IBM i on Power - Performance FAQ

Best practices and solutions for common performance challenges

December, 2024

Table of Contents

Table of Figures

Preface

This document is intended to address most frequently asked questions concerning IBM i performance on Power and provide best practice guidelines for most seen performance issues.

The following is a list of key IBM® reference and documents. The References section contains additional references.

Key IBM references:
- End to End performance management on IBM i Redbook: Comprehensive guidance on monitoring and managing IBM i system performance
- Performance optimization and tuning techniques for IBM Power systems processors including IBM Power8 Redbook: Includes performance tuning for Power8 processor
- IBM documentation on performance: Dedicated sections for managing and optimizing system performance
- Managing system performance: A focused subsection on system performance management
- IBM developer resource: Covers a wide range of IBM i topics, including performance
- IBM support for performance tools: Offers insights and resources on performance tools for IBM i

Acknowledgments

We would like to thank the many people who made invaluable contributions to this document. Contributions included authoring, insights, ideas, reviews, critiques and reference documents

Our special thanks to key contributors from IBM Power Performance:
- Grace Bowser – IBM i Performance
- Dirk Michel – Power Performance
- Rick Peterson – IBM i Performance

Our special thanks to key contributors from IBM i Development Support:
- Colin Devilbiss – IBM i Development
- Scott Forstie – IBM i Development
- Chris Francois – IBM i Development
- Bruce Hansel – IBM i Development
- Chad Olstad – IBM i Development
- Lora Powell – IBM i Development
- Shauna Rollings – IBM i Development

Our special thanks to key contributors from IBM i Global Support Center:
- Kevin Chidester - IBM i Performance
- Brad Menges – IBM i Performance

Our special thanks to key contributors from IBM Technology Expert Labs:
- Eric Barsness – IBM i Performance
- Stacy Benfield – IBM i Performance
- Terry Ford – IBM i Security
- Kent Milligan - Db2 for i Performance

1 Introduction

1.1 Purpose
The purpose of this document is to provide a basic understanding of IBM® i on Power® performance concepts, workloads and benchmarks on Power, capacity planning, performance monitoring and analysis, frequently asked questions and guidelines addressing common performance issues.

This document is not intended to replace performance management documentation or performance white papers.

1.2 Overview
This document covers a variety of Power performance including:
- What Is Performance?
- Performance Benchmarks
- Sizing a System and Capacity Planning
- Performance Tools
- Performance Analysis and Tuning
- Frequently Asked Questions
- Reporting a Performance Problem

1.2 What is new in the latest version?
This version includes updates and additions in the following areas:
- Updated frequently asked questions.
- Revised links throughout the document.
- Update to Section 7.3.9: "Why doesn't the CPU utilization reported by the Hardware Management Console (HMC) match the CPU utilization reported by IBM i?"
- Update to section 7.12: "How do I tune access path recovery performance?"

1.3 Responsibilities
The IBM Systems Power Performance organization is responsible for editing and maintaining the IBM i on Power - performance FAQ document. Any contributions or suggestions for additions or edits should be forwarded to Stacy Benfield, stacylb@us.ibm.com.

2 What is performance

2.1 Introduction
The purpose of this chapter is to explain what exactly computer performance is.

2.2 Performance
Computer performance is largely determined by a combination of response time and throughput. Other aspects associated with computer performance are availability of computer systems and their power efficiency.

2.3 Response time
The response time of a computer system is the elapsed time between the end of a transaction inquiry or demand and the beginning of a response to that transaction. For interactive users, the response time is the time from when the user presses the <enter> button to see the result displayed. The response time often is seen as a critical aspect of performance because of its potential visibility to end users or customers.

Let us take the example of a computer system that is running a web server with an online store. The response time here is the elapsed time between clicking the submit button to place an order and the beginning to receive the order confirmation.

2.4 Throughput
The throughput of a computer system is a measure of the amount of work performed by a computer system over the period. Examples for throughput are megabytes per second read from a disk, database transactions per minute, megabytes transmitted per second through a network adapter.

Let us go back to the previous example with the online store. The computer system on which the online store is running might be one out of many computers that are all connected to the same database server. While response time of the database server is an important factor, its throughput may be more important because it processes many requests from the web servers in parallel.

2.5 Throughput versus response Time
Throughput and response time are related. As system resources (processors, memory, disk, and so on) are used by multiple, simultaneous transactions, there may be delays to individual transactions because other transactions are using those resources when they are demanded and the request for processing is temporarily delayed. System resources that are shared can cause transactions that rely on these resources to incur a response time increase due to this queuing.

However, Power processor-based systems are designed for high-performance sharing. As transaction load increases on the system, throughput increases. It follows that, as resource queuing increases, response time increases too. Therefore, high throughput may come at the cost of slower response times.

Let's assume you load a truck with 10,000 1TB disks, drive the truck 30 miles, and unload the disks within one hour. The throughput would be 2.8TB per second but at the cost of the response time, which would be an hour.

Now take a sports car instead of a truck. It's unlikely that 10,000 disks would fit into a small car, so let's assume we can load 100 1TB disks, drive the sports car 30 miles, and unload the disks within 20

minutes. The response time now is three times better compared to the truck, however, the throughput reduced to 0.083TB per second.

2.6 Components of response time
Wait accounting is the patented technology built into the IBM i operating system that tells you what a thread or task is doing when it appears that it is not doing anything. When a thread or task is not running, it is waiting. Wait accounting, a concept exclusive to IBM i, is a very powerful capability for detailed performance analysis. The following information is going to focus on waiting, why threads wait, and how you can use wait accounting to troubleshoot performance problems or to simply improve the performance of your applications.

A job is the basic mechanism through which work is done. Every job has at least one thread and may have multiple threads. Every thread is represented by a Licensed Internal Code (LIC) task, but tasks also exist without the IBM i thread-level structures. LIC tasks are generally not visible externally except through the IBM i performance or service tools. Wait accounting concepts apply to both threads, and tasks, thus, the terms thread and task are used when referring to a runnable piece of work.

A thread or task can exist in two primary states:
- Running: actively executing on the processor.
- Waiting: paused or idle, pending processor availability.

There are three key wait conditions:
- Ready to run, waiting for the processor: This is a special wait state, generally referred to as *CPU queuing*. It indicates that the thread or task is queued, waiting to run on the CPU. CPU queuing can occur for various reasons. For instance, if the partition is overloaded and cannot accommodate all tasks, work may be queued to wait for CPU access. This is similar to a highway with ramp meters; when congested, ramp meters display a red signal, causing cars to stop and wait before entering traffic. Logical partitioning and simultaneous multithreading can also result in CPU queuing.
- Idle waits: Idle waits are a normal and expected wait condition. They occur when the thread is waiting for external input, which may come from a user, the network, or another application. Until this input is received, there is no work for the thread to perform.
- Blocked waits: Blocked waits result from serialization mechanisms used to synchronize access to shared resources. These waits can be normal and expected, as in cases of serialized access to update a row in a table, perform disk input/output (I/O) operations, or conduct communications I/O. However, blocked waits can sometimes be abnormal, with unexpected block points where wait accounting can help analyze the wait conditions.

You can envision the lifetime of a thread or task in a graphical format that segments the time spent running or waiting. This graphical description is called the *run-wait time signature*. At a high level, this signature represents the cumulative time the job spent in each wait group during its runtime, with each colored segment indicating a specific wait group.

Traditionally, efforts to improve application performance focused on maximizing CPU efficiency. On IBM i, with wait accounting, we can analyze the time spent waiting and determine the factors contributing to that wait time. Reducing or eliminating certain wait elements can lead to overall performance improvement.

Nearly all wait conditions in the IBM i operating system are identified and enumerated - each unique wait point is assigned a numerical value. This is possible because IBM fully controls both the licensed internal code and the operating system. As of IBM i 6.1 and 7.1 releases, there are 268 unique wait conditions. Tracking over 250 unique wait conditions for every thread and task would require excessive storage, so a grouping approach is used. Each unique wait condition is assigned to one of 32 groups, or *buckets*. As threads or tasks enter and exit wait conditions, the task dispatcher maps the wait condition to the appropriate group.

Using the run-wait time signature with wait accounting, we can now identify the components that contribute to the time the thread or task spent waiting.

For example:
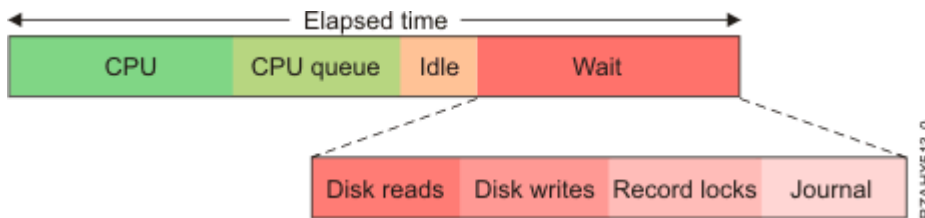
If the thread's wait time is due to reading and writing data to disk, locking records for serialized access, and journaling the data, these waits would appear broken out above. Understanding the types of waits involved allows you to consider some questions. For the previous example, possible questions include:
- Are disk reads causing page faults? If so, are my pool sizes appropriate?
- Which programs are causing the disk reads and writes? Is there unnecessary I/O that can be reduced or eliminated, or could the I/O be done asynchronously?
- Is my record locking strategy optimal, or am I locking records unnecessarily?
- What files are being journaled? Are all the journals required, and are they optimally configured?

2.6.1 Transaction characteristics

Transactions that involve substantial disk wait times are referred to as *I/O-limited transactions*. These transactions benefit most from an efficient disk I/O subsystem. High-speed disks, large disk caches, and high-speed I/O links can enhance the response time and throughput of these transactions. Additionally, consider solid-state drives (SSDs) or nonvolatile memory express (NVMe) as alternatives to HDDs for this type of transaction. For further tips on improving, I/O performance, refer to I/O questions section of this white paper.

I/O-limited Transaction:
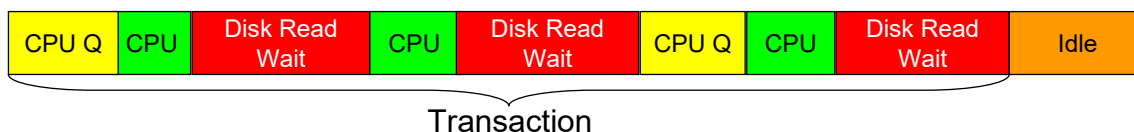
Transactions that consist of a considerable mix of waits and processor activity are referred to as *resource-limited transactions*. These transactions typically benefit most from improving application efficiency by reducing the impact of critical shared resources between applications or program threads. Collection Services and Job Watcher data can help identify critical resources and waiters. Use Performance Data Investigator or IBM iDoctor to analyze the data.

Resource-limited transactions:
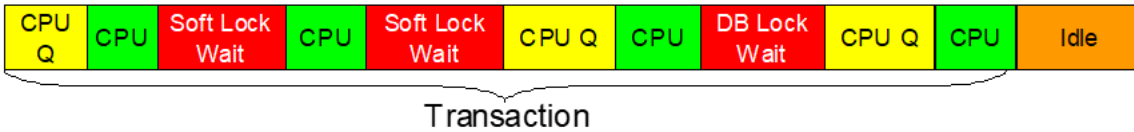
Transactions that consume significant processor cycles with minimal I/O or wait times are called *processor-bound transactions*. These transactions tend to benefit most from higher processor frequency and the efficiency of the processor pipeline stages. They may also benefit from improved processor and memory nodal affinity, as well as better partition placement (Example: Dynamic platform optimizer).

Processor-bound transaction:
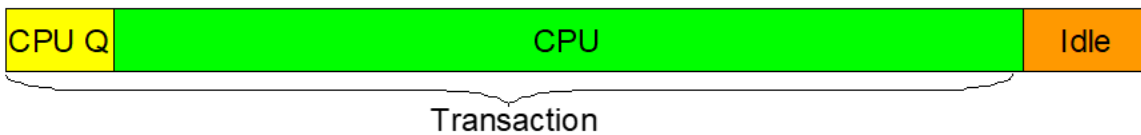
2.6.2 IBM i wait buckets
The following 32 wait groups or *buckets* are defined.
Note: The definition of the wait groups varies from release to release and may change in the future.
1   Time dispatched on a CPU
2   CPU queuing
3   Reserved
4   Other waits
5   Disk page faults
6   Disk non-fault reads
7   Disk space usage contention
8   Disk operation start contention
9   Disk writes
10   Disk other
11   Journaling
12   Semaphore contention
13   Mutex contention
14   Machine level gate serialization
15   Seize contention
16   Database record lock contention
17   Object lock contention
18   Ineligible waits
19   Main storage pool contention
20   Journal save while active
21   Reserved
22   Reserved
23   Reserved

24 Socket transmits
25 Socket receives
26 Socket other
27 IFS
28 PASE
29 Data queue receives
30 Idle/waiting for work
31 Synchronization Token contention
32 Abnormal contention

There are many wait groups that may surface during wait analysis of your application. Understanding what your application is doing and why it is waiting in those situations can help reduce or eliminate unnecessary waits.

For more information on these wait groups, refer to the following white papers:
- Job Waits Whitepaper
- IBM i Wait Accounting

2.6.3 Holders and waiters
IBM i not only tracks the resource a thread or task is waiting on, but also monitors the thread or task that currently holds the resource. This is a powerful feature. A *holder* is the thread or task that is using the serialized resource, while a *waiter* is the thread or task requesting access to that resource.

2.6.4 Call stacks
IBM i also manages call stacks for each thread or task, independent of the wait accounting information. The call stack shows the programs invoked and can help understand the wait condition, providing insight into the logic leading to either holding or requesting access to a resource. The combination of holder, waiter, and call stacks provides a powerful capability for analyzing wait conditions. Starting with IBM i 7.3, PASE stack frames will be collected and reported in Job Watcher when the call stack data collection option is enabled.

2.6.5 Collect and analyze the data
Collection Services and Job Watcher are two performance data collection mechanisms on IBM i that gather wait accounting information. Job Watcher also collects holder and waiter information, as well as call stacks. Once the performance data is collected, it can be graphically analyzed. The iDoctor product has a Microsoft® Windows® client for viewing performance data, while the IBM Navigator for i web console includes the performance data investigator feature for viewing performance data from a web browser interface.

2.7 Acceptable performance
The performance of a system should ideally be evaluated through objective measurements, such as application log files or batch job runtimes. Acceptable performance is defined by customer expectations, which may be based on benchmarks, modeling, or experience. If actual users find the response time or throughput unacceptable, it indicates a potential issue. Incorrect assumptions when architecting the system may result in a situation where acceptable performance cannot be achieved.

2.8 Partition scaling considerations
IBM Power hardware is highly versatile and scalable. Additional hardware resources (CPU, memory, disk, adapters, and so on) can be easily added to an IBM i partition to scale the partition's hardware. In many

cases, application throughput scales linearly with hardware, meaning that increasing or decreasing hardware resources results in a corresponding increase or decrease in throughput. However, not all environments are the same, and the software stack or allocated hardware resources may have performance limitations that prevent linear scaling. This could manifest in various ways, such as a need for more specific hardware resources when scaling up or requiring more resources than expected when sizing a partition based on CPW ratings. In some cases, throughput may significantly drop despite exponential demand for specific hardware resources.

These considerations are not specific to IBM i or Power hardware. While terminology may differ, the following AIX document discusses considerations from their perspective:
Considerations in software design for multi-core multiprocessor architectures

When investigating application scalability, it's important to recognize that all work requested of the hardware begins with the application software. Improving software (e.g., eliminating bottlenecks or inefficient coding) or altering the workload can often be done more quickly than waiting for future hardware enhancements or OS updates. The best approach to ensure linear application scalability includes:

- Test the environment: The test environment should consist of the same hardware resources, application code, and OS release running the expected workload. Creating a test environment or benchmark helps assess whether an application can scale as workload increases. IBM has benchmark centers to assist with this.
- Engage IBM Technology Expert Labs: Chapter 7 provides more information on engaging IBM Technology Expert Labs.
- Review resource usage: Use the tools in Chapter 6 to understand how system resources are used.
- Plan for performance as workload increases: Use the tools in Chapter 5 to monitor performance trends. Chapter 4 offers tools to plan for adding additional hardware resources as workload grows.
- Avoid common pitfalls: Chapter 7 and subsequent chapters provide best practices, such as reducing full opens, program activations, and high logical I/O.

Examples of considerations include:
- Memory: Each job has a working set that grows over time. The application's working set size may increase at a rate inconsistent with the workload, requiring more memory than expected.
- Disk I/O: A high volume of disk operations may eventually exceed hardware replication (for example, global mirroring) capabilities, significantly increasing response times.
- CPU: A shift from slower to faster processors may result in more jobs concurrently executing a single shared data object (for example, a counter), causing unexpected latency and increased resource consumption.
- Commercial processing workload (CPW) rating: An equivalent partition size on a different hardware architecture platform may not perform as expected for a given application.

3 Performance benchmarks

3.1 Introduction
This chapter covers performance benchmarks and how they should be used to compare system performance.

3.2 What are performance benchmarks?
Performance benchmarks are well-defined tests that evaluate and compare the performance of computer systems. These tests use representative sets of programs and data designed to assess the performance of hardware and software in a given configuration.

3.3 How are IBM i systems rated for performance?
IBM publishes CPW ratings for all IBM i systems to help customers compare the relative throughput capacity of systems.

3.4 What is the CPW rating of a system?
The CPW rating of a system is generated by measuring the performance of a specific workload, which is maintained internally within the IBM i Systems Performance group. The CPW rating evaluates a computer system and its associated software in a commercial environment. It is defined as a relative capacity metric for model comparisons and CPU consumption. The CPW rating is not representative of any specific environment, but it is generally applicable to the commercial computing environment.

What the CPW rating is:
- A test of a range of database applications, including various complexity updates and queries with commitment control and journaling.
- A test of concurrent data access by users running a single group of programs.
- A reasonable approximation of the relative performance of a steady-state, database-oriented commercial application.

What the CPW rating is not:
- An indication of a system's performance capabilities for any specific customer situation.
- A test of *ad hoc* (query) database performance.
- A test of single-threaded (batch) application throughput (for example, batch processing steps per minute).
- A test of single-threaded (batch) application run time or *batch window* (for example, job completes in a 4-hour batch window).

When to use the CPW rating results:
- To approximate product positioning between different systems running IBM i, where the primary application is expected to be oriented toward traditional commercial business uses (for example, order entry, payroll, billing).
- The CPW metric should be used in terms of system capacity potential (throughput), not in terms of the response time of transactions, batch jobs, or queries.

3.4.1 CPW rating versus public benchmarks

Specific choices were made in creating the CPW rating to best represent the relative positioning of IBM i systems. Some of the differences between the CPW rating and public benchmarks are:

- The code base for public benchmarks is constantly changing to obtain the best possible results, while the CPW rating attempts to keep the base constant to better represent relative improvements across releases and systems.
- Public benchmarks typically do not require full security, but since IBM customers tend to run on secure systems, Security Level 50 is specified for the CPW rating.
- Public benchmarks are optimized to achieve the best results for a specific benchmark, whereas the CPW rating uses more system defaults to better represent the system as shipped to customers.
- Public benchmarks can use different applications for different-sized systems and take advantage of all available resources, while the CPW rating runs the same application at all levels with approximately the same disk and memory resources per simulated user on all systems.
- Public benchmarks require extensive, sophisticated driver and middle-tier configurations. To simplify the environment, all required components for the CPW rating are included as part of the overall workload.

The net result is that the CPW rating is an application model that IBM believes provides a good indicator of multi-user transaction processing performance when comparing members of the IBM i system families. It is not intended to guarantee performance but serves as a reliable metric for multi-user transaction processing workloads.

3.5 Comparing benchmark results

When comparing performance benchmark results, it is crucial to compare the same benchmark test(s) to ensure consistency. The result of one benchmark test may not represent the performance of a computer system for a different workload.

For example, the result of a floating-point intensive benchmark does not provide information about the performance of the system running an integer-intensive benchmark or OLTP workload, and vice versa.

A common pitfall is to apply the results of one benchmark to a different workload. For instance, if an OLTP workload improves by 50% when upgrading from system A to system B, it would be invalid to assume that a compute-intensive workload would see the same improvement. Likewise, improvements in single-threaded workloads may not necessarily translate into the same improvements for OLTP workloads.

3.6 Custom benchmarks and proof of concept

Custom benchmarks are used when a customer needs to confirm the sizing requirements of a specific application on hardware configurations, especially when sizing tools do not offer sufficient confidence. This often involves testing application scalability across multiple hardware configurations.

Custom benchmarks can range in complexity from simple tests, such as end-of-day batch processing, to complex simulations of thousands of online users accessing a multi-tiered web application.

When performing a custom benchmark to measure performance for a customer's production workload, it is critical that the benchmark test represents the real workload to obtain meaningful data. For example, testing a single database job on an idle server gives useful data under ideal conditions but may not represent the performance under a medium or heavy production workload.

3.6.1 Considerations

When conducting a custom benchmark or proof of concept (PoC), it is important to simulate the production environment, especially as hardware evolves into multi-core systems with more focus on the cache/memory hierarchy.

3.6.2 Common pitfalls for custom benchmarks

A common pitfall in custom benchmarks or PoCs is failing to simulate the real production environment. This can lead to benchmark results that differ significantly from actual production performance. A benchmark test might yield better results than what will be achieved in production, leading to performance problems later. Conversely, if the benchmark does not simulate real workloads, it could cause delays or failures in the PoC.

3.6.3 IBM i Performance and Scalability Services Center

The IBM i Performance and Scalability Services Center in Rochester, MN, USA offers facilities, hardware, and expertise to assist with custom benchmarks or PoCs. Some examples of services provided include:

- Proof of concept (for example, HA alternatives, SSD analysis, external storage)
- Stress testing hardware configurations
- Evaluating application scalability
- Performance optimization and tuning
- Assessing application performance when migrating to a new release of IBM i
- Determining the impact of application changes
- Virtualization, consolidation, and migration services
- Capacity planning

To request a custom benchmark or PoC, submit a request by using the following link:
Contact IBM Technology Expert Labs

4 Sizing a system

4.1 Introduction
Sizing a system and its components to support a production environment can be complex. It requires understanding the characteristics of the workload(s) and the load they place on system components.

Before beginning the sizing process, consider these questions:
1 What are the primary metrics (example: throughput, latency) used to validate that the system meets performance requirements?
2 Does the workload run at a steady state, or is it bursty, causing spikes in load on system components? Are there specific criteria (example: maximum response time) that must be met during peak loads?
3 What are the average and maximum loads that need to be supported on various system components (example: CPU, memory, network, storage)?

4.2 Sizing best practices
Following are some of the best practices involved in sizing a system:
- Know your bottlenecks: If the workload is I/O constrained, upgrading to the latest processor technology may not help. Use tools like Performance Data Investigator (PDI) and IBM iDoctor for IBM i (iDoctor) to identify bottlenecks at the partition and job levels.
- Size for peaks: Ensure enough capacity to handle performance during end-of-month or year-end activity spikes.
- Focus on important jobs/subsystems: Adding cores will reduce CPU utilization but may not improve performance for specific batch jobs.
- Be careful with I/O sizing tools: IBM i application performance is sensitive to disk write response time and the number of disk units or LUNs. Switching to certain external storage solutions or reducing the number of disk units/LUNs may negatively affect performance.

4.3 System sizing tools
The following sections explain the system sizing tools in detail.

4.3.1 IBM Systems Workload Estimator
The IBM Systems Workload Estimator (WLE) is a web-based sizing tool for IBM Power and IBM z® systems.

You can use this tool to size a new system, to size an upgrade to an existing system, or to size a consolidation of several systems. WLE characterizes your projected workload either with customer measurement data or by using one of the many workload plug-ins (also known as sizing guides). Virtualization can be reflected in the sizing to yield a more robust solution by using various types of partitioning and virtual I/O. WLE provides current and growth recommendations for processor, memory, and disk (either internal or SAN) that satisfy the overall customer performance requirements.

WLE can support sizing dealing with multiple systems, multiple partitions, multiple operating systems, and multiple sizing intervals. These features can be coordinated by using the functions on the Workload Selection screen.

WLE recommends the system model, processor, memory, and disk requirements that are necessary to handle the overall workload with reasonable performance expectations. To use WLE, you select one or more workloads and answer a few questions about each workload. Based on the answers, WLE

generates a recommendation and shows the predicted CPU utilization of the recommended system in graphical format. The results can be viewed and saved as a PDF. The visualize solution function can be used to better understand the recommendation in terms of time intervals and virtualization. For many systems, WLE can also provide an energy consumption estimate to help with your "go green" initiatives. There is an optional integration point with planning tools (System Planning Tool, and the IBM Pre-sales Advisor Tool) so that the configuration planning and validation may continue.

The WLE recommendation is based on processing capacity, which assumes that the system can handle the aggregate transaction rate, and that the application can fully scale on the system. Although WLE does not model response times specifically, it abides by the best practice utilization guidelines to help minimize potential negative impacts to response time. Beyond what is recommended here, additional system resources may be required for additional workloads not sized here, growth resulting from workload changes, version/release changes not already considered, minimum memory to support I/O or virtualization configurations, minimum disk to support multiple ASP or RAID configurations, and all other resources beyond the scope of WLE (CPU, memory, and disk).

The WLE recommendation assumes that your system is well tuned in terms of performance (including the system hardware configuration, operating system settings, virtualization configuration and settings, and the application). The WLE scaling algorithms assume that the sized applications are multi-thread capable and can exploit and scale with multiple cores and simultaneous multithreading (SMT); otherwise, the sizing is invalid. So, do not use WLE to size single-threaded applications or for single-threaded time critical batch jobs. For these, it is important to also consider the performance per thread and per core, as well as GHz.

Sizing tool input starts with well-defined, realistic, consistent workloads and not with industry benchmarks (that many times avoid customer-like functions like logging and random non-cached accesses). WLE workload requirements come from performance data from existing customer systems or from WLE workload plug-ins. In the case of existing systems, it is important to consider peak data to fully support your business along with a growth trend. For plug-ins, it is important to answer the questionnaires with responses to adequately cover your peak requirements.

As with every performance estimate (whether a rule of thumb or a sophisticated model), you always need to treat it as an estimate. This is particularly true with robust IBM Systems that offer so many different capabilities where each installation will have unique performance characteristics and demands. The typical disclaimers that go with any performance estimate (for example: "your experience might vary...") are especially true. We provide these sizing estimates as general guidelines but cannot guarantee their accuracy in all circumstances.

4.3.1.1 How to access the IBM Workload Estimator
IBM Workload Estimator can be found at: <u>Workload Estimator</u>

4.3.1.2 How to get assistance in sizing a system
Use the IBM Workload Estimator or contact an IBM Business Partner or IBM Sales for assistance. IBM Post Sales Support does not provide sizing help.

4.3.2 IBM Storage Modeller
The IBM Storage Modeller is a cloud-based tool designed to model IBM SAN storage solutions for customer applications. It is available to technical sellers of IBM storage systems, including IBM Sales and Business Partners.

For more information about IBM Storage Modeller tool, refer to [IBM Storage Modeller](#) documentation.

To access the IBM Storage Modeller tool, visit [IBM Storage Modeller](#).

5 Proactive performance monitoring

5.1 Introduction

This chapter covers the proactive performance monitoring process from a high-level perspective. Its purpose is to provide guidelines and best practices on addressing performance issues using a top-down approach.

5.2 Performance monitoring before a problem occurs

Application performance should be recorded using log files, batch run times, or other objective measurements. General system performance should also be recorded and should include as many components of the environment as possible.

Performance statistics should be collected based on the typical period of activity in your environment. For example, a one-week period should be considered the minimum in an interactive system, as performance may vary based on the day of the week. A batch processing system that typically runs large end-of-month reports should consider a month as the minimum period. Data should be collected and stored over multiple periods to allow for comparison.

