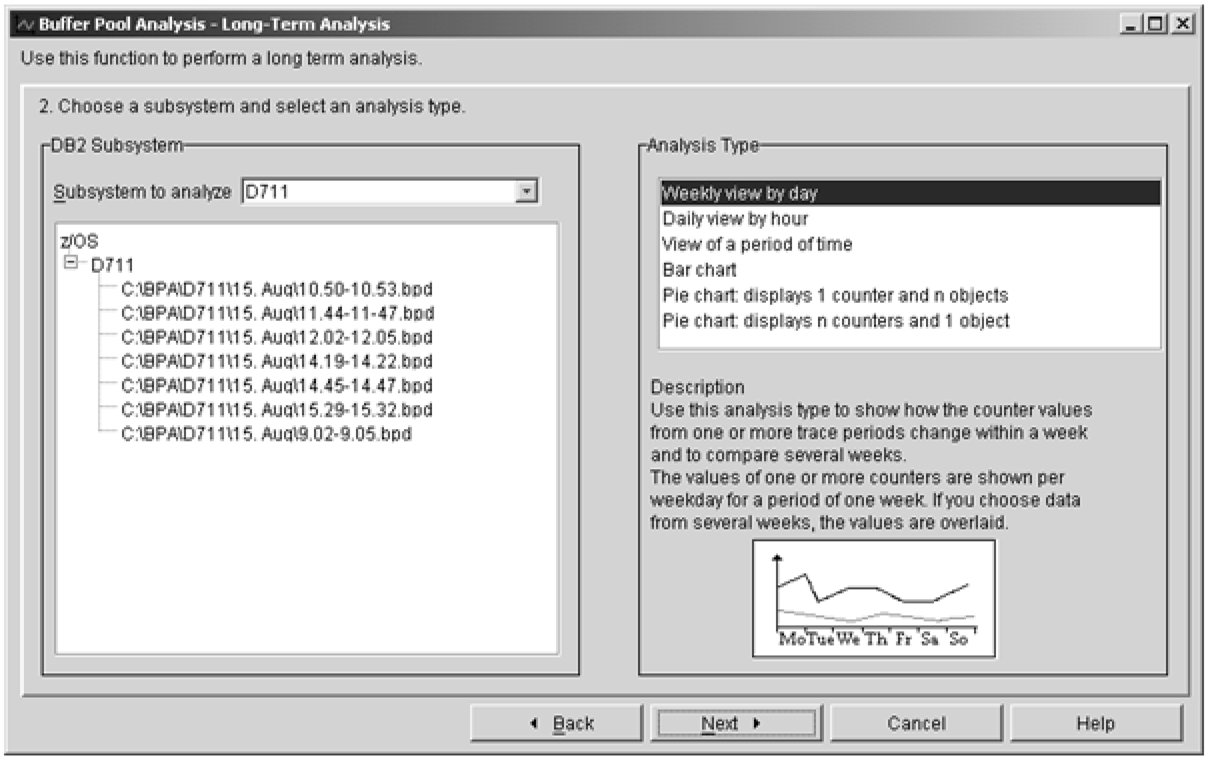
   - The subsystems are determined from which performance data was collected.
   - The counters and active objects are determined.
   - The earliest and the latest timestamp in each bpd file is determined.

3. Continue with "Step 2: Choosing a subsystem and specifying an analysis type" on page 123.

## Step 2: Choosing a subsystem and specifying an analysis type

### About this task

When you have performed "Step 1: Selecting and opening buffer pool data files" on page 122, the second page of the long-term analysis wizard is displayed:


Buffer Pool Analysis - Long-Term Analysis

Use this function to perform a long term analysis.

2. Choose a subsystem and select an analysis type.

DB2 Subsystem

Subsystem to analyze D711

```
z/OS
⊟ D711
    C:\BPA\D711\15. Aug\10.50-10.53.bpd
    C:\BPA\D711\15. Aug\11.44-11-47.bpd
    C:\BPA\D711\15. Aug\12.02-12.05.bpd
    C:\BPA\D711\15. Aug\14.19-14.22.bpd
    C:\BPA\D711\15. Aug\14.45-14.47.bpd
    C:\BPA\D711\15. Aug\15.29-15.32.bpd
    C:\BPA\D711\15. Aug\9.02-9.05.bpd
```

Analysis Type

- Weekly view by day
- Daily view by hour
- View of a period of time
- Bar chart
- Pie chart: displays 1 counter and n objects
- Pie chart: displays n counters and 1 object

Description

Use this analysis type to show how the counter values from one or more trace periods change within a week and to compare several weeks.

The values of one or more counters are shown per weekday for a period of one week. If you choose data from several weeks, the values are overlaid.

*Figure 25. Long-Term Analysis – The Subsystem and Analysis Type Selection window*

Use this page to verify your selection of bpd files and to specify the type of analysis you want to perform.

- The bpd files you selected in the previous step might contain performance data from different subsystems. However, a meaningful long-term analysis can only be performed with data from one subsystem. The **DB2 Subsystem** group box on this page shows from which subsystems the various selected bpd files were created. (This information was acquired when the selected bpd files were preprocessed.)

  The subsystem folder with the most bpd files is opened, and the corresponding subsystem is preselected from the **Subsystem to analyze** list. You can expand or collapse the tree items by clicking the plus sign (+), respectively the minus sign (-), or by double-clicking the corresponding tree item.

  Use this information to decide from which subsystem you want you use the data for the long-term analysis.

- The **Analysis type** group box on this page shows a list of selectable presentation types, with one type being preselected. The analysis type determines how the information will be presented in the graphical report.

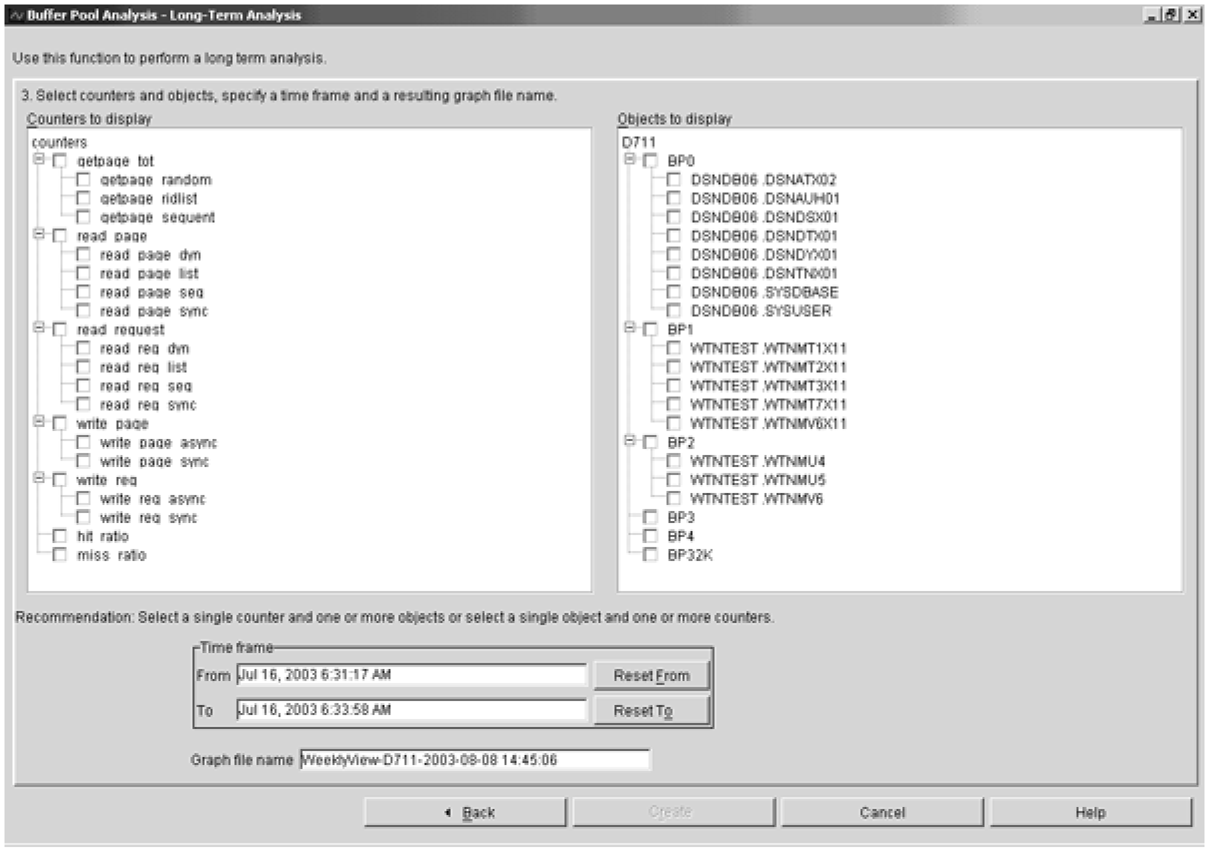
  Use this list to decide about the analysis type. For the selected type, a brief description and an illustration of the chart type is shown on this page. You can select the types in any order to examine them. "Viewing the result of a long-term analysis" on page 128 provides a more detailed description and explanation of the output from these analysis types.

1. Select the bpd files you want to use by selecting the appropriate subsystem from the **Subsystem to analyze** list.
2. Under **Analysis type**, select an appropriate analysis type.
3. Click **Next** and continue with "Step 3: Specifying counters, objects, time frame, and output" on page 125, or click **Back** to return to the previous page.

## Step 3: Specifying counters, objects, time frame, and output

### About this task

When you have performed "Step 2: Choosing a subsystem and specifying an analysis type" on page 123, the third page of the long-term analysis wizard is displayed. This page differs somewhat depending on the analysis type you specified in the previous step. At first, this section shows and describes the more flexible specifications applicable to the first four analysis types ("Weekly view by day", "Daily view by hour", "View of a period of time", and "Bar chart"), and the similarities of this page that apply to all analysis types. "Characteristics of the pie chart analysis types" on page 127 details the specifications for remaining pie chart presentations (which is basically a more restrictive use of *1-to-n* and *n-to-1* relations between counters and objects).


Buffer Pool Analysis - Long-Term Analysis

Use this function to perform a long term analysis.

3. Select counters and objects, specify a time frame and a resulting graph file name.

Counters to display

```
counters
⊟ ☐ getpage tot
    ☐ getpage random
    ☐ getpage ridlist
    ☐ getpage sequent
⊟ ☐ read page
    ☐ read page dyn
    ☐ read page list
    ☐ read page seq
    ☐ read page sync
⊟ ☐ read request
    ☐ read req dyn
    ☐ read req list
    ☐ read req seq
    ☐ read req sync
⊟ ☐ write page
    ☐ write page async
    ☐ write page sync
⊟ ☐ write req
    ☐ write req async
    ☐ write req sync
  ☐ hit ratio
  ☐ miss ratio
```

Objects to display

```
D711
⊟ ☐ BP0
    ☐ DSNDB06 .DSNATX02
    ☐ DSNDB06 .DSNAUH01
    ☐ DSNDB06 .DSNDSX01
    ☐ DSNDB06 .DSNDTX01
    ☐ DSNDB06 .DSNDYX01
    ☐ DSNDB06 .DSNTNX01
    ☐ DSNDB06 .SYSDBASE
    ☐ DSNDB06 .SYSUSER
⊟ ☐ BP1
    ☐ WTNTEST .WTNMT1X11
    ☐ WTNTEST .WTNMT2X11
    ☐ WTNTEST .WTNMT3X11
    ☐ WTNTEST .WTNMT7X11
    ☐ WTNTEST .WTNMV6X11
⊟ ☐ BP2
    ☐ WTNTEST .WTNMU4
    ☐ WTNTEST .WTNMU5
    ☐ WTNTEST .WTNMV6
  ☐ BP3
  ☐ BP4
  ☐ BP32K
```

Recommendation: Select a single counter and one or more objects or select a single object and one or more counters.

Time frame

From: Jul 16, 2003 6:31:17 AM

To: Jul 16, 2003 6:33:58 AM

Graph file name: WeeklyView-D711-2003-08-08 14:45:06

*Figure 26. Long-Term Analysis – The Counter and Object Selection window*

Use this page to specify which counters and objects to consider for the analysis, the time frame to include, and the name to be used for saving the analysis result.

- For a long-term analysis usually only a subset of counters and objects is of interest. This page is used to specify which of them are to be included in the analysis.
  - **Counters to display** shows a tree view of selectable counters. You can expand or collapse the tree items by clicking the plus sign (+), respectively the minus sign (-), or by double-clicking the corresponding tree item.

    You use this list to specify which counters to consider for the long-term analysis, provided they pertain to selected active objects (see later in this topic).
  - **Objects to display** shows a tree view of selectable objects. You can expand or collapse the tree items by clicking the plus sign (+), respectively the minus sign (-), or by double-clicking the corresponding tree item. The list of objects contains those objects that were identified as active objects in the selected bpd files during preprocessing, and which belong to the selected subsystem.

You use this list to specify which objects to consider for the long-term analysis, provided they pertain to selected counters (see previous information).

In both trees you can select counters, respectively objects, on different levels and in any combination. The tree hierarchies denote the counter and object hierarchies, not selectable groups of counters or groups of objects. For example, the Getpage total count contains the sum of the Getpage random, Getpage ridlist, and Getpage sequential counts; nevertheless, you can select Getpage total and Getpage random only, which will only show these two counters in the result.

As indicated, the selections of counters and objects are interrelated. Technically, both selections act as filters, which means that the analysis result includes only information about selected counters pertaining to selected objects. Practically, deliberate selections of counters and objects on this page permit two different but powerful views:

- If you are interested how a single counter behaves in several objects (for example, to compare the Getpage sequential counts of several objects), you select one counter and several objects. This is basically a *1-to-n* relation between counters and objects. However, you can also set up a *few-to-many* relation, for example, to compare a few counters in several objects.
- If you are interested how several counters behave in one object (for example, to analyze the key counters of an object), you select the counters and a single object. This is basically a *n-to-1* relation between counters and objects. However, you can also set up a *many-to-few* relation.

As pointed out, you are not restricted in your choice of *n-to-m* relations between counters and objects. The long-term analysis function processes any of your selections, independently of whether they make sense or how complex the graphical representation of the result becomes. Start with simple *1-to-1*, *1-to-n* or *n-to-1* counter-to-object relationships until you become familiar with the result. Refine these relations step by step.

Note that this degree of freedom in selecting counters and objects for the long-term analysis applies to the analysis types "Weekly view by day", "Daily view by hour", "View of a period of time", and "Bar chart". The selection of counters and objects for pie chart results is more restrictive (*1-to-n* and *n-to-1*) to avoid complex results. The details are described in "Characteristics of the pie chart analysis types" on page 127.

- The **Time frame** group box shows, by default, the earliest and the latest timestamp of data found in the selected bpd files. The controls can be used to limit the time frame by changing one or both timestamps. Only data from the selected bpd files with timestamps between the default, or specified, **From** and **To** dates is used for the long-term analysis. **Reset From** and **Reset To** reset any specified values to the default values. These push buttons are helpful if you change your mind; you do not need to remember the earliest and latest timestamps.
- The **Graphic file name** field shows the proposed name under which the result from the long-term analysis will be saved for later viewing. The proposed name can be changed, if required. The syntax of proposed names is explained in "Viewing the result of a long-term analysis" on page 128.

1. Select the **Counters to display** by selecting or clearing the check boxes as required. At least one counter must be selected.
2. Select the **Objects to display** by selecting or clearing the check boxes as required. At least one object must be selected.
3. Review the information under **Time frame**. If required, change one or both timestamps. To enter a different timestamp, overwrite the timestamp. Adhere to the format, otherwise, this function might not be able to process the appropriate records from the bpd files, or might not be able to continue.
4. Review the proposed name in the **Graphic file name** field. Change it as required.
5. Click **Create**, or click **Back** to return to the previous page.

   The wizard creates the graphical result and saves it under the specified name. If the name already exists, you are given the choice to replace the existing graphic file or to save the new one under a different name.

When the long-term analysis wizard finishes, the result is immediately shown in a new browser window and in the right pane of the Buffer Pool Analysis main window. In addition, the result is saved in the **Results** subfolder of the **Long Term Analysis** folder for later viewing.

6. Continue with "Viewing the result of a long-term analysis" on page 128.

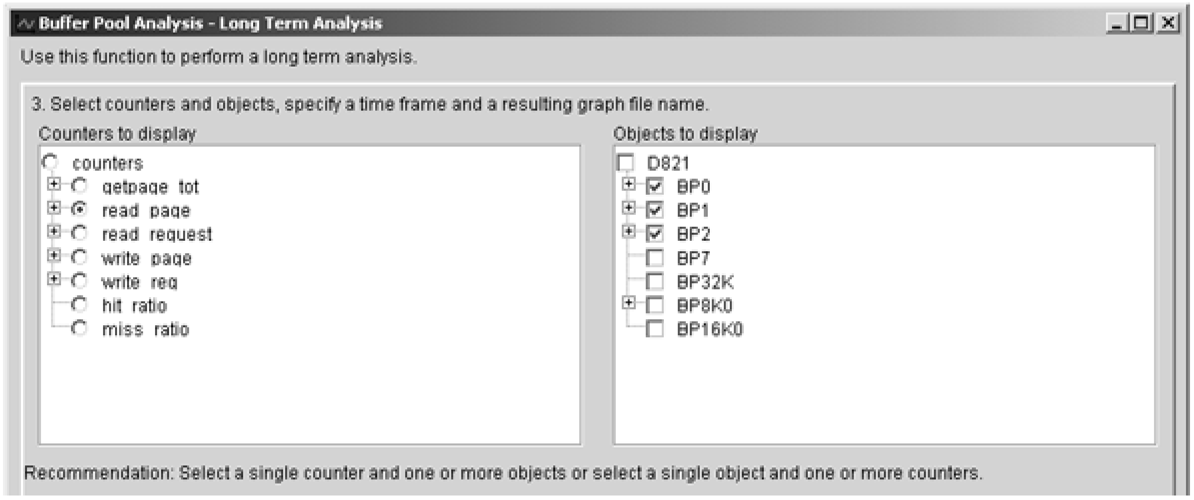
## Characteristics of the pie chart analysis types

### About this task

In "Step 2: Choosing a subsystem and specifying an analysis type" on page 123, you were asked to select one of six analysis types. For the first four types, which are "Weekly view by day", "Daily view by hour", "View of a period of time", and "Bar chart", the previous description outlined your choice of specifying *n-to-m* relations between counters and objects, restricted only by its usefulness.

For the remaining analysis types "Pie chart: display 1 counter and n objects" and "Pie chart: display n counters and 1 object" this degree of freedom is not useful. Therefore, when you choose one of these analysis types, the third page of the long-term analysis wizard (the upper part) looks as follows:

- For the analysis type "Pie chart: display 1 counter and n objects":


Buffer Pool Analysis - Long Term Analysis

Use this function to perform a long term analysis.

3. Select counters and objects, specify a time frame and a resulting graph file name.

Counters to display

- counters
  - getpage tot
  - [✓ read page]
  - read request
  - write page
  - write req
  - hit ratio
  - miss ratio

Objects to display

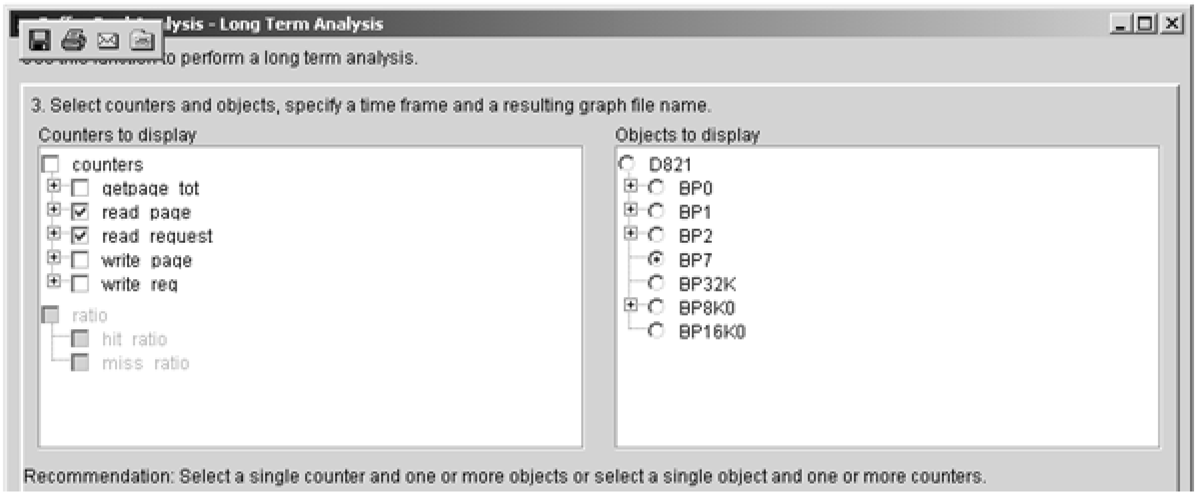
- D821
  - [✓ BP0]
  - [✓ BP1]
  - [✓ BP2]
  - BP7
  - BP32K
  - BP8K0
  - BP16K0

Recommendation: Select a single counter and one or more objects or select a single object and one or more counters.

*Figure 27. Characteristics of long-term analysis type: "Pie chart: display 1 counter and n objects"*

In the left pane, **Counters to display** shows a tree view of selectable counters. Radio buttons indicate that you can select only one item. In the right pane, **Objects to display** shows a tree view of selectable objects. The check boxes indicate that you can select multiple objects. This restricts your selections to a *1-to-n* relation (which includes *1-to-1*) between counters and objects for this analysis type. Note that you can expand the tree items on both sides to select a counter, respectively one or more objects.

- For the analysis type "Pie chart: display n counters and 1 object":



lysis - Long Term Analysis

to perform a long term analysis.

3. Select counters and objects, specify a time frame and a resulting graph file name.

Counters to display

- counters
  - getpage tot
  - [✓] read page
  - [✓] read request
  - write page
  - write req
- ratio
  - hit ratio
  - miss ratio

Objects to display

- D821
  - BP0
  - BP1
  - BP2
  - [●] BP7
  - BP32K
  - BP8K0
  - BP16K0

Recommendation: Select a single counter and one or more objects or select a single object and one or more counters.

*Figure 28. Characteristics of long-term analysis type: "Pie chart: display n counters and 1 object"*

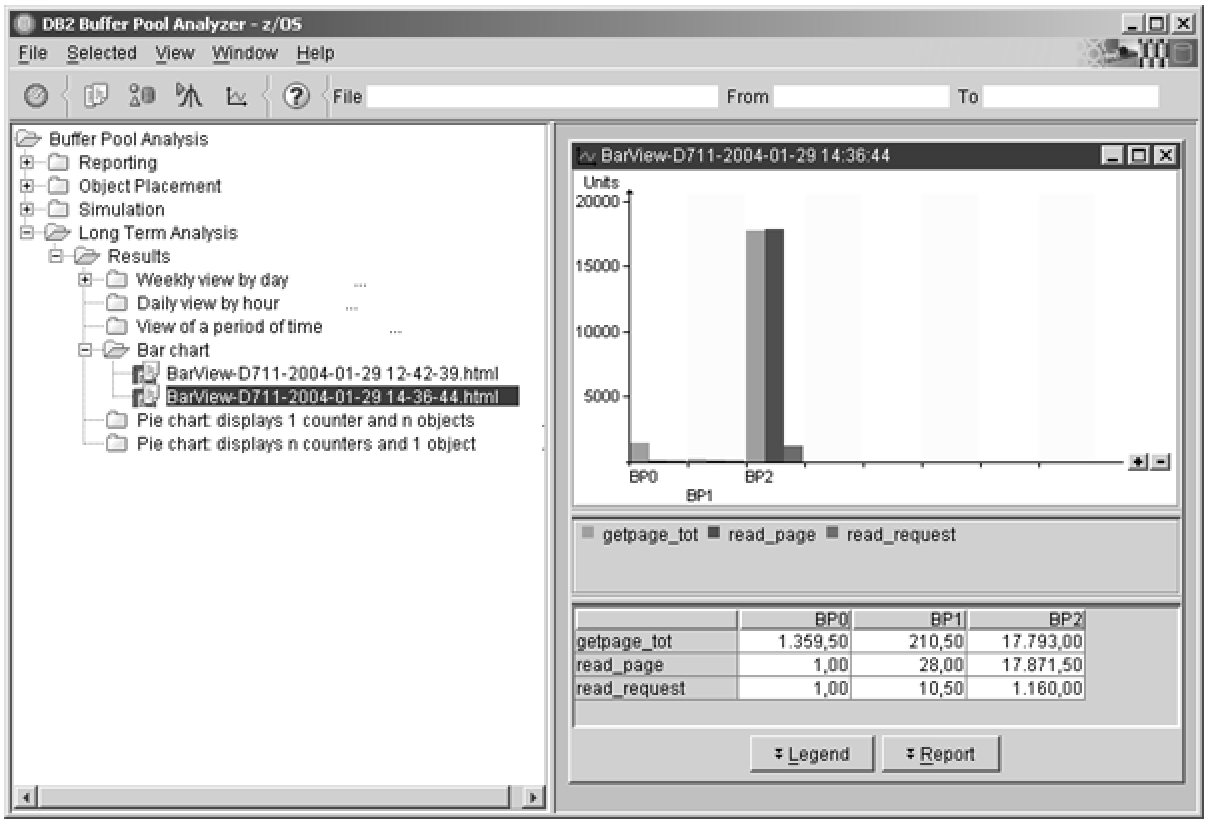
In the left pane, **Counters to display** shows a tree view of selectable counters. The check boxes indicate that you can select one or more counters. In the right pane, **Objects to display** shows a tree view of selectable objects. Radio buttons indicate that you can select only one object. This restricts your selections to an *n-to-1* relation (which includes *1-to-1*) between counters and objects. Note that you can expand the tree items on both sides to select one or more counters, respectively one object.

When you compare the list of selectable counters Figure 27 on page 127 and Figure 28 on page 128, you will notice on the left side that ratios are shown differently in context with counters. In Figure 27 on page 127 ratios are treated identical with counters because you can select only one counter or ratio (*1-to-n* relation to objects). Opposed to this, in Figure 28 on page 128 ratios are shown apart from counters to express their differences (ratios versus absolute values) and to emphasize that you can select multiple counters *or* multiple ratios (*n-to-1* relation to objects). Counters and ratios are mutually exclusive for this analysis type.

## Viewing the result of a long-term analysis

### About this task

When the long-term analysis wizard finishes, the Buffer Pool Analyzer main window shows the result in one of the **Results** subfolders of the **Long-Term Analysis** folder. The subfolders can contain results from several long-term analyses. The result from the most recent analysis is highlighted.


DB2 Buffer Pool Analyzer - z/OS
File Selected View Window Help
File From To
Buffer Pool Analysis
Reporting
Object Placement
Simulation
Long Term Analysis
Results
Weekly view by day
Daily view by hour
View of a period of time
Bar chart
BarView-D711-2004-01-29 12-42-39.html
BarView-D711-2004-01-29 14-36-44.html
Pie chart: displays 1 counter and n objects
Pie chart: displays n counters and 1 object
BarView-D711-2004-01-29 14:36:44
Units
20000
15000
10000
5000
BP0
BP1
BP2
getpage_tot
read_page
read_request
Legend
Report

| | BP0 | BP1 | BP2 |
|---|---|---|---|
| getpage_tot | 1.359,50 | 210,50 | 17.793,00 |
| read_page | 1,00 | 28,00 | 17.871,50 |
| read_request | 1,00 | 10,50 | 1.160,00 |

*Figure 29. Long-Term Analysis – The Results Selection window*

Results are named *<analysis_type>-<subsystem>-<date> <time>*, whereby *<analysis_type>* and *<subsystem>* correspond to your specifications in "Step 2: Choosing a subsystem and specifying an analysis type" on page 123, and *<date>* and *<time>* stand for the date and time when the long-term analysis result was generated and saved.

1. If you want to delete results from the folder, select a specific result by clicking it. Then press the Delete key. To delete all results, right-click **Results**. Then click **Delete all**. You are asked to confirm the deletion.

   Note that results remain on the hard disk drive and take up space until they are deleted. They are usually located in folder `C:\Documents and Settings \`*<userid>*` \db2pev`*<version>*` \bpa-zos-reports \longterm-analysis`. However, because of their special format, do not manipulate the folder contents manually.

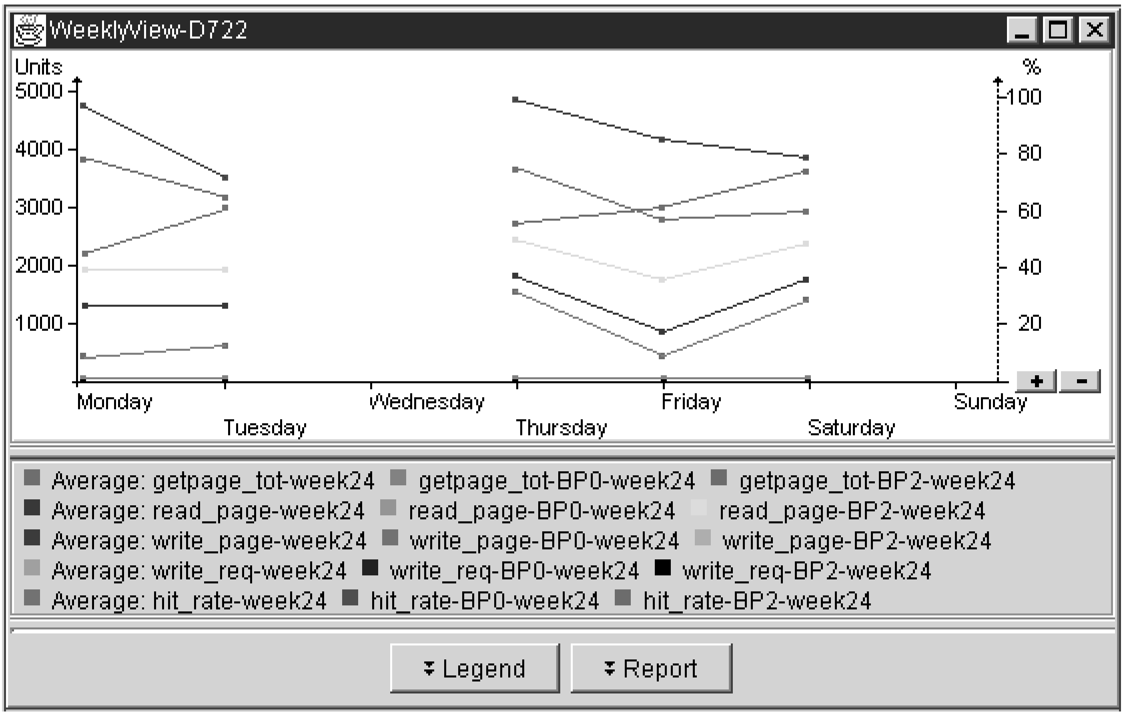
2. To view the result from a long-term analysis, double-click it, or select it and press Enter.

   The result is shown in the right pane of the Buffer Pool Analysis main window. The result consists of a chart and a corresponding legend and report. The legend contains symbols and text that explain the chart. The report lists the information in table form and shows the values represented in the chart. The legend and the report can be switched on or off by using the **Legend** and **Report** push buttons.

   All results can also be shown in your web browser. Right-click into a graphic and choose **Open in browser**.

   The long-term analysis function generates results that differ depending on the analysis type that is specified in "Step 2: Choosing a subsystem and specifying an analysis type" on page 123. The following list shows and describes examples of charts from each analysis type to help you understand how your specifications (mainly the counters and objects) and the performance data from the bpd files are reflected in the result.

**A "Weekly view by day" analysis result:**



■ Average: getpage_tot-week24 ■ getpage_tot-BP0-week24 ■ getpage_tot-BP2-week24
■ Average: read_page-week24 ■ read_page-BP0-week24 ■ read_page-BP2-week24
■ Average: write_page-week24 ■ write_page-BP0-week24 ■ write_page-BP2-week24
■ Average: write_req-week24 ■ write_req-BP0-week24 ■ write_req-BP2-week24
■ Average: hit_rate-week24 ■ hit_rate-BP0-week24 ■ hit_rate-BP2-week24

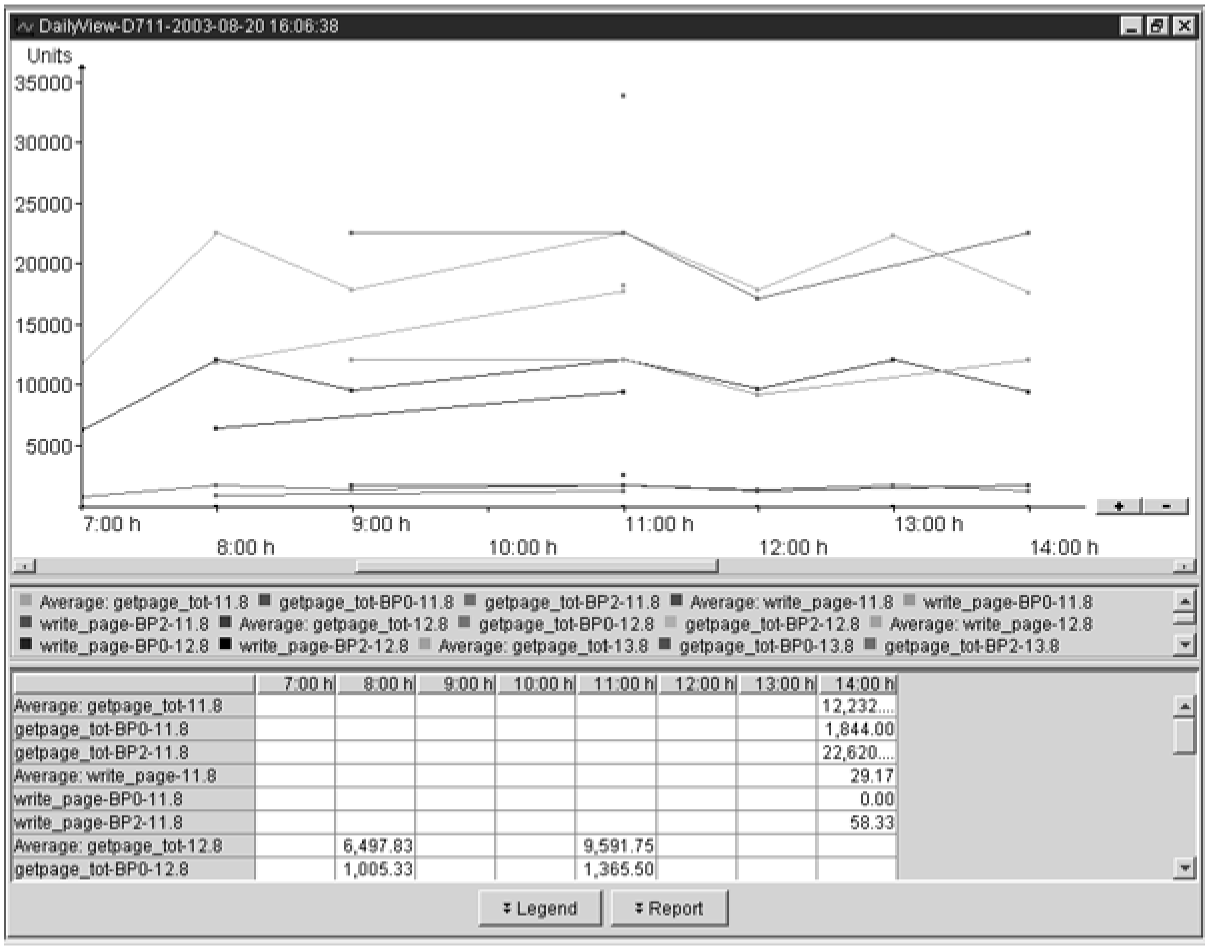
*Figure 30. Long-Term Analysis – Example of "Weekly view by day" result*

This analysis type shows counter values per weekday of selected counters and objects. Counter values represent per-minute values, for example 5 000 Getpage total operations per minute on average over a day. One counter value per counter, object, and weekday is shown, for example, 3 000 Read page operations (the counter) on BP0 (the object) on Monday (the weekday), 2 500 for the same combination on Tuesday, and so on. The counter values for the seven weekdays are connected by lines for better readability. (The lines themselves do not represent interim values.) The "Average" counts show the calculated averages over all affected objects for each counter per weekday, for example, the average of the Write page operations (the counter) of buffer pool BP0 and BP1 (the objects). The interpretation of these average counts is only reasonable if the affected objects are of the same type, for example only buffer pools or only page sets. If you selected objects of different types, the average values are calculated over all objects and do not yield helpful results. Note that counters and ratios are treated equally in this graphic, except that they have their dedicated y-axis.