Proactive performance monitoring of the environment provides many benefits. Application performance metrics offer a valuable method for quantifying performance changes, while a history of acceptable system performance directs analysis to any components that have changed. Additionally, continuous performance monitoring reveals performance and capacity trends before they become system-wide issues.

5.2.1 Real-time interactive monitoring

Basic system commands provide a real-time view of data and can help identify potential issues early. The most common commands include:

- `WRKACTJOB`: Shows all jobs with key system resources used, including CPU %, response time, I/O information, and temporary storage used by each job.
- `WRKSYSSTS`: Displays system utilization and pool usage.
- `WRKDSKSTS`: Shows statistics for all disk units.
- `WRKSYSACT`: Displays tasks, primary threads, and secondary threads that have consumed CPU in the refresh period. It also shows temporary storage used.

Except for `WRKSYSACT`, equivalent GUI task versions of these commands are available in IBM Navigator for i (Active Jobs, System Status, and Disk Status).

Additionally, corresponding IBM i Services are available - ACTIVE_JOB_INFO, SYSTEM_STATUS, and SYSDISKSTAT.

5.2.2 Dashboard on navigator for i

The dashboard on IBM Navigator for i provides real-time information for multiple systems. Data updates automatically by default. Each system added displays three primary metrics: CPU percent used, System ASP percent used, and the number of active jobs.
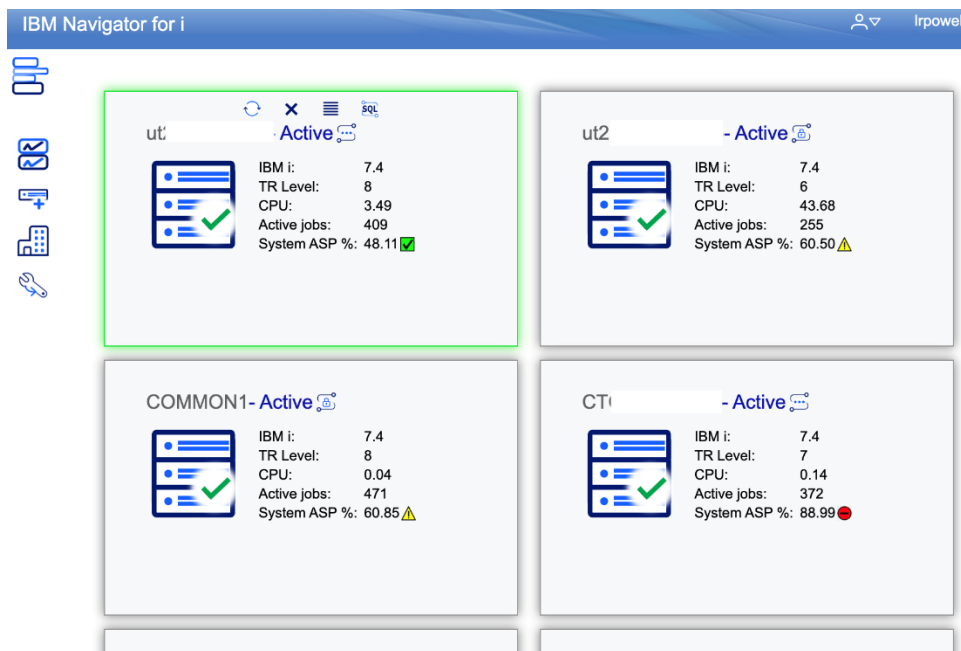
Figure 1. IBM Navigator for i dashboard

Adding more nodes allows more systems to appear on the managing node GUI dashboard. Select the upper-left icon to switch from tile view to table view.
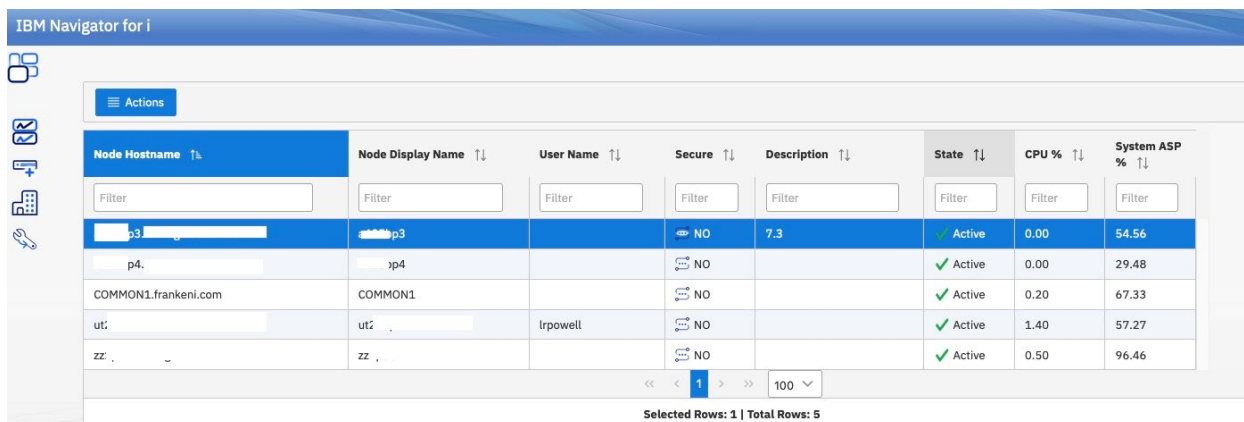
Figure 2. IBM Navigator for i dashboard table view

For additional details on this dashboard, refer to Navigator for i - Dashboard.

5.2.3 Automated monitoring
There are monitoring services available in IBM Navigator for i.
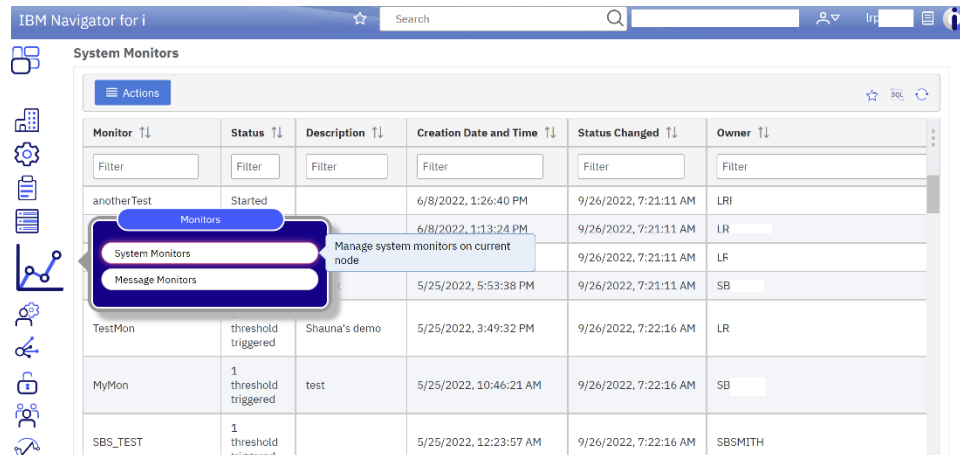
IBM Navigator for i
IBM Navigator for i is a modern web-based interface for managing and monitoring one or more IBM i instances from a single location.

System monitors and Message monitors are available in IBM Navigator for i.

Monitors can be used to pro-actively monitor system health and message queues, alerting you of potential performance problems before they become serious issues.

Thresholds are set for various metrics on these monitors and provide a command action (optional) which is executed when a monitor detects that a threshold is triggered. For a command example, you can run an IBM i command or start a program when the threshold is triggered.

System monitors have a collection interval that is set from 15-300 seconds (5 minutes). Generally, for system monitors, 60 seconds provides a good balance between timeliness and the size of the monitor data collection. 15-second collections can get large quickly.

Figure 3. IBM Navigator for i system monitors

The system monitors in IBM Navigator for i use performance data investigator to display the monitor performance data as a graph or multiple graphs on a dashboard view. These system monitors are limited to a single partition.

IBM Navigator for i system monitor metrics are collected, calculated, and managed by collection services and the data is stored in database files (no longer stored in a private *MGTCOL as done in management central).

Because collection services calculate and manages the system monitor metric data, the metric data can be generated for your collection services collection without ever starting an IBM Navigator for i system monitor. To do this, specify `CRTPFRSUM(*ALL)` on either the `CFGPFRCOL` or `CRTPFRDTA` command. Or you can start a system monitor collection by using the `CFGPFRCOL` command and specifying `ENBSYSMON(*YES)`.

For more information, refer to the following documentation:
IBM Docs - IBM Navigator for i Monitors
System Monitors Overview
Message Monitor

5.2.4 Monitoring messages
The system provides a watch-for-event function that enables monitoring for specific messages. With this function, you can specify messages the system should watch for. When these messages occur, a user exit program is triggered to take any necessary action. You must specify the message queue or job log

where you expect the message to be sent. You can also define text strings to compare against message data, specify the originating program, or the destination program. Selecting watches by message type or severity allows you to focus on only certain messages. Watch sessions can be started, ended, and viewed using IBM Navigator for i. For more information, refer to [Watching for Messages](#) – IBM support documentation.

5.2.5 IBM i services
IBM i provides many services to help monitor key aspects of your partition. These services can be listed and launched directly from IBM Navigator for i.
For more information, refer to:
- [Chapter 9: IBM i Services](#)
- [IBM i Services (SQL)](#)

5.3 Defining a performance problem (what is slow?)
Before gathering additional data or making configuration adjustments, clearly define the specific aspect that is slow. A precise problem definition helps reduce the time required to address a performance issue and guides the performance analyst on what data to collect and analyze.

A good way to define a performance issue is to consider questions such as:
- Is everything slow or only a specific task?
- Which application log file indicates the performance issue?
- Is it slow when initiated remotely but fast locally?
- Is the issue experienced by everyone or just a single user?

6 Performance data collectors and analysis tools

6.1 Introduction
This chapter provides information on available tools for collecting data on IBM i systems and toolsets for visualizing data to support easier interpretation and analysis.

6.2 Manage performance on IBM i
Managing performance requires various specialized applications, each providing specific insights into system performance. Some applications collect data, while others display, analyze, monitor, or manage the data collected. For more information, refer to IBM i Performance IBM documentation.

6.3 Performance Data Collectors
A key strength of IBM i in performance management is its built-in collection technologies, providing robust metrics that can aid in data analysis.

The primary data collectors include:
- Collection Services
- Disk Watcher
- Job Watcher
- Performance Explorer
- SQL Plan Cache

Each collector offers unique benefits, such as sample data or trace data collection. Sample data is gathered at specific intervals, while trace data records events in detail as they occur.
For information on performance data collection tools, refer to the Performance Data Collectors section in IBM Docs. Recommendations for configuring performance data collectors are outlined in the sections below.

6.3.1 Collection Services
Collection Services collects system and job-level performance data and is the primary system data collector, recommended to be always active. It gathers data from resources like CPU, memory pools, disk (internal and external), and communications.

Collection Services samples data at intervals from 15 seconds to 1 hour, with a 15-minute default. Often, Collection Services is used to:
- Monitor system performance
- Investigate reported problems
- Identify areas for improvement

It provides insights into resource usage, changes, timing, and affected components.

Best practices for Collection Services include:
- Set collection intervals to 5 minutes
- Maintain at least one week of data
- Keep historical data for comparison
- Enable cross-partition performance collection on the HMC (refer to collect and View performance data for entire physical system)

Collection Services can be managed by the IBM Navigator for i Performance interface, IBM iDoctor for i, or CL commands. For additional information, refer to [Collection Services](#) – IBM documentation.

6.3.2 IBM i Disk Watcher

IBM i Disk Watcher collects disk performance data for diagnosing disk-related issues, obtaining data on I/O operations, and identifying accessed objects, files, processes, threads, and tasks. Disk Watcher provides data beyond standard tools like `WRKDSKSTS`, `WRKSYSSTS`, and `WRKSYSACT` and supports both short and long-duration traces.

Some potential uses of this tool include:
- Evaluating the performance of I/O operations on multi-path disk units
- Evaluating the performance of I/O queuing
- Determining how performance may be improved by respreading data across units
- Determining the optimal placement of devices, IOAs, or buses

Disk Watcher can be managed by the IBM Navigator for i Performance interface, IBM iDoctor for i, or CL commands. For more information, refer to [IBM i Disk Watcher](#) – IBM documentation.

6.3.3 IBM i Job Watcher

IBM i Job Watcher collects job data for any or all jobs, threads, and tasks, providing call stacks, SQL statements, Java™ JVM statistics, wait statistics, and more. This tool is essential for diagnosing job-related performance issues. Similar to `WRKACTJOB` and `WRKSYSACT`, Job Watcher calculates delta information at each interval in a non-intrusive manner.

Best practices for Job Watcher:
- Consider running Job Watcher continuously using [QMGTOOLS: Job Watcher Monitor Function](#) or the [iDoctor Monitors Guide](#)
- Use a 5 or 10-second interval definition, collect SQL data, and gather call stacks (for example, Q10SECSQL)
- Maintain historical data for comparison

Job Watcher can be configured by IBM Navigator for i Performance, IBM iDoctor for i, QMGTOOLS, or CL commands. For more information, refer to [IBM i Job Watcher](#) – IBM documenatation.

6.3.4 Performance Explorer

Performance Explorer gathers detailed information on specific applications, programs, or system resources to address complex performance problems. Performance Explorer performs various trace types and levels and provides detailed reports.

Performance Explorer is a data collection tool that helps the user identify the causes of performance problems that cannot be identified by collecting data by using Collection Services, Job Watcher, Disk Watcher or by doing general trend analysis. Two reasons to use Performance Explorer include:
- Isolating performance problems to the system resource, application, program, procedure, or method that is causing the problem
- Analysing the performance of applications

As your computer environment grows both in size and in complexity, it is reasonable for your performance analysis to gain in complexity as well. Performance Explorer addresses this growth in complexity by gathering data on complex performance problems.

Note: Use Performance Explorer after other tools, as it captures specific data that may impact system performance if collected improperly.

This tool is designed for developers aiming to improve application performance and for users skilled in performance management to identify complex issues. Performance Explorer can be configured and managed through IBM iDoctor for i or CL commands.

For more information, refer to IBM i Performance Explorer – IBM documentation.

In 2007, an IBM Redbooks® was written on analysing application performance using PEX. Although it covers V5R3 and V5R4 releases, much of the content remains relevant (excluding outdated screenshots and GUI elements). For more information, refer to Application and Program Performance Analysis Using PEX Statistics – IBM Redbooks.

Collector Data Characteristics

Collection Services	Disk Watcher	Job Watcher	Performance Explorer

▪ Sample data
▪ Designed to collect data 24x7
▪ Support for small intervals
▪ No information concerning specific I/O operations
▪ Wait time information
	▪ Statistics and Trace data
	▪ Detailed information focused on I/O operations to disk units
		▪ Sample data
		▪ Support for very small intervals
		▪ Focus on job data
		  - Call Stacks
		  - SQL statements
		▪ Wait time information
			▪ Statistics, Profile, or Trace data
			▪ Information collection for every I/O event
			▪ Collection and analysis can be complex

Figure 4. Collector data characteristics

6.3.5 SQL Plan Cache
The SQL plan cache contains access plans for SQL queries that have already run or are currently running on the partition, as well as performance information such as query run times, CPU usage, and I/O usage. Since the plan cache does not persist after an IPL, it is best practice to generate a snapshot of the plan cache before each IPL and prior to any major application or software changes. A snapshot can be created using the stored procedure `CALL QSYS2.DUMP_PLAN_CACHE` or through the IBM i Access Client Solutions (ACS) SQL performance center.

6.4 Analysis tools

6.4.1 Overview
It is essential to understand the distinction between performance data collectors and tools for visualizing or analyzing that data. Performance data collectors are integrated into the operating system and allow for collecting a variety of performance data, but visualizing or analyzing this data may require additional interfaces.

IBM offers several tools with different user interfaces for viewing, analyzing, reporting, graphing, and managing performance data.

6.4.2 Performance tools for i
The IBM Performance Tools (PT1) includes various tools, such as performance reports for Collection Services. While these reports are text-based and may not always be the most accessible method for performance analysis, graphical analysis is often simpler.

The requirement to install PT1 to access certain Performance Data Investigator (PDI) graphical analysis features has been removed, allowing any authorized user to access and use Disk Watcher, Performance Explorer (PEX), Database, and Job Watcher.

IBM Performance Tools (PT1) LPP is now available at no charge; for more information, refer to IBM i Portfolio Simplification.

6.4.3 IBM i Access Client Solutions (ACS)
IBM i Access Client Solutions (ACS) provides a platform-independent, Java-based interface compatible with Linux®, Mac®, and Microsoft® Windows®. It consolidates commonly used IBM i management tasks into a single location and includes critical database functions like SQL Performance Center and Run SQL Scripts. ACS also allows for launching the IBM Navigator for i.

6.4.4 IBM Navigator for i
IBM Navigator for i is the strategic tool for IBM i administrative tasks, supporting most functions required for managing and monitoring one or more IBM i instances from a central interface.

Navigator includes features like Performance Data Investigator (PDI) for performance data analysis (detailed in section 6.4.5). This web-based application does not require any installation on your workstation; it is part of the base IBM i operating system. Access Navigator by entering `http://hostname:2002/Navigator/login` (where *hostname* is your server's name and *2002* is the port number) in a browser for an insecure connection. For a secure (TLS) connection (recommended), use `http://hostname:2003/Navigator/login`.

Navigator is available on IBM i 7.3, IBM i 7.4, and IBM i 7.5, with PTFs for IBM i 7.3 and 7.4 initially released in September 2021.

For more information on this topic, refer to the following documentation:
- Navigator for i
- IBM Docs: IBM Navigator for i
- IBM Support Documentation: IBM Navigator for i - TLS Encryption

6.4.5 Performance Data Investigator (PDI)
Performance Data Investigator (PDI), a feature within IBM Navigator for i, provides a web-based GUI that allows interactive analysis of performance data through charts and tables. Using PDI, you can view and analyze data collected by Collection Services, IBM i Job Watcher, and IBM i Disk Watcher. Additionally, PDI offers chart options for Health Indicators, Database, System Monitors, and Graph History (available in IBM i 7.3 and later).

Within the Performance section, PDI also provides access to configuration commands for Collection Services, Disk Watcher, Job Watcher, and tools for managing collections.

To access PDI, select the Performance category in IBM Navigator for i and open the **Investigate Data** task. This opens various perspectives (charts and tables), organized by package name:
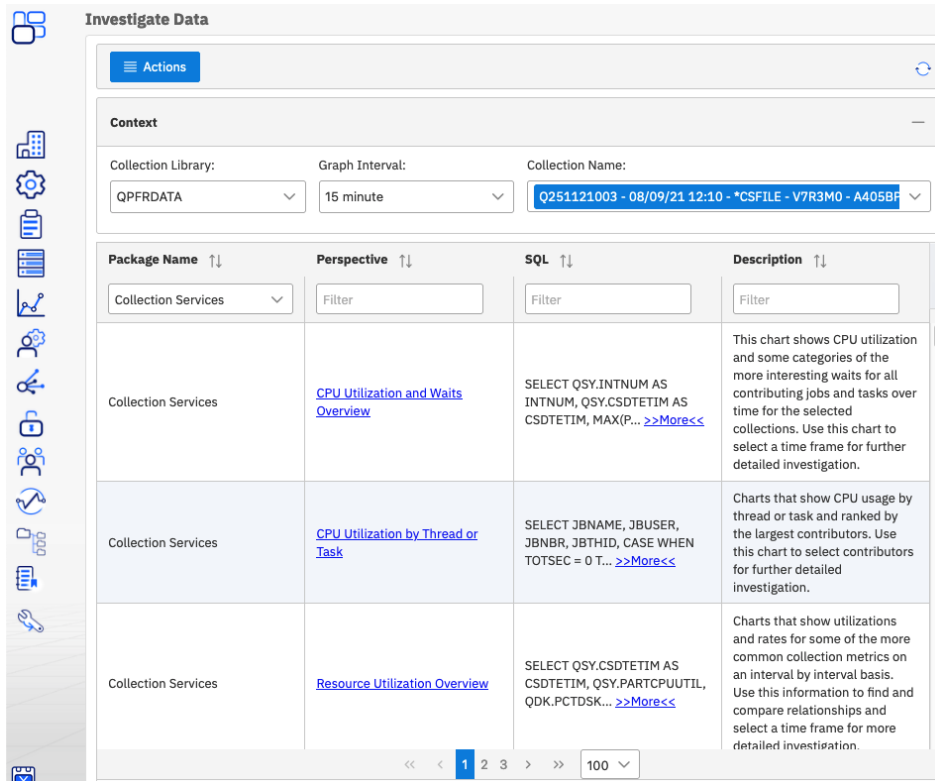
Figure 5. Performance Data Investigator

The capability to access perspectives across all packages is integrated into the base operating system, including views for Collection Services, Health Indicators, System Monitors, Job Watcher, Disk Watcher, Database, and Graph History.
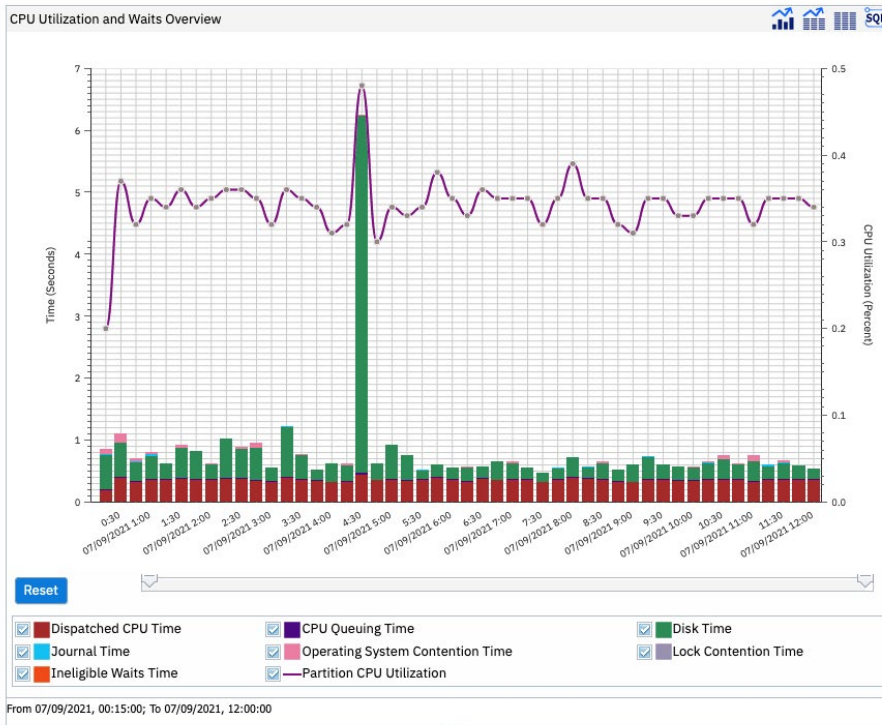
Figure 6. Performance Data Investigator example

Graph History charts, available starting with IBM i 7.3, utilize Collection Services' historical data to provide long-term insights (multiple days, months, and years) and allow comparison with real-time data in the System Monitor.

In the following stacked chart example, a sample layout displays one week's worth of data stacked for each day's comparison. The layout includes two drill-down panels on the right:
The top-right panel lists the top contributors for a selected data point.
The bottom-right panel displays the properties for an object from the top contributors list.
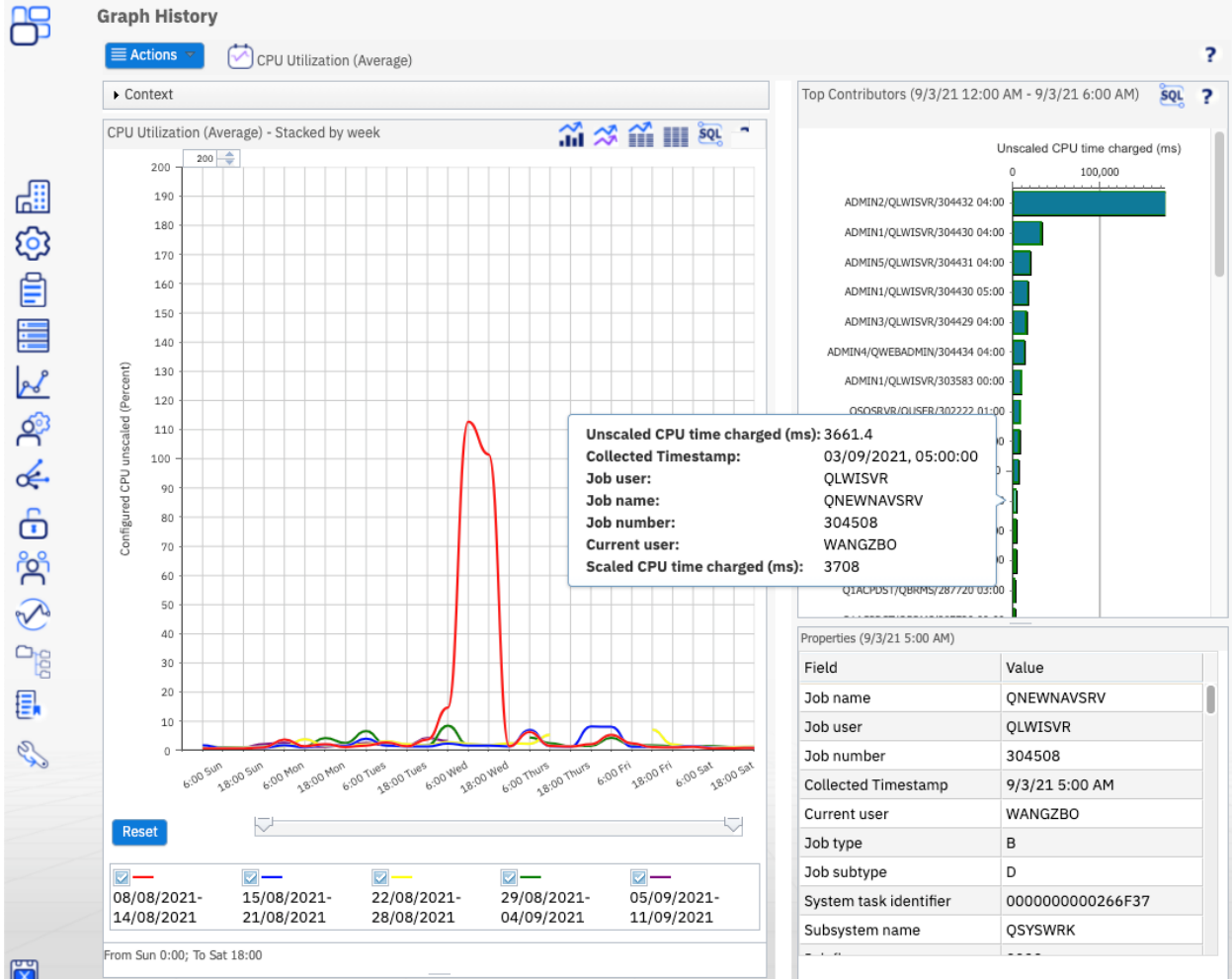
Figure 7. Graph history example

For more information on this topic, refer to the following documentation:
- [A new way to analyze historical performance data on IBM i](#)
- [IBM Docs: Performance Data Investigator](#)
- [Performance on the web - Performance tools for IBM i](#)

6.4.6 IBM iDoctor for IBM i

IBM iDoctor for IBM i is a Windows-based suite of performance tools that can be used for analysis of Collection Services, Job Watcher, PEX, Disk Watcher, and SQL Plan Cache Snapshots.