If the data from the bpd file spans several weeks, the values are overlaid, which means that the described graphic for one week is overlaid with a similar graphic for the second week (having different values), and so on. This example clarifies what is already described in "Step 3: Specifying counters, objects, time frame, and output" on page 125: You can easily overload the graphic by selecting too many counters and objects for longer periods.

You can use this analysis type to analyze how certain counters develop over a week (if the time frame covers a week), or to compare how counters develop over several weeks. This type helps to identify counters that show conspicuously high or low values at specific weekdays or show a trend toward lower or higher values over weeks.

**A "Daily view by hour" analysis result:**



■ Average: getpage_tot-11.8 ■ getpage_tot-BP0-11.8 ■ getpage_tot-BP2-11.8 ■ Average: write_page-11.8 ■ write_page-BP0-11.8
■ write_page-BP2-11.8 ■ Average: getpage_tot-12.8 ■ getpage_tot-BP0-12.8 ■ getpage_tot-BP2-12.8 ■ Average: write_page-12.8
■ write_page-BP0-12.8 ■ write_page-BP2-12.8 ■ Average: getpage_tot-13.8 ■ getpage_tot-BP0-13.8 ■ getpage_tot-BP2-13.8

| | 7:00 h | 8:00 h | 9:00 h | 10:00 h | 11:00 h | 12:00 h | 13:00 h | 14:00 h |
|---|---|---|---|---|---|---|---|---|
| Average: getpage_tot-11.8 | | | | | | | | 12,232.... |
| getpage_tot-BP0-11.8 | | | | | | | | 1,844.00 |
| getpage_tot-BP2-11.8 | | | | | | | | 22,620.... |
| Average: write_page-11.8 | | | | | | | | 29.17 |
| write_page-BP0-11.8 | | | | | | | | 0.00 |
| write_page-BP2-11.8 | | | | | | | | 58.33 |
| Average: getpage_tot-12.8 | | 6,497.83 | | | 9,591.75 | | | |
| getpage_tot-BP0-12.8 | | 1,005.33 | | | 1,365.50 | | | |

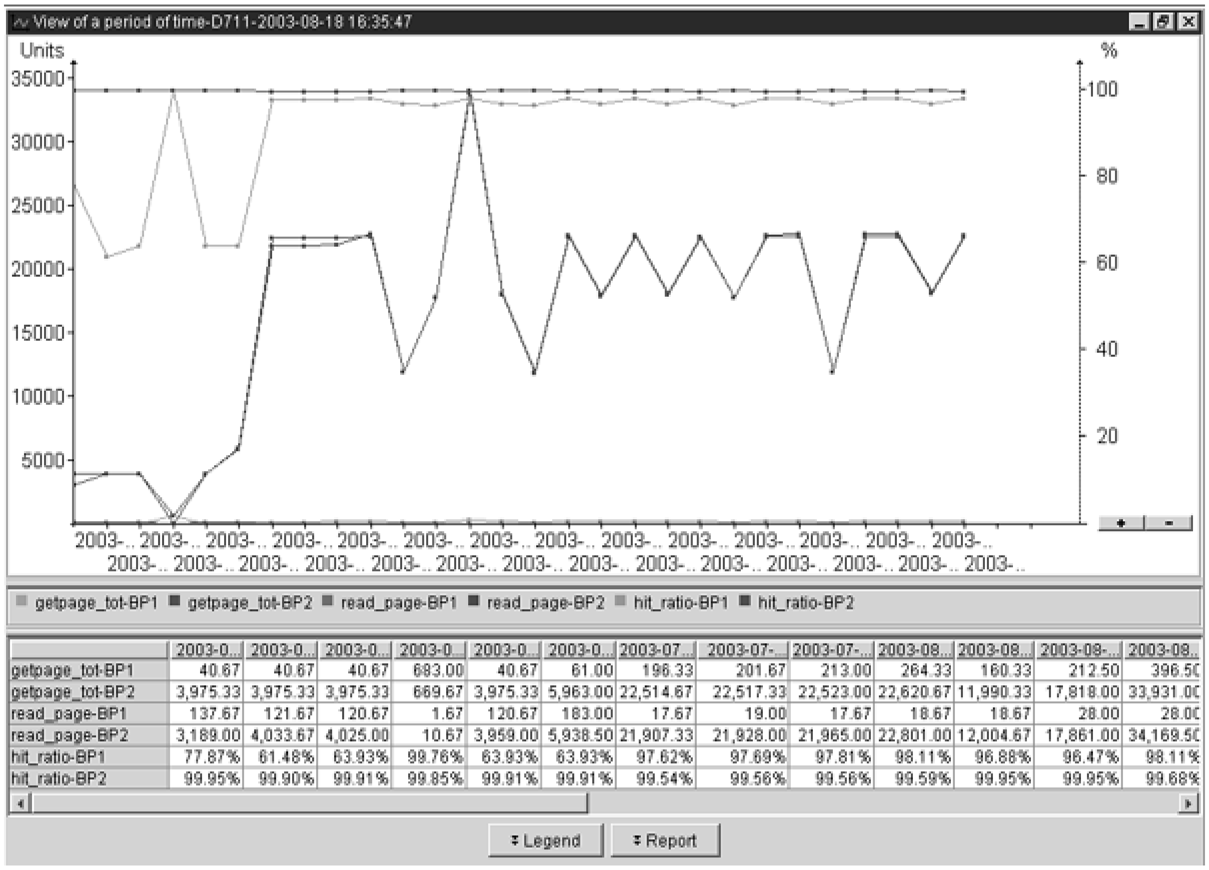
*Figure 31. Long-Term Analysis – Example of "Daily view by hour" analysis result*

This analysis type shows counter values per hour of selected counters and objects. Counter values represent per-minute values, for example 5 000 Getpage operations per minute on average over an hour. One counter value per counter, object, and hour of the day is shown, for example, 7 000 Read page operations (the counter) on BP0 (the object) between 4:00 p.m. and 5:00 p.m. (the hour), 3 000 for the same combination during the next hour, and so on. The counter values for the 24 hours of a day are connected by lines for better readability. (The lines themselves do not represent interim values.) The "Average" counts show the calculated averages over all affected objects for each counter per hour, for example, the average of the Write page operations (the counter) of buffer pool BP0 and BP1 (the objects). The interpretation of these average counts is only reasonable if the affected objects are of the same type, for example only buffer pools or only page sets. If you selected objects of different types, the average values are calculated over all objects and do not yield helpful results. Note that counters and ratios are treated equally in this graphic, except that they have their dedicated y-axis.

If the data from the bpd file spans several days, the values are overlaid, which means that the described graphic for one day is overlaid with a similar graphic for the second day (having different values), and so on. The same precautions should be taken as with the "Weekly view by day" analysis to avoid overloaded results.

You can use this analysis type to analyze how certain counters develop over a day (if the time frame covers a day), or to compare how counters develop over several days. It is basically a more detailed analysis than the "Weekly view by day" analysis.

**A "View of a period of time" analysis result:**



| | 2003-0... | 2003-0... | 2003-0... | 2003-0... | 2003-0... | 2003-0... | 2003-07... | 2003-07-... | 2003-07-... | 2003-08... | 2003-08... | 2003-08-... | 2003-08... |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| getpage_tot-BP1 | 40.67 | 40.67 | 40.67 | 683.00 | 40.67 | 61.00 | 196.33 | 201.67 | 213.00 | 264.33 | 160.33 | 212.50 | 396.5 |
| getpage_tot-BP2 | 3,975.33 | 3,975.33 | 3,975.33 | 669.67 | 3,975.33 | 5,963.00 | 22,514.67 | 22,517.33 | 22,523.00 | 22,620.67 | 11,990.33 | 17,818.00 | 33,931.0 |
| read_page-BP1 | 137.67 | 121.67 | 120.67 | 1.67 | 120.67 | 183.00 | 17.67 | 19.00 | 17.67 | 18.67 | 18.67 | 28.00 | 28.0 |
| read_page-BP2 | 3,189.00 | 4,033.67 | 4,025.00 | 10.67 | 3,959.00 | 5,938.50 | 21,907.33 | 21,928.00 | 21,965.00 | 22,801.00 | 12,004.67 | 17,861.00 | 34,169.5 |
| hit_ratio-BP1 | 77.87% | 61.48% | 63.93% | 99.76% | 63.93% | 63.93% | 97.62% | 97.69% | 97.81% | 98.11% | 96.88% | 96.47% | 98.11% |
| hit_ratio-BP2 | 99.95% | 99.90% | 99.91% | 99.85% | 99.91% | 99.91% | 99.54% | 99.56% | 99.56% | 99.59% | 99.95% | 99.95% | 99.68% |

*Figure 32. Long-Term Analysis – Example of "View of a period of time" analysis result*

This analysis type shows counter values of selected counters and objects from several bpd files in chronological order. Counter values represent per-minute values, as in the previous analysis types. One counter value per counter, object, and bpd file is shown, for example, 1 000 Read request operations (the counter) on BP0 (the object) on average from data from the first bpd file, 1 050 for the same counter and object from the second bpd file, and so on. The counter values are connected through lines for better readability. The y-axis on the left side is applicable to counter values, the one on the right side to ratios, if those were selected.

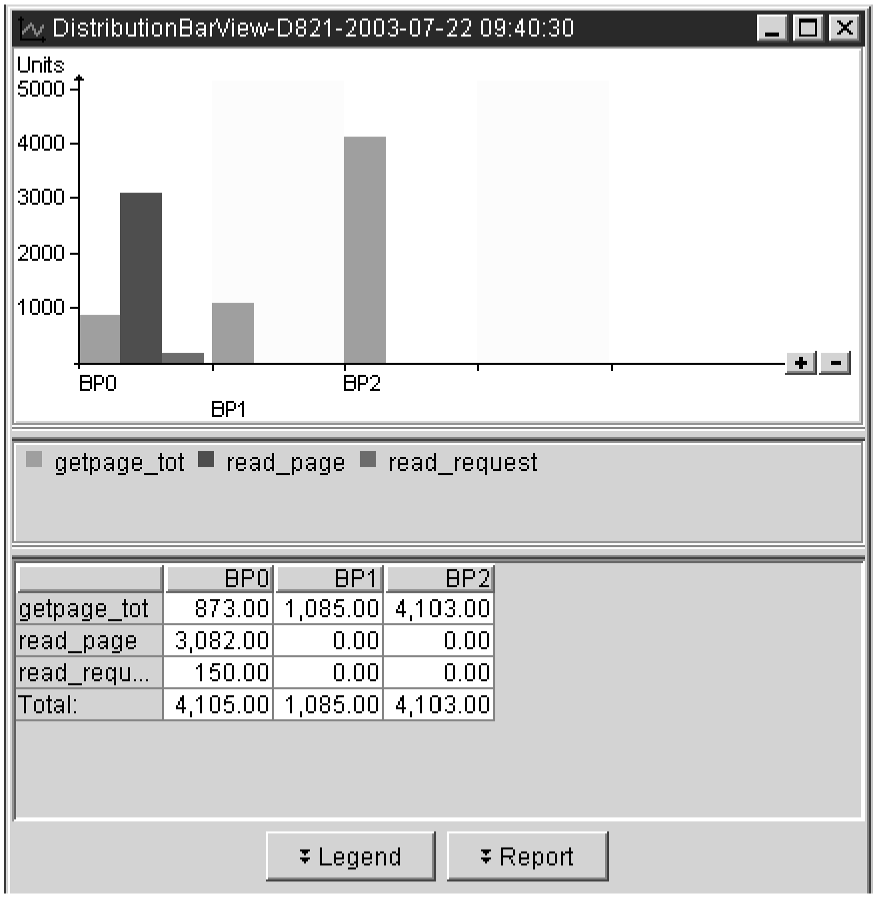
This analysis type provides meaningful information when several bpd files were selected, and if the effect of this selection was not canceled by a restrictive specification of a time frame. For example, if you have selected seven bpd files, whereby each file contains performance data of one subsequent day, but you have restricted the time frame to the second and third day, only these two bpd files are effectively used in this analysis.

Each effectively used bpd file is identified on the x-axis by a timestamp, and the files are shown in ascending order from left to right. The identifying timestamp is the timestamp of the latest performance record found in a bpd file (which might not necessarily be used in this analysis if the time frame restricts the use to some time before the latest record).

A further clarification on the y-values is appropriate: The calculated per-minute values of counters and ratios of effectively used bpd files are based on the specified (or default) time frame. For example, if the latest bpd file contains performance records of one day between 8:00 a.m. and 12:00 a.m., but you have specified a time frame limit of 9:00 a.m. (for whatever reason), the values for selected counters and ratios are calculated based on performance records between 8:00 a.m. and 9:00 a.m. (Nevertheless, the identifying timestamp for this bpd file on the x-axis shows 12:00 a.m.)

You can use this analysis type to analyze how certain counters develop over long periods, by using your portfolio of historical bpd files.

**A "Bar chart" analysis result:**


DistributionBarView-D821-2003-07-22 09:40:30
Units
5000
4000
3000
2000
1000
BP0
BP1
BP2
■ getpage_tot ■ read_page ■ read_request
Legend
Report

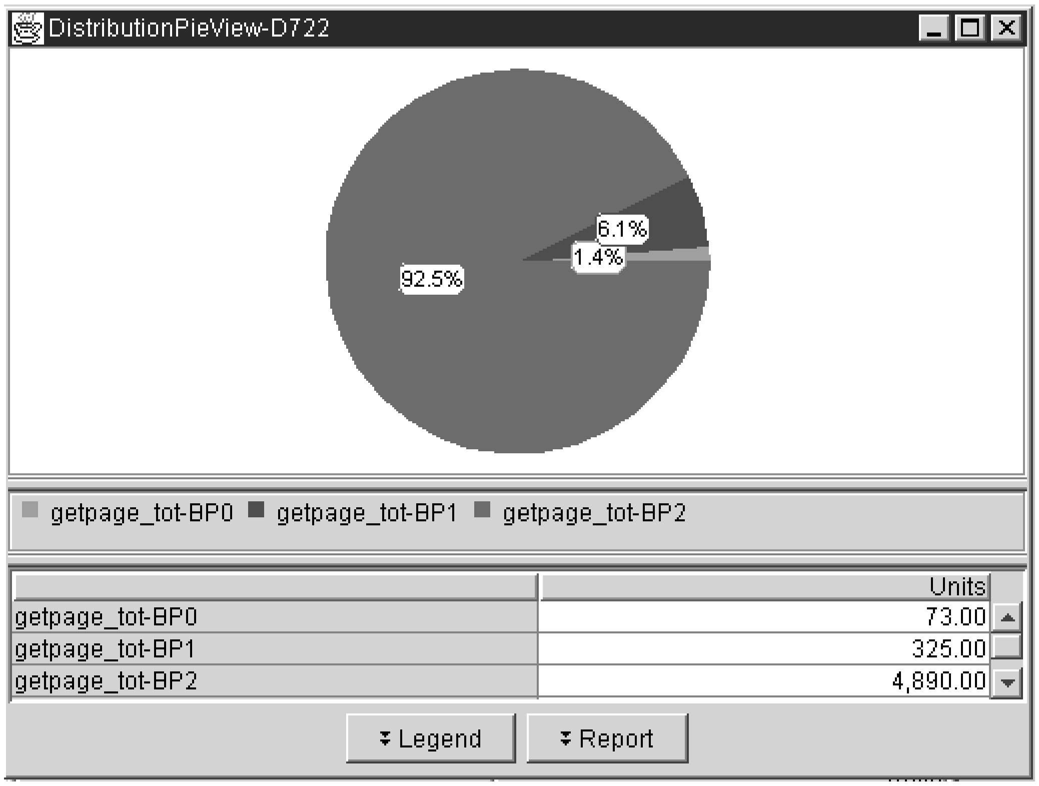
| | BP0 | BP1 | BP2 |
|---|---|---|---|
| getpage_tot | 873.00 | 1,085.00 | 4,103.00 |
| read_page | 3,082.00 | 0.00 | 0.00 |
| read_requ... | 150.00 | 0.00 | 0.00 |
| Total: | 4,105.00 | 1,085.00 | 4,103.00 |

*Figure 33. Long-Term Analysis – Example of "Bar chart" analysis result*

This analysis type shows the distribution of counter values of selected counters over selected objects as bar chart. As usual, counter values are per-minute values. The selected counters in this example are the Getpage total counter, the Read page counter, and the Read request counter. The selected objects are either one or more objects of one or more buffer pools, or all objects of one or more buffer pools, dependent on your selections in "Step 3: Specifying counters, objects, time frame, and output" on page 125. In this example, buffer pools BP0, BP1, and BP2 were selected, which means that the three counters encompass the activities of all objects in these buffer pools. The x-axis reflects the selected objects, here the buffer pools.