IBM iDoctor consists of following components:
- Collection Services Investigator
- Job Watcher
- PEX Analyzer
- Disk Watcher
- Plan Cache Analyzer
- NMON

IBM iDoctor for IBM i is generally used by the performance expert to collect, investigate, and analyze performance data on IBM i. The tools are used to monitor overall system health at a high *overview* level

or to drill down to the performance details within job(s), disk unit(s) or programs over data collected during performance situations.
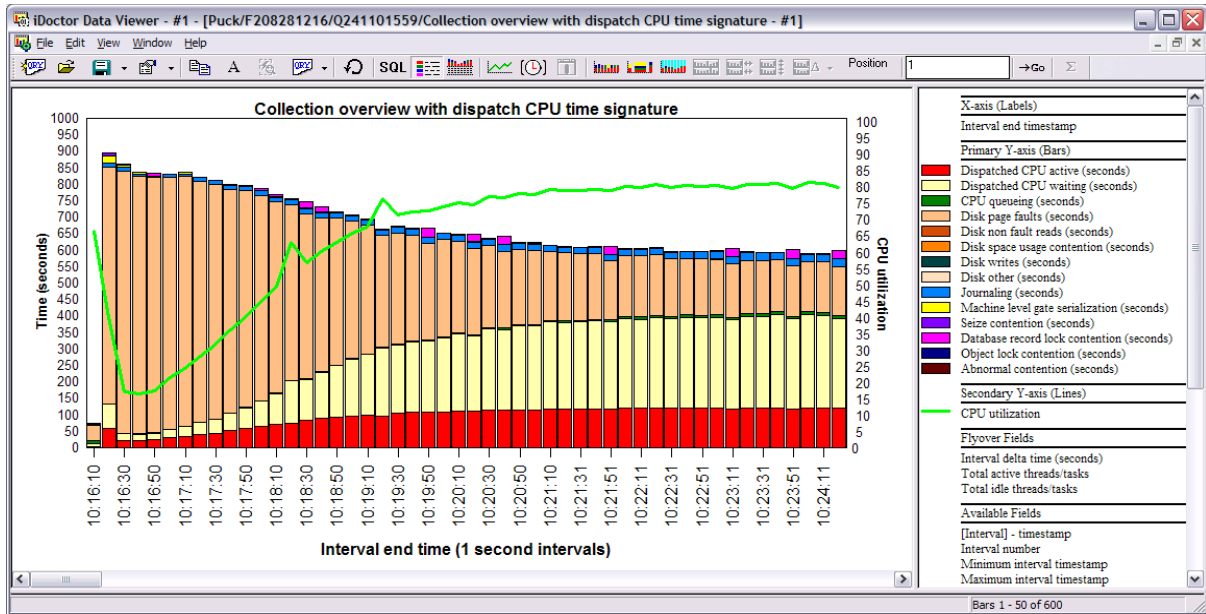
Figure 8. IBM iDoctor example

iDoctor is constantly evolving to meet the needs of our clients. Therefore, it is not a formal licensed program product. The software is licensed for use under a service agreement. For more information on free 45-day trial license, or a price quote, refer IBM iDoctor for IBM i website.

Training for iDoctor is available. For more information, refer iDoctor Workshop. For additional information on education and training, refer to section of chapter 16.

6.4.7 Tools matrix
The following chart shows at a high-level the data that can be analyzed by the various IBM i performance analysis tools.
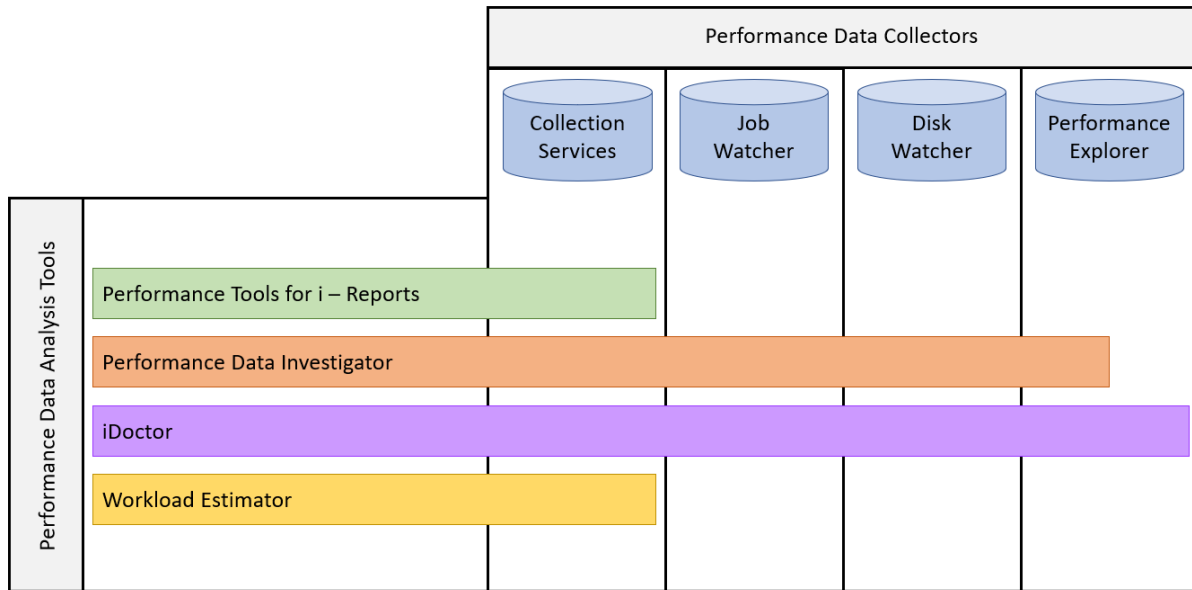
Figure 9. Relationship between collector and major functions or interfaces

6.4.8 Open source languages

All of the analysis tools discussed in this section can be used to investigate programs written in open source languages, including those running in PASE. Examples of such languages include JavaScript (Node.js), Python, Ruby, PHP, and others. Performance investigation of these programs can also leverage language-specific tools. The open source community often maintains tools that can perform method profiling, memory analysis, tracing, and so on, and these tools can be used complementary to IBM i tooling.

7 Frequently asked questions

7.1 Introduction
This chapter covers frequently asked questions.

7.2 General questions

7.2.1 What are the general performance tuning recommendations for best IBM i performance?
There are no universal performance tuning recommendations. Adjustments should be based on system-specific analysis.

7.2.2 I heard something on the internet; should I change?
No, never apply any tuning changes based on information from unofficial channels and always consider the author or source of that information. Make sure the advice applies to IBM i, especially when related to Java and Db2. Configuration and application changes should be done based on performance analysis and tested before being made in production.

7.2.3 My <CPU, disk, memory> usage is high; should I add more resources?
Before adding hardware, confirm that existing resources are used efficiently through system tuning, database, and application analysis.

7.2.4 Why didn't the performance of my partition improve after I added <CPU, memory, disk arms, SSDs>?
The bottleneck may not be related to the resource you added (CPU, disk, memory). Proper wait time analysis should be done to identify the bottleneck and how to best resolve it. Refer to the Components of response time section of this document.

7.2.5 Should I update PTFs before starting a performance analysis project?
Yes, install the latest cumulative and group PTFs, along with the latest technology refresh (TR) levels, as these may resolve performance issues.

For additional information, refer to the following documentation:
- IBM i Technology Refreshes
- IBM i Support: Recommended fixes

Some PTFs are required before certain performance data can be collected. In addition to the latest performance tools group PTFs (SF99723 for 7.3, SF99663 for 7.4, and SF99953 for 7.5), the following lists should be checked for additional PTFs:
- 7.5: Job Watcher, Performance Explorer (PEX), and Disk Watcher PTFs for IBM i 7.5
- 7.4: Job Watcher, Performance Explorer (PEX), and Disk Watcher PTFs for IBM i 7.4
- 7.3: Job Watcher, Performance Explorer (PEX), and Disk Watcher PTFs for IBM i 7.3

You can find information on the latest Db2 for IBM i fixes and enhancements at the following link. IBM recommends staying current on fixpacks, which can include performance enhancements.
Db2 for i - Technology Updates

You should also keep the firmware level current as this can also affect overall system performance. Additionally, certain performance data is only collected if you are at a minimum firmware level.

Released PTFs enhance the functionality of the features in Navigator, including the performance tasks.

Recommended PFT groups for full functionality of the Navigator tasks:
- Performance Tools group
- HTTP Server group
- Java group
- Database group
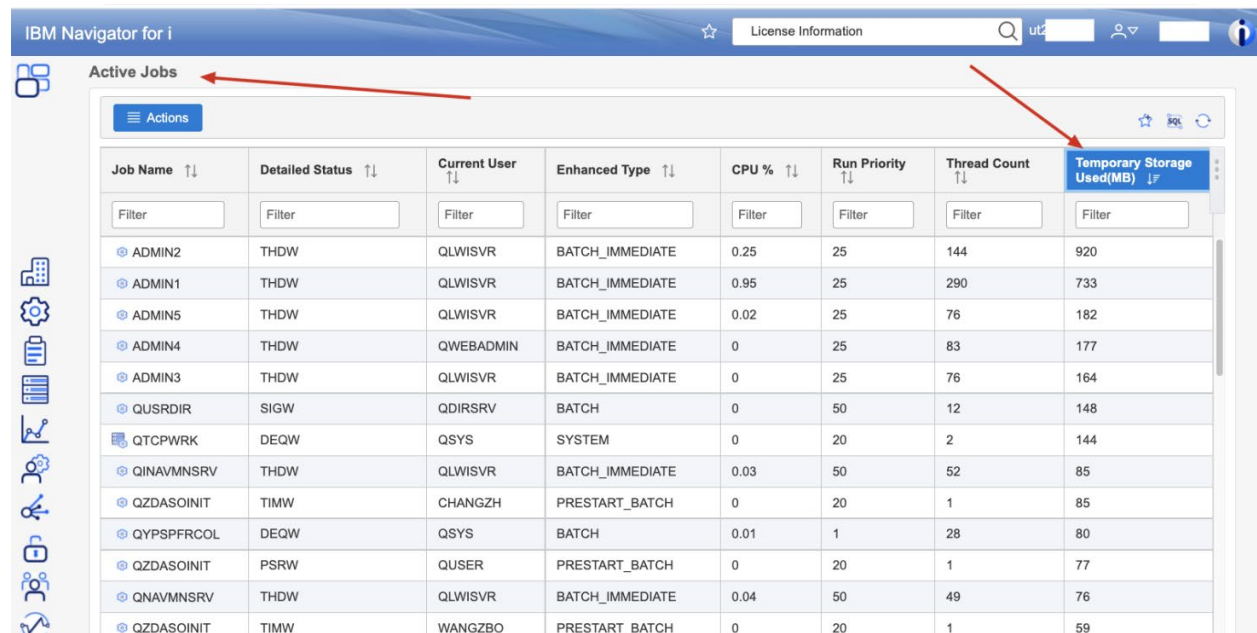
7.2.6 How can I find jobs consuming the most temporary storage?
Prior to IBM i 7.2, identifying top consumers of temporary storage was challenging and time-consuming. IBM i 7.2 introduced significant enhancements to streamline this process, particularly in how temporary storage usage is tracked across jobs.

Key Enhancements in IBM i 7.2:
- Temporary Storage Buckets: IBM i now categorizes temporary storage into "buckets," enabling a detailed view of storage consumption across jobs.
- Enhanced Visibility: You can now view temporary storage used by each job directly in Active Jobs in IBM Navigator for i, WRKACTJOB, or by using the System Temporary Storage Service (SYSTMPSTG).

In Navigator for i Active Jobs, by adding in this new column and sorting in a descending fashion, you can now see the top consumers of temporary storage:

Figure 10. Active jobs temporary storage used - Navigator for i

The System Temporary Storage Service provides a programmable interface to display which active jobs are the top consumers of temporary storage:

```
 1  -- category:  IBM i Services
 2  -- description:  Storage - Temporary storage consumption, by active jobs
 3  --
 4  -- Which active jobs are the top consumers of temporary storage?
 5  --
 6  SELECT bucket_current_size,
 7         bucket_peak_size,
 8         RTRIM(job_number) CONCAT '/' CONCAT RTRIM(job_user_name) CONCAT '/' CONCAT RTRIM(job_name)
 9           AS q_job_name
10    FROM QSYS2.SYSTMPSTG
11    WHERE job_status = '*ACTIVE'
12    ORDER BY bucket_current_size DESC;
```

BUCKET_CURRENT_SIZE	BUCKET_PEAK_SIZE	Q_JOB_NAME
696217600	805888000	032834/QLWISVR/ADMIN2
184573952	306057216	032835/QLWISVR/ADMIN3
183795712	336728064	032836/QLWISVR/ADMIN5
178180096	285876224	032833/QWEBADMIN/ADMIN4
154488832	154525696	032812/QDIRSRV/QUSRDIR
154148864	258433024	032832/QLWISVR/ADMIN1
151515136	151576576	032695/QSYS/QTCPWRK
83443712	97673216	032821/QYPSJSVR/QYPSJSVR
80506880	82591744	033738/QSYS/QYPSPFRCOL
74809344	94429184	032918/QLWISVR/QINAVMNSRV

Figure 11. System temporary storage consumption by active jobs - IBM i Services

In addition, the system temporary storage global buckets can be viewed in Navigator for i by selecting the **System** icon on the left panel and selecting **Temporary Storage** from the pop up menu.

Figure 12. Temporary storage used - Navigator for i

7.2.7 Which interface should I use for performance data visualization: PDI or iDoctor?

Like all performance-related questions, it depends. IBM i is an industry leader with its rich integrated Performance Data Collectors. To visualize all the various data collected in meaningful ways, IBM has two graphical user interfaces available, Performance Data Investigator (PDI) and iDoctor. Both GUI tools provide visualization over the various types of IBM i performance data.

In general, PDI is geared toward the average user and iDoctor is geared to the expert user who does performance diagnostics frequently. PDI is more limited in features and functions but is easier to learn and use. PDI is included with the operating system, and you can visualize Collection Services data with no additional products or cost. iDoctor has many advanced features and capabilities and tends to be a

more complex tool. However, for deep diagnostics, iDoctor can provide insights into the data where PDI is more limited.

A key advantage of PDI is that it is a task within Navigator for i. This means everyone has PDI and it has valuable integration with other IBM i management tasks. For example, you can launch PDI from active jobs, database tasks, system monitors, and graph history. iDoctor is a separate user interface that does not have this level of integration with navigator management tasks.

The following table summarizes the key differences between PDI and iDoctor to help you determine which interface is best for you.

PDI and iDoctor features		
Feature	PDI	iDoctor
User interface	Browser No client OS restrictions	Windows application
IBM i Performance Data Supported	Collection Services Job Watcher Disk Watcher Graph History	Collection Services Job Watcher Disk Watcher Performance Explorer Graph History
Update Frequency	Quarterly	Quarterly
Support	Normal product support	Email idoctor@us.ibm.com for support
National Language Support	Translated	English only
License	PDI is part of the base OS	Yearly license terms and charges

Table 1. PDI and iDoctor feature comparison

7.2.8 What is QDBFSTCCOL and why does it use resources?
QDBFSTCCOL is an IBM i system job responsible for collecting statistics for the SQL Query Optimizer. It is enabled by the QDBFSTCCOL system value, with the default value being *ALL. This job runs continuously at a low priority on your partition collecting valuable information on how best to implement your queries. In general, IBM does not recommend that you turn it off (a value of *NONE can degrade query performance). There are times when you may notice this job taking more CPU than usual. There are several reasons for this, including:
- A new file being introduced to the partition
- Existing file statistics are inaccurate due to records being added or changed
- Running a new set of queries on the partition

Viewing the job log for the QDBFSTCCOL job can provide clues on activity.

For more information, refer to Why is the QDBFSTCCOL Taking CPU? - IBM documentation.

7.2.9 How can I better isolate PASE jobs for analysis?
PASE jobs are commonly launched from SSH sessions, QSHell, or Qp2Term. This can result in many jobs having the same name, for instance QP0ZSPWP. This can make it more difficult to isolate analysis to a single set of workloads. To combat this, it is recommended to route jobs to relevant subsystems or to assign more descriptive job names with the PASE_FORK_JOBNAME environment variable.

7.3 Questions related to CPU
Following are the frequently asked questions related to CPU usage:

7.3.1 How can I reduce CPU usage on the partition?
Typically, this requires collecting and analyzing performance data from running jobs. Sometimes you can shift scheduled jobs to run during periods of lower CPU utilization. The best approach depends on which workloads or applications are dominating system performance.
Refer to the following sections for tuning specific application types:
- Tune SQL performance: SQL performance section
- Tune RPG/COBOL application performance: RPG/COBOL performance section
- Tune Java/WebSphere application performance: Java/WAS section

7.3.2 What can I do if I can't reduce my CPU consumption?
If CPU usage remains high after system tuning, perform the following steps:
- Change shared processor partitions to be uncapped and set virtual processors (VPs) higher than number of configured processing units.
- Don't configure more VPs than are needed for uncapping and workload demand (from spare capacity).
- Don't set the number of VPs higher than the number of licensed processors in the machine (or processors in the shared pool if dedicated processor partitions are not donating unused cycles).
- Don't exceed your software licenses.
- Add more processing capacity if system, database, and application tuning has not reduced CPU usage sufficiently.
- Use powerful performance tools to understand processor utilization and logical partitioning effects (e.g. CPU queuing, latencies, and shared processor behavior). The iDoctor tool is a valuable tool to understand these effects.

7.3.3 Why does my partition's CPU utilization show a value greater than 100%?
Normally a partition's CPU utilization ranges from 0% to 100%. There are several reasons why the CPU utilization would be reported to be greater than 100%:
1. If the partition is configured as shared and uncapped, the CPU utilization can exceed 100%. The amount the partition can exceed 100% depends on multiple settings including the number of physical and virtual processors, the size of the shared processor pool, and the partition's uncapped weight. This occurs when additional available capacity is allocated to the partition.
2. If performance data is refreshed rapidly, performance tools such as WRKACTJOB may not have enough time to gather all the required counter values, and the resulting CPU utilization may inaccurate. As a result, CPU can be incorrectly reported as more than 100%. To improve accuracy, allow some time before refreshing the performance counters.
3. The IBM i operating system provides "helper threads" to assist with functions such as database. If these helper threads consume considerable CPU, there may be intervals in which the partition's CPU utilization is reported as greater than 100%.

7.3.4 Why doesn't the system CPU utilization equal the sum of the CPU utilization of the individual jobs in WRKACTJOB?

The CPU utilization information for individual jobs is tracked differently than the overall system CPU utilization. Each process has counters associated with that process to track a wide array of statistics, including CPU. The system utilization comes from counters that do not keep track of each individual process. Each value should only be used to compare to itself - System CPU in one period to System CPU in another; Job CPU to either the CPU of another job or the job CPU of another period.

Keep in mind that IBM i system tasks are not included in `WRKACTJOB` command output. They are included in the `WRKSYSACT` command output.

7.3.5 How can I keep certain workloads from impacting my whole partition?

To reduce the impact of selected workloads from affecting the entire partition consider the following tips:

- Configure workload groups to limit the amount of CPU being used. For more information on workload groups, refer to Managing Workload Groups – IBM documentation.
- Reduce the priority of the jobs and increase the priority of other jobs.
- Assign jobs to a dedicated subsystem to control memory consumption, then set shared pool parameters to define maximum and minimum limits for each memory pool.
- Limit parallel processing for Db2 SMP by restricting certain jobs, especially long-running SQL queries, from using Db2 Symmetric Multiprocessing capabilities. Use the *PARALLEL_MIN_TIME* option in IBM i 7.5 to focus parallel processing on relevant queries. For more information, refer to New QAQQINI control PARALLEL_MIN_TIME – IBM documentation.
- Collect and analyze workload data using Collection Services, then review it through Performance Data Investigator or iDoctor for insights and optimizations.

7.3.6 The performance of my system is inconsistent. What could cause that?

Verify your system's energy management settings. IBM Power processor-based systems can be managed by an HMC. These tools allow users to implement power-capping or power-saving modes. When you use power-capping or power-savings modes and these technologies are in effect, the frequency of the processor may change.

For more information on EnergyScale, refer to the following documentation:

- IBM EnergyScale for POWER9 Processor-Based Systems
- IBM EnergyScale for Power10 Processor-Based Systems

Be aware of energy-saving settings that may slow down the processor, as this can impact performance. The scaled CPU time metric reveals whether energy-management features affected that partition.

7.3.7 Why do I see CPU queuing time when my CPU utilization is not excessive?

CPU queuing occurs when more jobs, threads, or tasks want to be dispatched to the CPUs in a partition than there are CPUs currently available to run on. Generally, we see CPU queuing time when CPU utilization is high, and it often shows that there is more work than what the partition processing resources can accommodate.

However, CPU queuing time can occur when CPU utilization is low for other reasons. One reason is that the workload is *bursty*. Each job, threads, or task may not use much processing time, but they are still all attempting to run at the same time, and thus some of them must wait for a CPU to become available, causing CPU queuing time.

Another reason CPU queuing can occur at low utilizations, is when the workload has many very short requests for CPU. This leads to threads constantly switching in and out of the processor. Every switch into the processor has at least some small amount of CPU queuing time, so if there is a high rate of switching in and out of the processor, this shows up as CPU queuing time. Collection Services reports the average CPU utilization for the entire collection interval, typically set to 5 or 15 minutes. This can mask short bursts of high CPU activity that result in CPU queuing. Using a smaller Collection Services interval, or collecting Job Watcher data, can allow you to better see bursts of high CPU.

Also, if Workload Groups are implemented, any dispatch latency time that may be introduced by limiting the work to a particular number of cores is reflected as CPU queuing time. Both PDI and iDoctor have graphs available to better understand the latency effects.

7.3.8 Why didn't my performance improve when I added cores to my partition?

Adding cores to a partition should reduce the overall CPU utilization but may not increase performance. There are two main reasons this can happen. The first case is when the workload is not CPU constrained. Meaning the jobs running are mostly waiting on I/O or some other resource rather than CPU. In this situation, the real bottleneck was not addressed. An analysis tool such as iDoctor or PDI can be used to identify the correct bottleneck.

The second case is when there are more cores than active jobs or threads. For example, if your batch process runs by using a single threaded job, it cannot take full advantage of more than one hardware thread in a single core. Changing the workload to use multiple jobs or threads allows more work to be completed in parallel, reducing the overall run time.

7.3.9 Why doesn't the CPU utilization reported by the HMC match the CPU utilization reported by IBM i?

The HMC and IBM i are measuring CPU utilization differently. IBM i uses CPU utilization to report how much compute capacity remains in the partition. For example, if the partition is 50% utilized, there is approximately half the compute capacity of the partition remaining. The HMC is measuring how much of a core's capacity remains for other partitions to use. When a core is dispatched to a partition, the HMC considers it to be 100% busy even if the partition only dispatches work to 1 of the 8 hardware threads on the core. In this example, IBM i would report the core to be around 30% busy vs. the HMC reporting it to be 100% busy.

The CPU utilization percentage is typically the ratio of utilized time to available time over an interval of interest, expressed as a percentage. The traditional IBM i CPU utilization that is reported by green screen utilities such as WRKSYSACT and WRKACTJOB is the ratio of the delta processor utilized time and the delta processor available time, for an interval. The processor utilized time originates from MATRMD Hex 26 and is known as SYSPTU in Collection Service's QAPMSYSTEM file. Similarly, processor available time originates from MATRMD Hex 26 and is known as SYSCTA in Collection Service's QAPMSYSTEM file. The traditional IBM i CPU utilization is `100 * SYSPTU / SYSCTA`.

Available IBM i performance data can be used to calculate an alternative CPU utilization percentage, such as that reported by the HMC. Non-idle processor virtual time originates from MATRMD Hex 26 and is known as SYTRUNVTB in Collection Service's QAPMSYSTEM file. The HMC view of a shared processor partition's processing entitled capacity utilization percentage can be calculated as `100 * SYTRUNVTB / SYSCTA`. This calculation is used by iDoctor to report the ENTUSED_HMC percentage for a shared processor partition.

For more information on differences in CPU reporting, Refer to [Understanding Processor Utilization on PowerVM](#) - IBM PowerVM blog post.

7.4 Questions related to memory

7.4.1 How much memory should I configure for my partition?
The minimum partition memory size is 2GB. However, the recommended minimum partition memory size is 4GB. For more information, refer to [Miscellaneous Limits](#) – IBM documentation.

7.4.2 How should I configure my partition's memory pools?
Every partition is different, but here are some general guidelines:
- In general, enable Expert Cache on most user pools (i.e. *CALC). For more information, refer to [Expert Cache - How It Works](#) – IBM documentation.
- Add memory to the machine pool if the fault rate of that pool is more than 10 faults per second and the amount of memory used in the pool is high.
- Set the maximum activity level high enough to avoid transitions to ineligible, but not higher than necessary.

7.4.3 Should I turn on the Performance Adjuster (QPFRADJ)?
Most customers can benefit from using the performance adjuster. The following experience report is helpful for understanding QPFRADJ: [Performance Adjuster](#).
Make sure you configure the shared pool settings properly to limit how large (and small) each memory pool can become. If you chose not to use QPFRADJ, make sure the max activity level is high enough to avoid transitions to ineligible. Java and Open Source applications have special considerations when it comes to QPFRADJ. For more details, refer to [Java chapter](#) of this white paper.

7.4.4 How can I tell what is in my partition's memory pools?
The `DMPMEMINF` command can provide the necessary information. For more information, refer ti [Dump Main Memory Information (DMPMEMINF)](#) – IBM documentation.

7.4.5 When should I add more memory to my partition?
As a general guideline you should limit the amount of time spent waiting on disk faults to 25% of the runtime of an average job on the system. If application optimization cannot reduce jobs below this level, consider adding more memory. Also, check the performance of the I/O subsystem to ensure disk requests are being completed with good response times.

7.4.6 What are the *memory page fault guidelines*?
With the wide variety of applications that now run on IBM i it is not possible to come up with guidelines that would satisfy every scenario. As a result, IBM no longer publishes general faulting guidelines for user pools.

7.5 Questions related to I/O operations
Many of the approaches that reduce CPU usage can also reduce the number of I/O operations, especially those related to improving database and SQL performance. There are also opportunities to improve the performance of the I/O subsystem.

7.5.1 What are some common storage subsystem configuration changes that can hurt performance?
Here are key actions to avoid storage subsystem configuration changes that can hurt performance:
- Mix drive sizes carefully to avoid inconsistent performance in the same ASP.

- Balance units across IOAs to prevent uneven workloads on specific IOAs.
- Limit the number of units assigned to a single IOA.
- Use paired (dual) SAS IOAs with separate parity sets instead of relying on a single parity set.
- Maintain consistent drive sizes and stop protection before adding new drives to existing arrays. Restart protection to ensure all drives include parity data and avoid performance issues.
- Select the *Add and Balance* option when adding new drives to an ASP to evenly distribute workloads.
- Retain the number of drives during storage migration to sustain performance levels.
- Configure high availability replication solutions with adequately sized links between source and target storage systems.
- Assess the potential performance impact of introducing Real-time Compression on systems like SVC, Storwize, or FlashSystem V840/V9000. For more information, refer to chapter 7.6 of Introducing and Implementing IBM FlashSystem V9000 – IBM Redbooks.
- Choose dedicated storage options over Virtual I/O Server (VIOS) to achieve faster performance and enhanced reliability.

7.5.2 How can I analyze I/O subsystem performance?
- Use Collection Services and Disk Watcher to collect and analyze disk performance.
- Use the SQL Plan Cache and Visual Explain to identify SQL causing many I/O operations.

Refer to RPG/COBOL and Native I/O section for more tips.

7.5.3 I can't modify my applications. What can I do to improve my I/O performance?
- Install more memory or disk drives.
- Install SAS Solid State Drives (SSDs) or NVMe devices and move hot objects to them.
- Use the Workload Estimator or contact an IBM business partner or sales representative to size for a storage solution that delivers faster performance.