You can use this analysis type to easily compare selected counters in selected objects, for example, to compare the workload in selected buffer pools.

**A "Pie chart: display 1 counter and n objects" analysis result:**



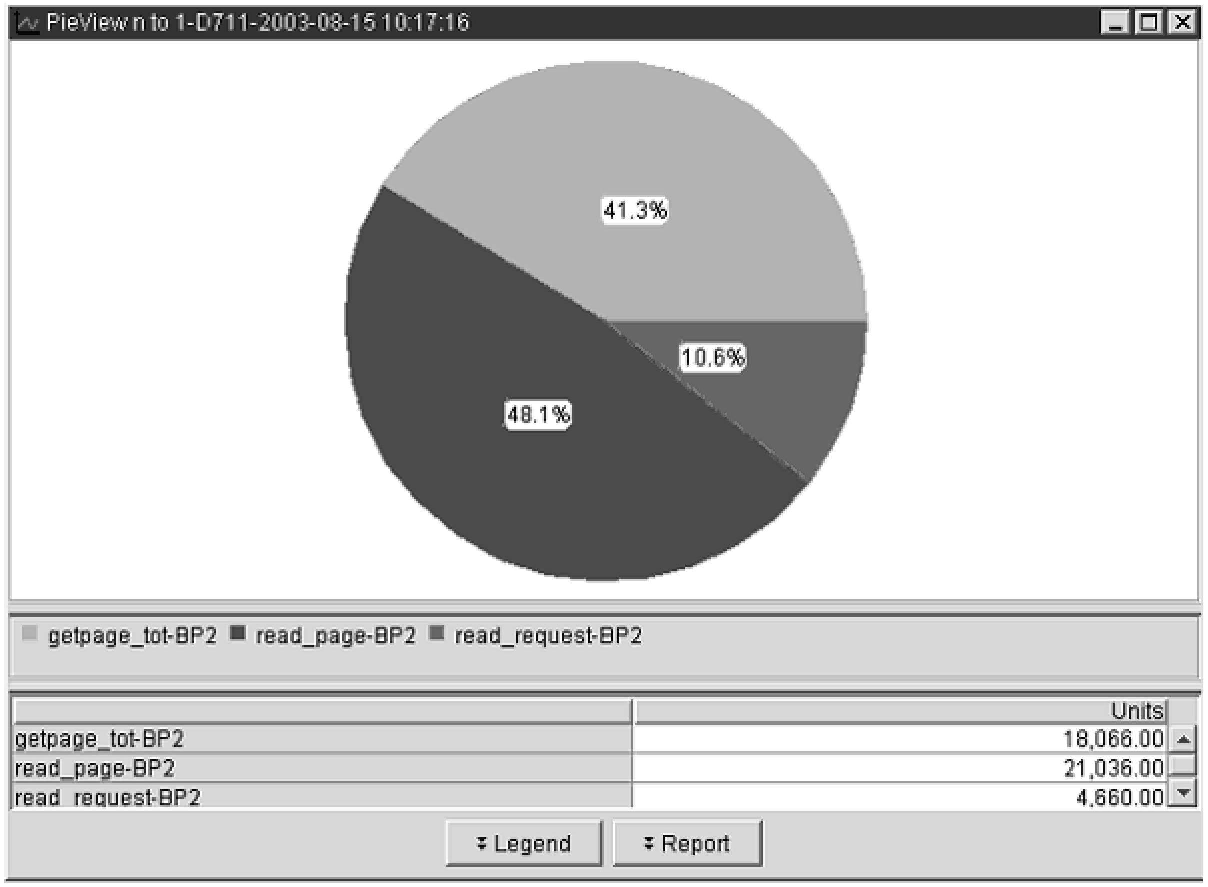
| | Units |
|---|---|
| getpage_tot-BP0 | 73.00 |
| getpage_tot-BP1 | 325.00 |
| getpage_tot-BP2 | 4,890.00 |

*Figure 34. Long-Term Analysis – "Pie chart: display 1 counter and n objects" analysis result*

This analysis type shows a *1-to-n* relationship of a selected counter to several selected objects as pie chart. Each slice of the pie represents one of the selected objects; the size of the slice corresponds to the percentage of the total of all selected objects. In this example, the selected counter is the Getpage total counter, and the selected objects are the objects in buffer pools BP0, BP1, and BP2. The percentages are shown in the graphic; the corresponding values per object (as per-minute values) are shown in the report following the graphic.

You can use this analysis type to compare a few values to a total, for example, to determine how much of the Getpage total activity happens in the most important buffer pools.

**A "Pie chart: display n counters and 1 object" analysis result:**



| | Units |
|---|---|
| getpage_tot-BP2 | 18,066.00 |
| read_page-BP2 | 21,036.00 |
| read_request-BP2 | 4,660.00 |

*Figure 35. Long-Term Analysis – "Pie chart: display n counters and 1 object" analysis result*

This analysis type shows a *n-to-1* relationship of several selected counters to a selected object as pie chart. Each slice of the pie represents one of the selected counters; the size of the slice corresponds to the percentage of the total of all selected counters. In this example, the selected counters are Getpage total, Read page, and Read request, and the selected object is BP2. The percentages are shown in the graphic; the corresponding values per counter (as per-minute values) are shown in the report following the graphic.

You can use this analysis type to compare a few values to a total, for example, to determine which counters have the most activity in a buffer pool.

# Chapter 10. Example of a use case

This section describes an example of a use case that explains how Buffer Pool Analyzer tools can be used.

The example supports the generalized approach given in "Buffer pool analysis and tuning processes" on page 8. It is assumed that you already have a reasonable understanding of host-based activity reports (described in Chapter 5, "Interpreting activity reports," on page 47) and the use of the object placement and simulation functions (described in and Chapter 8, "Simulating buffer pool behavior," on page 111). The example is applicable for Buffer Pool Analyzer for z/OS, respectively the Buffer Pool Analysis function of Db2 Performance Expert for z/OS.

The example shows:

1. How summary reports are used to quickly identify major buffer pool performance characteristics.
2. How detail reports are used to identify the most active and most expensive objects (in terms of synchronous operations).
3. How object placement and simulation are used to analyze the effects of different buffer pool attributes.

## Using reports to analyze trace data

The following steps show how summary reports and detail reports are used to identify objects of interest.

The reports are shown partially; certain issues are highlighted.

1. Creating host reports and a buffer pool data file with batch job BPOQBTCH (described in "Specifying a JCL command stream" on page 38):

   Data is collected over 30 minutes, with short record format. The trace data file is used for activity reports and the bpd file generation. The trace data file and the bpd file are used in later steps for object placements and simulations on the client.

2. Analyzing the activity reports:

   a. The summary report, ordered by `BPID-QPAGESET` and sorted by `ASYNCPAGE`, shows the buffer pool configuration (in the Buffer Pool Characteristics report section) and high-level activity (in the Buffer Pool Statistics report section):

```
                    =========   Buffer Pool Characteristics   =========
BPID                          BP0      BP1      BP2      BP3      BP10    BP32K
------------------------- -------- -------- -------- -------- -------- --------
General
 Virtual pool size           1000     2000   297525   297525     3000      100


                     =======     Buffer Pool Statistics     =======
BUFFER  POOL ID                BP0        BP1        BP2        BP3       BP10
------------------------ ---------- ---------- ---------- ---------- ----------
Reached threshold

 Deferred write                  0          0          0          0          0
 Vertical deferred write         0          0          0          0          0
 Data manager                    0          0          0          0          0
System hit ratio             39.73      97.15      75.03      99.24     100.00
Application hit ratio        83.15     100.00      97.77      99.68     100.00
Getpage request                813        561     321601     665989       1018
 Sequential                      6        435       6026         41         56
 Random                        807        126     315575     665948        962
Read
 Sequential prefetch
  Pages read                    49         16       2857         59          0
 Dynamic prefetch
  Pages read                   304          0      70262       2876          0
Write
 Page write                      3          0      10785       1222          0
```

   The report shows:

   - The system hit ratio is low for buffer pool BP2. The number of pages that are written to disk is moderate.

- The applications are doing a lot of random Getpage operations in buffer pools BP2 and BP3.
- Many Getpage operations in buffer pool BP2 are converted to Dynamic prefetch operations.
- Dynamic prefetches are significant higher than Sequential prefetches, which implies that the application is causing unneeded prefetches. In this example, it would be advantageous if the applications could be modified to move away from the current random scanning of tables or indexes to obvious scanning. This would change the prefetch behavior from dynamic to sequential and allow the optimizer to plan prefetches in a much more efficient way.

b. The detail report, ordered by `BPID-QPAGESET` and sorted by `BPID` and `GETPAGE`, shows the most active objects in the Detail Activity report section:

```
                     =======     Detail Activity     =======
   BPID                  BP2        BP2        BP2        BP2        BP2        BP2
   QPAGESET          WTNTEST    WTNTEST    WTNTEST    WTNTEST    WTNTEST    WTNTEST
                      WTNSEC     WTNACT     WTNSEC     WTNSE2     WTNSE3     WTNSE1
   ------------- ---------- ---------- ---------- ---------- ---------- ----------
   BP Hit ratio(%)
    System             100.0       44.0       65.5       98.4       99.1       72.3
    Application        100.0       98.4       95.0       99.8       99.9       90.8

   Getpage           2037600     790940     761760     552562     496460     230257
    Sequential       1358310          0          0          0          0          0
    Random            679288     790940     761760     552562     496460     230257
    Miss random            0      12462      37987       1085        552      21277

…

   BPID                  BP2        BP2        BP2        BP2        BP2        BP2
   QPAGESET          WTNTEST    WTNTEST    WTNTEST    WTNTEST    WTNTEST    WTNTEST
                     WTNMMMB     WTNFRD     WTNHLD     WTNACT     WTNBND     WTCSE4
   ------------- ---------- ---------- ---------- ---------- ---------- ----------
   BP Hit ratio(%)
    System              97.7       38.1       37.8       49.5       64.4       65.1
    Application        100.0       92.6       53.0       90.1       99.2       93.6

   Getpage            152187     151998      76591      56897      48059      48048
    Sequential             0      85459          0          0          0          0
    Random            152187      66539      76591      56897      48059      48048
    Miss random           34      11227      35964       5620        367       3085



                          **********  TOTAL  **********
   BPID                  BP0        BP1        BP2        BP3       BP10 |    GRAND
                                                                         |    TOTAL
   ------------- ---------- ---------- ---------- ---------- ---------- ----------
   Getpage              6311     125821    5782260    5476585      14488 | 11405465
    Sequential            56      63134    1490275        883       2670 |  1557018
    Random              6255      62687    4291981    5475702      11818 |  9848443
    Miss random         1810          0     175602      59389          0 |   236801
```

The report shows:

- A very high proportion of the total system activity is concentrated in buffer pool BP2 in only a few table spaces.
- An even larger proportion of the random misses is concentrated in a subset of these table spaces.

c. The detail report, ordered by `BPID-QPAGESET` and sorted by `BPID` and `READSYNC`, shows the most "expensive" objects in terms of I/O:

```
   BPID                  BP2        BP2        BP2        BP2        BP2        BP2
   QPAGESET          WTNTEST    WTNTEST    WTNTEST    WTNTEST    WTNTEST    WTNTEST
                      WTNSEC     WTNHLD     WTNSE1     WTNRCK     WTNACT     WTNFRD
   ------------- ---------- ---------- ---------- ---------- ---------- ----------
   Getpage            761760      76591     230257      31457     790940     151998
    Sequential             0          0          0          0          0      85459
    Random            761760      76591     230257      31457     790940      66539
    Miss random        37987      35964      21277      14439      12462      11227

   Read request        47920      36408      24113      14610      43143      14103
    Synchronous        37984      35961      21274      14439      12458      11296
    Dyn prefetch        9936        447       2839        171      30685        137
    Delay(msec)
     Synchronous        10.8       34.9       25.8        4.2      148.7        6.5
     Dyn pref           12.9       21.3       17.3       25.7       19.2       12.9

   Read page          262868      47611      63832      18897     442639      94019
    Synchronous        37984      35961      21274      14439      12458      11296
    Dyn prefetch      224884      11650      42558       4458     430181       2691
```

The report shows:

- Again, a very large proportion of all disk accesses is concentrated in a relatively small number of objects.
- One object stands out by a different average delay time on synchronous read operations.

The conclusion so far is: It is worth simulating the effect of moving such objects into a buffer pool of their own.

The recommendation so far is: Objects with frequent misses should be moved to a faster disk, if possible.

## Analyzing effects of different buffer pool attributes

The following steps show the effects of using object placement and simulation, based on the information found in the activity reports.

1. Performing a simulation using the actual object placements:
   a. The four most active buffer pools BP1, BP2, BP3, and BP10 are chosen.
   b. A minimum and maximum page size for all buffer pools of 25 000 and 1 000 000 and a simulation interval of 25 000 is chosen (the actual buffer pool size for BP2 and BP3 is approximately 300 000 pages each).
   c. The object to buffer pool assignments remain unchanged (the actual object placements are to be simulated).

*Table 12. Comparing separate Buffer Pools versus a single combined Buffer Pool*

| | **Separate Buffer Pools** | | | **Combined Buffer Pool** | | |
|---|---|---|---|---|---|---|
| **Total Pages** | **Misses** | **Application Hit Ratio** | **Global Miss Ratio** | **Misses** | **Application Hit Ratio** | **Global Miss Ratio** |
| 100000 | 2418999 | 79.1 | 20.9 | 463512 | 96.0 | 4.0 |
| 125000 | 880442 | 92.4 | 7.6 | 436713 | 96.2 | 3.8 |
| 150000 | 464793 | 96.0 | 4.0 | 412713 | 96.4 | 3.6 |
| 175000 | 439478 | 96.2 | 3.8 | 389354 | 96.6 | 3.4 |
| 200000 | 415460 | 96.4 | 3.6 | 366904 | 96.8 | 3.2 |
| 225000 | 397097 | 96.6 | 3.4 | 353614 | 96.9 | 3.0 |
| 250000 | 376870 | 96.7 | 3.3 | 333667 | 97.1 | 2.9 |
| 275000 | 354308 | 96.9 | 3.1 | 319326 | 97.2 | 2.8 |
| 300000 | 333677 | 97.1 | 2.9 | 308144 | 97.3 | 2.7 |
| 325000 | 317252 | 97.3 | 2.7 | 298350 | 97.4 | 2.6 |
| 350000 | 305352 | 97.4 | 2.6 | 290182 | 97.5 | 2.5 |
| 375000 | 295169 | 97.5 | 2.5 | 284812 | 97.5 | 2.5 |
| 400000 | 287248 | 97.5 | 2.5 | 277367 | 97.6 | 2.4 |
| 425000 | 280833 | 97.6 | 2.4 | 273104 | 97.6 | 2.4 |
| 450000 | 274582 | 97.6 | 2.4 | 268341 | 97.7 | 2.3 |
| 500000 | 264242 | 97.7 | 2.3 | 257447 | 97.8 | 2.2 |
| ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ |

Table 12. Comparing separate Buffer Pools versus a single combined Buffer Pool (continued)

| | Separate Buffer Pools | | | Combined Buffer Pool | | |
|---|---|---|---|---|---|---|
| **Total Pages** | **Misses** | **Application Hit Ratio** | **Global Miss Ratio** | **Misses** | **Application Hit Ratio** | **Global Miss Ratio** |
| 975000 | 210329 | 98.2 | 1.8 | 208484 | 98.2 | 1.8 |
| 1000000 | 209556 | 98.2 | 1.8 | 208140 | 98.2 | 1.8 |

- For a total buffer pool size of less than 300 000 pages the hit ratio is better if BP2 and BP3 are combined in a single buffer pool. Above 300 000 pages the improvement is marginal. An object placement should be performed to determine if there is a more favorable distribution of objects.
- It is noticeable that even around 1 000 000 pages the buffer pool hit ratio still improves slowly with further memory.

Table 13. Recommended sizing for separate Buffer Pools

| Total Pages | BP1 pages | BP2 pages | BP3 pages | BP10 pages |
|---|---|---|---|---|
| 100000 | 25000 | 25000 | 25000 | 25000 |
| 125000 | 25000 | 50000 | 25000 | 25000 |
| 150000 | 25000 | 50000 | 50000 | 25000 |
| 175000 | 25000 | 75000 | 50000 | 25000 |
| 200000 | 25000 | 100000 | 50000 | 25000 |
| 225000 | 25000 | 125000 | 50000 | 25000 |
| 250000 | 25000 | 150000 | 50000 | 25000 |
| 275000 | 25000 | 175000 | 50000 | 25000 |
| 300000 | 25000 | 200000 | 50000 | 25000 |
| 325000 | 25000 | 225000 | 50000 | 25000 |
| 350000 | 25000 | 250000 | 50000 | 25000 |
| 375000 | 25000 | 275000 | 50000 | 25000 |
| 400000 | 25000 | 275000 | 75000 | 25000 |
| 425000 | 25000 | 300000 | 75000 | 25000 |
| 450000 | 25000 | 325000 | 75000 | 25000 |
| 475000 | 25000 | 325000 | 100000 | 25000 |
| 500000 | 25000 | 350000 | 100000 | 25000 |
| 525000 | 25000 | 375000 | 100000 | 25000 |
| 550000 | 25000 | 425000 | 75000 | 25000 |
| 575000 | 25000 | 425000 | 100000 | 25000 |
| 600000 | 25000 | 425000 | 125000 | 25000 |
| 625000 | 25000 | 425000 | 150000 | 25000 |
| 650000 | 25000 | 425000 | 175000 | 25000 |
| 675000 | 25000 | 450000 | 175000 | 25000 |

Table 13. Recommended sizing for separate Buffer Pools (continued)

| Total Pages | BP1 pages | BP2 pages | BP3 pages | BP10 pages |
|---|---|---|---|---|
| 700000 | 25000 | 450000 | 200000 | 25000 |
| 725000 | 25000 | 450000 | 225000 | 25000 |
| 750000 | 25000 | 600000 | 100000 | 25000 |
| 775000 | 25000 | 575000 | 150000 | 25000 |
| 800000 | 25000 | 600000 | 150000 | 25000 |
| 825000 | 25000 | 600000 | 175000 | 25000 |
| 850000 | 25000 | 625000 | 175000 | 25000 |
| 875000 | 25000 | 650000 | 175000 | 25000 |
| 900000 | 25000 | 650000 | 200000 | 25000 |
| 925000 | 25000 | 675000 | 200000 | 25000 |
| 950000 | 25000 | 675000 | 225000 | 25000 |
| 975000 | 25000 | 675000 | 250000 | 25000 |
| 1000000 | 25000 | 725000 | 225000 | 25000 |

- The most important result is the allocation of memory between the two main buffer pools BP2 and BP3 (each one actually using approximately 300 000 pages): For a total of 600 000 pages, the simulation recommends to allocate three times as much memory to BP2 as BP3.
- The values for buffer pools BP1 and BP10 are wasteful and caused by the high interval value simulated (25 000 pages). An additional simulation between 500 and 5 000 pages, only for BP0, BP1, BP4 and BP10, will give a better idea of the amount of memory they really require, but this is absolutely insignificant compared to BP2 and BP3.