7.5.4 How do I virtualize IBM Serial-Attached SCSI (SAS) adapters for the best performance?
For the best performance, consider using IBM i as the storage host rather than using a VIOS.

For more information on special configuration considerations, refer to section 3.9 titled VIOS vSCSI disks and IBM i client partitions of IBM Power Systems RAID Solutions Introduction and Technical Overview – IBM Redbooks.

For more information on requirements and setup of the performance boost, refer to SAS Adapter Performance Boost with VIOS – IBM documentation.

7.5.5 How many and what type of storage I/O products meet performance requirements for an upgrade or new system or workload?
To determine the number and type of storage I/O products that meet performance requirements for a system upgrade or a new workload, refer the following resources for insights on performance management, documented comparisons, and tuning recommendations:

- SAP on IBM i Solid State Disk (SSD) Usage Recommendations: Guidance on optimizing SAP workloads with IBM i and SSD.
- IBM Power Systems with IBM i using Solid State Drives to boost your Oracle's JD Edwards EnterpriseOne performance: Information on using SSDs to enhance performance for Oracle JD Edwards EnterpriseOne.

- [IBM Storage](#): Information in detailed specifications and supported SAN storage products
- Appendix A of [IBM FlashSystem Best Practices and Performance Guidelines](#): Information on performance guidelines for IBM FlashSystem.
- For information on direct attached storage products:
    - [Performance Capabilities Reference – February 2013](#): Information on previous performance capabilities, review the Performance Capabilities Reference before April 2014 version.
    - [Performance Study of 2nd Generation IBM Power Systems SAS RAID Adapters Designed for Solid State Storage](#) : Information on recent SAS Adapter Technology Comparison

7.5.6 How can I tell if a FlashSystem solution is the best storage configuration for my environment?

There are many websites that explain the performance, functionality, and cost advantages of an IBM FlashSystem over both HD and SSD. For more information, refer to [IBM Storage FlashSystem](#).

When moving to a FlashSystem solution, certain factors can prevent expected performance improvements, such as average read response times reaching as low as 0.5 milliseconds per I/O. Common issues include:

- Resource constraints or wait conditions: Other limitations within the IBM Power processor-based system configuration can mask improvements, unrelated to the I/O subsystem. These constraints can cause faster FlashSystem response times to push additional work, potentially worsening overall performance due to increased thrashing.
- Write-intensive workloads: Although Flash offers good write response times, it cannot match the speed of writing to cache in an I/O adapter (IOA). Internal drives configured with large write caches in IOAs handle most write IOPs asynchronously or near-asynchronously, resulting in faster apparent response times than Flash storage.

These conditions can result in slower performance despite the upgrade to Flash, particularly if the system isn't optimized to address these limitations.

To help customers identify situations that could prevent the unacceptable response times, and also to help project the run time improvements of the most I/O intensive jobs, IBM Technology Expert Labs offers an *IBM i Workload Assessment for FlashSystem Solution*. For more information, complete this form: [IBM Technology Expert Labs Contact Form](#)

Also, note that the SSIC (System Storage Interoperability Center) states the following regarding 4096 sector size used for natively attached FlashSystems. Also reference section 8.5.9 regarding the performance implications of 4096 sector drives on IBM i.

Certain unique workloads show lower than expected performance for IBM i hosts when attached to IBM FlashSystem 840/900 storage without SAN Volume Controller (SVC). It is recommended these opportunities be vetted with a PoC to ensure the clients' workload does not encounter this performance issue. In addition, IBM i is maintaining a list of [recommended PTFs](#) for any clients running in a 4096 sector size configuration which is required with an attached FlashSystem 840/900.

7.5.7 How can I tune my Virtual Fibre Channel attached external storage?

Following is a list of recommendations that can be used to ensure vFC/NPIV environments perform as well as possible when connected to supported External Storage Systems:

- [IBM i Virtual Fibre Channel Performance Best Practices](#): Information on how to take greater advantage of the high command throughput rates of vFC attached solid state storage, 7.5 TR1, 7.4 TR6 and 7.3 TR12 utilize a multiple queue (MQ) feature that was added to VIOS version 3.1.2.
- [Modernization of PowerVM NPIV stack: from Single to Multiple Queue Support (ibm.com)](#): provides a description of MQ and some impacts observed with AIX as a client partition. The increased parallelism MQ provides to an IBM i partition can increase command throughput for workloads that need high io/sec and have relatively short (<16KB/io) average storage io lengths.

7.5.8 How do I analyze I/O performance if I'm using external storage?

There are many components to consider when external storage is being used. Latency can be added in the virtualization layer (VIOS and Power Hypervisor) and SAN which are located between the IBM i and storage. A good strategy is to compare the IBM i drive service times with the response times measured on the storage. If the service times and response times are close to the same, then the problem is likely in the storage. If the IBM i service time is higher than the storage, then latency is likely being added in the SAN or virtualization layer. Collection Services can be used to measure the service times on the IBM i. There are several ways to measure the response times on the storage depending on what is being used.

- IBM iDoctor for i – Collection Services Investigator can be used to analyze data that is automatically collected from attached DS8000 Storage systems. More information, refer to [iDoctor Documentation](#).
- IBM Storage Insights provides an unparalleled level of visibility across your storage environment to help you manage complex storage infrastructures and make cost-saving decisions. Benefits include enhanced performance monitoring, capacity planning and capability to troubleshoot issues faster. Two editions are available, including a no charge offer to monitor the basic health of your storage. For more information, refer to [IBM Storage Insights](#).
- The Storage Monitoring feature in IBM Spectrum Control (formerly known as IBM Tivoli Storage Productivity Centre (TPC)) can be used to analyze performance of most IBM Storage Systems. For more information, refer to [IBM Spectrum Control](#).

7.5.9 What are the IBM i performance effects of using 4096 byte sector drives?

There are options for 4096 byte formatted internal HDDs and SSDs both natively and virtualized through VSCSI from VIOS or an iHost. Flash Systems external storage is also formatted for 4096 byte sectors. Ensure the latest relevant PTFs are applied before implementation. For information on IBM i 7.2 and 7.3 list, refer to [PTF listing for 4096 disk sector support](#).

If implementing 4096 byte sectors, there is additional overhead associated with storage management cleanup tasks. To lessen the impact, review the workload for inefficient application issues, specifically high rates of database full opens and high rates of activation group creates and destroys. These are discussed in section 8.16. Consider engaging IBM Technology Expert Labs for assistance in reducing open rates before implementing 4096 byte sectors. For more information on SMFSCLEAN* tasks, refer to [SMFSCLEAN* tasks and machine level gating in SMFREESPACEMAP4K](#)

7.5.10 Should I utilize NVMe drives?

IBM i 7.5, 7.4, 7.3 and 7.2 various Technology Refreshes support NVMe devices that are available in multiple capacity points, multiple form factors, with various speeds, for enterprise workloads on selected Power servers with Power technology. The [IBM i I/O Support Summary](#) describes what releases support which devices and under what configurations (native, VIOS and iVirt). These low latency devices

can be used as IBM i load sources and are able to provide a high number of io/s and enhanced virtualization capabilities.

The PCIe gen4 devices can also provide a significant amount of data throughput (GB/s). Mirroring is required. Certain Power Servers can include only Add-In-Card (AIC) form factor NVMe devices while some Power Servers can accept both AIC and U.2 form factor devices.

NVMe devices can be virtualized into multiple namespaces. Each namespace is treated as a storage device by Storage Management. It is recommended to allocate as many namespaces as SAS or FC LUNs deemed to be sufficient to meet application throughput requirements. For more information on how to configure and create namespaces, refer to [NVMe - IBM Documentation](#).

If more space, or more throughput, is required from NVMe devices than can be installed within CECs slots, a NED24 NVMe Expansion Drawer (feature ESR0) that fits into the identical rack space as the latest SAS remote I/O enclosures can be used to house up to 24 U.2 NVMe SSDs. Each ESR0 can provide up to 10x more data throughput (GB/s) than a comparable ESLS SAS drawer and up to 3-4X more command throughput (io/s).

7.6 Questions related to processor virtualization

7.6.1 How do I determine if my workload is being impacted by processing capacity entitlement delay?
A shared processor LPAR is configured with a number of virtual processors and a processor capacity entitlement. When executing instructions, a virtual processor is assigned a physical processor of the shared processor pool, and processing capacity usage is charged to the LPAR at a rate of 1.0 unit of capacity per elapsed time. PowerVM meters the processing capacity usage of a LPAR's virtual processors over a 10 millisecond window, so that over the long term, the partition is guaranteed access to its entitled capacity. When a capped LPAR has consumed its capacity entitlement within a window, its virtual processors are preempted and queued until the next window, at which time the capacity entitlement is replenished. If an uncapped LPAR exhausts entitlement, it may continue to claim a share of unused capacity from the shared processor pool, so its entitlement delay can be reduced. Field SYPTREADY in Collection Services file QAPMSYSTEM records the entitlement delay for a shared processor LPAR, and field JBVPDLY in Collection Services file QAPMJOBMI records the combined entitlement and virtual processor dispatch delays at the job-level.

The following questions demonstrate how to use this data to answer entitlement delay questions using iDoctor. Virtual processor delay time is also available in Performance Data Investigator in the perspective CPU utilization & waits overview with JBVPDLY. The active time on the view is dispatched CPU minus virtual processor delay time.

7.6.2 How can processing capacity entitlement delays be reduced?
Partition-level processing capacity entitlement delays are indicated by field `SYPTREADY` in Collection Services file `QAPMSYSTEM`, and job-level virtual processor delays are indicated by field JBVPDLY in Collection Services file `QAPMJOBMI`. Entitlement delays can be reduced by increasing the ratio of processing capacity units to virtual processors in the LPAR configuration. This is typically accomplished by increasing the processing capacity units or by reducing the virtual processors, or both.

Figures 13 and 14 illustrate the relationship between entitlement delay and the ratio of processing capacity units to virtual processors in a LPAR configuration. In this example, the workload primarily consists of a single-threaded CPU-intensive SAVLIB operation, with the LPAR configuration being 1

processor, 0.1 processor units (capped) until 15:41:00, when the processor units were increased to 1.0. In this example, notice that the Entitlement Used from HMC viewpoint (%) indicates that the LPAR is consuming its entire entitlement both before and after the change, and that the entitlement delay (CPU thread ready wait time) at partition-level and processor virtualization delay (Dispatched CPU wait) at job-level are mostly eliminated by the change. This example also illustrates how processing capacity entitlement can be entirely consumed despite the partition CPU utilization being much lower, owing the relative inactivity of the processor's other SMT threads.
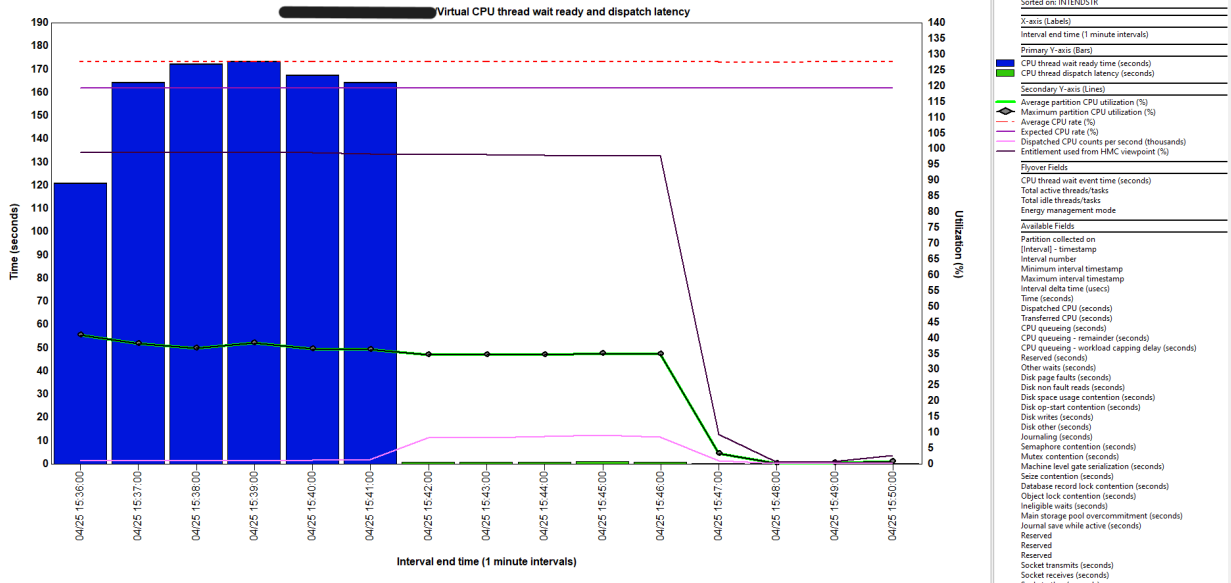
Figure 13. Partition-level illustration of entitlement delay reduction - IBM iDoctor for i

Figure 14. Job-level illustration of processor virtualization delay reduction - IBM iDoctor for i

If a shared processor LPAR is capped or soft-capped (uncapped with weight 0), then changing it to uncapped with non-zero weight can also reduce entitlement delays, with the extent of relief being dependent on the availability of unused capacity in the shared processor pool.

7.6.3 How do I determine if my workload is being impacted by over-commit of the shared processor pool?

When actively executing instructions, a virtual processor is assigned a physical processor of the shared processor pool, and all threads of the processor are reserved for exclusive use by the LPAR to which it is assigned. When the number of virtual processors demanded (and not waiting for entitlement) by the shared processor pool's LPARs exceeds the number of physical processors assigned to the pool, then some virtual processors will temporarily experience virtual processor dispatch latency. Field SYPTLATEN in Collection Services file QAPMSYSTEM records the dispatch latency delays for a shared processor LPAR, and field JBVPDLY in Collection Services file QAPMJOBMI records the combined entitlement and virtual processor dispatch delays at the job-level

7.6.4 How can virtual processor dispatch latency delays be reduced?

This is a challenging question from the LPAR perspective because numerous factors impact virtual processor dispatch latency, and in many cases, these factors are beyond the control of a single LPAR. Specifically, the pattern of virtual processor demands across *all* the shared processor pool's LPARs determines the virtual processor dispatch latency. The short answer is to reduce the ratio of virtual processors to physical processors of the shared processor pool. This can be accomplished by reducing the number of virtual processors, or by allocating additional capacity to the shared processor pool, or by configuring dedicated processor LPARs to temporarily donate their inactive or idle processors to the shared processor pool. The ratio can also be optimized by observing best practices regarding the configuration of shared processor LPARs, but this discipline needs to be employed across all LPARs for best results. OS-level scheduling behavior and features such as processor folding can impact a LPAR's efficiency with regard to virtual processor demands. In some cases, the problem may require higher-level coordination across LPARs, such as sequencing peak-usage periods.

Figures 15 and 16 illustrate the complex relationship between processor virtualization delay from a single LPAR's perspective and *other* activity in the LPAR's shared processor pool. The shared processor pool has 22 processors initially, with 2 shared processor LPARs of 22 processors, 11 units (uncapped) configured. The LPAR under consideration (LPAR 1) is running COPR, which is the OLTP workload used to establish CPW ratings of IBM Power system models running the IBM i operating system.

The approximate timeline of actions is as follows:
- 14:50 – 15:00 : LPAR 1 is running a COPR workload. LPAR 2 is idle.
  During this period, LPAR 1 benefits greatly from idle capacity in the shared processor pool, driving average partition CPU utilization to 180% while incurring minimal processor virtualization delays.
- 15:00 – 15:20 : LPAR 1 and LPAR 2 are running a COPR workload.
  With LPAR 2 also running a COPR workload, the processor pool is heavily over-committed, with 44 virtual processors sharing 22 physical processors. Processor virtualization delays at the system-level and job-level are significantly impacting workload performance in both LPARs.
- 15:20 : Processor pool expands by 11 processors from 22 to 33.
  A dedicated processor LPAR has deactivated 11 of its processors, causing the 11 processors to be available to the partitions of the shared processor pool. The additional processors has reduced the degree of over-commitment, with 44 virtual processors sharing 33 physical processors. The additional processing capacity is used by the uncapped LPARs, resulting in a reduction of system-level and job-level virtualization delays.
- 15:40 : Processor pool expands by 10 processors from 33 to 43.

Additional processors were made available to the shared processor pool, to the point where the pool is barely over-committed with 44 virtual processors sharing 43 physical processors. Workload performance in LPAR 1 has nearly returned to the level of interval 14:50 – 15:00, where LPAR 2 was idle.
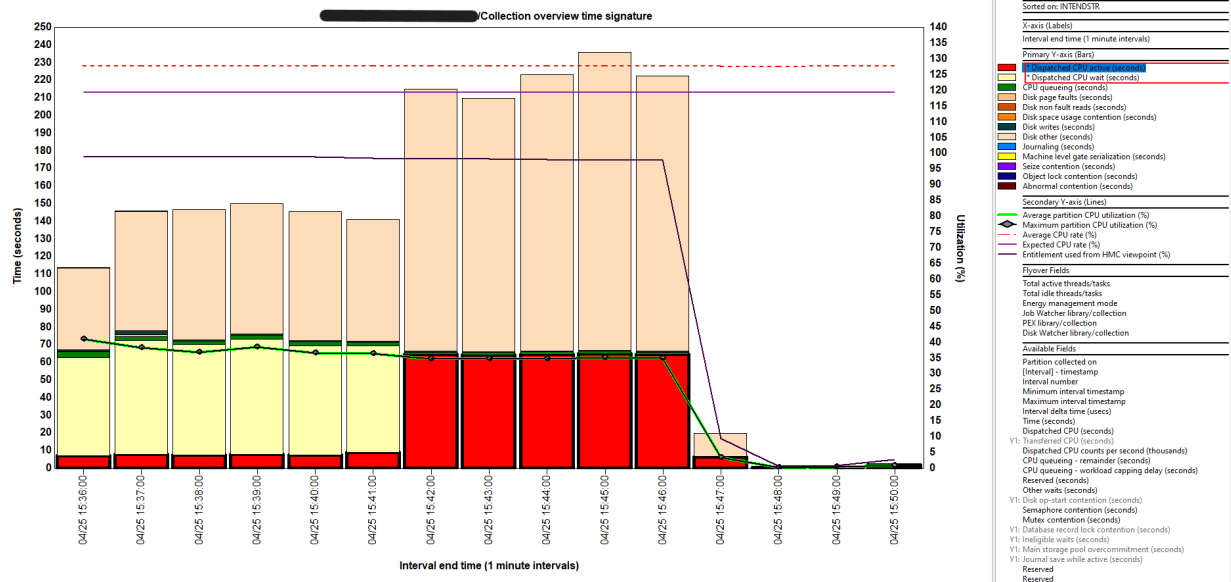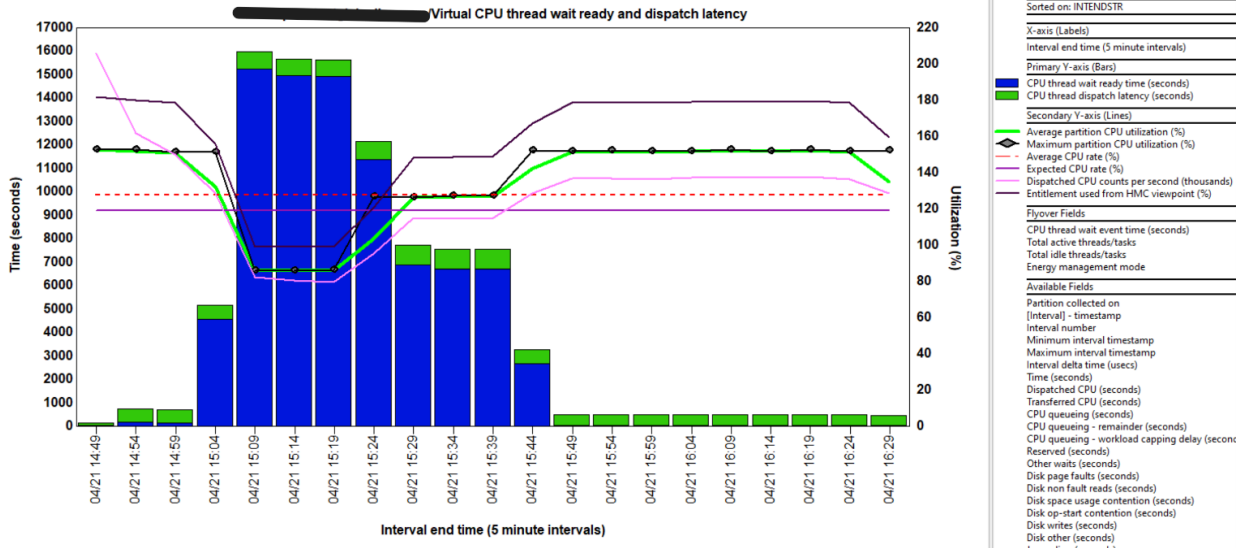

Figure 15. Partition-level illustration of processor virtualization delay - IBM iDoctor for i
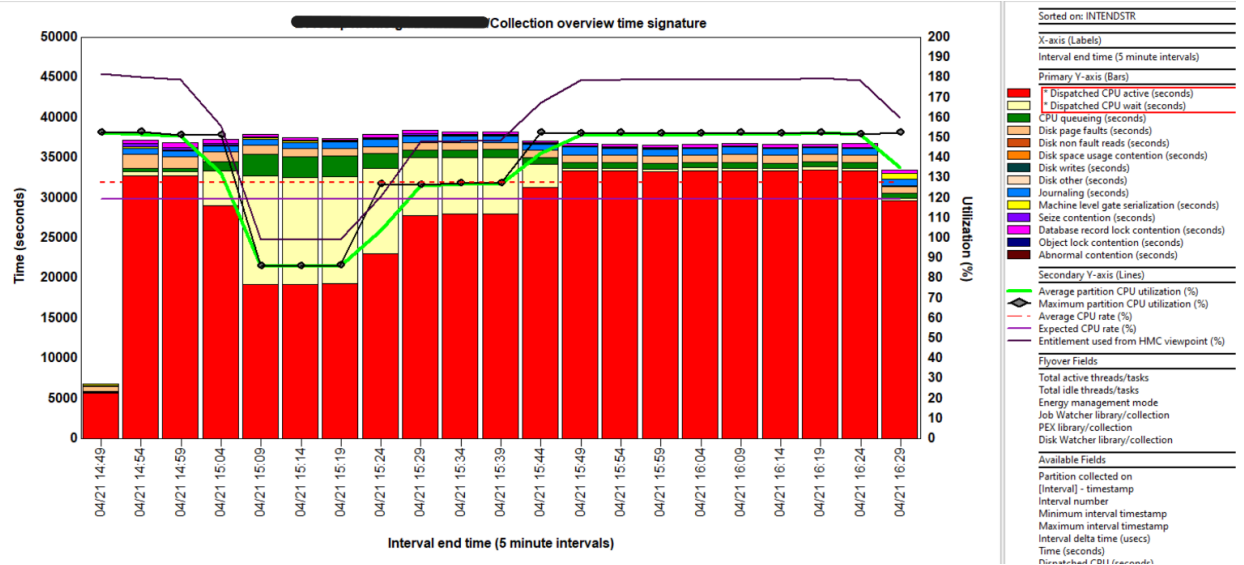
Figure 16. Job-level illustration of processor virtualization delay - IBM iDoctor for i

7.7 How do I tune IBM i database performance?
Refer to chapter 8 for details on database performance tuning.

7.8 How can I improve backup and recovery times?
Refer to the following information to understand how to improve backup and recovery performance:
Backup and recovery frequently asked questions

For improved IFS SAV performance information:

[Achieve faster IFS save times by using SAV with ASYNCBRING](#)

7.9 How can I improve IPL time?

Although IPL duration is highly dependent on hardware and software configuration, there are tasks that can be performed to reduce the amount of time required for the system to perform an IPL. The [Experience Report: Reducing System IPL Time](#) report contains many suggestions for reducing IPL time.

Following are some more recent suggestions to consider are:

- IBM i 7.3 includes a design change to the IPL process to help reduce IPL time for partitions with very large amounts of memory. This improvement is not available for earlier releases.
- The performance of the I/O subsystem can be an important component of IPL time. Customers who upgraded their storage from HDDs to SSDs or Flash storage saw substantial improvements to IPL time. However, not all systems will see improved IPL times through I/O upgrades due to other factors.
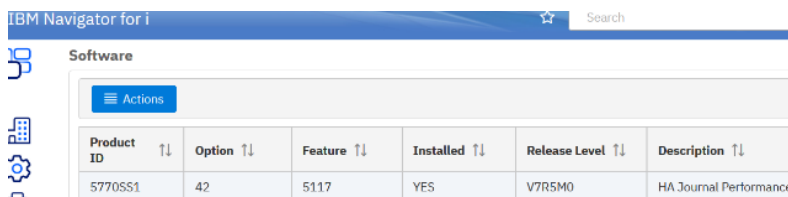- Internal measurements show that moving only the load source from an HDD to SDD does not significantly improve IPL time.

7.10 What are considerations for Main Store Dump (MSD) IPL performance?

The [Best Practices for Managing Time Needed for Main Storage Dump (MSD)](#) Technote has the latest recommendations, enhancements, and PTFs to help manage and reduce time required for MSD.

7.11 How do I tune journal and recovery performance?

- For IBM i releases before 7.1 make sure that the journal threshold for all user journals is at least 6.4GB. As of IBM i 7.1 this is no longer necessary.
  The journal receiver size options (RCVSIZOPT) should preferably specify at least MAXOPT2 and RMVINTENT.
  For more information on these commands, refer to [Journal Management Guide.](#)
- Consider enabling IBM i Journal Cache. Journal caching is a feature that, per journal, allows the system to bundle or cache journal entries in main storage before writing them to disk. Journal caching modifies the behavior of traditional non-cached journaling in batch. Without journal caching, a batch job waits for each new journal entry to be written to disk. Journal caching allows most operations to no longer be held up waiting for synchronous disk writes to the journal receiver. Journal Caching (HA Journal Performance) is Option 42 of the IBM i operating system.

IBM Navigator for i	☆	Search			
Software					
≡ Actions					
Product ID ↑↓	**Option** ↑↓	**Feature** ↑↓	**Installed** ↑↓	**Release Level** ↑↓	**Description** ↑↓
5770SS1	42	5117	YES	V7R5M0	HA Journal Performance

Journal caching was previously a chargeable feature, but as of June 2022, it is no-charge option. It is distributed with IBM i 7.5 but will have to be downloaded and installed on versions 7.4 and 7.3.

Performance results of using Journal Cache will vary. Batch applications which perform large numbers of changes to the data portion of the journaled objects often see the most performance benefit of journal caching. Applications using commitment control will see less improvement as commitment control already performs some journal caching.

You can leverage the power of IBM i Wait Accounting technology in Collection Services (and Job Watcher) to get an idea whether Journal wait time is an issue. The Journal wait time bucket can be viewed at the partition level, as well as an individual job level.   Use the wait graphs in either Navigator PDI or iDoctor to view the data.

For additional usage information and considerations, refer to following IBM documentations:
- o   IBM i
- o   Basics on Journal Cache

7.12 How do I tune access path recovery performance?
System-managed access path protection (SMAPP) offers system monitoring of potential access path rebuild time and automatically starts and stops journaling of system selected access paths. In the unlikely event of an abnormal IPL, this allows for faster access path recovery time.

An estimation of how long access path recovery takes is provided by SMAPP, and SMAPP provides a setting for the acceptable length of recovery. For most customers, the default value minimizes the performance impact, while at the same time provides a reasonable and predictable recovery time.   But the overhead of SMAPP varies from system to system and application to application.