"Simulated behavior of each separate Buffer Pool" of the simulation result shows:

Table 14. Results of simulated behavior of separate buffer pools

| | Buffer Pool BP1 | | | Buffer Pool BP2 | | | Buffer Pool BP3 | | | Buffer Pool BP10 | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Buffer Pool Pages | Misses | Applic. Hit Ratio | Global Miss Ratio | Misses | Applic. Hit Ratio | Global Miss Ratio | Misses | Applic. Hit Ratio | Global Miss Ratio | Misses | Applic. Hit Ratio | Global Miss Ratio |
| 25000 | 359 | 99.8 | 0.0 | 1909336 | 67.3 | 16.5 | 509028 | 90.8 | 4.4 | 276 | 98.1 | 0.0 |
| 50000 | 359 | 99.8 | 0.0 | 370779 | 93.6 | 3.2 | 93379 | 98.3 | 0.8 | 276 | 98.1 | 0.0 |
| 75000 | 359 | 99.8 | 0.0 | 345464 | 94.1 | 3.0 | 85458 | 98.5 | 0.7 | 276 | 98.1 | 0.0 |
| 100000 | 359 | 99.8 | 0.0 | 321446 | 94.5 | 2.8 | 79923 | 98.6 | 0.7 | 276 | 98.1 | 0.0 |
| 125000 | 359 | 99.8 | 0.0 | 303083 | 94.8 | 2.6 | 76671 | 98.6 | 0.7 | 276 | 98.1 | 0.0 |
| 150000 | 359 | 99.8 | 0.0 | 282856 | 95.2 | 2.4 | 7352 | 98.7 | 0.6 | 276 | 98.1 | 0.0 |
| 175000 | 359 | 99.8 | 0.0 | 260294 | 95.5 | 2.2 | 70005 | 98.7 | 0.6 | 276 | 98.1 | 0.0 |
| 200000 | 359 | 99.8 | 0.0 | 239663 | 95.9 | 2.1 | 68202 | 98.8 | 0.6 | 276 | 98.1 | 0.0 |
| 225000 | 359 | 99.8 | 0.0 | 223238 | 96.2 | 1.9 | 67029 | 98.8 | 0.0 | 276 | 98.1 | 0.0 |
| 250000 | 359 | 99.8 | 0.0 | 211338 | 96.4 | 1.8 | 66103 | 98.8 | 0.6 | 276 | 98.1 | 0.0 |
| 275000 | 359 | 99.8 | 0.0 | 201155 | 96.6 | 1.7 | 65542 | 98.8 | 0.6 | 276 | 98.1 | 0.0 |
| 300000 | 359 | 99.8 | 0.0 | 194740 | 96.7 | 1.7 | 63988 | 98.8 | 0.6 | 276 | 98.1 | 0.0 |
| 325000 | 359 | 99.8 | 0.0 | 188489 | 96.8 | 1.6 | 63473 | 98.9 | 0.5 | 276 | 98.1 | 0.0 |
| 350000 | 359 | 99.8 | 0.0 | 183684 | 96.9 | 1.6 | 63102 | 98.9 | 0.5 | 276 | 98.1 | 0.0 |

Table 14. Results of simulated behavior of separate buffer pools (continued)

| | Buffer Pool BP1 | | | Buffer Pool BP2 | | | Buffer Pool BP3 | | | Buffer Pool BP10 | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Buffer Pool Pages | Misses | Applic. Hit Ratio | Global Miss Ratio | Misses | Applic. Hit Ratio | Global Miss Ratio | Misses | Applic. Hit Ratio | Global Miss Ratio | Misses | Applic. Hit Ratio | Global Miss Ratio |
| 375000 | 359 | 99.8 | 0.0 | 179376 | 96.9 | 1.5 | 62749 | 98.9 | 0.5 | 276 | 98.1 | 0.0 |
| 400000 | 359 | 99.8 | 0.0 | 174842 | 97.0 | 1.5 | 62528 | 98.9 | 0.5 | 276 | 98.1 | 0.0 |
| 425000 | 359 | 99.8 | 0.0 | 168940 | 97.1 | 1.5 | 62024 | 98.9 | 0.5 | 276 | 98.1 | 0.0 |
| 450000 | 359 | 99.8 | 0.0 | 165805 | 97.2 | 1.4 | 60979 | 98.9 | 0.5 | 276 | 98.1 | 0.0 |
| 475000 | 359 | 99.8 | 0.0 | 164905 | 97.2 | 1.4 | 60853 | 98.9 | 0.5 | 276 | 98.1 | 0.0 |
| 500000 | 359 | 99.8 | 0.0 | 163878 | 97.2 | 1.4 | 60850 | 98.9 | 0.5 | 276 | 98.1 | 0.0 |
| 525000 | 359 | 99.8 | 0.0 | 162486 | 97.2 | 1.4 | 60850 | 98.9 | 0.5 | 276 | 98.1 | 0.0 |
| 550000 | 359 | 99.8 | 0.0 | 159330 | 97.3 | 1.4 | 60850 | 98.9 | 0.5 | 276 | 98.1 | 0.0 |
| 575000 | 359 | 99.8 | 0.0 | 154300 | 97.4 | 1.3 | 60850 | 98.9 | 0.5 | 276 | 98.1 | 0.0 |
| 600000 | 359 | 99.8 | 0.0 | 150934 | 97.4 | 1.3 | 60850 | 98.9 | 0.5 | 276 | 98.1 | 0.0 |
| 625000 | 359 | 99.8 | 0.0 | 148159 | 97.5 | 1.3 | 60850 | 98.9 | 0.5 | 276 | 98.1 | 0.0 |
| 650000 | 359 | 99.8 | 0.0 | 145146 | 97.5 | 1.3 | 60850 | 98.9 | 0.5 | 276 | 98.1 | 0.0 |
| 675000 | 359 | 99.8 | 0.0 | 143591 | 97.5 | 1.2 | 60850 | 98.9 | 0.5 | 276 | 98.1 | 0.0 |
| 700000 | 359 | 99.8 | 0.0 | 143263 | 97.5 | 1.2 | 60850 | 98.9 | 0.5 | 276 | 98.1 | 0.0 |
| 725000 | 359 | 99.8 | 0.0 | 141892 | 97.6 | 1.2 | 60850 | 98.9 | 0.5 | 276 | 98.1 | 0.0 |
| 750000 | 359 | 99.8 | 0.0 | 141783 | 97.6 | 1.2 | 60850 | 98.9 | 0.5 | 276 | 98.1 | 0.0 |
| 775000 | 359 | 99.8 | 0.0 | 141569 | 97.6 | 1.2 | 60850 | 98.9 | 0.5 | 276 | 98.1 | 0.0 |
| 800000 | 359 | 99.8 | 0.0 | 141491 | 97.6 | 1.2 | 60850 | 98.9 | 0.5 | 276 | 98.1 | 0.0 |
| 825000 | 359 | 99.8 | 0.0 | 141377 | 97.6 | 1.2 | 60850 | 98.9 | 0.5 | 276 | 98.1 | 0.0 |
| 850000 | 359 | 99.8 | 0.0 | 141331 | 97.6 | 1.2 | 60850 | 98.9 | 0.5 | 276 | 98.1 | 0.0 |
| 875000 | 359 | 99.8 | 0.0 | 141281 | 97.6 | 1.2 | 60850 | 98.9 | 0.5 | 276 | 98.1 | 0.0 |
| 900000 | 359 | 99.8 | 0.0 | 141275 | 97.6 | 1.2 | 60850 | 98.9 | 0.5 | 276 | 98.1 | 0.0 |
| 925000 | 359 | 99.8 | 0.0 | 141275 | 97.6 | 1.2 | 60850 | 98.9 | 0.5 | 276 | 98.1 | 0.0 |
| 950000 | 359 | 99.8 | 0.0 | 141275 | 97.6 | 1.2 | 60850 | 98.9 | 0.5 | 276 | 98.1 | 0.0 |
| 975000 | 359 | 99.8 | 0.0 | 141275 | 97.6 | 1.2 | 60850 | 98.9 | 0.5 | 276 | 98.1 | 0.0 |
| 1000000 | 359 | 99.8 | 0.0 | 141275 | 97.6 | 1.2 | 60850 | 98.9 | 0.5 | 276 | 98.1 | 0.0 |

- The "Misses" columns for buffer pools BP2 and BP3 show that, with unlimited memory, no improvement would be achieved with more than 900 000 pages for BP2 and 500 000 pages for BP3. These are the absolute limits, although economical limits will always be lower.

The other tables in this simulation result are only required for very detailed analysis, especially for application tuning.

2. Performing object placement with default rule set and object placements:

   a. The default rule set is used; in this case `pattern_large`.

   b. No changes are made to rules or placements. The object placement wizard is started by simply clicking Next.

   "ALTER BUFFERPOOL COMMANDS" of the object placement result shows the new buffer pools and the recommended sizes:

```
ALTER BUFFERPOOL(BP3) VPSIZE(119555) HPSIZE(0) VPSEQT(20) DWQT(39) VDWQT(10,0)
ALTER BUFFERPOOL(BP2) VPSIZE(392018) HPSIZE(0) VPSEQT(20) DWQT(0) VDWQT(0,0)
ALTER BUFFERPOOL(BP10) VPSIZE(15046) HPSIZE(0) VPSEQT(40) DWQT(10) VDWQT(3,0)
ALTER BUFFERPOOL(BP0) VPSIZE(15046) HPSIZE(0) VPSEQT(20) DWQT(25) VDWQT(6,0)
ALTER BUFFERPOOL(BP32K) VPSIZE(1881) HPSIZE(0) VPSEQT(40) DWQT(10) VDWQT(3,0)
ALTER BUFFERPOOL(BP4) VPSIZE(15046) HPSIZE(0) VPSEQT(98) DWQT(2) VDWQT(0,0)
ALTER BUFFERPOOL(BP1) VPSIZE(15046) HPSIZE(0) VPSEQT(100) DWQT(70) VDWQT(50,0)
```

"ALTER TABLESPACE AND ALTER INDEX STATEMENTS" of the object placement result shows the object placement commands (for those objects being allocated to a different buffer pool):

```
ALTER TABLESPACE WTNTEST.WTNADD BUFFERPOOL BP3;
ALTER INDEX WTNTEST.WTNADR01 BUFFERPOOL BP2;
ALTER TABLESPACE WTNTEST.WTNARC BUFFERPOOL BP3;
ALTER TABLESPACE WTNTEST.WTNHST BUFFERPOOL BP3;
ALTER INDEX WTNTEST.WTNTRD01 BUFFERPOOL BP2;
ALTER INDEX WTNTEST.WTNAST01 BUFFERPOOL BP2;
ALTER INDEX WTNTEST.WTNBAL01 BUFFERPOOL BP2;
```

- The actual object assignments (at data collection time) separate data from indexes.
- The default rule set in the selected pattern file has separated objects primarily according to the amount of sequential and dynamic access.

"OBJECT PLACEMENT OVERVIEW" of the object placement result shows attributes and placement of objects:

*Table 15. Results of object placement*

| Object Name | Type | Page | Used | Catalog / Directory | Seq. Access [%] | Change Rate [%] | Size [pages ] | Current | Recommended | User-defined |
|---|---|---|---|---|---|---|---|---|---|---|
| WTNTEST.DBD01 | TABLESPACE | 4K | YES | DIR | 0 | 133 | ? | BP0 | BP0 | BP0 |
| WTNTEST.DSNLLX01 | INDEX | 4K | YES | DIR | 0 | 6 | ? | BP0 | BP0 | BP0 |
| WTNTEST.DSNLLX02 | INDEX | 4K | YES | DIR | 0 | 28 | ? | BP0 | BP0 | BP0 |
| WTNTEST.DSNLUX01 | INDEX | 4K | YES | DIR | 0 | 9 | ? | BP0 | BP0 | BP0 |
| WTNTEST.DSNLUX02 | INDEX | 4K | YES | DIR | 0 | 21 | ? | BP0 | BP0 | BP0 |
| ⋮ | | | | | | | | | | |

3. Performing a simulation with results from object placement:
   a. The two most active buffer pools BP2 and BP3 are chosen (they contain almost all database activity).
   b. A minimum and maximum page size for all buffer pools of 100 000 and 1 000 000 and a simulation interval of 25 000 is chosen.

"Simulated behavior of each separate Buffer Pool" of the simulation result shows:

*Table 16. Results of simulated behavior of most active buffer pools*

| | Buffer Pool BP2 | | | Buffer Pool BP3 | | |
|---|---|---|---|---|---|---|
| **Buffer Pool Pages** | **Misses** | **Application Hit Ratio** | **Global Miss Ratio** | **Misses** | **Application Hit Ratio** | **Global Miss Ratio** |
| 100000 | 437337 | 96.0 | 3.8 | 12904 | 97.9 | 0.1 |
| 125000 | 408394 | 96.3 | 3.5 | 12886 | 97.9 | 0.1 |
| 150000 | 366438 | 96.5 | 3.3 | 12886 | 97.9 | 0.1 |
| 175000 | 362053 | 96.7 | 3.1 | 12886 | 97.9 | 0.1 |
| 200000 | 341920 | 96.9 | 2.9 | 12886 | 97.9 | 0.1 |
| 225000 | 325018 | 97.0 | 2.8 | 12886 | 97.9 | 0.1 |
| 250000 | 305094 | 97.2 | 2.6 | 12886 | 97.9 | 0.1 |
| 275000 | 293311 | 97.3 | 2.5 | 12886 | 97.9 | 0.1 |
| 300000 | 282036 | 97.4 | 2.4 | 12886 | 97.9 | 0.1 |

| Table 16. Results of simulated behavior of most active buffer pools (continued) | | | | | | |
|---|---|---|---|---|---|---|
| | Buffer Pool BP2 | | | Buffer Pool BP3 | | |
| **Buffer Pool Pages** | **Misses** | **Application Hit Ratio** | **Global Miss Ratio** | **Misses** | **Application Hit Ratio** | **Global Miss Ratio** |
| 325000 | 273809 | 97.5 | 2.4 | 12886 | 97.9 | 0.1 |
| 350000 | 266564 | 97.6 | 2.3 | 12886 | 97.9 | 0.1 |
| 375000 | 259972 | 97.6 | 2.2 | 12886 | 97.9 | 0.1 |
| 400000 | 250910 | 97.7 | 2.2 | 12886 | 97.9 | 0.1 |
| ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ |

- The application hit ratio for buffer pool BP2 is marginally better than in the previous simulation (look at misses for a given number of total buffer pool pages).
- Buffer pool BP3 is already optimal at 100 000.

4. Performing object placement with modified rule set:
   a. The default rule set `pattern_large` is selected and edited as follows:
      i) A duplicate BP2, called BP12, is added directly under BP2.
      ii) A duplicate BP3, called BP13, is added directly under BP3.
      iii) BP2 and BP3 should only be used for table spaces.
      iv) BP12 and BP13 should only be used for index spaces.
   b. The edited pattern file is saved with a new name. This pattern will be chosen as the default for all future placements with this subsystem.
   c. No changes are made to placements.
5. Performing simulation with results from object placement with modified rule set:
   a. Buffer pools BP2, BP3, BP12, and BP13 are chosen.
   b. A minimum and maximum page size for all buffer pools of 25 000 and 1 000 000 and a simulation interval of 25 000 is chosen.