As the target access path recovery time is lowered, the performance impact from SMAPP may increase as the SMAPP background tasks have to work harder to meet this target. There is a balance of recovery time requirements vs. the system resources required by SMAPP.

Although SMAPP may start journaling access paths, for performance-sensitive workloads, it is recommended that the most important/large/critical/performance-sensitive access paths be journaled explicitly with `STRJRNAP`. This eliminates any unexpected overhead of SMAPP evaluating these access paths and implicitly starting journaling at an undesirable time. This overhead can include significant seize contention while large/active access paths are written to disk when starting journaling. The `SMAPP_ACCESS_PATHS` view can be used to identify which access paths should be explicitly journaled, or IBM Technology Expert Labs can be engaged for assistance.

There are 3 sets of tasks which do the SMAPP work. These tasks work in the background but can impact user jobs while performing their required functions:
- JO-EVALUATE-TSKn (where n is 0 or 1): Evaluates indexes, estimates rebuild time for an index and may start or stop implicit journaling of an index.
- JO-TUNING-TASK: Periodically wakes up to consider where the user recovery threshold is set and manages which indexes should be implicitly journaled.
- JO-RECRA-Dx-xx-x and JORECRA-USR-xxx tasks are the worker tasks which sweep aged journal pages from main memory to minimize the amount of recovery needed during IPL.

Here are guidelines for reducing the amount of work for each of these tasks:
- If the JO-TUNING-TASK or JO-EVALUATE tasks are busy, consider increasing SMAPP recovery target time or explicitly journaling additional access paths.
- If the JORECRA tasks seem busy, consider increasing the journal recovery ratio.
- Also, if the target recovery time is not being met there may be SMAPP ineligible access paths. These should be modified to become SMAPP eligible. You can use the Display Recovery for Access Paths (DSPRCYAP) command to see a list of access paths that are not eligible for SMAPP.

To monitor the performance impacts of SMAPP there is a substantial set of Collection Services counters which provide information on the SMAPP work.

For more information on reducing access path recovery time, refer to [Reducing time in access path recovery](#) – IBM documentation.

7.13 How do I tune network performance?

The network adapters being used determine the maximum speed which could be reached, e.g. 1Gb, 10Gb, and so on. (at least in theory). However, the protocol being used, the networking parameters set and the quality of the connections as well as other systems in the same subnet in the network determines the actual performance in terms of network throughput and speed. Performance Collection Services collect vital data from the use of networking resources and can be used to determine the actual speed and throughput numbers of the network.

- Consider increasing the TCP/IP buffer for send and receive operations by the `CHGTCPA` command to a value greater than the default of 64KB. This can significantly reduce the network traffic.
- Ensure the parameters for current line speed reflect what the adapter is being capable of, e.g. 1Gb, 10Gb, and so on. A single device with a lower line speed capability will force the whole subnet in the network to run at the lower speed and can severely degrade network performance.
- Ensure the current DUPLEX parameter is set to *FULL so the connection can be used for send and receive at the same time.
- Consider increasing the maximum frame size from 1496 bytes to 8996 bytes as this can significantly reduce the traffic between the system and the next network router and speed up the connections especially when virtual Ethernet connections are used.
- Consider using multiple network connections together and use link aggregation to bundle as shown in the following diagram for example four 1Gb connections into one 4Gb link. This aggregated link provides a significant bandwidth improvement.

For more information, refer to [Ethernet Link Aggregation](#) – IBM documentation.
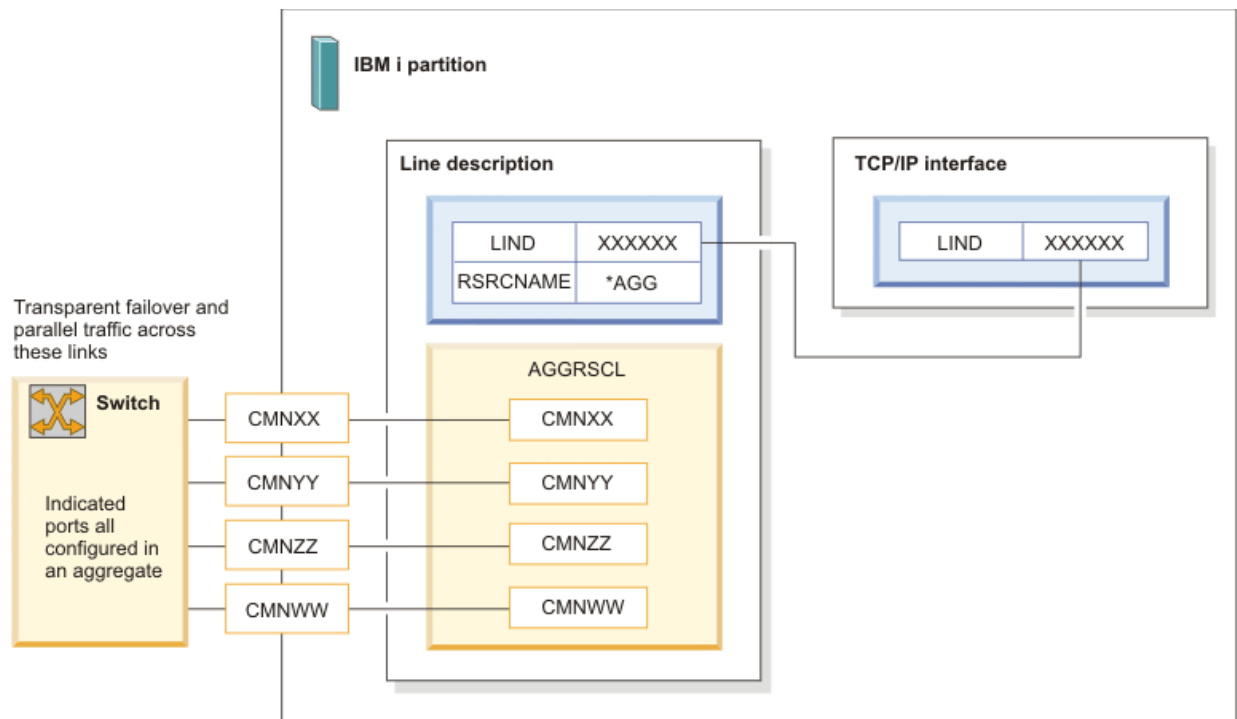
Figure 17. Network link aggregation

More information on optimizing communications performance, refer to Optimizing communications performance – IBM documentation.

7.14 How can I improve the performance of Java and/or WebSphere Application Server applications?

For more information, refer to Java and WebSphere chapter.

7.15How do I tune RPG/COBOL application performance and native I/O file access?

For more information, refer to RPG/COBOL and native I/O chapter.

7.16 Can security settings affect performance?

Protecting information is an important part of most applications. Security should be considered, along with other requirements, at the time the application is designed. When security is integrated properly into application design, it will result in the best performance. It is when security is not implemented correctly that performance problems typically occur.

Defining many private authorities can negatively affect performance. The following links to IBM Docs show how authority checking is done. A properly architected solution will balance security concerns with performance.

Flowchart 1: Main authority checking process
Flowchart 2: Fast path for object authority checking

When implemented correctly, the application fast path will generally be taken through program adoption of the owner. When public and private authorities are added, complications can arise, impacting performance.

Collection Services includes two metrics that can be useful for monitoring authority activity. The metrics are `SYAUTH` and `SYNUAL` in the file `QAPMSYSTEM`. `SYAUTH` counts the number of object authority checks system wide. `SYNUAL` counts the number of non-cached user authority lookups.

7.17 What are some inefficient application issues I can identify at a system level?

7.17.1 Look for high rates of database full opens

Having a high rate of database file full opens can lead to poor performance. Full opens can come from SQL or programs using native I/O. Collection Services collects the number of database file full opens at both a system level and a thread level. This data can be visualized using either PDI or iDoctor.
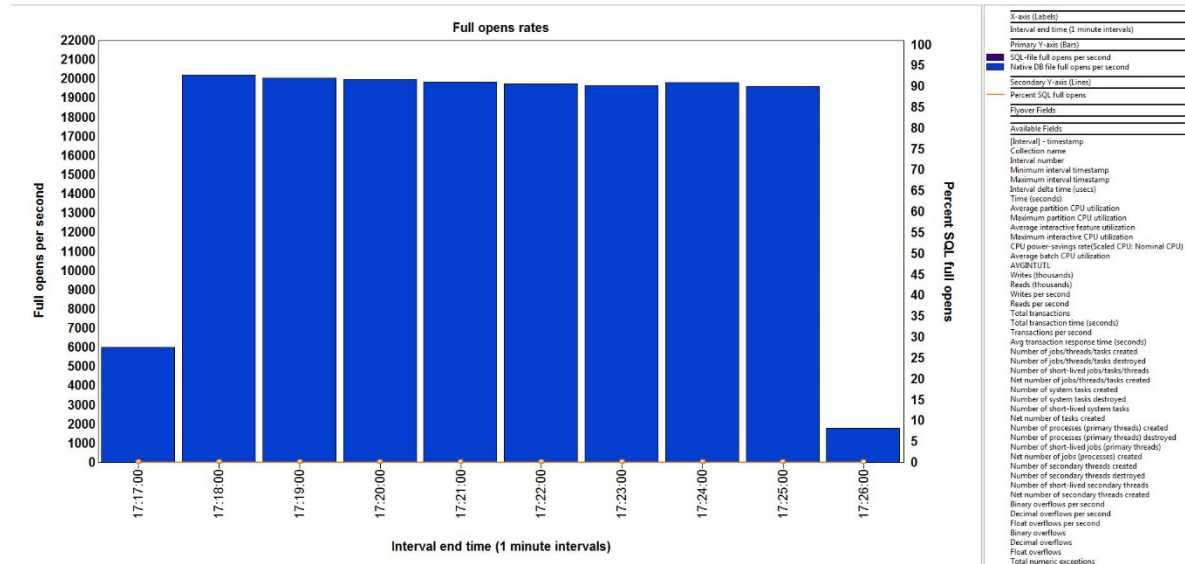

Figure 18. Full Opens in iDoctor Collection Services Investigator

To identify the programs causing the full opens, a PEX trace can be collected using the *DBOPEN OS event. The PEX trace can be analyzed using the PEX Analyzer component of iDoctor

A high rate of database file full opens is a common cause for poor performance on IBM i partitions. Full file opens are heavy-weight operations that should be minimized. It is not uncommon for jobs with high full open rates to be wasting one third or more of their CPU time doing full file opens, when they typically don't need to.

A general guideline for partition wide full open rates is less than 1000 per second on small to medium sized partitions, and low single digit thousands per second on large partitions. Additionally, keep the full open rate to well under one hundred opens per second for any single job. Regardless of the size of the partition, the lower the number the better for performance and scalability.

These full open guidelines are independent of disk storage technology.

7.17.2 Look for high rates of activation group creates/destroys

Having a high rate of activation group creations and destroys is also inefficient. Starting with IBM i 7.2, Collection Services collects the number of groups and programs activated at a system and thread level. This data can be visualized using iDoctor.
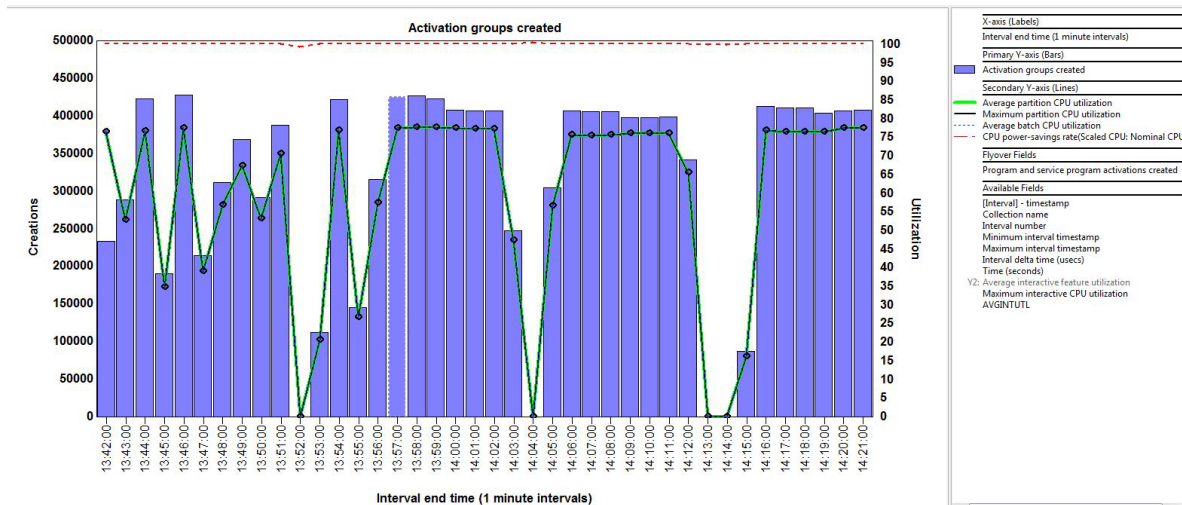
Figure 19. Activation Groups Created in iDoctor Collection Services Investigator

To identify the programs causing the activation group creates, a PEX trace can be collected using the *ACTGRPCRT Base event. The PEX trace can be analyzed using the PEX Analyzer component of iDoctor.

7.18 What is the largest number of cores that can be assigned to an IBM i partition?
There are two partition core limits for IBM i. The first is the standard limit which is 48 cores for all supported IBM i releases. Customers who need to exceed this limit can go through a no-charge assessment process to validate that their partitions will benefit from having additional cores. The maximum number of cores allowed depends on the level of IBM i and Power hardware, as well as the results of the assessment.

Refer to the following matrix showing the maximum single-partition core limits. For more information, contact Eric Barsness (ericbar@us.ibm.com) or Mike Gordon (mgordo@us.ibm.com).

OS level	Max Allowed w/Assessment		
	Power8	Power9	Power10
IBM i 7.2	96	96	96
IBM i 7.3	96 SMT8 192 SMT4	96 SMT8 192 SMT4	96 SMT8 192 SMT4
IBM i 7.4	96 SMT8 192 SMT4	192	192
IBM i 7.5	-	192	240

Figure 20. Maximum cores in partition

7.19 How can I improve the scalability of my large IBM i partition?
Something that is done inefficiently but infrequently may not be a problem. If that activity is done more and more frequently it can become a major problem. It is not uncommon for a program written 10 years ago to access a small file accessed by a few users be an issue today because that file has grown to millions of records and is being accessed by hundreds of users. Avoid these application *worst practices*:
- Issuing a high rate of full file opens and closes
- Creating and destroying activation groups at a high rate
- Frequently destroying large activation groups
- Causing a high rate of job initiations and terminations

The rate metrics for these scaling issues are collected by Collection Services and can be viewed using either Performance Data Investigator (PDI) or iDoctor Collection Services Investigator (CSI). See section 7.17 for more information on some of these metrics.

In addition to application coding practices, the following partition configuration areas are important for scalability:
- Partitions with many cores should use dedicated processors for best performance
- Placement of partition resources (memory and cores) on the physical hardware

Chapter 6 of the IBM POWER Virtualization Best Practices Guide provides information on how to improve partition placement.

7.20 How can I proactively ensure that batch performance will improve with my system upgrade?
When upgrading to a newer processor model or feature, batch performance may not be improved unless some upfront work is done to determine what limitations may be currently constraining the batch jobs. Here are some considerations for batch performance:
1. Determine if there are application constraints such as locking on an object or resource. Proper wait time analysis should be done to identify the bottleneck and how to best resolve it. See the Components of Response Time section earlier in this document.
2. Determine if storage I/O response times will be improved with the new system.   When systems are upgraded, often the storage I/O is migrated as well.   If this is the case, I/O response times will be the same or worse due to heavier stress with faster processor/memory. When moving to new storage subsystems, consider I/O response time differences and cache differences. The I/O response time can be analyzed with tools such as Collection Services.
3. Determine if memory constraints are not tightened due to more work on the system. Memory performance can be analyzed with tools such as Collection Services.
4. Determine if the batch workload is truly single-threaded.   If this is the case, simply adding more cores to the partition (or just a higher system CPW rating) may not help. Faster processor frequency is the key to improving many well-behaved single-threaded applications. Better yet, overlapping work by utilizing multi-processing or multiple jobs may help even more.
5. IBM Technology Expert Labs can assist with analysis of batch jobs and help to reduce batch window processing times.

7.21 How can I check the *interactive capacity* setting on my system?
Some IBM i systems allow Interactive Capacity to be limited. This reduces the performance for non-batch jobs (also known as 5250 interactive jobs).
To avoid this limitation, the '100% 5250' feature must be enabled.

Without the *100% 5250 features*, a `CPI1479` message will appear in the `QSYSOPR` message queue and interactive performance will be poor while batch jobs remain unaffected. The Interactive Capacity values shown below will be '0'.

Interactive Capacity values are stored in the Collection Services QAPMCONF file and can be easily viewed using either Performance Data Investigator or iDoctor.

To view in Performance Data Investigator, display the QAPMCONF perspective for the Collection Services collection of interest.

Interactive Limit (%):	100
Time Interval (seconds):	300
Interactive Threshold (%):	100

To view in iDoctor: launch Collection Services Investigator, then right on the collection of interest and select Properties.

Interactive limit	100%
Interactive threshold	100%

You can also view in ASMI by opening the **On Demand Utilities** menu on the left column. Then click **CoD VET Information**. Interactive capacity is listed as item 11 on this list.

VET Capability Settings

Capability bit information	Capability bit status/count
0 AIX allowed (license needed)	False
1 AIX allowed (license not needed)	True
2 RPA I/O hosting	True
3 AIX diagnostics	True
4 Subprocessor partitioning	True
5 LPAR creation allowed	True
6 Greater than 128 processors per partition	True
7 CAPI	True
8 i5/OS partitions allowed	True
9 Reserved	True
10 Live partition mobility	True
11 i5/OS 100% interactive capacity	True
12 Active Memory Expansion	True
13 Reserved	True
14 NEBS enabled	False

Figure 21. Interactive capacity

If this option is set to **False**, your interactive/5250 session performance will be affected.
For more information, refer to Power9 IBM i P20 (and greater) systems 5250 interactive performance and CPI1479 – IBM documentation.

7.22 What are considerations for IBM Power Virtual Server partitions
IBM i supports IBM Power Virtual Servers and can be configured much like a normal partition. Larger system configuration options such as IBM Power System E880 and IBM Power System E980 and smaller system configurations such as IBM Power System S922 are provided.

Processors can be configured as dedicated or shared. Dedicated processors provide the most consistent processing performance. For more tips on virtualization performance, refer to Virtualization Best Practices.

IBM Power Virtual Servers provide two storage options: Tier 1 and Tier 3. The Tier 1 storage type is best for customers who require higher storage throughput. Customers who do not require high storage throughput and are looking to minimize costs can select Tier 3. The storage types cannot be changed once the volume is created. A virtual machine (VM) cannot have disks from both storage types.

The storage tiers in Power Virtual Server are based on I/O operations per second (IOPS). The performance of your storage volumes is limited to the maximum number of IOPS based on volume size and storage tier. Although, the exact numbers might change over time, the Tier 3 storage is currently set to 3 IOPS/GB, and the Tier 1 storage is currently set to 10 IOPS/GB. For example, a 100 GB Tier 3 storage volume can receive up to 300 IOPS, and a 100 GB Tier 1 storage volume can receive up to 1000 IOPS. For more IOPS for a volume you can either increase the size of the volume or choose a tier that provides more IOPS/GB. In some cases, it may be more cost efficient to increase the size and use Tier 3 than using a smaller size volume and using Tier 1. After the IOPS limit is reached for the storage volume, the I/O latency increases. Even brief bursts in IOPs that exceeds the configured Tier throughput limit can trigger IOs with much higher latency that can impact application response times.

While not unique to this environment, consideration should also be given to the network. Be sure to understand how a change in network latency between the client and server may affect client-server application performance. While client-server applications designed requiring a high volume of network traffic may perform well when both the client and server are physically close together, application changes can be required as additional network latency is introduced.

It is important to properly size your partition before you configure your virtual server partition. Ensure that the processor configuration can meet your peak performance needs.   Closely analyze your storage performance needs before choosing Tier 3.   If you are not confident that your peak storage needs can be satisfied with the Tier 3 performance offering, choose Tier 1 instead. Be sure to size your partition processing and storage performance capacity for peak throughput that you may experience during the year.

7.23 What are some considerations for use of data compression in IBM i?
IBM i 7.5 adds support for compressing data using the DEFLATE compression format (which is the underlying format for compressed files). That support is available as a new compression format option on the SAVLIB command's DTACPR parameter (*ZLIB) and a new format option for the CPRDATA MI instruction. In addition, an IBM i partition running on a Power 10 system can use the Power processor's Nest Accelerator (NX) to accelerate handling of DEFLATE data in those interfaces.

The performance of any compression algorithm varies drastically based on the nature of the data that is being compressed. All data compression works by reducing the redundancy in the input; data that is completely unpredictable cannot be compressed, and data with predictable patterns can be compressed very well. DEFLATE compressors will also vary in their throughput depending on the input data.

On the data that IBM i has tested, IBM i's DEFLATE implementation typically compresses better than LZ1 (which is the compression offered by SAVLIB DTACPR(*HIGH)); that is, DEFLATE typically eliminates more redundancy, leading to smaller output than LZ1. If NX support is not available, DEFLATE typically

gives similar CPU and wall-clock time throughput to LZ1. When NX is used, DEFLATE compression operations typically offer at least 5 times faster throughput than LZ1.

Given that, there are significant performance gains to be had by ensuring that a partition has access to NX resources. There is a limited amount of NX capacity available system wide for DEFLATE operations, which is shared by all of the partitions in the system. Partitions can reserve NX resources ("Quality of Service credits") for DEFLATE from the Hardware Management Console; each credit is the capability to have one request outstanding to an NX. Any credits reserved this way reduce the number of requests that other partitions can have outstanding. The credits that are not reserved by any partition are shared by any partitions that attempt to send more requests at a time than they have reserved.

To ensure NX resources are available to your partition, ensure that your partition is in Power 10 compatibility mode, and reserve credits at the HMC partition General Properties, Advanced section, Supported Hardware Accelerator Types. For more information on devices that can be used for IBM i's DEFLATE implementation, refer to sections 8.22a and 8.22b.

A. How do I maximize data compression performance when saving to *SAVF files?
    Section 7.23 outlines IBM i's DEFLATE implementation and its performance benefits on P10. DEFLATE can be used for saving to *SAVF. In addition to DEFLATE (DTACPR (*ZLIB)), there is support for *MEDIUM, *HIGH, *LOW, and *NO compression as well.

    For more information on creating *SAVF files and data compression using *SAVF files, refer to the following IBM documentation:
    CRTSAVF
    SAVLIB

DTACPR	Compression Type
*NO	None
*LOW	SNA
*MEDIUM	TERSE
*HIGH	LZ1
*ZLIB	ZLIB

Figure 22. Compression options

    Above is a list of the current options for the DTACPR argument on SAV operations, and the compression type each of the options maps to.

B. How do I maximize data compression performance when saving to virtual optical media?
    If you're looking to save backups of files or compress data for transfer, you may want to consider saving to Virtual Optical Media. Virtual Optical Media supports all compression types offered on IBM i (*ZLIB, *HIGH, *MEDIUM, *LOW, and *NO). As mentioned in section 7.23, an IBM i partition running on a Power 10 system can use the NX accelerator for DEFLATE (*ZLIB) compression. When saving to a virtual optical device on IBM i, SAVLIB DTACPR(*ZLIB) can run

up to 20 times faster on Power 10 than Power 9 (for both small and large libraries). If using Power 9, SAVLIB DTACPR(*MEDIUM) will offer the best ratio between compressibility and time taken to complete compression. More information on setting up a virtual optical device and saving a library to a virtual optical device, refer to the following IBM documentation:
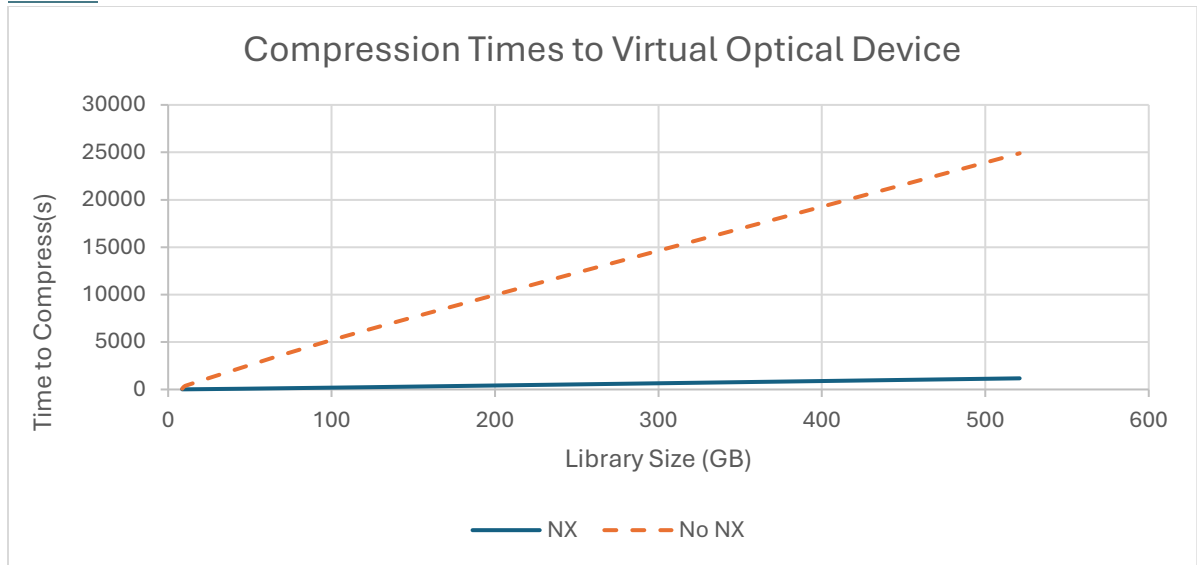Virtual Optical Media
SAVLIB

Figure 23. Compression times by library size

The chart above follows the save times for four libraries ranging from 8 GB to 500 GB. The top line outlines the time to save without NX support, while the bottom lines outline saves times with NX. Notice how the difference can be up to 20x faster when running with NX.

7.24 Should I utilize ASP Encryption?
New hardware-assist function recently added to IBM i 7.5 and the latest Power systems (Power9 and Power10) provides improved performance for storage access operations when ASP encryption is enabled. Workloads that require frequent storage accesses (read/write/update operations) can experience significantly improved storage I/O response times and reduced CPU consumption from this new functional enhancement. In addition, the encrypted ASP storage I/O throughput rate may approach the same rate as unencrypted ASP storage I/O operations. The CPU usage may be reduced by as much as half for large record I/O operations.

For more information on how to create encrypted ASPs, refer to Creating an encrypted auxiliary storage pool – IBM documentation.

7.25 Where can I find more information on performance tuning?
For more information on performance tuning - the top-level link for tuning system performance, refer to Monitoring and Managing Performance – IBM documentation.

The References chapter has many more links related to performance.

7.26 IBM Technology Expert Labs can help
The IBM Technology Expert Labs organization is available to assist customers with resolving system, application, and database performance problems. Formal and informal training opportunities are also

available where customers learn how to use the performance tools and resolve performance problems on their own. You can contact IBM i performance team, send an email to Eric Barsness at ericbar@us.ibm.com.

The IBM Db2 for i team in IBM Technology Expert Labs is also available to help clients in all areas of Db2 on i including Business Intelligence (BI), Analytics, database modernization, Very Large Database (VLDB) design, performance, and more. Contact Kent Milligan at kmill@us.ibm.com.