   "Recommended sizing for separate Buffer Pools" of the simulation result shows:

*Table 17. Recommended sizing for separate buffer pools with modified rule set*

| Total Pages | BP2 pages | BP3 pages | BP12 pages | BP13 pages |
|---|---|---|---|---|
| 100000 | 25000 | 25000 | 25000 | 25000 |
| 125000 | 50000 | 25000 | 25000 | 25000 |
| 150000 | 50000 | 25000 | 50000 | 25000 |
| 175000 | 75000 | 25000 | 50000 | 25000 |
| 200000 | 100000 | 25000 | 50000 | 25000 |
| 225000 | 125000 | 25000 | 50000 | 25000 |
| 250000 | 150000 | 25000 | 50000 | 25000 |
| 275000 | 175000 | 25000 | 50000 | 25000 |
| 300000 | 200000 | 25000 | 50000 | 25000 |
| 325000 | 225000 | 25000 | 50000 | 25000 |
| 350000 | 250000 | 25000 | 50000 | 25000 |
| 375000 | 275000 | 25000 | 50000 | 25000 |
| 400000 | 300000 | 25000 | 50000 | 25000 |
| 425000 | 300000 | 25000 | 75000 | 25000 |

Table 17. Recommended sizing for separate buffer pools with modified rule set (continued)

| Total Pages | BP2 pages | BP3 pages | BP12 pages | BP13 pages |
|---|---|---|---|---|
| 450000 | 300000 | 25000 | 100000 | 25000 |
| 475000 | 325000 | 25000 | 100000 | 25000 |
| 500000 | 350000 | 25000 | 100000 | 25000 |
| 525000 | 350000 | 25000 | 125000 | 25000 |
| 550000 | 400000 | 25000 | 100000 | 25000 |
| 575000 | 425000 | 25000 | 100000 | 25000 |
| 600000 | 425000 | 25000 | 125000 | 25000 |
| 625000 | 425000 | 25000 | 150000 | 25000 |
| 650000 | 425000 | 25000 | 175000 | 25000 |
| 675000 | 425000 | 25000 | 200000 | 25000 |
| 700000 | 550000 | 25000 | 100000 | 25000 |
| 725000 | 575000 | 25000 | 100000 | 25000 |
| 750000 | 575000 | 25000 | 125000 | 25000 |
| 775000 | 600000 | 25000 | 125000 | 25000 |
| 800000 | 625000 | 25000 | 125000 | 25000 |
| 825000 | 625000 | 25000 | 150000 | 25000 |
| 850000 | 625000 | 25000 | 175000 | 25000 |
| 875000 | 650000 | 25000 | 175000 | 25000 |
| 900000 | 650000 | 25000 | 200000 | 25000 |
| 925000 | 675000 | 25000 | 200000 | 25000 |
| 950000 | 675000 | 25000 | 225000 | 25000 |
| 975000 | 675000 | 25000 | 250000 | 25000 |
| 1000000 | 650000 | 25000 | 300000 | 25000 |

- Most of the memory is allocated to BP2 followed by BP12.

"Simulated behavior of each separate Buffer Pool" of the simulation result shows:

Table 18. Results of simulated behavior of separate buffer pools with modified rule set

| | Buffer Pool BP2 | | | Buffer Pool BP3 | | | Buffer Pool BP12 | | | Buffer Pool BP13 | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Buffer Pool Pages | Misses | Applic. Hit Ratio | Global Miss Ratio | Misses | Applic. Hit Ratio | Global Miss Ratio | Misses | Applic. Hit Ratio | Global Miss Ratio | Misses | Applic. Hit Ratio | Global Miss Ratio |
| 25000 | 1870289 | 66.5 | 16.1 | 3529 | 99.2 | 0.0 | 490692 | 90.8 | 4.2 | 10037 | 94.7 | 0.1 |
| 50000 | 362552 | 93.5 | 3.1 | 3320 | 99.2 | 0.0 | 79115 | 98.5 | 0.7 | 9817 | 94.8 | 0.1 |
| 75000 | 337339 | 94.0 | 2.9 | 3320 | 99.2 | 0.0 | 72303 | 98.6 | 0.6 | 9686 | 95.0 | 0.1 |
| 100000 | 313468 | 94.4 | 2.7 | 3320 | 99.2 | 0.0 | 65437 | 98.8 | 0.6 | 9566 | 95.0 | 0.1 |
| 125000 | 294921 | 94.7 | 2.5 | 3320 | 99.2 | 0.0 | 62273 | 98.8 | 0.5 | 9566 | 95.0 | 0.1 |
| 150000 | 274273 | 95.1 | 2.4 | 3320 | 99.2 | 0.0 | 59767 | 98.9 | 0.5 | 9666 | 95.0 | 0.1 |
| 175000 | 251441 | 95.5 | 2.2 | 3320 | 99.2 | 0.0 | 58299 | 98.9 | 0.5 | 9566 | 95.0 | 0.1 |
| 200000 | 226048 | 95.9 | 1.9 | 3320 | 99.2 | 0.0 | 56990 | 98.9 | 0.5 | 9566 | 95.0 | 0.1 |
| 225000 | 211886 | 96.2 | 1.8 | 3320 | 99.2 | 0.0 | 56176 | 98.9 | 0.5 | 9566 | 95.0 | 0.1 |
| 250000 | 202081 | 96.4 | 1.7 | 3320 | 99.2 | 0.0 | 55614 | 99.0 | 0.5 | 9566 | 95.0 | 0.1 |
| 275000 | 193639 | 96.5 | 1.7 | 3320 | 99.2 | 0.0 | 55162 | 99.0 | 0.5 | 9566 | 95.0 | 0.1 |
| 300000 | 185904 | 96.7 | 1.6 | 3320 | 99.2 | 0.0 | 53560 | 99.0 | 0.5 | 9566 | 95.0 | 0.1 |
| 325000 | 180557 | 96.8 | 1.6 | 3320 | 99.2 | 0.0 | 52788 | 99.0 | 0.5 | 9566 | 95.0 | 0.1 |

Table 18. Results of simulated behavior of separate buffer pools with modified rule set (continued)

| | Buffer Pool BP2 | | | Buffer Pool BP3 | | | Buffer Pool BP12 | | | Buffer Pool BP13 | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Buffer Pool Pages | Misses | Applic. Hit Ratio | Global Miss Ratio | Misses | Applic. Hit Ratio | Global Miss Ratio | Misses | Applic. Hit Ratio | Global Miss Ratio | Misses | Applic. Hit Ratio | Global Miss Ratio |
| 350000 | 175611 | 96.9 | 1.5 | 3320 | 99.2 | 0.0 | 51485 | 99.0 | 0.4 | 9566 | 95.0 | 0.1 |
| 375000 | 172480 | 96.9 | 1.5 | 3320 | 99.2 | 0.0 | 51319 | 99.0 | 0.4 | 9566 | 95.0 | 0.1 |
| 400000 | 167135 | 97.0 | 1.4 | 3320 | 99.2 | 0.0 | 51299 | 99.0 | 0.4 | 9566 | 95.0 | 0.1 |
| 425000 | 161517 | 97.1 | 1.4 | 3320 | 99.2 | 0.0 | 51299 | 99.0 | 0.4 | 9566 | 95.0 | 0.1 |
| 450000 | 160539 | 97.1 | 1.4 | 3320 | 99.2 | 0.0 | 51299 | 99.0 | 0.4 | 9566 | 95.0 | 0.1 |
| 475000 | 159454 | 97.1 | 1.4 | 3320 | 99.2 | 0.0 | 51299 | 99.0 | 0.4 | 9566 | 95.0 | 0.1 |
| 500000 | 157582 | 97.2 | 1.4 | 3320 | 99.2 | 0.0 | 51299 | 99.0 | 0.4 | 9566 | 95.0 | 0.1 |
| 525000 | 153173 | 97.3 | 1.3 | 3320 | 99.2 | 0.0 | 51299 | 99.0 | 0.4 | 9566 | 95.0 | 0.1 |
| 550000 | 149690 | 97.3 | 1.3 | 3320 | 99.2 | 0.0 | 51299 | 99.0 | 0.4 | 9566 | 95.0 | 0.1 |
| 575000 | 146377 | 97.4 | 1.3 | 3320 | 99.2 | 0.0 | 51299 | 99.0 | 0.4 | 9566 | 95.0 | 0.1 |
| 600000 | 143229 | 97.4 | 1.2 | 3320 | 99.2 | 0.0 | 51299 | 99.0 | 0.4 | 9566 | 95.0 | 0.1 |
| 625000 | 140383 | 97.5 | 1.2 | 3320 | 99.2 | 0.0 | 51299 | 99.0 | 0.4 | 9566 | 95.0 | 0.1 |
| 650000 | 139060 | 97.5 | 1.2 | 3320 | 99.2 | 0.0 | 51299 | 99.0 | 0.4 | 9566 | 95.0 | 0.1 |
| 675000 | 137805 | 97.5 | 1.2 | 3320 | 99.2 | 0.0 | 51299 | 99.0 | 0.4 | 9566 | 95.0 | 0.1 |
| 700000 | 137718 | 97.5 | 1.2 | 3320 | 99.2 | 0.0 | 51299 | 99.0 | 0.4 | 9566 | 95.0 | 0.1 |
| 725000 | 137590 | 97.5 | 1.2 | 3320 | 99.2 | 0.0 | 51299 | 99.0 | 0.4 | 9566 | 95.0 | 0.1 |
| 750000 | 137416 | 97.5 | 1.2 | 3320 | 99.2 | 0.0 | 51299 | 99.0 | 0.4 | 9566 | 95.0 | 0.1 |
| 775000 | 137324 | 97.5 | 1.2 | 3320 | 99.2 | 0.0 | 51299 | 99.0 | 0.4 | 9566 | 95.0 | 0.1 |
| 800000 | 137268 | 97.5 | 1.2 | 3320 | 99.2 | 0.0 | 51299 | 99.0 | 0.4 | 9566 | 95.0 | 0.1 |
| 825000 | 137240 | 97.5 | 1.2 | 3320 | 99.2 | 0.0 | 51299 | 99.0 | 0.4 | 9566 | 95.0 | 0.1 |
| 850000 | 137239 | 97.5 | 1.2 | 3320 | 99.2 | 0.0 | 51299 | 99.0 | 0.4 | 9566 | 95.0 | 0.1 |
| 875000 | 137239 | 97.5 | 1.2 | 3320 | 99.2 | 0.0 | 51299 | 99.0 | 0.4 | 9566 | 95.0 | 0.1 |
| 900000 | 137239 | 97.5 | 1.2 | 3320 | 99.2 | 0.0 | 51299 | 99.0 | 0.4 | 9566 | 95.0 | 0.1 |
| 925000 | 137239 | 97.5 | 1.2 | 3320 | 99.2 | 0.0 | 51299 | 99.0 | 0.4 | 9566 | 95.0 | 0.1 |
| 950000 | 137239 | 97.5 | 1.2 | 3320 | 99.2 | 0.0 | 51299 | 99.0 | 0.4 | 9566 | 95.0 | 0.1 |
| 975000 | 137239 | 97.5 | 1.2 | 3320 | 99.2 | 0.0 | 51299 | 99.0 | 0.4 | 9566 | 95.0 | 0.1 |
| 1000000 | 137239 | 97.5 | 1.2 | 3320 | 99.2 | 0.0 | 51299 | 99.0 | 0.4 | 9566 | 95.0 | 0.1 |

- The results are further improved again (look at misses for a given number of total buffer pool pages).

6. Performing object placement with the default rule set and moving "expensive" objects:
   a. The default rule set `pattern_large` is used again (not the saved pattern from step “4” on page 144).
   b. No changes are made to rules.
   c. When the object placement is performed, the identified problem table spaces and index spaces (from the detail report) are assigned to buffer pool BP10.
7. Performing simulation with results from the object placement:
   a. Buffer pools BP2, BP3, and BP10 are chosen.
   b. A minimum and maximum page size for all buffer pools of 25 000 and 1 000 000 and a simulation interval of 25 000 is chosen.

   "Recommended sizing for separate Buffer Pools" of the simulation result shows:

Table 19. Recommended sizing for separate buffer pools with moved objects

| Total Pages | BP2 pages | BP3 pages | BP10 pages |
|---|---|---|---|
| 75000 | 25000 | 25000 | 25000 |
| 100000 | 50000 | 25000 | 25000 |
| 125000 | 50000 | 25000 | 50000 |
| 150000 | 50000 | 25000 | 75000 |
| 175000 | 50000 | 25000 | 100000 |
| 200000 | 50000 | 25000 | 125000 |
| 225000 | 50000 | 25000 | 150000 |
| 250000 | 50000 | 25000 | 175000 |
| 275000 | 75000 | 25000 | 175000 |
| 300000 | 75000 | 25000 | 200000 |
| 325000 | 75000 | 50000 | 200000 |
| 350000 | 100000 | 50000 | 200000 |
| 375000 | 100000 | 50000 | 225000 |
| 400000 | 100000 | 50000 | 250000 |
| 425000 | 100000 | 50000 | 275000 |
| 450000 | 100000 | 50000 | 300000 |
| 475000 | 100000 | 50000 | 325000 |
| 500000 | 125000 | 50000 | 325000 |
| 525000 | 150000 | 50000 | 325000 |
| 550000 | 175000 | 50000 | 325000 |
| 575000 | 175000 | 50000 | 350000 |
| 600000 | 200000 | 50000 | 350000 |
| 625000 | 175000 | 50000 | 400000 |
| 650000 | 175000 | 50000 | 425000 |
| 675000 | 175000 | 50000 | 450000 |
| 700000 | 175000 | 50000 | 475000 |
| 725000 | 200000 | 50000 | 475000 |
| 750000 | 225000 | 50000 | 475000 |
| 775000 | 250000 | 50000 | 475000 |
| 800000 | 275000 | 50000 | 475000 |
| 825000 | 300000 | 50000 | 475000 |
| 850000 | 325000 | 50000 | 475000 |
| 875000 | 325000 | 50000 | 500000 |
| 900000 | 350000 | 50000 | 500000 |

*Table 19. Recommended sizing for separate buffer pools with moved objects (continued)*

| Total Pages | BP2 pages | BP3 pages | BP10 pages |
|---|---|---|---|
| 925000 | 375000 | 50000 | 500000 |
| 950000 | 400000 | 50000 | 500000 |
| 975000 | 425000 | 50000 | 500000 |
| 1000000 | 450000 | 50000 | 500000 |

- The memory is mainly shared between BP2 and BP10 (BP10 gets somewhat more).

"Simulated behavior of each separate Buffer Pool" of the simulation result shows:

*Table 20. Results of simulated behavior of separate buffer pools with moved objects*

| | **Buffer Pool BP2** | | | **Buffer Pool BP3** | | | **Buffer Pool BP10** | | |
|---|---|---|---|---|---|---|---|---|---|
| **Buffer Pool Pages** | **Misses** | **Application Hit Ratio** | **Global Miss Ratio** | **Misses** | **Application Hit Ratio** | **Global Miss Ratio** | **Misses** | **Application Hit Ratio** | **Global Miss Ratio** |
| 25000 | 1977916 | 77.8 | 17.1 | 22826 | 96.3 | 0.2 | 384203 | 80.9 | 3.3 |
| 50000 | 157728 | 98.2 | 1.4 | 13192 | 97.9 | 0.1 | 277984 | 86.2 | 2.4 |
| 75000 | 146042 | 98.4 | 1.3 | 12939 | 97.9 | 0.1 | 255630 | 87.3 | 2.2 |
| 100000 | 137354 | 98.5 | 1.2 | 12904 | 97.9 | 0.1 | 231980 | 88.5 | 2.0 |
| 125000 | 132345 | 98.5 | 1.1 | 12886 | 97.9 | 0.1 | 208635 | 89.6 | 1.8 |
| 150000 | 127540 | 98.6 | 1.1 | 12886 | 97.9 | 0.1 | 182152 | 91.0 | 1.6 |
| 175000 | 122628 | 98.6 | 1.1 | 12886 | 97.9 | 0.1 | 148533 | 92.6 | 1.3 |
| 200000 | 120617 | 98.6 | 1.0 | 12886 | 97.9 | 0.1 | 137183 | 93.2 | 1.2 |
| 225000 | 119323 | 98.7 | 1.0 | 12886 | 97.9 | 0.1 | 129862 | 93.6 | 1.1 |
| 250000 | 117911 | 98.7 | 1.0 | 12886 | 97.9 | 0.1 | 123232 | 93.9 | 1.1 |
| 275000 | 116349 | 98.7 | 1.0 | 12886 | 97.9 | 0.1 | 117433 | 94.2 | 1.0 |
| 300000 | 115100 | 98.7 | 1.0 | 12886 | 97.9 | 0.1 | 108989 | 94.6 | 0.9 |
| 325000 | 113892 | 98.7 | 1.0 | 12886 | 97.9 | 0.1 | 102452 | 94.9 | 0.9 |
| 350000 | 112813 | 98.7 | 1.0 | 12886 | 97.9 | 0.1 | 100332 | 95.0 | 0.9 |
| 375000 | 112392 | 98.7 | 1.0 | 12886 | 97.9 | 0.1 | 98409 | 95.1 | 0.8 |
| 400000 | 111509 | 98.7 | 1.0 | 12886 | 97.9 | 0.1 | 95495 | 95.3 | 0.8 |
| 425000 | 111000 | 98.8 | 1.0 | 12886 | 97.9 | 0.1 | 92747 | 95.4 | 0.8 |
| 450000 | 110719 | 98.8 | 1.0 | 12886 | 97.9 | 0.1 | 90016 | 95.5 | 0.8 |
| 475000 | 110378 | 98.8 | 1.0 | 12886 | 97.9 | 0.1 | 87894 | 95.6 | 0.8 |
| 500000 | 110040 | 98.8 | 0.9 | 12886 | 97.9 | 0.1 | 86791 | 95.7 | 0.7 |
| 525000 | 109411 | 98.8 | 0.9 | 12886 | 97.9 | 0.1 | 86778 | 95.7 | 0.7 |
| 550000 | 107470 | 98.8 | 0.9 | 12886 | 97.9 | 0.1 | 86778 | 95.7 | 0.7 |
| 575000 | 105709 | 98.8 | 0.9 | 12886 | 97.9 | 0.1 | 86778 | 95.7 | 0.7 |
| 600000 | 104477 | 98.8 | 0.9 | 12886 | 97.9 | 0.1 | 86778 | 95.7 | 0.7 |
| 625000 | 103580 | 98.8 | 0.9 | 12886 | 97.9 | 0.1 | 86778 | 95.7 | 0.7 |
| 650000 | 102072 | 98.9 | 0.9 | 12886 | 97.9 | 0.1 | 86778 | 95.7 | 0.7 |
| 675000 | 101881 | 98.9 | 0.9 | 12886 | 97.9 | 0.1 | 86778 | 95.7 | 0.7 |
| 700000 | 101765 | 98.9 | 0.9 | 12886 | 97.9 | 0.1 | 86778 | 95.7 | 0.7 |
| 725000 | 101760 | 98.9 | 0.9 | 12886 | 97.9 | 0.1 | 86778 | 95.7 | 0.7 |
| 750000 | 101760 | 98.9 | 0.9 | 12886 | 97.9 | 0.1 | 86778 | 95.7 | 0.7 |
| 775000 | 101760 | 98.9 | 0.9 | 12886 | 97.9 | 0.1 | 86778 | 95.7 | 0.7 |

*Table 20. Results of simulated behavior of separate buffer pools with moved objects (continued)*

| | Buffer Pool BP2 | | | Buffer Pool BP3 | | | Buffer Pool BP10 | | |
|---|---|---|---|---|---|---|---|---|---|
| **Buffer Pool Pages** | **Misses** | **Application Hit Ratio** | **Global Miss Ratio** | **Misses** | **Application Hit Ratio** | **Global Miss Ratio** | **Misses** | **Application Hit Ratio** | **Global Miss Ratio** |
| 800000 | 101760 | 98.9 | 0.9 | 12886 | 97.9 | 0.1 | 86778 | 95.7 | 0.7 |
| 825000 | 101760 | 98.9 | 0.9 | 12886 | 97.9 | 0.1 | 86778 | 95.7 | 0.7 |
| 850000 | 101760 | 98.9 | 0.9 | 12886 | 97.9 | 0.1 | 86778 | 95.7 | 0.7 |
| 875000 | 101760 | 98.9 | 0.9 | 12886 | 97.9 | 0.1 | 86778 | 95.7 | 0.7 |
| 900000 | 101760 | 98.9 | 0.9 | 12886 | 97.9 | 0.1 | 86778 | 95.7 | 0.7 |
| 925000 | 101760 | 98.9 | 0.9 | 12886 | 97.9 | 0.1 | 86778 | 95.7 | 0.7 |
| 950000 | 101760 | 98.9 | 0.9 | 12886 | 97.9 | 0.1 | 86778 | 95.7 | 0.7 |
| 975000 | 101760 | 98.9 | 0.9 | 12886 | 97.9 | 0.1 | 86778 | 95.7 | 0.7 |
| 1000000 | 101760 | 98.9 | 0.9 | 12886 | 97.9 | 0.1 | 86778 | 95.7 | 0.7 |

- There is an improvement at 600 000 pages, but it is worse at 1 000 000 pages.