8 Database performance

8.1 Are there some basic suggestions for optimal SQL Performance?
The following suggestions apply to any SQL workload including embedded SQL in high-level language (HLL) programs, JDBC, ODBC, Db2 Web Query for i, and so on:
- Make sure you are using the latest level of Db2 for i. This ensures that most or all the SQL processing will use the newer SQL Query Engine, as well as many other performance enhancements delivered in recent IBM i OS releases and technology refreshes. Review recent performance enhancements in Db2 for i at Recent Performance Enhancements - Db2 for i Technology Updates.
- Implement an appropriate indexing strategy based on index advice from the query optimizer and the overall profile of the database. Index advice is available from Visual Explain, Db2 for i catalogs, the SQL plan cache, and the SQL performance monitor. For more information, refer to IBM Db2 for i indexing methods and strategies - IBM white paper.
- Utilize the MTI_INFO service to identify temporary indexes that have been created by Db2 for i. The information returned by this service enables the creation of permanent indexes to replace the temporary indexes which will reduce temporary storage usage and will provide more consistent query performance.
- Verify that the available *fair share of memory* is large enough to ensure efficient query implementation. This value is calculated by the optimizer based on the size of the memory pool, the number of active jobs, and sometimes by the configured activity level depending on the Db2 for i level. The number of active jobs calculation for a memory pool is more accurate when the memory pool is using Expert Cache.
- Ensure that the optimization goal is consistent with the intended use of the query (*ALLIO, *FIRSTIO).
- For environments dominated by short running queries, determine if large numbers of full opens and/or access plan rebuilds are impacting performance.
- For environments featuring a high number of SQL procedural objects (functions, procedures, and triggers), review the object attributes and procedural coding to determine if more efficient options and coding practices can be utilized. For more information, refer to Improving SQL Procedure Performance.
- Consider selectively enabling Db2 Symmetric Multiprocessing feature of IBM i (5770SS1 Product option 26) to allow for parallel processing on longer-running queries. When selectively enabling the use of parallel processing, it's recommended to use a parallel degree setting of *OPTIMIZE. IBM i 7.5 provides additional function and tuning capabilities when using parallel processing. For more information, refer to SQE - Improved SMP processing

This product (5770SS1 Product option 26) is now available at no charge.

The Query Supervisor feature can be used to limit the amount of time that a query can run or the amount of system resources a query can consume. This tool can also be used to just track the queries that are running long or consuming significant resources. For more information, refer to IBM i 7.5.

The following steps apply to applications which use an explicit connection model such as JDBC, ODBC, and CLI:
- Review the prestart job entries for QZDASOINIT and QSQSRVR and make appropriate adjustments. Prestart job activity can be monitoring using the DSPACTPJ command or the PRESTART_JOB_STATISTICS table function.

- If necessary, selected QZDASOINIT and QRWTSRVR jobs can be rerouted to different subsystems using the SET_SERVER_SBS_ROUTING procedure.
- Ensure you are using connection pooling. Review the QHST log or run database monitor to determine if there is a problem. The HISTORY_LOG_INFO table function can be used to do this.

8.2 How can I improve performance of applications that use Record Level Access (native I/O)?
Refer to chapter RPG/COBOL and Native I/O

8.3 What tools are available to monitor and tune SQL?
IBM i provides several integrated tools to collect and visualize database specific performance information. They can help you identify and tune performance problem areas. Key tools available include:
- IBM i Access Client Solutions (ACS) contains several components that a database engineer, administrator, or developer should leverage to address tuning of SQL.
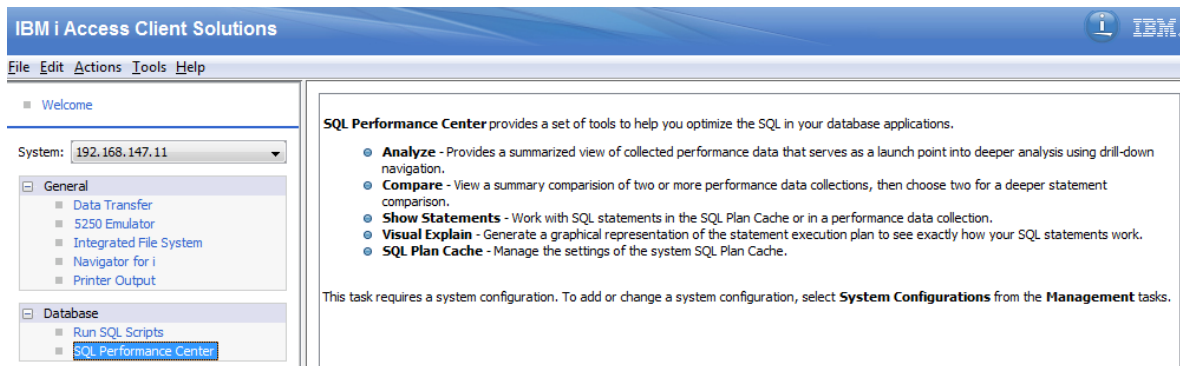
Figure 24. SQL Performance Center

- **SQL Plan Cache** collects plans in real time for SQL statements executing queries including non-SQL query interfaces such as Open Query File (OPNQRYF). It is continuously collecting information and provides a "moving picture" of query activity. You can view the "live" plan cache or save point in time snapshots of the plan cache for later analysis or comparisons. Plan Cache analysis is available from the SQL Performance Center link in ACS.

Figure 25. SQL plan cache

The Show Statements option provides an ability to filter on the plan cache data and displays *plans* based on your filter criteria.
For more information refer to SQL Plan Cache – IBM documentation.

- **SQL database monitor**: This monitor is a collection process that must be started and then turned off. It gathers real-time information about all SQL statements and non-SQL queries. The collection information includes details about optimization and runtime behavior. Its contents are viewable through the SQL Performance Center in the IBM i Access Client Solutions.

  For more information, refer to Database Monitor – IBM documentation.

Figure 26. SQL database monitor

- **Visual explain**: Within Show Statements (either the SQL Plan Cache or a database monitor) you can select Visual Explain to analyze a query. Visual Explain provides a picture of how Db2 is processing aa query request. A tremendous amount of information is provided here to diagnose a poor performing query.

  For more information, refer to Visual Explain – IBM documentation.
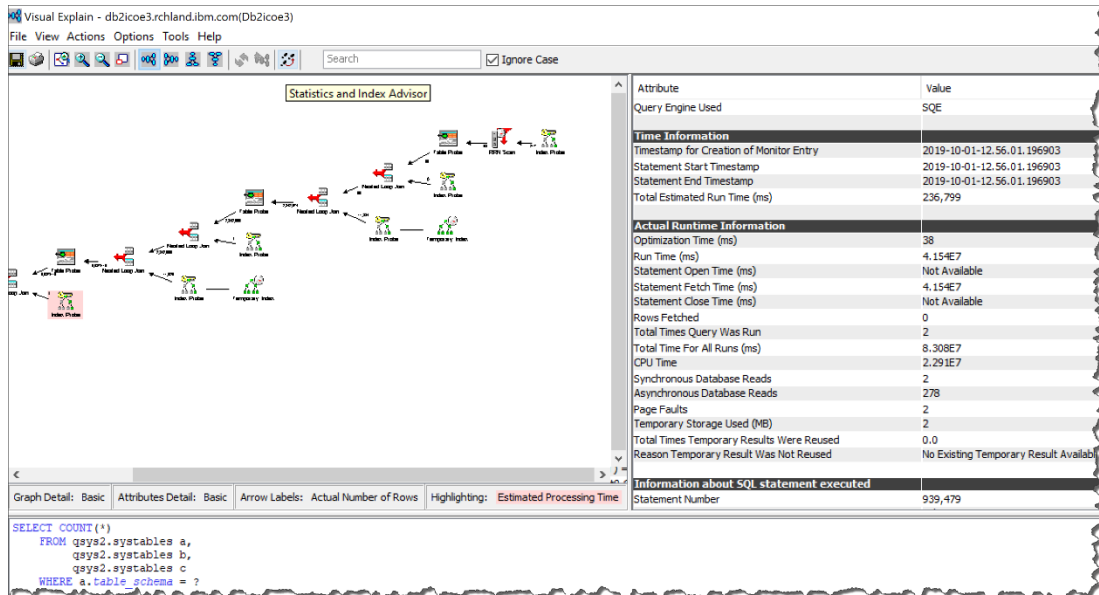
Figure 27. Visual explain

- **Index Advisor (of a statement)**: Within Visual Explain of a statement (either the SQL Plan Cache or a database monitor collection) you can select Index Advisor. Indexing is a critical performance enhancement technique in Db2 for i, and Db2 provides recommendations for indexes that would help the selected query request.

  For more information, refer to Index Advisor – IBM documentation.
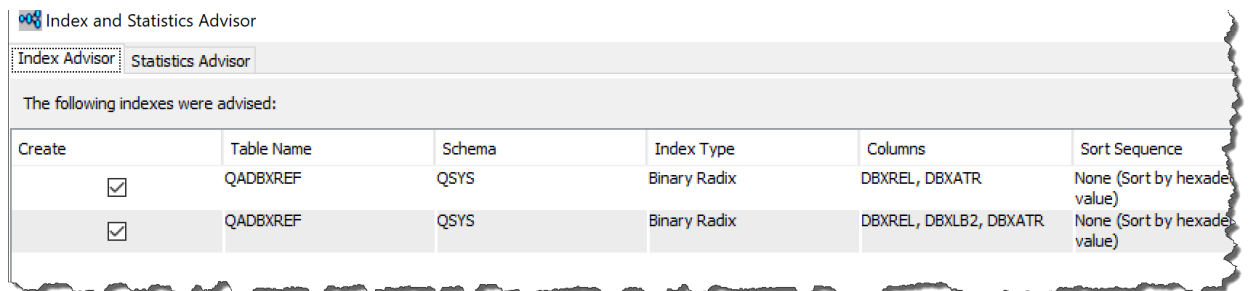
Figure 28. Index advisor

- **Index Evaluator** : With the ACS Schemas tool, you can evaluate what indexes are providing value to the query optimizer and query engine. The Index Evaluator provides usage information on all the indexes that are available for the optimizer to use. The Index Evaluator is launched by right clicking a table object and selecting the **Work With → Indexes** task to produce the following index usage output.
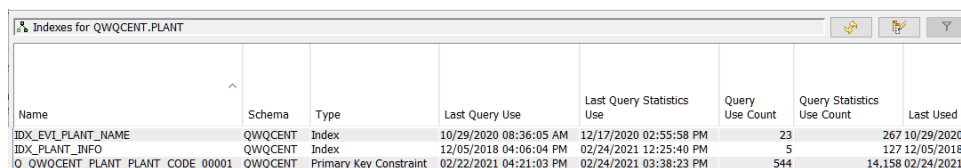
Figure 29. Index evaluator

- **Collection services**: Many database and SQL related metrics are collected by Collection Services (Collection Services should be automatically running on your partition 24x7).   Several job level SQL metrics are captured. Metrics include:
    - SQL clock time
    - SQL unscaled / scaled CPU time
    - SQL synchronous / asynchronous database reads
    - SQL synchronous / asynchronous non-database reads
    - SQL synchronous / asynchronous database writes
    - SQL synchronous / asynchronous non-database writes
    - Number of high-level SQL statements per thread

These metrics provide information regarding how much resource usage (such as CPU and I/O) was due to SQL work being performed by the thread. This data can be visualized in Performance Data Investigator (PDI). Launch to Investigate Data and then filter by the Database Package:
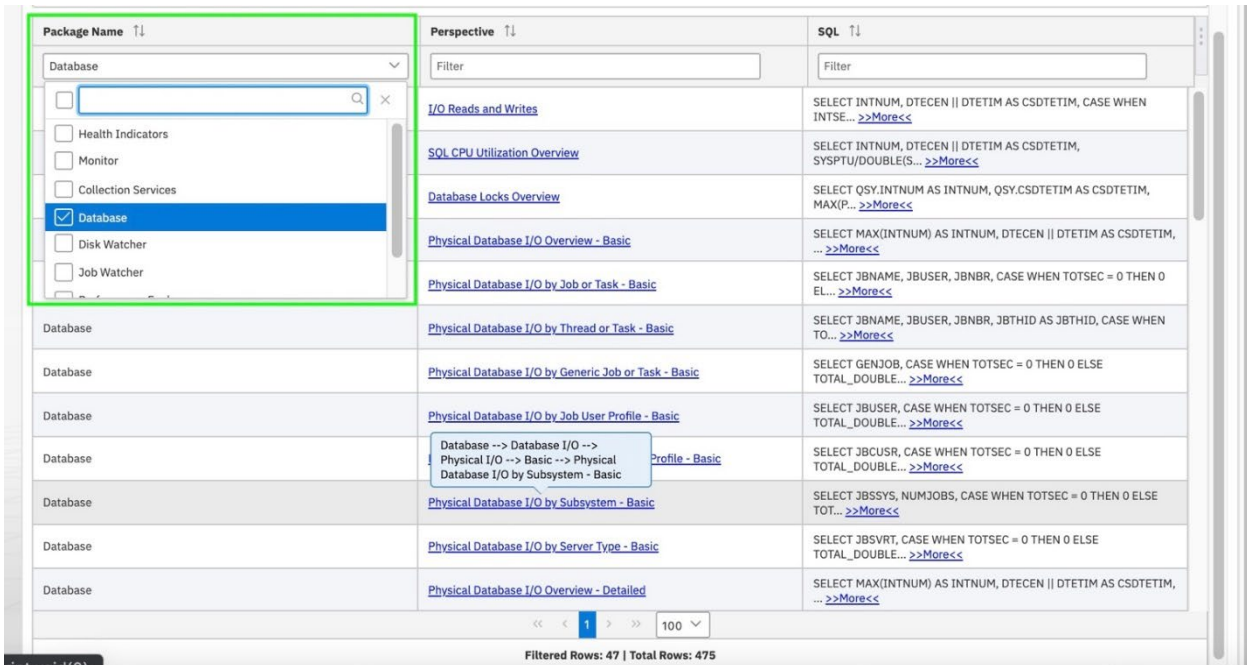
Figure 30. PDI database package

The following example chart of Detailed Physical Database I/O by Job or task shows the number of reads/write events per second that are SQL related versus non-SQL related activities:
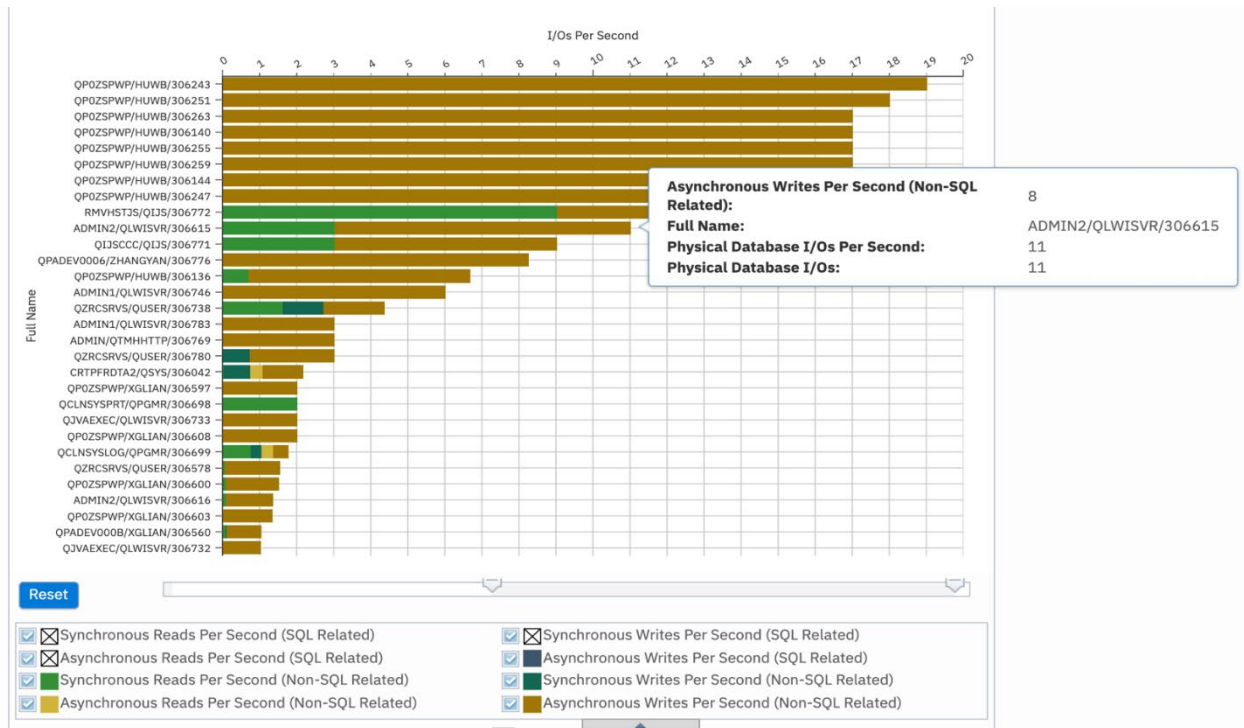
Figure 31. PDI physical database I/O by Job or Task

Starting with IBM i 7.2, Collection Services has included a new SQL category; this SQL category includes SQL Plan Cache data. This category is part of the standard collection category, and this data is collected by default. This new data can be viewed in PDI in the Database package as shown here:
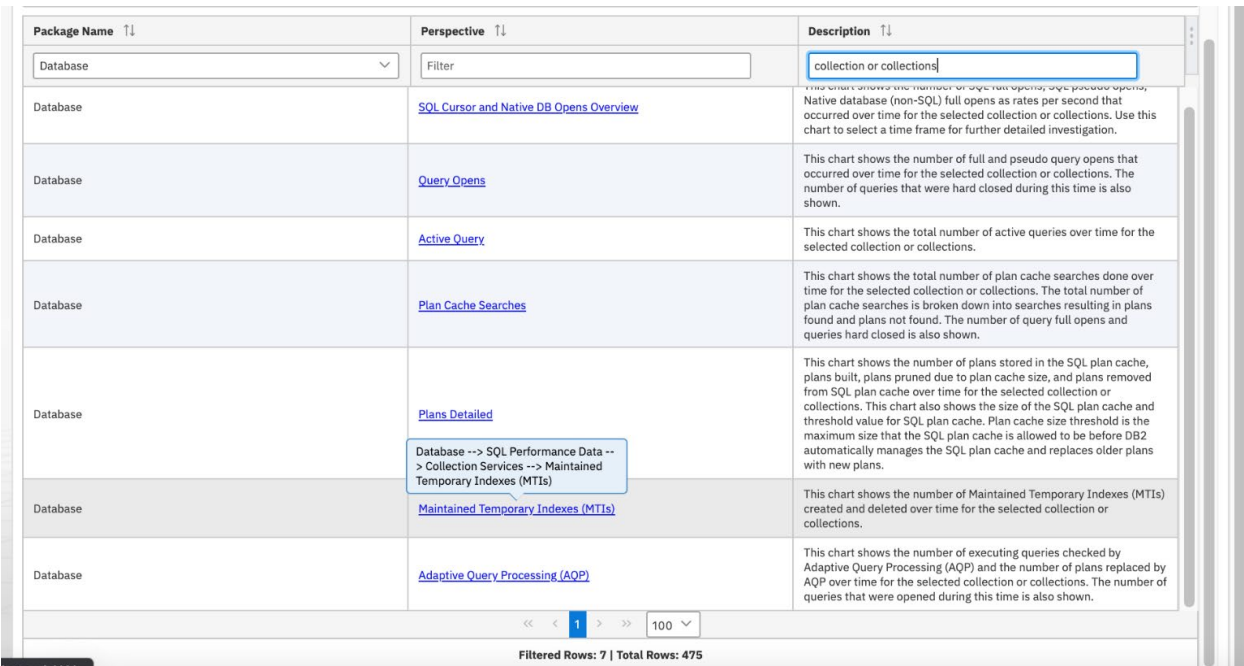
Figure 32. PDI - Collection Services perspectives in Database Package

The following is an example Query Opens chart showing number of Opens being done across the Collection Services time intervals:
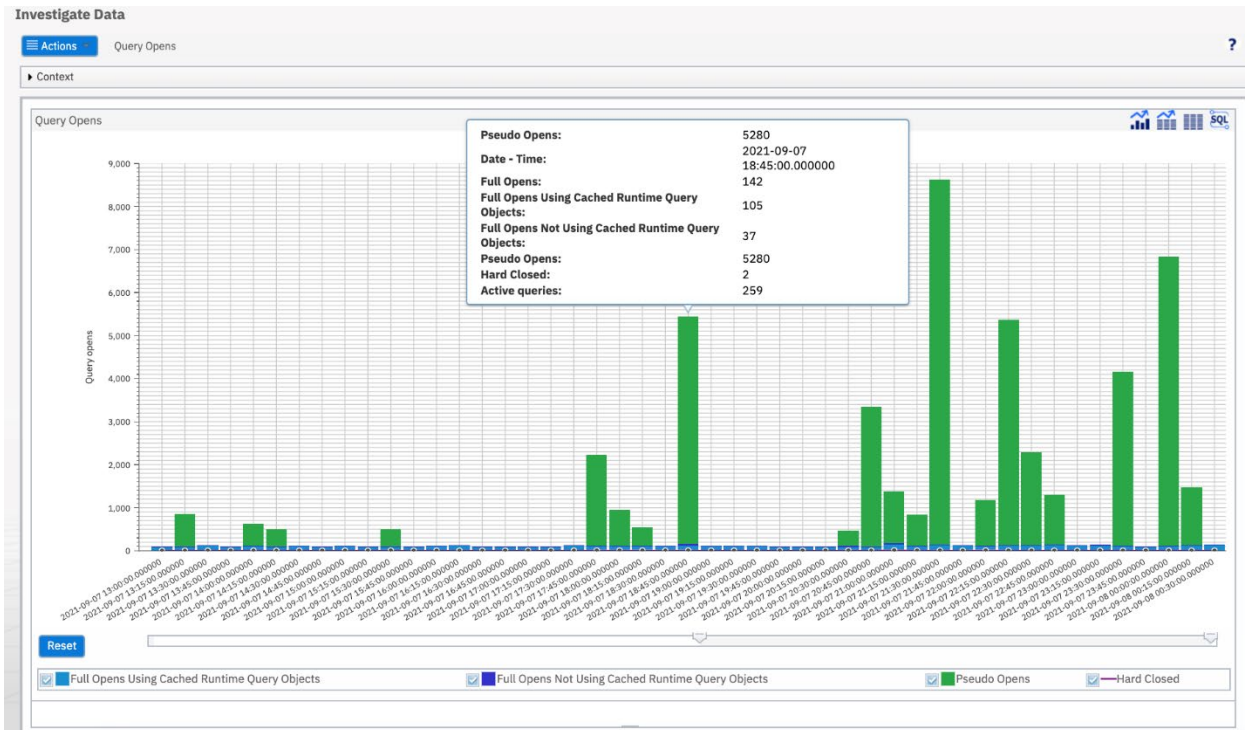
Figure 33. Collection Services

For more information on other available database and SQL charts, refer to Database Package and Perspectives.

8.4 IBM Db2 Mirror for IBM i
Db2 Mirror is a continuous availability solution available in IBM i 7.4 and IBM i 7.5. More information about Db2 Mirror, refer to the following IBM documentation:
- IBM Docs - Db2 Mirror
- IBM Db2 Mirror for i: Performance Considerations

Db2 Mirror requires a high-speed network connection to perform synchronous data operations. The high-speed connection interface being used is Remote Direct Memory Access (RDMA) over Converged Ethernet (RoCE). Network Redundancy Groups (NRGs) are used to automate the physical network redundancy, failover, and load balancing of the high-speed connections.

The new Collection Services collection category *RDMA collects performance information about NRGs and RDMA links. The *RDMA category is collected when Collection Services is configured to collect the *STANDARD collection profile. The new Collection Services data files are:
- QAPMLINK
- QAPMNRG
- QAPMNRGL

8.5 More information on Db2 for i performance and support
For more information on the Db2 for IBM i, refer to the following documentation:

- [IBM Support - Db2 for i](#)
- [IBM Docs - Db2 for i](#)

The IBM Docs has a section on SQL performance:
[SQL Performance](#)

For more information on database performance, refer to [References](#) chapter.

IBM Technology Expert Labs Db2 for i consulting team can also provide guidance around performance-related issues or information transfer. For the list of Db2 for I, IBM Technology Expert Labs offerings, refer to [IBM Technology Expert Labs - Db2 for i Offerings](#).

9 IBM i services

9.1 Introduction
There are many system services that can be accessed through system-provided SQL views, procedures, and functions. These provide an SQL interface to access, transform, order, and subset the information without needing to code to a system API.

To take advantage of the IBM i Services, you need a basic understanding of SQL. The Db2 for i SQL Reference is a good place for getting started with SQL as well as a great reference for those who are experts.

Some IBM i Services provide alternative methods to access information available through commands or APIs. However, others grant access to data that would otherwise be challenging to retrieve. Examples include:
- System Health Services provide the capability to programmatically get at system limit information. Are you nearing the system limits for files or jobs or journals (or many other limits)? How do you know? System health services can help you answer that question.
- System temporary storage (SYSTMPSTG) service provides a programmatic interface to identify where the temporary storage allocations have been made.
- PTF Group Currency service provides a capability to check your installed PTF groups with the levels that are available on Fix Central

9.2 IBM i services currently available
The following IBM i services are currently available:
- Application services: These procedures functions, and views provide interface information that can be used by applications.
- Backup and recovery services: These table functions and views provide information about BRMS.
- Communication services: These services provide communication information.
- Configuration services: These services provide configuration information.
- IFS services: These services provide information about the integrated file system (IFS).
- Java services: This view and procedure provide Java information and JVM management options.
- Journal services: This function and view provide journal information.
- Librarian services: These services provide object and library list information.
- Message handling services: These views and functions provide system message information.
- Performance services: These services provide information related to system performance.
- PowerHA services: These table functions and views provide information about PowerHA®
- Product services: These services provide information about licensed products.
- PTF services: These views provide PTF information.
- Security services: These views, procedures, and functions provide security information.
- Spool services: This view and function provide information about spooled files.
- Storage services: These views provide information about storage and storage devices.
- System health services: For the most important system resources, the IBM i operating system automatically tracks the highest consumption and consumers.
- Work management services: These views and functions provide system value and job information.

9.3 Db2 for i services currently available
The following Db2 for i services are currently available:

- Application Services: These procedures provide interfaces that are useful for application development.
- Performance services: These services include procedures that provide interfaces to work with indexes and a view to see information about database monitors. In addition, the QSYS2.OVERRIDE_QAQQINI procedure can be used to establish a temporary version of the QAQQINI file.
- Plan cache services: These services include procedures to assist you in performing database administration (DBA) and database engineering (DBE) tasks.
- Utility services: These procedures provide interfaces to monitor and work with SQL in jobs on the current system or to compare constraint and routine information across systems.

9.4 IBM i Access Client Solutions

IBM i Access Client Solutions is the newest member of the IBM i Access family. It provides a Java based, platform-independent interface that runs on most operating systems that support Java, including Linux, Mac, and Windows.

IBM i Access Client Solutions consolidates the most used tasks for managing your IBM i into one simplified location. The latest version of IBM i Access Client Solutions is available to customers with an IBM i software maintenance contract. For more information, refer to Product Download Information.