Further object placements and simulations could be performed for a combination of the saved rule set and putting problem table spaces into a separate buffer pool.

# Chapter 11. Downloading files from the host to the client

This section describes how to download the input data for these functions from the host to the client.

## About this task

The following Buffer Pool Analyzer functions are performed on a Windows-based client and require input data from the host.

- Chapter 6, "Viewing performance data on the client," on page 89
- Chapter 7, "Optimizing object placements and initial buffer pool sizes," on page 97
- Chapter 8, "Simulating buffer pool behavior," on page 111
- Chapter 9, "Analyzing long-term buffer pool performance," on page 121

You can use any file transfer method or product, for example, the File Transfer Protocol (FTP) or IBM Personal Communications. You must have at least one of these products installed on your client. The following procedure describes the basic steps to download files. For more information, see the product documentation.

If you have the choice, you should use FTP, because it is faster.

If you need to download large trace data files (for simulations), consider using compressed trace data sets to reduce download times. See Chapter 3, "Collecting data," on page 27 for instructions how to create compressed data sets. The simulation function automatically extracts compressed trace data files.

1. On the client, create a folder where you want to store the files to be downloaded. For example, in the Windows Explorer file list:
   a. Click the **C** drive.
   b. Click **File —► New —► Folder**. A new folder icon is displayed.
   c. Rename the folder icon to a meaningful name, for example `bp_data`.

      It is recommended to keep buffer pool data files (used for viewing performance data and optimizing the object placements) and trace data files (used for simulations) in a single folder. This helps to keep together bpd files and trace data files that were created on the host from the same buffer pool trace data. You can distinguish them by their file name extensions (bpd or `trace`).

   Note that Buffer Pool Analyzer does not provide means to delete downloaded files. They remain on the hard disk drive, and take up space, until you delete them.
2. If you want to use FTP to download files:

   If your File Transfer Protocol (FTP) program provides the RDW and NORDW command options, ensure that the NORDW command option is active.

   a. Open a Command Prompt window on your client and start an FTP dialog with one of the following commands:
      - `ftp` *`<hostname>`*, whereby *<hostname>* denotes the name of your host
      - `ftp` *`<IP address>`*, whereby *<IP address>* denotes the IP address of your MVS host
   b. Enter your MVS user ID and password and wait until the current client folder is displayed.
   c. Enter `binary` to set the transfer type to binary.
   d. Enter `lcd` *`<client_directory>`*, whereby *<client_directory>* denotes the local directory where you want to store the files (the folder that you created in step 1). If you do not specify a directory, the current directory on the client is used.

e. If you do not remember the data set name you want to download, enter `dir` or `ls` to get a list of your data sets.

f. Enter `get` *<file_name> <new_name>*, whereby *<file_name>* denotes the fully qualified host data set name in quotes, and *<new_name>* denotes the client file name without quotes.

   This starts the download.

g. Enter `quit` to leave the FTP program, or download more files as required.

Example: The following figure shows an example of an FTP session.

```
C:\bpa>ftp boepm01
Connected to boepm01.boeblingen.de.ibm.com
220-FTPD1 IBM FTP CS V1R2 at BOEPM01.boeblingen.de.ibm.com, 13:53:23 on 2003-05-
20.
220 Connection will close if idle for more than 60 minutes.
User (boepm01.boeblingen.de.ibm.com:(none)): wtn
331 Send password please.
Password:
230 WTN is logged on.  Working directory is "WTN.".
ftp> bin
200 Representation type is Image
ftp> lcd c:\bpa
Local directory now C:\bpa.
ftp> get 'WTN.TEST.TRACE' test.trace
200 Port request OK.
125 Sending data set WTN.TEST.TRACE
250 Transfer completed successfully.
ftp: 1768197 bytes received in 1.34Seconds 1317.58Kbytes/sec.
ftp> get 'WTN.TEST.BPD' test.bpd
200 Port request OK.
125 Sending data set WTN.TEST.BPD
250 Transfer completed successfully.
ftp: 149487 bytes received in 0.43Seconds 347.64Kbytes/sec.
ftp> quit
221 Quit command received. Goodbye.

C:\bpa>dir test*
 Volume in drive C is C_DRIVE
 Volume Serial Number is 6C10-18AA

 Directory of C:\bpa

20.05.2003  16:06              149 487 test.bpd
20.05.2003  16:06            1 768 197 test.trace
```

3. If you want to use IBM Personal Communications to download files:

   a. Log on to your Multiple Virtual Storage (MVS) session on the z/OS or OS/390 system. Ensure that your MVS terminal is in READY mode. Note that keyboard entries in lowercase are converted to uppercase on the host.

   b. Start IBM Personal Communications on your client and click Receive.

   c. In the **Host file** field, type the name of the data set that contains the data to be downloaded.

      Example: `'sample.bpd'`, a buffer pool data file. Do not forget the quotes.

      Example: `'bpasim.trace'`, a trace data file. Do not forget the quotes.

   d. In the **PC** field, type the destination folder and file name.

      Example: `c:\bp_data\sample.bpd` for a buffer pool data file

      Example: `c:\bp_data\bpasim.trace` for a trace file

   e. In the **Transfer type** field, select **BINARY**.

      This step starts the download.

# Chapter 12. Concatenating trace data for activity reports and bpd files

This section outlines some possibilities of using trace data from SMF or GTF data sets and concatenating multiple data sets as input for activity reports and bpd files.

You should be familiar with batch jobs that create reports and bpd files, especially the use of the **INPUTDD** statement in these batch jobs, as described in "Specifying a JCL command stream" on page 38.

## Concatenating trace data from SMF and GTF data sets

The Collect Report Data (CRD) function of Buffer Pool Analyzer or an equivalent batch job are the recommended methods of making Db2 trace data available as input for activity reports and bpd files.

Other methods, like the Db2 trace facility, also provide usable Db2 trace data in standard SMF and GTF data sets.

If you intend to exploit SMF or GTF data, ensure that it contains the IFCIDs required by Buffer Pool Analyzer, as described in "Determining what to collect" on page 14. Otherwise some of the functionality that Buffer Pool Analyzer normally provides will be missing.

If you are accustomed to Db2 trace classes instead of individual IFCIDs, ensure that the appropriate trace classes, covering the required IFCIDs, are included in the data. Note that IFCID 198, which is required by Buffer Pool Analyzer for data of data type `Detail`, does not belong to any specific trace class, and corresponding data might therefore be missing in SMF or GTF data. If `detail` data is required in your SMF or GTF data, and if it is to be collected by means other than the Collect Report Data (CRD) function of Buffer Pool Analyzer, you can collect this data explicitly by means of the START TRACE command, as follows:

```
-START TRACE(PERFM) CLASS(30) IFCID(198) DEST(SMF)
```

This command uses the generic multi-purpose trace class 30, which has no predefined IFCIDs assigned to it. The required IFCID 198 is explicitly specified.

Even so collecting performance data through GTF or SMF is attractive especially for long-term collection and larger volumes, be aware that GTF, when the destination data set has filled up with trace data, proceeds recording data by overwriting the oldest data in the data set. Allocate a data set large enough to hold the expected amount of trace data for the collection period.

Note that SMF and GTF data do not provide data in the `short` record format, as it is required for the simulation function. Therefore, for simulations, you must collect trace data by means of the functions provided by Buffer Pool Analyzer. The other Buffer Pool Analyzer functions accept the `standard` record format, even so `short` is recommended in "Determining what to collect" on page 14 for several other reasons.

Details about trace data in SMF and GTF data sets are described in the *Db2 11 Administration Guide*.

Assuming that the SMF or GTF data contains the necessary IFCIDs, you can use it alternatively or together with trace data being collected through Buffer Pool Analyzer. The JCL command stream in Chapter 4, "Creating activity reports and bpd files," on page 37 describes how input data sets are specified. You can concatenate multiple data sets with the INPUTDD statement to create one logical data set and continue processing the trace data as usual. The rules for concatenating data sets apply.

If you use DFSORT, see the *z/OS DFSORT Application Programming Guide* for rules that apply to the concatenation of data sets.

If the trace data from other sources misses catalog information, you can use the `Catalog only` option of the CRD function (or the corresponding parameter in a batch job) to collect only catalog information (see "Configuring a collect task" on page 31, if required). Catalog data is used to enhance trace data that is collected through SMF or GTF. When you concatenate both data sets with the INPUTDD statement, the database identifiers (DATABASE) and object identifiers (OBJECT) from the SMF and GTF data are mapped to the actual database and object names.

Restriction: The possibilities outlined so far are provided for experienced and interested users wanting to use trace data from other sources. Their use requires detailed knowledge about trace data, involved IFCIDs, and tools. Because of endless variations and possible drawbacks, these options are formally not recommended and not supported in the current version of Buffer Pool Analyzer.

## Effects from concatenated input data sets

If you concatenate multiple data sets containing buffer pool performance data (no matter whether they were created by means of the Collect Report Data (CRD) function or by any other means), and if you use this data as input to create activity reports or bpd files, you should be aware of some side effects concerning the results.

- The input data sets you are concatenating might contain performance data from overlapping or segregative collection time frames.
- Summary information in activity reports is based on data that is collected at so-called *statistics intervals* (opposed to actual counts of events for detail reports). This means, a counter in a summary report is computed as the difference between the first and latest value covered during the collection of data.

Knowing this, it becomes obvious that summary information created from concatenated data can become imprecise or even useless because of various events.

- A Db2 system might be restarted between two collection time frames, which resets the counters being sampled at statistics intervals.
- The performance data in different data sets might be collected with different statistics interval settings or with overlapping time frames, which makes it impossible to compute valid results.
- Objects in buffer pools might be created or dropped during different collection time frames, or new objects might be assigned to buffer pools with previously used identifiers, which makes object related counter values invalid.
- Catalog information, correlation data, and aliases might be different and might not match the concatenated data, which also causes invalid results.

In summary, concatenating performance data to create summary activity reports is not recommended. This method, if used at all, is more suitable to detail activity reports, which are based on actual event counts in the concatenated data sets.

"Preliminary remarks about the accuracy of summary and detail reports" on page 48 discloses further details about statistics-based and event-based data collection and the use of this data for activity reports.

# Chapter 13. Loading a bpd file into a Db2 table

This section briefly describes how to store trace data from bpd files into Db2 tables.

Db2 tables can be used by administrators to extract performance-related data with self-written SQL queries. The further utilization of this data is outside the scope of this information. See the *Report Reference* for more information about possible uses of performance data.

Before you can store trace data into a table, you must create a table with an appropriate layout that can accept the data from a bpd file. This is done with the SQL CREATE TABLE statement. When you determine the table layout, you need to consider:

- Whether summary or detailed data was collected. Each data type requires a different table layout.
- Whether the bpd file was created with the `Summary` or `Detail` option of the **`BPACTIVITY FILE`** command. If detailed data was collected, but the `Summary` option was used, the bpd file contains only summary data. Consequently, the table layout must be appropriate for summary data.

When you have created the appropriate table, you can use the Db2 LOAD utility to load data from a bpd file into the table. LOAD requires the specification of the data elements that are to be stored into the table. For more information, see the *IBM Db2 11 for z/OS: Utility Guide and Reference*.

Buffer Pool Analyzer provides several samples of CREATE and LOAD statements that store data into Db2 tables in the following formats:

- Summary data, from IFCID 002
- Summary data, from IFCID 230
- Summary data, from IFCID 251
- Summary data, from IFCID 254
- Detail data, from IFCIDs 6, 7, 8, 9, 10, and 198
- Detail data, but aggregated by object
- Detail data, but aggregated by buffer pool
- Detail data, but aggregated by system

The sample statements are in members of the partitioned data set `*prefix*.TKO2SAMP`. The data set also contains members that contain descriptions of the individual Db2 table columns used with CREATE and LOAD. The *italic* characters and numbers in the following table show the naming associations for easier identification.

*Table 21. Member names holding the sample statements and associated column descriptions*

| **Samples for** | **CREATE statements are in member** | **For CREATE: column descriptions are in member** | **LOAD statements are in member** | **For LOAD: column descriptions are in member** |
|---|---|---|---|---|
| Summary data, from IFCID 00*2* | BPOQFC*2*F | BPOQFB*2*F | BPOQFL*2*F | BPOQFD*2*F |
| Summary data, from IFCID 23*0* | BPOQFC*0*F | BPOQFB*0*F | BPOQFL*0*F | BPOQFD*0*F |
| Summary data, from IFCID 25*1* | BPOQFC*1*F | BPOQFB*1*F | BPOQFL*1*F | BPOQFD*1*F |
| Summary data, from IFCID 25*4* | BPOQFC*4*F | BPOQFB*4*F | BPOQFL*4*F | BPOQFD*4*F |
| *D*etail data | BPOQFC*D*F | BPOQFB*D*F | BPOQFL*D*F | BPOQFD*D*F |
| Detail data, aggregated by *o*bject | BPOQFC*O*F | BPOQFB*O*F | BPOQFL*O*F | BPOQFD*O*F |
| Detail data, aggregated by *b*uffer pool | BPOQFC*B*F | BPOQFB*B*F | BPOQFL*B*F | BPOQFD*B*F |

*Table 21. Member names holding the sample statements and associated column descriptions (continued)*

| **Samples for** | **CREATE statements are in member** | **For CREATE: column descriptions are in member** | **LOAD statements are in member** | **For LOAD: column descriptions are in member** |
|---|---|---|---|---|
| Detail data, aggregated by *s*ystem | BPOQFC*S*F | BPOQFB*S*F | BPOQFL*S*F | BPOQFD*S*F |
| Names of i*n*dex space objects | BPOQFC*N*F | BPOQFB*N*F | BPOQFL*N*F | BPOQFD*N*F |

Note that the sample CREATE and LOAD statements work independently of whether the bpd file actually contains relevant data. For example, if you create a Db2 table for storing group buffer pool related data, but the bpd file does not contain such data because the data was collected from a Db2 system that is not a data sharing group member, the table remains empty after the LOAD statement is executed. Generally, if you encounter difficulties with missing data, verify the parameters that were used for the data collect and the bpd file creation tasks. See especially the job summary logs (JOBSUMDD) and the DPMLOG execution logs. The latter might contain information about record types that were not available for processing.

Example: This example (from member BPOQFCDF) shows a partial CREATE TABLE statement that creates a table for detail data:

```
--**Start of Specifications********************************************
--*                                                                   *
--* MODULE-NAME      = BPOQFCDF                                       *
--* DESCRIPTIVE-NAME = SQL for creating table for detail activity     *
--*                    data from IFCIDs 6, 7, 8, 9, 10, 198           *
--* COPYRIGHT = IBM Db2 Buffer Pool Analyzer for z/OS  V5R4M0         *
--*             Licensed Material - Property of IBM                   *
--*             5655-W35 (C) Copyright IBM Corp. 2001, 2016           *
--* STATUS    = Version 5.4.0                                         *
--*                                                                   *
--* FUNCTION = Sample SQL for creating table for detail activity      *
--*                    data from IFCIDs 6, 7, 8, 9, 10, 198           *
--**End of Specifications**********************************************
 CREATE TABLE DB2PE_BPA_DETAIL
  (DB2PM_RELEASE        SMALLINT          NOT NULL WITH DEFAULT,
   DB2_RELEASE          CHAR(2)           NOT NULL WITH DEFAULT,
   LOCAL_LOCATION       CHAR(16)          NOT NULL WITH DEFAULT,
   GROUP_NAME           CHAR(8)           NOT NULL WITH DEFAULT,
   SUBSYSTEM_ID         CHAR(4)           NOT NULL WITH DEFAULT,
   MEMBER_NAME          CHAR(8)           NOT NULL WITH DEFAULT,
   INTERVAL_TSTAMP      CHAR(26),
   INTERVAL_ELAPSED     DECIMAL(15,6),
   BEGIN_REC_TSTAMP     CHAR(26),
   END_REC_TSTAMP       CHAR(26),
   REQ_LOCATION         CHAR(16)          NOT NULL WITH DEFAULT,
   PRIMAUTH             CHAR(8)           NOT NULL WITH DEFAULT,
   .
   .
   .
   GETPAGE_MISS_A       INTEGER           NOT NULL WITH DEFAULT,
   GETPAGE_NOREAD       INTEGER           NOT NULL WITH DEFAULT,
   READ_REQUEST         INTEGER           NOT NULL WITH DEFAULT,
   READ_REQ_SYNC        INTEGER           NOT NULL WITH DEFAULT,
   READ_REQ_SEQ         INTEGER           NOT NULL WITH DEFAULT,
   READ_REQ_LIST        INTEGER           NOT NULL WITH DEFAULT,
   READ_REQ_DYN         INTEGER           NOT NULL WITH DEFAULT,
   READ_DEL_SYNC        DECIMAL(7,1),
   READ_DEL_SEQ         DECIMAL(7,1),
   READ_DEL_LIST        DECIMAL(7,1),
   READ_DEL_DYN         DECIMAL(7,1),
   READ_PAGE            INTEGER           NOT NULL WITH DEFAULT,
   READ_PAGE_SYNC       INTEGER           NOT NULL WITH DEFAULT,
   PRIMAUTH             CHAR(8)           NOT NULL WITH DEFAULT,
   .
   .
   .
   WRITE_PAGE           INTEGER           NOT NULL WITH DEFAULT,
   WRITE_PAGE_SYNC      INTEGER           NOT NULL WITH DEFAULT,
   WRITE_PAGE_ASYNC     INTEGER           NOT NULL WITH DEFAULT)
   PARTITION_NUMBER     INTEGER           NOT NULL WITH DEFAULT)    IN GRPBP
```

The following LOAD statement (from member BPOQFLDF) loads data into the previously created table:

```
LOAD DATA INDDN(SYSREC)
  REPLACE LOG NO
  INTO TABLE DB2PE_BPA_DETAIL
  WHEN (13:13) = 'D'
 (DB2PM_RELEASE         POSITION(7) SMALLINT,
  DB2_RELEASE           POSITION(14) CHAR(2),
  LOCAL_LOCATION        POSITION(17) CHAR(16),
  GROUP_NAME            POSITION(33) CHAR(8),
  SUBSYSTEM_ID          POSITION(41) CHAR(4),
  MEMBER_NAME           POSITION(45) CHAR(8),
  PRIMAUTH             CHAR(8)           NOT NULL WITH DEFAULT,
  .
  .
  .
  WSNAME                POSITION(291) CHAR(18),
  BUFFERPOOL_ID         POSITION(309) CHAR(8),
  PAGESET_QUAL          POSITION(317) CHAR(27),
  PAGESET_TYPE          POSITION(344) CHAR(1),
  SYSTEM_HIT_RATIO      POSITION(345) DECIMAL
                           NULLIF SYSTEM_HIT_RATIO=X'FFFFFFFF',
  APPL_HIT_RATIO        POSITION(349) DECIMAL
                           NULLIF APPL_HIT_RATIO=X'FFFFFFFF',
  GETPAGE_TOT           POSITION(353) INTEGER,
  GETPAGE_SEQUENT       POSITION(357) INTEGER,
  PRIMAUTH             CHAR(8)           NOT NULL WITH DEFAULT,
  .
  .
  .
  READ_PAGE_SEQ         POSITION(429) INTEGER,
  READ_PAGE_LIST        POSITION(433) ICNTEGER,
  READ_PAGE_DYN         POSITION(437) INTEGER,
  UPD_WRT_PAGE          POSITION(441) DECIMAL
                           NULLIF UPD_WRT_PAGE=X'FFFFFFFF',
  PAGE_WRITE_REQ        POSITION(445) DECIMAL
                           NULLIF PAGE_WRITE_REQ=X'FFFFFFFF',
  BUFFER_UPDATE         POSITION(449) INTEGER,
  WRITE_REQ             POSITION(453) INTEGER,
  WRITE_REQ_SYNC        POSITION(457) INTEGER,
  WRITE_REQ_ASYNC       POSITION(461) INTEGER,
  WRITE_REQ_DEL_SYNC    POSITION(465) DECIMAL
                           NULLIF WRITE_REQ_DEL_SYNC=X'FFFFFFFF',
  WRITE_REQ_DEL_ASYN    POSITION(469) DECIMAL
                           NULLIF WRITE_REQ_DEL_ASYN=X'FFFFFFFF',
  WRITE_PAGE            POSITION(473) INTEGER,
  WRITE_PAGE_SYNC       POSITION(477) INTEGER,
  PARTITION_NUMBER      POSITION(485) INTEGER)
```

# Chapter 14. The TRSMAIN terse utility

This section provides information about the TRSMAIN terse utility that is used to compress collected trace data, and it shows batch job examples using the utility.

The utility is mainly provided to reduce the sizes of output data sets with raw trace data, as used for the simulation function of Buffer Pool Analyzer, and to reduce the download times of these data sets. Compressed trace data files are automatically uncompressed when opened by the simulation function. If you use the compression facility only for this purpose, you can ignore the following information.

Nevertheless, when you collect performance data, you are not limited to the compression of raw trace data for simulations. You can compress all data that is collected by means of the Collect Report Data (CRD) function or an equivalent batch job. But no Buffer Pool Analyzer function, except the simulation function, processes compressed input data. Therefore, if you want to compress collected data on the host for whatever reason, you need to uncompress data before it can be used as input to other Buffer Pool Analyzer functions. Note that the following information pertains to TRSMAIN on the host; the uncompress component of the simulation function on the client remains transparent and is not accessible.

**Related tasks**

“Collecting data” on page 27
This topic describes how to collect the performance data that is used by Buffer Pool Analyzer. It describes two methods to collect buffer pool trace data. The first method uses ISPF and the Collect Report Data (CRD) function to configure and control a collect task, the second method uses a batch job that contains equivalent specifications for a collect task.

## About the TRSMAIN terse utility

The terse utility is prerequisite on z/OS platforms, if you want to compress collected data.

It might already be installed because it is often used during the installation of operating system fix packs. The utility is freeware and can be downloaded from this IBM Support website. The terse utility is based on US patent number US04814746. TRSMAIN provides two compression methods, PACK and SPACK. Only SPACK is used with Buffer Pool Analyzer because it provides the highest compression ratio.

IBM supports only version 2 of TRSMAIN, shipped in 1993, and later versions. Further, support is limited to the SPACK option.

When used with Buffer Pool Analyzer trace data files, the compression ratio is approximately 75 percent.

## Compressing trace data using the batch JCL

The following batch job shows an example of how TRSMAIN is used to compress trace data residing in data set `NKA.COLLECT.TRACE` and to write it to data set `NKA.COLLECT.TRACE.TERSE`.

You can use this batch job together with the batch job described in “Collecting data by using the batch JCL” on page 35, if you do not want to use ISPF.

Example:

```
//*********************************************************************
//* DESCRIPTION:  JCL for compressing data from Collect Report Data    *
//*********************************************************************
//*
//NKA$D711 JOB (DE03704),'NKA',CLASS=A,MSGCLASS=X,
//         REGION=0M,MSGLEVEL=(1,1),PRTY=5,NOTIFY=NKA,TIME=8
//*
//BPACRD   EXEC PGM=BPOMAB00
//*
//STEPLIB  DD DSN=SYS1.DSN.V910.SDSNLOAD,DISP=SHR
//         DD DSN=SYS1.FPE.V540.RKANMOD,DISP=SHR
//*
//SYSPRINT DD SYSOUT=*
//SYSOUT   DD SYSOUT=*
```

```
//SYSUDUMP DD SYSOUT=*
//*
//SYSIN    DD DSN=NKA.BPACRD.CNTL(CRD#IN),DISP=SHR
//DPMLOG   DD SYSOUT=*
//JOBSUMDD DD SYSOUT=*
//DPCOLLDD DD DSN=NKA.COLLECT.TRACE,
//            DISP=(NEW,CATLG),
//            DCB=(RECFM=VBS,BLKSIZE=9076,LRECL=32756),
//            SPACE=(TRK,(500,100)),UNIT=3390
//RC1OK    IF (BPACRD.RC LT 4) THEN
//TERSE    EXEC PGM=TRSMAIN,PARM=SPACK
//SYSPRINT DD  SYSOUT=*
//SYSOUT   DD  SYSOUT=*
//INFILE   DD  DISP=SHR,DSN=NKA.COLLECT.TRACE
//OUTFILE  DD  DSN=NKA.COLLECT.TRACE.TERSE,
//             DISP=(NEW,CATLG,DELETE),
//             DCB=(RECFM=VBS,BLKSIZE=9076,LRECL=32756),
//             SPACE=(TRK,(500,100),RLSE),UNIT=3390
//ENDRC1OK    ENDIF
```

## Uncompressing trace data using the batch JCL

The following batch job shows an example of how TRSMAIN is used to uncompress trace data residing in data set `NKA.COLLECT.TRACE.TERSE` and to write it to data set `NKA.COLLECT.TRACE`.

This batch job is shown only for the sake of completeness, if you want to use TRSMAIN for your own purposes. Compressed trace data for simulations is automatically uncompressed by the simulation function.

Example:

```
//**********************************************************************
//* DESCRIPTION:  JCL for uncompressing trace data                     *
//**********************************************************************
//NKATRSE  JOB (DE03704,),'NKA',CLASS=A,MSGCLASS=X,
//              MSGLEVEL=(1,1),NOTIFY=NKA,REGION=5M
//TERSE    EXEC PGM=TRSMAIN,PARM=UNPACK
//SYSPRINT DD   SYSOUT=*
//SYSOUT   DD   SYSOUT=*
//INFILE   DD   DISP=SHR,DSN=NKA.COLLECT.TRACE.TERSE
//OUTFILE  DD   DSN=NKA.COLLECT.TRACE,
//              DISP=(NEW,CATLG,DELETE),
//              DCB=(RECFM=VB,BLKSIZE=9076,LRECL=9072),
//              SPACE=(TRK,(500,100),RLSE),UNIT=3390
```

# Chapter 15. Troubleshooting for Buffer Pool Analyzer

This section categorizes possible Buffer Pool Analyzer problems and describes steps to solve them. The intention of this section is to provide a fast problem determination.

## Problems with a collect task

**Collect task terminates with message FPEM0802E and a reference to SQL code -805**
Contact your Db2 administrator. One or more packages might need to be rebound. This error can occur if the product or a product update was improperly installed.

**Collect task with compression shows TERSEMVS messages**
These messages indicate errors reported from the TRSMAIN utility. See "Interpreting trace status summary and trace messages" on page 33 about how to diagnose these errors.

## Problems with batch jobs

**Not enough records read or too many records lost**
If the job summary log (JOBSUMDD) indicates these problems, your batch job might not have sufficient priority. See the recommendation in "Collecting data by using the batch JCL" on page 35 for details.

**Abend B37 (out of space) when creating reports or bpd files**
Buffer Pool Analyzer cannot store temporary data in a temporary work data set because of virtual storage constraints. Use the BPWORK DD statement as described in "Specifying a JCL command stream" on page 38.

**FPEU0020E or FPEC4085U insufficient virtual storage**
Check the REGION statement in the affected batch job.

## Missing information in reports or bpd files

**Sections are missing in reports**
Ensure that you have collected the appropriate data type (`Summary` or `Detail`). See "Determining what to collect" on page 14, "Configuring a collect task" on page 31, and "Collecting data by using the batch JCL" on page 35.

For reports, ensure that you have specified the appropriate **LEVEL** option of the **BPACTIVITY REPORT** command. See "Specifying reports and bpd files with BPACTIVITY" on page 40.

Check the job summary log JOBSUMMDD for more details. See "Specifying a JCL command stream" on page 38.

**Object placement function lacks information about unused objects**
If you miss certain objects in one of the windows, or if only active objects are listed (the Used column always shows `Yes`), or if the **Show only objects with activity** or **Assign objects not accessed during data collection** check boxes seem to have no effect, it is likely that you have used a bpd file that was created with the **BPACTIVITY FILE ACTIVEOBJECTS** command. The **ACTIVEOBJECTS** subcommand option excludes all information about inactive, respectively unused objects.

This behavior is not an error. If you want inactive objects to be considered during your object placement optimization, you need to use a bpd file that is created without the **ACTIVEOBJECTS** subcommand option.

## Problems with client functions

**`Out of memory` message**
: Your client might not have enough available physical memory (random access memory) to start or to perform a function. See the General remarks topics at the beginning of the corresponding sections for specific memory requirements.

**Unable to select a bpd or trace data file**
: The file name extension of the bpd or trace file should be bpd, respectively `trace`, as described in "File and data set naming conventions" on page 18. If you have used other file name extensions, select `All files (*.*)` from the **File of type** list in the Open dialog box to see all files and select the file with your custom extension.

**Unable to open a bpd or trace data file**
: Ensure that the file size is less that 2 GB. If required, modify the data collection parameters (described in "Configuring a collect task" on page 31).

**Message BPOK6000 - Internal error - is displayed**
: You tried to open a buffer pool data (bpd) file with one of the client-based functions, but the bpd file could not be opened or properly preprocessed. The file might be damaged, does not contain data in the required format (`Short` or `Standard`) or data type (`Summary` or `Detail`), or was not created by means of the **`BPACTIVITY FILE`** command. Create a new bpd file, or use another bpd file. See Table 2 on page 15 and "Specifying reports and bpd files with BPACTIVITY" on page 40, if required.

  If this error occurs when you use the File Transfer Protocol (FTP), also check whether your file transfer program provides the RDW and NORDW command options. If the default is RDW, it might cause a four-byte record descriptor record to be included in the data set being downloaded from the host to the client (which might then cause message BPOK6000). Specify the NORDW command option to avoid the creation of the descriptor record.

**Unintentional characters are displayed during data entry**
: Check and correct the regional settings of the Windows operating system.

**Simulation does not finish - progress indicator stops at approximately 99%**
: The number of simulated buffer pool sizes, determined by the minimum and maximum buffer pool size and the interval, should not exceed 40. See "Step 1: Setting simulation parameters" on page 113 for more details.

## Hard disk drive space management

Buffer Pool Analyzer needs to keep its input files (the buffer pool data files and trace data files) on the client's local hard disk drive to perform its functions. Further, Buffer Pool Analyzer keeps its results (from object placements, simulations, and long-term analyses) also on its local hard disk drive. Over time, you might have collected a multitude of input files and results that unnecessarily use up hard disk space.

Buffer Pool Analyzer does not perform any cleanup. All files and results remain on the hard disk until *you* delete them. Use the following guidelines to delete no longer required files and results:

**Input files**
: Buffer pool data files (`*.bpd`) and trace data files (`*.trace`) are stored in optional folders, which are determined by you when these files are downloaded.

  Delete these files by using the client's functions (Windows Explorer).

**Output files**
: Results from different Buffer Pool Analyzer functions are stored in folder `C:\Documents and Settings \<userid> \db2pev<version> \… \…` and various subfolders. Do not delete individual files from these folders; they are also used by other Db2 Performance Expert functions and as temporary work area. If required, use them only to determine how much hard disk space they use (Windows' Properties function).

  To delete results, use the Buffer Pool Analyzer main window. Refer to one or more of the following sections for a description:

- “Viewing the result of an optimization cycle” on page 107
- “Viewing the result of a simulation cycle” on page 116
- “Viewing the result of a long-term analysis” on page 128

This process ensures that all files belonging to a report or result are properly deleted.

# Product legal notices

## Programming interface information

This publication documents intended Programming Interfaces that allow the customer to write programs to obtain the services of OMEGAMON for Db2 Performance Expert.

This publication documents information that is NOT intended to be used as Programming Interfaces of OMEGAMON for Db2 Performance Expert.

This publication primarily documents intended Programming Interfaces that allow the customer to write programs to obtain the services of OMEGAMON for Db2 Performance Expert.

This publication also documents information that is NOT intended to be used as Programming Interfaces of OMEGAMON for Db2 Performance Expert. This information is identified where it occurs by an introductory statement to a topic or section.

This publication primarily documents information that is NOT intended to be used as Programming Interfaces of OMEGAMON for Db2 Performance Expert.

This publication also documents intended Programming Interfaces that allow the customer to write programs to obtain the services of OMEGAMON for Db2 Performance Expert. This information is identified where it occurs by an introductory statement to a topic or section.

## Trademarks

## Terms and conditions for product documentation

## Privacy policy considerations

IBM Software products, including software as a service solutions, ("Software Offerings") may use cookies or other technologies to collect product usage information, to help improve the end user experience, to tailor interactions with the end user or for other purposes. In many cases no personally identifiable information is collected by the Software Offerings. Some of our Software Offerings can help enable you to collect personally identifiable information. If this Software Offering uses cookies to collect personally identifiable information, specific information about this offering’s use of cookies is set forth below.

This Software Offering does not use cookies or other technologies to collect personally identifiable information.

If the configurations deployed for this Software Offering provide you as customer the ability to collect personally identifiable information from end users via cookies and other technologies, you should seek your own legal advice about any laws applicable to such data collection, including any requirements for notice and consent.

For more information about the use of various technologies, including cookies, for these purposes, see IBM’s Privacy Policy at http://www.ibm.com/privacy and the section titled "Cookies, Web Beacons, and Other Technologies" in IBM’s Online Privacy Statement at http://www.ibm.com/privacy/details. Also, see the "IBM Software Products and Software-as-a-Service Privacy Statement" at http://www.ibm.com/software/info/product-privacy.

# Index

Product Number: 5655-W37

SH12-7075