This tool provides an interface to access the IBM i Solution functions. It also includes examples to show how to build SQL statements to retrieve results.
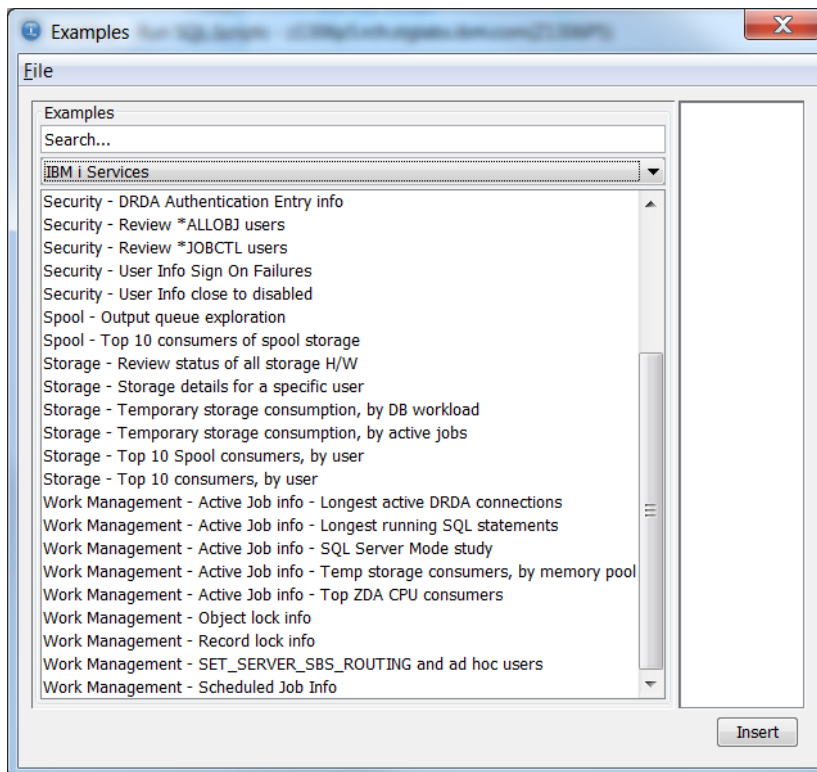
Figure 34. IBM i Access Client Solutions - IBM i services examples

The following SQL window shows how to use the IBM i Access Client Solutions tool to find the longest running SQL statements using one of the previous examples:

```
 1  -- category: IBM i Services
 2  -- description: Work Management - Active Job info - Longest running SQL statements
 3  --
 4  -- Find the jobs with SQL statements executing and order the results by duration of SQL statement execution
 5  --
 6  WITH ACTIVE_USER_JOBS (Q_JOB_NAME, AUTHORIZATION_NAME, CPU_TIME, RUN_PRIORITY) AS (
 7      SELECT JOB_NAME,
 8             AUTHORIZATION_NAME,
 9             CPU_TIME,
10             RUN_PRIORITY
11        FROM TABLE (
12             ACTIVE_JOB_INFO('NO', '', '', '')
13             ) x
14        WHERE JOB_TYPE <> 'SYS'
15  )
16  SELECT Q_JOB_NAME,
17         AUTHORIZATION_NAME,
18         CPU_TIME,
19         RUN_PRIORITY,
20         V_SQL_STATEMENT_TEXT,
21         ABS(CURRENT TIMESTAMP - V_SQL_STMT_START_TIMESTAMP) AS SQL_STMT_DURATION,
22         B.*
23      FROM ACTIVE_USER_JOBS,
```

Q_JOB_NAME	AUTHORIZATION_NAME	CPU_TIME	RUN_PRIORITY	V_SQL_STATEMENT_TEXT
034400/QUSER/QZDASOINIT	PDITEST	851	20	SELECT COUNT(*) FROM (SELECT MAX(JBNAME) AS JBNAME, MAX(JBUSER) AS JBUSER, MAX(JBNBR) AS JBNBR, MAX(JBT..
034361/QUSER/QZDASOINIT	STACYB	796	20	-- category: IBM i Services-- description: Work Management - Active Job info - Longest running SQL ..

Figure 35. IBM i Access Client Solutions - Active Job - Longest running SQL statements

9.5 More information
Here's the launch point for IBM i Services and Db2 for i services:
- IBM i Services
- Db2 for i Services

The IBM Support website provides details about the enabling operating system level and Db2 PTF Group. For more information, refer to Db2 for i Services.

10 RPG/COBOL and native I/O

10.1 Introduction
This section discusses methods for improving the performance of applications written in RPG or COBOL and those programs that are accessing data using native IO operations.

10.2 CPU performance tips
RPG and COBOL programs are optimized to run on IBM i with every release and will adopt the latest processor enhancements when compiled or when re-translated as required, for example when moving from IBM i 5.4 to 6.1 or 7.1.

However, additional optimization can be done over and above what the compilers and the translator provide to improve performance in terms of runtime or CPU. Some of the following optimization techniques will trade off better optimization with the capability to debug the code, so be aware when deciding to use program optimization. It is generally good practice to select a few highly used programs with intense use of CPU cycles and perform optimization and when debugging is not required any longer. Some of these performance optimization techniques can be very significant and worth the effort for highly used programs in the interactive or batch environment.

10.2.1 Compiler options
Optimize the use of processors by specifying the `OPTIMIZE` parameter on the `CRTRPGMOD` and `CRTBNDRPG` command, for example - `OPTIMIZE(*BASIC or *FULL)`. You can also use the commands `CHGMOD`, `CHGPGM` or `CHGSRVPGM` and the parameter `OPTIMIZE(*YES)` to optimize instructions being executed within your program. Note that debug mode function is reduced as optimization is added. For more information, refer to Create Bound RPG Program.

Optimize the use of program arguments by using the parameter ARGOPT(*YES) on the CRTPGM and CRTSRVPGM commands. For more information , refer to Improve Performance with Argument Optimization.

Control program activation by the activation keyword (*DEFER) with the bind service program parameter on the `CRTPGM`, `CRTSRVPGM`, `UPDPGM` and `UPDSRVPGM` commands. For more information, refer to Activate Your Programs On Demand.

10.2.2 Profiling
Optimize your ILE RPG or COBOL programs by profiling and collecting procedure execution information and applying them to optimize the program structures and instruction flows. For more information read: Program Profiling

Identifying hot programs (programs which use most of the resources) within your application with Performance Explorer and the PEX definition *STATS. For more information, refer to the following IBM Redbooks:
Application and Program Performance Analysis using PEX Statistics on IBM i5/OS

Identifying hot statements of programs (actual statements of programs which use most of the resources) within your application with Performance Explorer and the PEX definition *PROFILE. For more information, refer to the following IBM Redbooks:
Application and Program Performance Analysis Using PEX Statistics on IBM i5/OS

10.3 I/O performance tips

10.3.1 General tips

Use higher blocking for input/output only operations – `SEQONLY((*YES #rcds)) >= 128KB`. For more information on increasing logical page size in DB operations, refer to Sequential-only processing of database files.

Avoid open/close of files, use RETRN vs SETON *LR in RPG. For more information, refer to Tips: Run and call IBM i procedures.

Eliminate FRCRATIO(1) to prevent every update/write to be forced out to disk. For more information, refer to FRCRATIO parameter.

The number of physical I/Os can be significantly reduced by using the SETOBJACC command and pinning a database file or index into a private memory pool which is not used by anything else. For more information, refer to SETOBJACC/CLRPOOL Commands.

10.3.2 Database modernization

For applications which are dominated by database processing, performance can often be improved by converting existing high-level language programs from native IO (record level access) to SQL set-based processing. IBM Technology Expert Labs has a well-established process for this kind of database modernization which takes advantage of the SQL query optimizer's advanced capabilities while minimizing the impacts to existing programs.

11 C program compiler optimization

C and C++ programs are optimized to run on IBM i with every release and will adopt the latest processor enhancements when compiled or when re-translated, e.g. when going from 5.4 to 6.1 to 7.1. Additional optimization can be done over and above what the compilers and the translator provide in order to improve performance in terms of runtime or CPU use and path length.

Be aware that some of the following optimization techniques trade off improved performance with less capability to debug the code. It is generally a good practice to focus optimization on a few highly used programs with intense use of CPU cycles when debugging is not required any longer. Some of these performance optimization techniques can be very significant and worth the effort for highly used programs in the interactive or batch environment.

11.1 IBM i architecture – MI interface
The IBM i Architecture depicted below only allows predefined MI Instructions to go to the lower level of microcode. All user code, including the compilers as well as the operating system itself, is independent from the microcode and hardware layer below. All compiled code will have to get translated in order to get machine level instructions specific to the hardware being run on.
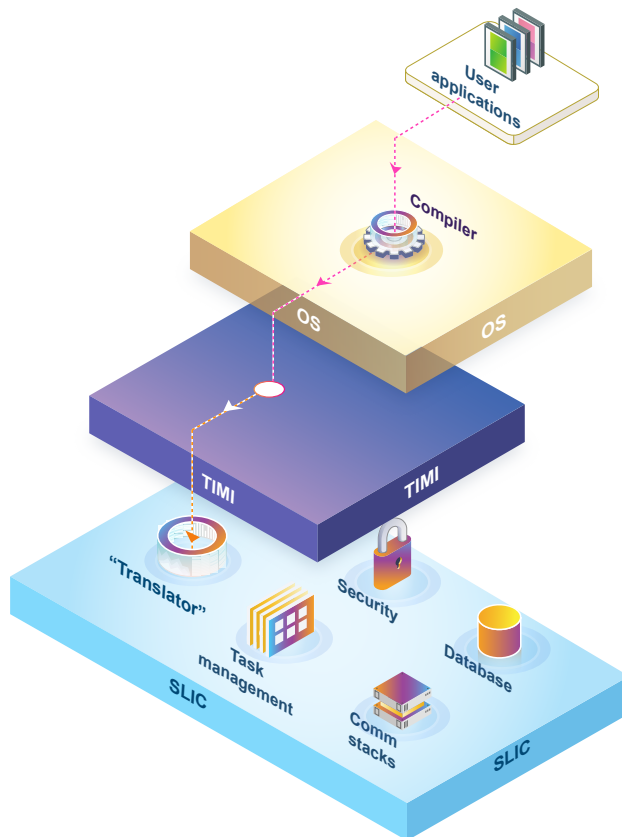
Figure 36. IBM i architecture – MI interface

11.2 IBM i program creation
During the compile of any program, including C and C++ programs, you will need to use the CRTPG MI instruction to generate machine executable code. This translation step can also be independently performed through the `CHGPGM`, `CHGMOD` commands.
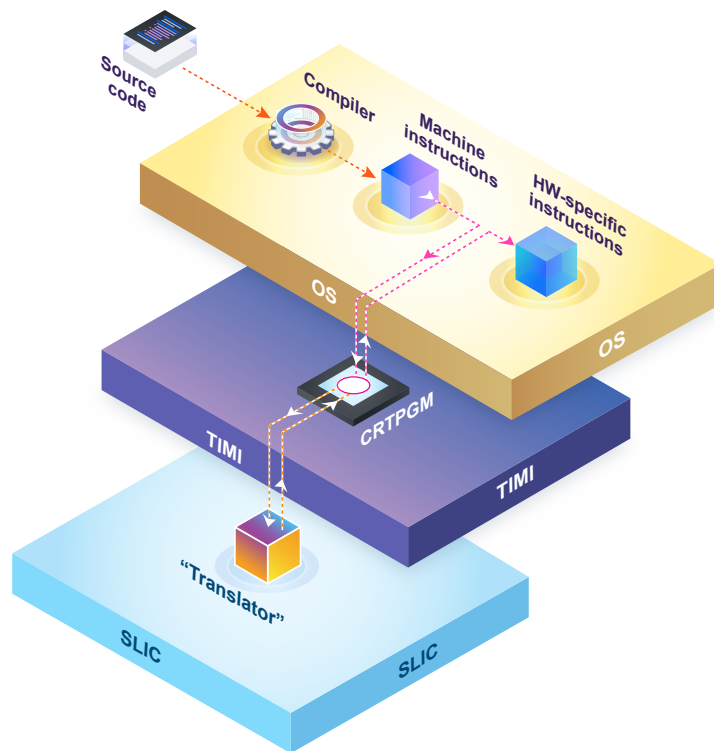
Figure 37. IBM i program creation

Key notes:
- Compile programs to an intermediate code level called Machine Interface (MI).
- Incorporate translation into the IBM i architecture using the Technology-independent machine interface (TIMI).
- Ensure the translator is hardware-specific to optimize performance.

11.3 Compiler option – OPTIMIZE parameter

Optimize the use of CPU by specifying the `OPTIMIZE` parameter on the `CRTCMOD` and `CRTBNDC` command, for example: `OPTIMIZE(20, 30 and 40)`.

Optimization may rearrange or eliminate some statements altogether. Consequently, field values presented by the debugger, as well as breakpoints and step locations, may be inaccurate.

11.3.1 CRTCMOD and CRTBNDC commands

- OPTIMIZE(10) is the default value. The compiler performs no optimization on the code. This option allows you to debug the program accurately, with the debugger presenting correct current field values.
- OPTIMIZE(20) performs some optimization on the compiled code. The debugger will let you display variables, but the displayed value may not be the current value. The debugger will also allow variable changes, but you may encounter unexpected results if you try to use the changed variables.
- OPTIMIZE(30) generates more efficient code than level 20. More optimizations are performed in addition to those performed at level 20, and the compile process takes longer. The debugger will let you see (but not change) field values, but the debugger may not show you the correct current field values.

- OPTIMIZE(40) generates the most efficient code. All optimizations done at level 30 are performed on the generated code. In addition, code is eliminated from procedure prologue and epilogue routines that enable instruction trace and call trace system functions. Eliminating this code enables the creation of leaf procedures. A leaf procedure is a procedure that does not require an invocation frame to be stacked. Procedure call performance to a leaf procedure is significantly faster than to a normal procedure.

10.3.2 CHGMOD, CHGPGM or CHGSRVPGM commands

You can also use the commands `CHGMOD`, `CHGPGM` or `CHGSRVPGM` and the parameter `OPTIMIZE(*YES)` to optimize instructions being executed within your program. The `DSPMOD`, `DSPPGM`, and `DSPSRVPGM` commands shows you the current level of optimization for an object.

You can change optimization levels with the `CHGMOD`, `CHGPGM`, or `CHGSRVPGM` commands; these commands support the levels already discussed, but also add other options:
- OPTIMIZE(*NONE|*NO|10): These options are equivalent and are discussed above. OPTIMIZE(*NO) is used only by the CHGPGM command.
- OPTIMIZE(*BASIC|20): These options are equivalent and are discussed above.
- OPTIMIZE(*FULL|30): These options are equivalent and are discussed above.
- OPTIMIZE(*YES|40): This is the highest possible optimization level. In addition to all level 30 optimization, level 40 performs optimization that reduces the capability to provide extensive call and instruction tracing.
- OPTIMIZE(*YES) is used only by the CHGPGM command.

Note: Debug mode function is reduced as optimization is added.

For more information on how to use the CRTBNDC command, refer to [Create Bound C Program (CRTBNDC)](#).

11.4 Program profiling

Optimize your ILE C programs by profiling and collecting procedure execution information and applying that information to optimize the program structures and instruction flows. This is an advanced optimization technique that reorders procedures, or code within procedures, in ILE programs and service programs based on statistical data gathered while running the program. This reordering can improve instruction cache utilization and reduce paging required by the program, thereby improving performance.

Note: The semantic behavior of the program is not affected by program profiling.

The performance improvement realized by program profiling depends on the type of application and you can expect more improvement from programs that spend most of the time in the application code itself, rather than spending time in the operating system or doing input/output processing.

The performance of program code produced when applying profile data depends upon the optimizing translator correctly identifying the most important portions of the program during typical use. Therefore, it is important to gather profile data while performing the tasks that will be performed by end users and using input data that is similar to that expected in the environment in which the program will be run.

Note: Program profiling is available only for ILE programs and service programs that are compiled using optimization level 30 or 40. Because of the optimization requirements, you should fully debug your programs before using program profiling.

To enable the program to collect profiling data:
- Compile your ILE program(s) using the PRFDTA(*COL) parameter option.

- Start the program profiling collection on the system with the Start Program Profiling (STRPGMPRF) command.
- Collect profiling data by running the program through its high-use code paths. Because program profiling uses statistical data gathered while running the program to perform these optimizations, it is critical that this data is collected over typical uses of your application.
- End the program profiling collection on the system with the End Program Profiling (ENDPGMPRF) command.
- Apply the collected profiling data to the program by requesting that code be reordered for optimal performance based on the collected profiling data using the PRFDTA(*APYALL) parameter option.

For more information on how to find program hot spotsm refer to Program Profiling – IBM documentation.

11.5 Advanced argument optimization

Advanced argument optimization is a cross-module optimization that is used to improve the performance of programs that contain frequently run procedure calls, including C++ applications that make mostly non-virtual method calls. Improved runtime performance is achieved by enabling the translator and binder to use the most efficient mechanisms to pass parameters and return results between procedures that are called within a program or service program.

The Argument optimization (ARGOPT) parameter, with *YES and *NO as possible values, is available on the CRTPGM and CRTSRVPGM commands to support advanced argument optimization. Specifying ARGOPT(*YES) causes the program or service program to be created with advanced argument optimization. The default is *NO.

Note: For programs that consist of hundreds or thousands of modules, creation time can be significantly longer using this option.

For more information on how to perform argument optimization, refer to Improve Performance with Argument Optimization.

11.6 Interprocedural analysis

At compile time, the optimizing translator performs both intraprocedural and interprocedural analysis (IPA). Intraprocedural analysis is a mechanism for performing optimization for each function within a compilation unit, using only the information available for that function and compilation unit. Interprocedural analysis is a mechanism for performing optimization across function boundaries. The optimizing translator performs interprocedural analysis, but only within a compilation unit. Interprocedural analysis that is performed by the IPA compiler option improves on the limited analysis described above. When you run interprocedural analysis through the IPA option, IPA performs optimizations across the entire program. It also performs optimizations not otherwise available at compile time with the optimizing translator.

To optimize your program or service program with the Inter-Procedural Analysis function, make sure that all the modules have the parameter MODCRTOPT(*KEEPILDATA) specified and are compiled with an optimization level of 20 or higher, and the parameter IPA(*YES) is specified on the CRTPGM or CRTSRVPGM command
For more information on additional optimization with IPA, refer to Interprocedural analysis (IPA) – IBM documentation.

12 Java and WebSphere

12.1 Introduction
This chapter covers best practices for Java and WebSphere Application Server performance. Most of the Java questions apply to WebSphere applications.

12.2 Does it matter if I use the 32-bit versus 64-bit JVM?
With current supported JVMs, there is little difference in performance between using 32 and 64-bit. When using a 64-bit JVM, compressed references (-Xcompressedrefs) can be used to reduce memory requirements. Newer versions of Java are 64 bit-only.

12.3 I'm using an old JDK level and am behind on PTFs, is that okay?
No, for best performance use the most current JDK level you can and should apply the latest PTFs.

12.4 Should I run my JVMs in a separate memory pool?
Yes, JVMs should run in a memory pool configured with enough memory to hold all JVMs at their maximum heap size. WRKSHRPOOL can be used to set the minimum pool size in percent to ensure the pool does not shrink too much if QPFRADJ is on. If there is not enough memory in the pool, the JVMs may become unresponsive due to high page faulting.

12.5 How do I determine the proper GC settings (heap sizes, collection policy, and so on) for my application?
For most customers, the GENCON policy results in the best performance. Discussion on how to set the minimum and maximum size is in the IBM SDK for Java: Diagnostics Guide, section "How to do heap sizing".

Collection Services collects JVM information by default. Use Performance Data Investigator or iDoctor to visualize Java heap usage system-wide and by job. This might be a good starting point to find the job(s) which are consuming the most Java heap.

Take care not to overcommit the number of garbage collection (GC) threads for your JVM. The default number of GC threads depends on the number of cores in the partition. It is calculated by the number of dedicated processors, or virtual processors (VPs) for shared processor partitions, multiplied by the maximum SMT level minus 1. A Power7 partition (with SMT level of 4) and with 20 VPs would have 79 helper threads per JVM by default. Each of these threads is concurrently active during garbage collection, which can lead to CPU queuing, and can limit the amount of CPU available for other work on the partition.

You can control the number of GC helper threads through the `-Xgcthreads##` setting. This sets the number of helper threads you want to be used during garbage collection. If you want to use 8 threads, then use `-Xgcthreads08`, which will result in one controlling thread and 7 GC helper threads.

Note that IBM i and the JVM determine the values for Java system properties by using the following order of precedence:
1   Command line or JNI invocation API
2   QIBM_JAVA_PROPERTIES_FILE environment variable
3   user.home SystemDefault.properties file
4   /QIBM/UserData/Java400/SystemDefault.properties

5   Default system property values

For more information, refer to Java system properties – IBM documentation.

12.6 How do I analyze JVM dumps and verbose GC output?
Download and install the IBM Support Assistant and install the IBM Monitoring and Diagnostic Tools for Java toolset: Monitoring and Post Mortem/

12.7 How do I tune Java code?
The Java and WebSphere Performance on IBM eServer iSeries Servers Redbook still is a good resource even though it is from 2002. The recommendations specific to the JVM mostly do not apply anymore, but the coding tips and techniques are still valid.

For more information, refer to Java and WebSphere Performance on IBM eServer iSeries Servers – IBM Redbooks

12.8 Where can I get more information on Java tuning and debugging?
For information on Java tuning and debugging, refer to the IBM support page collections.

12.9 Can I use Job Watcher to collect Java information?
By default, Job Watcher does not collect the call stacks of Java threads. The *IBM Technology for Java* data collection tab in iDoctor can be used to enable the collection of JVM statistics and Java thread call stacks. IBM supplied definitions ending in "J" include the collection of this additional information.

12.10 How can I improve the performance of WebSphere Application Server running on IBM i?
- Use the most recent version of WebSphere available.
- Turn off JSP reloading in production environments.
- Use the Fast Response Cache Accelerator (FRCA) for HTTP Server for caching static and dynamic content.
- Use the Apache HTTP server cache for secure (SSL) static content.

For more information, refer to the following IBM documentation:
- Fast Response Cache Accelerator (FRCA) for HTTP Server
- Tuning the application serving environment

12.11 How do I tune IBM Toolbox for Java performance?
For information on tuning IBM toolbox for Java performance, refer to IBM Toolbox for Java Performance – IBM documentation.

12.12 Where can I find additional information?
For additional information, refer to Java Performance for IBM i – IBM documentation.

13 Power processor-based systems

13.1 Introduction
This chapter covers best practices for Power processor-based system performance.

13.2 SMT and Intelligent Threads
Power8, Power9, and Power10 processors support SMT with eight hardware threads, which provides more transaction processing capacity per core. Power hardware has had SMT support for many years. Initially, it was two hardware threads of execution on a processor core. With Power7, the number of hardware threads was increased to four. With Power8, the number of hardware threads was increased to eight.

13.2.1 Single thread versus SMT2/SMT4/SMT8
Although it is not generally recommended for most customers, applications that are single process and single threaded may gain some benefit from running in single thread mode. Multithreaded and/or multi process applications typically benefit more running in higher SMT modes. Single thread mode may also be beneficial in the case of applications where the number of application threads is smaller than the number of cores assigned to the partition. Applications that do not scale with a larger number of CPUs may also benefit from running in SMT2 or single thread mode instead of SMT4 or SMT8 since lower number of SMT threads mean lower number of logical CPUs. This is a partition-wide setting and overall transaction processing capacity may be reduced by changing from the default mode.

The default mode on Power7 systems is SMT4. The default mode on Power8 systems running IBM i 7.1 TR8 is SMT4 and Power8 systems running IBM i 7.2 is SMT8. The default mode on Power9 and Power10 systems is SMT8.

For more information, refer to Under the Hood: Simultaneous Multi-Threading on POWER7 Processors.

13.2.2 Intelligent threads technology
The Power10 processor, like Power9, Power8 and Power7 processors before it, uses Intelligent Threads technology to maximize workload performance regardless of the processor's threading context. This technology has been enhanced to adapt more quickly and with greater efficiency to changes in the workload. If the Power10 processor is under-committed, meaning fewer hardware threads are dispatched than are available, the core performance is roughly the same independent of threading context. So, for example, if one thread is dispatched, performance will be similar in single thread, SMT2, SMT4, and SMT8 contexts; if two threads are dispatched, performance will be similar in SMT2, SMT4, and SMT8 contexts; if three or four threads are dispatched, performance will be similar in SMT4 and SMT8 contexts.

From a usability standpoint, Intelligent Threads means that system-level processor threading context adjustments typically aren't necessary to maximize workload performance. Highly multi-threaded workloads can benefit from the additional throughput offered by SMT8 technology, but multi-threading isn't necessary for a workload to achieve high-performance. It just works!

13.3 Flexible SMT controls
SMT settings can be adjusted specifically for each partition. While the default processor threading context is suitable for most commercial environments, IBM i offers manual controls that allow the system to be fine-tuned to the specific characteristics of the workload. In general, as the processor

threading context is decreased, single-thread performance may be improved at the expense of maximum aggregate throughput.

IBM i provides flexible, dynamic system-level processor threading controls. The processor threading contexts available for the partition IPL is determined by the Processor Compatibility Mode partition attribute as shown in table 2. Note that the PCM is established during partition activation.

13.3.1 Processor Compatibility Mode

Power10, Power9, Power8 and Power7 processor based systems can set the Processor Compatibility Mode to change the SMT behavior. Partitions running in Power6 or Power6+ compatibility mode can run in single thread or SMT2. Partitions running in Power7 compatibility mode can run in SMT4, SMT2 or single thread mode.

IBM i supported and default processor threading contexts				
Processor Compatibility Modes	Supported threading context	Default threading context		
		IBM i 7.3	IBM i 7.4	IBM i 7.5
Power7	Single thread, SMT2, SMT4	SMT4	Not available	Not available
Power8	ST, SMT2, SMT4, SMT8	SMT8	SMT8	SMT8
Power9	ST, SMT2, SMT4, SMT8	SMT8	SMT8	SMT8
Power10	ST, SMT2, SMT4, SMT8	SMT8	SMT8	SMT8

Table 2. IBM i supported and default processor threading contexts

The default thread context is selected by the operating system, but it can be easily changed by the system administrator. For all supported releases, the default SMT level is the maximum supported by the hardware.

Compatibility mode can be changed using the HMC when the partition is powered down.

Figure 38. Processor Compatibility Mode

On the HMC enhanced GUI, the **Processor Compatibility Mode** is found on the **Processor** tab, under **Advanced Settings**, after clicking on the **Advanced** button.

13.3.2 QPRCMLTTSK system value

To enable or disable the SMT behavior for each partition, refer to Processor multitasking (QPRCMLTTSK) system value – IBM documentation. The default for this system value is *system controlled* which means the operating system determines the optimal setting. For Power7 systems, it is dynamic switching with four hardware threads. For Power10, Power9, and Power8 processor based systems, it is dynamic switching with eight hardware threads.

You can explicitly turn off SMT support and always run the cores in a single-threaded mode by setting QPRCMLTTSK to *OFF. In general, the default setting will be the best for most applications. Switching to and from single-thread context may be accomplished using the IBM i processor multitasking mode system value, QPRCMLTTSK. QPRCMLTTSK changes are effective immediately and persist across system IPL.

Supported QPRCMLTTSK values are as follows:

- 0 - Processor multitasking is disabled. This value corresponds to ST context.
- 1 - Processor multitasking is enabled. This value corresponds to SMT2 context if the partition's Processor Compatibility Mode is Power6 or Power6+. Otherwise, the thread context is determined by the maximum SMT level control.
- 2 - Processor multitasking is system controlled. This is the default value. For Power7, Power8, Power9 and Power10, the implementation is identical to '1', processor multitasking enabled.

Examples:

- DSPSYSVAL SYSVAL(QPRCMLTTSK)
- CHGSYSVAL SYSVAL(QPRCMLTTSK) VALUE('0') /* ST context */
- CHGSYSVAL SYSVAL(QPRCMLTTSK) VALUE('1') /* SMTn context */
- CHGSYSVAL SYSVAL(QPRCMLTTSK) VALUE('2') /* SMTn context */

13.3.3 QWCCHGPR API

The QWCCHGPR API is available to switch between the various SMT settings without rebooting the partition. When processor multitasking is enabled, switching among the thread contexts available for the partition IPL may be accomplished using the change processor multitasking information API, QWCCHGPR. QWCCHGPR API changes are effective immediately and persist across system IPL. The QWCCHGPR API takes a single parameter, the maximum number of secondary threads per processor:

- 0 - No maximum is selected. The system uses the default number of secondary threads as determined by the operating system.
- 1-255 - The system may use the number of secondary threads specified.

The QWCCHGPR API may be called from the command line interface. Note that setting the maximum number of secondary threads does not establish the processor threading context directly. The maximum value will be accepted regardless of the processor threading contexts supported by the underlying hardware, and the OS will apply the configured maximum to the system. On a Power10, Power9, or a Power8 processor-based system, if a maximum value is specified by the QWCCHGPR API, the OS will try to establish the maximum thread context supported as shown in Table 1, subject to the limit specified.

Examples:
- CALL PGM(QWCCHGPR) PARM(X'00000000') /* No maximum */
- CALL PGM(QWCCHGPR) PARM(X'00000001') /* SMT2 context */
- CALL PGM(QWCCHGPR) PARM(X'00000003') /* SMT4 context */
- CALL PGM(QWCCHGPR) PARM(X'00000007') /* SMT8 context */

The maximum number of secondary threads can be obtained from the retrieve processor multitasking information API - QWCRTVPR.

For more details, refer to Change Processor Multitasking Information (QWCCHGPR) API – IBM documentation.

13.4 Power saving mode

Many customers can achieve cost savings by tuning their systems for energy management. By doing this tuning, system performance may be affected. On the positive side, tuning the energy setting allows some systems to be able to boost their processor frequency dynamically when running at lower processor utilization. On the other hand, tuning the energy setting can result in slower system performance at high utilization due to slower dynamic processor frequency changes.

Nominal energy settings provide the most consistent processor frequency behavior and allow consistent response time and throughput results. Use nominal settings if energy tuning is not required.

Power processor-based systems include intelligent energy features that help to dynamically optimize energy usage and performance so that the best possible balance is maintained. Intelligent energy features like EnergyScale™ dynamically optimize processor speed based on thermal conditions and system utilization. If the system's environment is favorable, the system can speed up processor cores to a frequency up to 10% higher than the normal rated speed to better handle demanding workloads. When utilization is lighter, or if environmental conditions are less favorable, the system can slow processor cores by as much as 50% to save energy. Users can select whether intelligent energy optimization favors maximizing performance or energy savings based on their needs.

For more information on Power9 processor-based systems, refer to IBM EnergyScale for Power9 Processor-Based Systems.

For more information on Power10 processor -based systems, refer to IBM EnergyScale for Power10 Processor-Based Systems.

13.5 Adapter placement
High-speed adapters should be placed following the recommendations of the PCI adapter placement recommendations for the individual Power model.

For detailed information on systems hardware, refer to the following IBM documentation:
- Power8 System Hardware
- Power9 System Hardware
- Power10 System Hardware

13.6 Affinitized partitions
The Power hypervisor was improved to maximize partition performance through affinization. It optimizes the assignment of CPU and memory to partitions based on system topology. This results in a balanced configuration when running multiple partitions on a system.

When a partition is first activated, the hypervisor will allocate CPUs as close as possible to where allocated memory is located to reduce remote memory access. For shared partitions the hypervisor assigns a home node domain, the chip where the partition's memory is located, for each virtual processor. The hypervisor dispatches the shared partition's virtual processor(s) to run on the home node domain whenever possible. If dispatching on the home node domain is not possible due to physical processor over commitment of the system, the hypervisor will dispatch the virtual processor temporarily on another chip.

Collection Services includes metrics that can be useful for monitoring affinity activity. The primary metric to monitor is AFSCORE (affinity score) in the file QAPMSYSAFN.

iDoctor Collection Services Investigator can be used to view the placement of an IBM i partition. Right click on the current Collection Services collection and select **System Graphs → Partition placement (affinity) → Partition placement**.

A report similar to the following is generated:

Interval end time	Shared LPAR	Affinity score	Book	Node	Partition processors	Partition memory (gigabytes)	Processors % of total	Memory % of total	Total processors	Total memory (gigabytes)
2015-09-16-00.0>	0	50	0000	000A	4	91.1592	12.50%	12.45%	32	732.1826
2015-09-16-00.0>	0	50	0000	000B	7	161.8076	21.87%	22.09%	32	732.1826
2015-09-16-00.0>	0	50	0000	0008	5	114.3369	15.62%	15.61%	32	732.1826
2015-09-16-00.0>	0	50	0001	000C	4	109.6865	12.50%	14.98%	32	732.1826
2015-09-16-00.0>	0	50	0001	000D	6	90.9434	18.75%	12.42%	32	732.1826
2015-09-16-00.0>	0	50	0001	000E	3	82.2822	9.37%	11.23%	32	732.1826
2015-09-16-00.0>	0	50	0001	000F	3	81.9668	9.37%	11.19%	32	732.1826

Figure 39. Partition placement

The report states whether the partition is shared or dedicated and shows which books or drawers (Book column) and chips (Node column) the partition's processors and memory are assigned to. The report also includes an affinity score that ranges from 0 to 100, where 100 is optimal. Dynamic logical

partitioning operations may cause a partition's affinity score to decrease over time. Therefore, it is a good idea to review your partition's placement and affinity score from time to time and after DLPAR operations.

13.7 System affinity settings

IBM i provides a system value that dynamically adjusts the affinity of threads. IBM i assigns each newly created task a Home Node ID, essentially suggesting for it its preferred Power chip. Each time that a task enters a dispatchable state, the IBM i's task dispatcher first attempts to dispatch the task on a core of this preferred node.

In the case of multithreaded jobs, it is possible to indicate either on a job basis (saying*GROUP or *NOGROUP on THDRSCAFN within a routing entry or prestart job entry) or system-wide (by using the QTHDRSCAFN system value) to control whether the multiple threads of a job should share the same Home Node, or whether each should be assigned an arbitrary Home Node.
System Affinity Setting - QTHDRSCADJ
System Affinity Setting - Routing Entry

For more details on affinity on Power hardware, refer to Under the Hood: NUMA on POWER7 in IBM i.

13.8 Collect and view performance data for entire physical system

IBM i customers using multiple partitions may find it very valuable to understand overall processing capability across all logical partitions on the physical system. Collection Services has the ability to collect select key performance metrics (such as CPU) for all partitions on a single server, regardless of whether the partition is running IBM i, AIX or Linux. It collects the data known to the Hypervisor and stores the data in the QAPMLPARH performance database file.

To collect this performance data, navigate to HMC classic GUI → Partition Properties → General, and select the **Enable performance information collection** checkbox as shown:
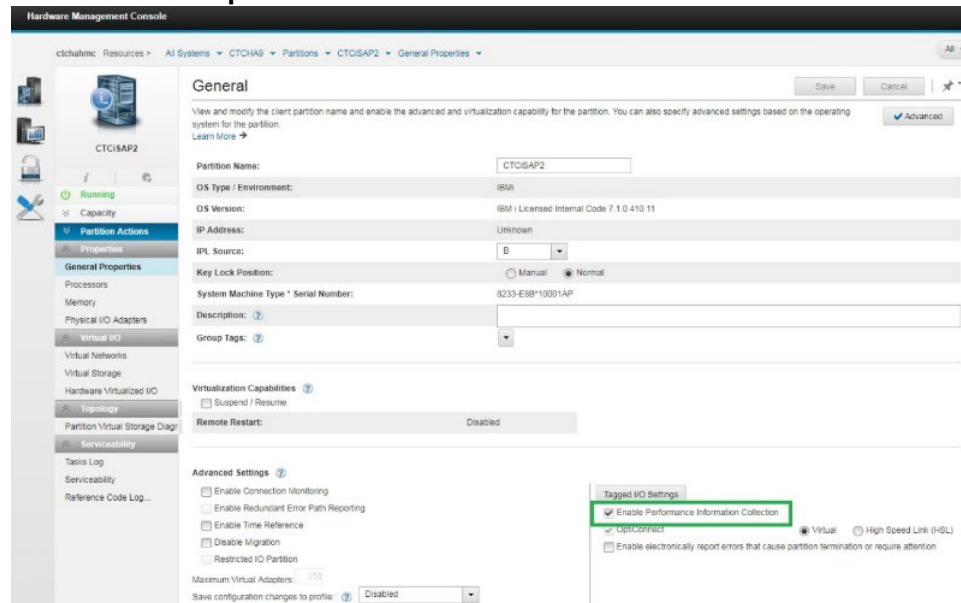

Figure 40. Partition properties

The **Enable performance information collection** setting in the HMC Enhanced GUI is located under the Advanced Settings section of the General Properties for each partition. Enable this option on one IBM i

partition to collect data reflecting work done in IBM i, AIX, and Linux partitions. Note that only IBM i supports this data collection. Visualize the collected data using charts available in both PDI and iDoctor.

In PDI, navigate to **Investigate Data** and select the **Collection Services** package. Multiple charts, including those listed and additional options, are available for analysis.

Package Name ⇅	Perspective ⇅	Description ⇅
Collection Services ⌄	Filter	processor
Collection Services	Logical Partitions Overview	including operating system, number of virtual processors, partition memory, donated CPU time, uncapped CPU time used and others.
Collection Services	Donated Processor Time by Logical Partition	This chart shows the processor time that has been donated by dedicated processor logical partitions that are configured to donate unused processor cycles.
Collection Services	Uncapped Processor Time Used by Logical Partition	This chart shows the uncapped processor time that has been used by logical partitions in excess of their entitled processing capacity configured.
Collection Services	Collection Services --> Physical System --> Virtual Shared Processor Pool Utilization

Virtual Shared Processor Pool Utilization | This chart shows processing capacity available and processing capacity utilization for virtual shared processor pools. |
Collection Services	Dedicated Processors Utilization by Logical Partition	This chart shows the dedicated processor utilization by logical partitions.
Collection Services	Physical Processors Utilization by Processor Status Overview	This chart shows a summary of processors utilization for the entire collection identifying utilization for dedicated processors and for shared processors.
Collection Services	Physical Processors Utilization by Processor Status Detail	This chart shows processors utilization over time identifying utilization for dedicated processors and for shared processors.

ɔt:void(0);

Figure 41. PDI physical system perspective

Following is an example of the logical partitions overview perspective showing various CPU related metrics across several partitions:

Figure 42. Logical partitions overview

In iDoctor, many charts are available in Collections Services Investigator under the System graphs folder:

Figure 43. iDoctor system graphs folder

For more information, refer to <u>Collecting and displaying CPU for all partitions</u> – IBM documentation.

<u>13.8.1 View performance data for multiple systems</u>

Prometheus and Grafana
<u>Prometheus</u> and <u>Grafana</u> are leading open source monitoring solutions. Prometheus delivers a time-series database to keep track of historical data and is often paired with Grafana for dashboards and visualization.

A Prometheus exporter is available for IBM i and other databases. It provides an interface for passive metrics collection. That is, Prometheus can scrape this exporter for metrics. It exports over 400 metrics (including Apache HTTP server metrics), is built around SQL and designed to be user-extensible, and lets you connect from several other monitoring stacks.
Additional information on the exporter, including installation and metrics, refer to <u>prometheus-exporter</u> – github.

Figure 44. Prometheus visualization with Grafana

Instana
Instana provides an industry-leading enterprise observability and monitoring solution, including capability for IBM i metrics.

For more information on Instana, monitoring IBM i instances, and monitoring Db2 on I, refer to:
- About Instana
- Monitoring IBM i instances
- Monitoring Db2 for IBM i

Additionally, the widespread adoption of the Prometheus protocol allows compatibility with numerous enterprise monitoring tools, such as Instana. More information on the Prometheus Exporter for IBM i, refer to Prometheus Exporter for IBM i.

Nagios
Nagios provides enterprise-class Open Source IT monitoring, network monitoring, server and applications monitoring. Multiple systems can be monitored in a central view. An IBM i plug-in is available, which provides monitoring and alerting capability for several key i metrics.

For more information on Nagios, refer to Nagios official website.

In addition, since the Prometheus protocol is so prevalent, it is understood by many enterprise monitoring tools, including Nagios.
Additional information regarding this integration can be found here:
https://techchannel.com/open-source-on-ibm-i/monitoring-ibm-i-with-prometheus/
Link to additional information on the Prometheus Exporter for IBM i can be found here:
https://github.com/ThePrez/prometheus-exporter-jdbc

The widespread adoption of the Prometheus protocol ensures compatibility with various enterprise monitoring tools, including Nagios. For more information on this integration, refer to Monitoring IBM i with Prometheus.

For more information on the Prometheus Exporter for IBM i, refer to <u>Prometheus Exporter for IBM i</u>.

13.9 How does IBM i report CPU utilization for SMT processors?

For historical reasons, processor utilization is usually calculated in the time domain. The processors used in early time-sharing computer systems were fixed-frequency and single-threaded; processing capacity per unit of time was constant. Because processor activity alternated between idle and run states, it was convenient to express processor utilization as the percentage of time that the processor was utilized (for run-state) over an interval of interest.

> **PROCESSOR UTILIZATION (%) = 100 * RUN-STATE TIME / AVAILABLE TIME**

Equation 1

When Power processors employed SMT, it was no longer possible to accurately report processor utilization based on the percentage of time that the processor was in run state. Consider the case of Power8 in the SMT8 context. When all processor threads are idle, utilization should be 0%. When no processor threads are idle, utilization should be 100%. What should processor utilization be for all other combinations of idle and run state? Let's consider some options:

- Treat the processor as idle (0% utilization). This is obviously wrong.
- Continue to treat the processor as non-idle (100% utilization). While this option may be appropriate from the system context, because a processor dispatched to one partition cannot be simultaneously dispatched to another, it is not ideal within the partition context, because it causes utilization to be grossly over-reported; it does not recognize the capacity available in the idle threads that is available to the partition to which the processor is dispatched.
- Treat processor threads as individual processors and report processor-thread based utilization in the time domain. This causes utilization to be under-reported because SMT efficiencies are much less than ideal. For example, when half of the threads are idle, there is much less than 50% additional capacity available in the processor.

To address this issue, the Power processor created the Processor Utilization Resource Register with following properties:

- The PURR is defined such that $\sum (\Delta PURR) = \Delta TIME$ over any interval. That is, the PURR apportions processor time among the processor's threads.
- The PURR assigned to the set of idle threads can be calibrated for combinations of idle and run state threads for each SMT context.

With these properties and with Processor Utilization Resource Register virtualization by the PowerVM hypervisor, processor utilization can continue to be reported in the time domain for shared and dedicated processors alike. The Processor Utilization Resource Register is what allows IBM i to report processor utilization that accurately reflects the additional compute capacity remaining in the SMT core. In terms of Processor Utilization Resource Register and time, <u>equation 1</u> becomes:

> **PROCESSOR UTILIZATION (%) = 100 * NON-IDLE PURR / AVAILABLE TIME**

Equation 2

Equation 2 is the standard partition processor utilization metric reported by WRKSYSACT and other IBM i performance tools.

13.10 How is IBM i CPU utilization calibrated for SMT processors?

The standard IBM i processor utilization metric is based on the Processor Utilization Resource Register of the Power processor. The Processor Utilization Resource Register is defined in section 13.9. PURR calibration is the key feature that enables IBM i processor utilization to accurately reflect the SMT scaling characteristics of a specific processor implementation.

For example, the following graph illustrates the difference in PURR calibration derived from Power8, Power9, and Power10 processor-based systems in SMT8 context:
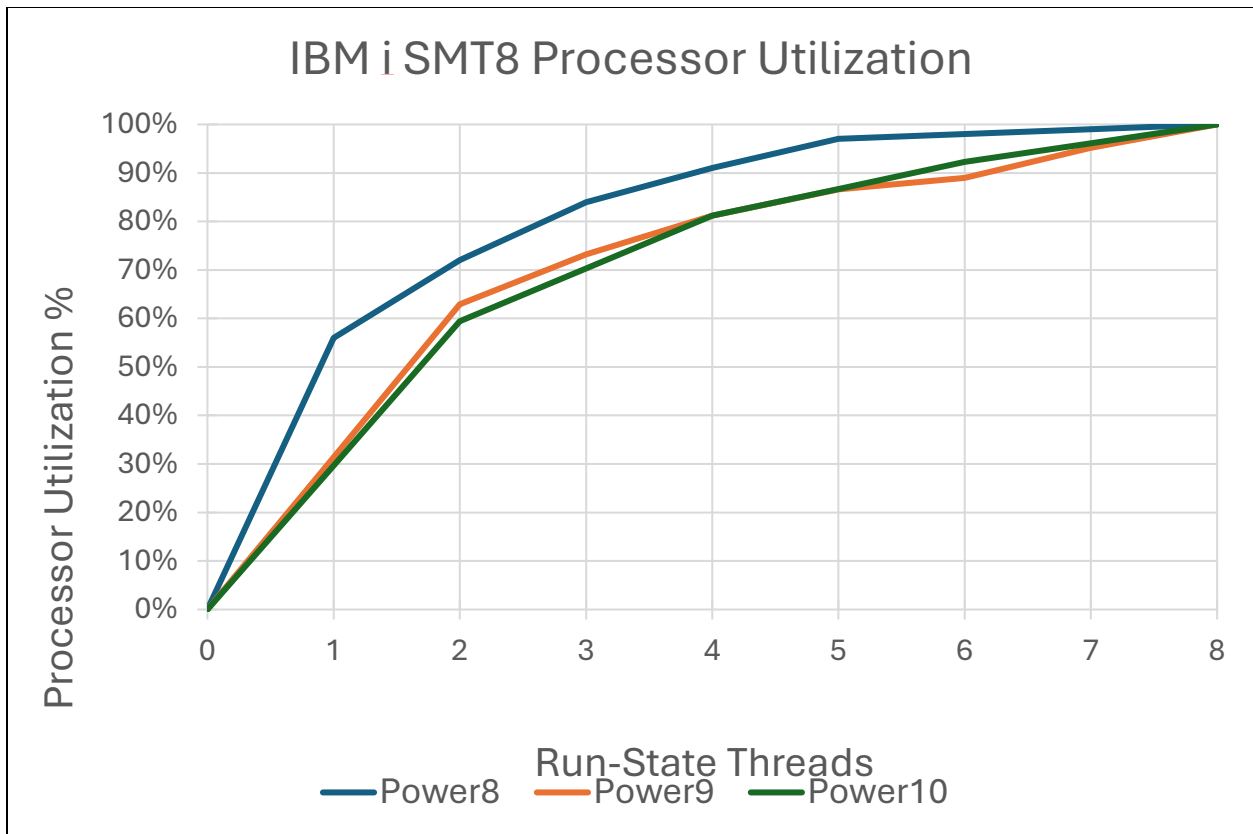
Figure 45. IBM i SMT8 CPU utilization reporting

IBM uses same workloads for Processor Utilization Resource Register calibration as those used to establish rPERF and CPW ratings, the objective being that the IBM i processor utilization metric accurately represents the percent of rated capacity utilized throughout its entire range. Ultimately, the IBM i processor utilization metric is an approximation, and actual computational utilization may be influenced by numerous factors such as workload, partition configuration, and energy management.

13.11 Power8 Redbooks

The Power8 Performance Redbooks is available at: Performance Optimization and Tuning Techniques for IBM Processors, including IBM Power8

13.12 Power9 and Power10 Redbooks

The Power10 Performance Redbooks are available at: IBM Power Systems Redbooks

14 Virtualization

14.1 Virtualization best practices
This section summarizes the Power virtualization best practices. For detailed description and latest updates, refer to the latest version of the IBM Power virtualization best practices guide: IBM Power Virtualization Best Practice Guide

For more information on how shared processor performance compares to dedicated processors, refer to How does Shared Processor Performance Compare to Dedicated Processors – IBM PowerVM community blog.

For more information on Power7 partitions and virtualization, refer to Under the Hood: Power7 Logical Partitions – IBM white paper.

For more information on PowerVM, refer to following documentation:
PowerVM Processor Virtualization 101
PowerVM Processor Virtualization 101 Part 2

14.2 Dynamic platform optimizer
The dynamic platform optimizer can be used to report affinity scores for partitions and to potentially improve partition placement for partitions with poor placement. On recent firmware DPO can be used to report the affinity score of individual partitions:

```
lsmemopt –m <managed_system> –o currscore –r lpar
lpar_name=mylpar1,lpar_id=1,curr_lpar_score=100
lpar_name=mylpar2,lpar_id=2,curr_lpar_score=80
```

Dynamic platform optimizer can also be used to predict potential placement improvements:

```
lsmemopt -m <managed_system> -o calcscore -r lpar
lpar_name=mylpar1,lpar_id=1,curr_lpar_score=100,predicted_lpar_score=100
lpar_name=mylpar2,lpar_id=2,curr_lpar_score=80,predicted_lpar_score=100
```

Dynamic platform optimizer can be used to re-assign partition resources, potentially improving affinity and partition performance. However, DPO may reduce partition performance while it is actively moving resources. For more information on using Dynamic platform optimizer, refer to Dynamic Platform Optimizer.

14.3 VIOS advisor
The VIOS advisor is an application that runs within the customer's VIOS for a user specified amount of time (hours). It polls and collects key performance metrics before analyzing results and providing a health check report. Additionally, it proposes changes to the environment or areas to investigate further. The goal of the VIOS advisor is to be an expert system that analyzes performance metrics and makes assessments and recommendations based on the expertise and experience available within the IBM systems performance group.

For more information on VIOS advisor, refer to VIOS Performance Advisor.

15 Report a performance problem

15.1 Introduction
The focus of this chapter is on what information and data should be provided when reporting a performance problem to post-sales support. However, the information provided in this chapter also applies when reporting performance problems to pre-sales and service organizations.

15.2 Define the performance problems
The first question support will ask when reporting a performance problem is, *what is slow?* A good understanding about what exactly is slow helps the performance analyst concentrate the analysis on the component(s) that is most likely causing the performance problem.

As a rule, the better the description of the performance problem the more likely there will be a quick identification of the root cause and potential resolution.

The IBM i Global Support Center (iGSC) has provided several must gather documents with detailed instructions for gathering the data required to analyze performance problems.
Master Document List for Performance

Over the last several years the iGSC has worked on automating the majority of the Must Gather Documents into a suite of tools known as QMGTOOLS. Many of the must gather documents have been updated to point at QMGTOOLS functions. When working with support, you will likely be asked to use QMGTOOLS to automate the collection of data needed to assist in debug. As a proactive measure, it is recommended to ensure the QMGTOOLS library is installed on your system and is periodically updated. For more information, refer to How To Obtain and Install QMGTOOLS.

You may also be directed to the iDoctor website to access iDoctor. If IBM Support requires iDoctor to assist with a case, a temporary license key will be provided.

15.3 Questions that help IBM diagnose the problem
The following is a list of questions that help to define a performance problem:
- Can you append more detail on the simplest, repeatable example of the problem?
- Can the problem be demonstrated with the execution of a specific command or sequence of events?
- Were any new IBM PTFs or groups applied just before the problem starting?
- Does the problem impact interactive users, batch jobs, or both?
- What cumulative PTF package is installed on the system?
- What is the level of the hiper, database, Java, tech refresh, and other applicable group PTFs?
- Were any application changes or vendor fixes applied recently?
- Has the CPU utilization changed since the problem started? Is it higher or lower than it was before?
- How has the disk utilization changed since the problem started?
- If the problem impacts a specific job, user, or subset of jobs, describe these in the greatest detail possible.
- When do any problem jobs or problem workloads run?
- Does the performance problem involve SQL access to the database?   If so, has a specific statement or type of SQL statement been identified? Has the SQL been analyzed with Visual Explain and indexed properly?
- Were there any hardware upgrades or changes since the problem happened?

- Is there any performance data available from before the problem started? If not, is there any previous data that can be retrieved from tape?

16 References

The following sections provide links to various types of online resource information and documentation for IBM i:

16.1 Performance topics
For more information on performance related topics, refer to the following blogs:
- PowerVM
- Db2 for IBM i
- IBM iSee Video Blog
- 'i Can' blog of blogs

16.2 Education and Training
The IBM Technology Expert Labs performance team can do customized formal or informal education or skills transfer on any IBM i performance-related topic. Contact the IBM i performance team leader, Eric Barsness, at ericbar@us.ibm.com for more information. For knowledge and skills transfer sessions that cover Db2 for i, SQL, and data-centric application performance, scalability monitoring, analysis, and tuning contact Kent Milligan at kmill@us.ibm.com.

16.3 IBM Redbooks
The following are selected IBM Redbooks related to IBM i performance tools:
End to End Performance Management on IBM i
IBM eServer iSeries Performance Management Tools

Although written in the V5R4 timeframe, the following Redbooks still contain applicable technical content:
- Application and Program Performance Analysis Using PEX Statistics on IBM i5/OS
- IBM iDoctor iSeries Job Watcher: Advanced Performance Tool
- Best Practices for Managing IBM i Jobs
- i5/OS Diagnostic Tools for System Administrators: An A to Z Reference for Problem Determination

The following are selected IBM Redbooks related to IBM Power performance and virtualization:
- IBM PowerVM Virtualization Introduction and Configuration
- IBM PowerVM Best Practices
- IBM PowerVM Virtualization Managing and Monitoring

16.4 Additional performance-related information
An internet search is a common and effective method for finding IBM i information. Following are selected performance-related links to key topics:
- IBM Workload Estimator: WLE
- IBM i performance tools: IBM Support: Performance Tools
- IBM i performance data investigator: IBM Support: PDI
- IBM iDoctor for IBM i: iDoctor
- IBM i wait accounting information:
  - Job Waits Whitepaper
  - The basics of Wait Accounting
  - IBM i Wait Accounting
- Best practices for managing IBM i jobs and output: Jobs Redpaper

- IBM Tivoli monitoring: [Tivoli Monitoring](#)
- SSD references:
    - [Concurrently Move Db2 for i Tables and Indexes to Solid State Disks](#)
    - [Customer Use of SSD (Solid State Drives)](#)

16.5 Database performance articles
- Db2 for i: [Database Performance and Query Optimization](#)
- DDS and SQL: [DDS and SQL - The Winning Combination for Db2 for i](#)
- Db2 tuning: [Get Your System Humming: 7 Great Tips for Tuning Db2 for i](#)
- Db2 for i indexes in parallel: [Process your Db2 for i indexes in parallel](#)

16.6 Key performance resources
The following are valuable resources to use as starting points when researching performance-related topics:
- IBM i performance and capacity: [Performance and Capacity](#)
- IBM 7.5: [7.5 Docs: Performance Topic](#)
- IBM 7.4: [7.4 Docs: Performance Topic](#)
- IBM 7.3: [7.3 Docs: Performance Topic](#)
- IBM systems hardware:
  [Power8](#)
  [Power9](#)
  [Power10](#)
- Power10 performance quick start guide: [Power10 Performance Quick Start Guides](#)
- IBM i: [IBM i](#)
- IBM i support software knowledge base: [Software Knowledge Base](#)

16.7 Performance capabilities reference note
The Performance capabilities reference manual (PCRM) was formerly the reference manual for all things related to IBM i performance considerations. Beginning in 2014, the PCRM only covers CPW rating information. This includes updates for new hardware models and CPW ratings. Previous versions are still available for download. For detailed IBM i performance information, refer to [IBM i Performance Capabilities Reference](#).

17 Summary

This white paper explored best practices for optimizing performance in IBM i environments, covering key topics such as resource management, system configuration, and performance monitoring tools. It highlighted techniques for improving scalability, managing workloads, and leveraging IBM i performance features, including storage solutions, partition configuration, and performance data collection. Practical guidance is provided for troubleshooting common performance issues and optimizing system resources for both IBM i and cross-platform applications.