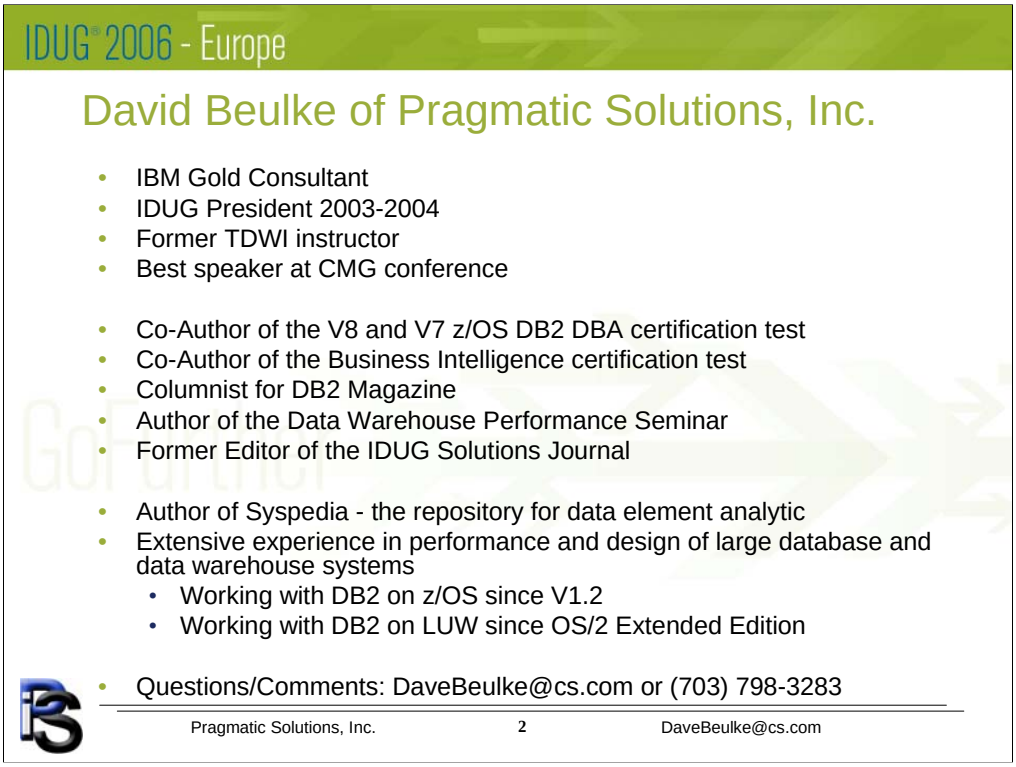
DaveBeulke@cs.com (703) 798-3283



David Beulke is an internationally recognized DB2 consultant, author and teacher. He is known for his extensive expertise in database performance, data warehouses and internet applications.

He is currently a member of the IBM Gold Consultant program, co-author of the V8 and V7 Data Administration Certification exams, co-author of the Business Intelligence Certification exam, past president of the International DB2 Users Group (IDUG), columnist for DB2 Magazine, former instructor for The Data Warehouse Institute (TDWI), and former editor of the IDUG Solutions Journal.

David is a well known speaker on data management topics at international and domestic conferences regularly speaking at the IDUG, DAMA, CMG conferences and is a recipient of the prestigious Mullen award.

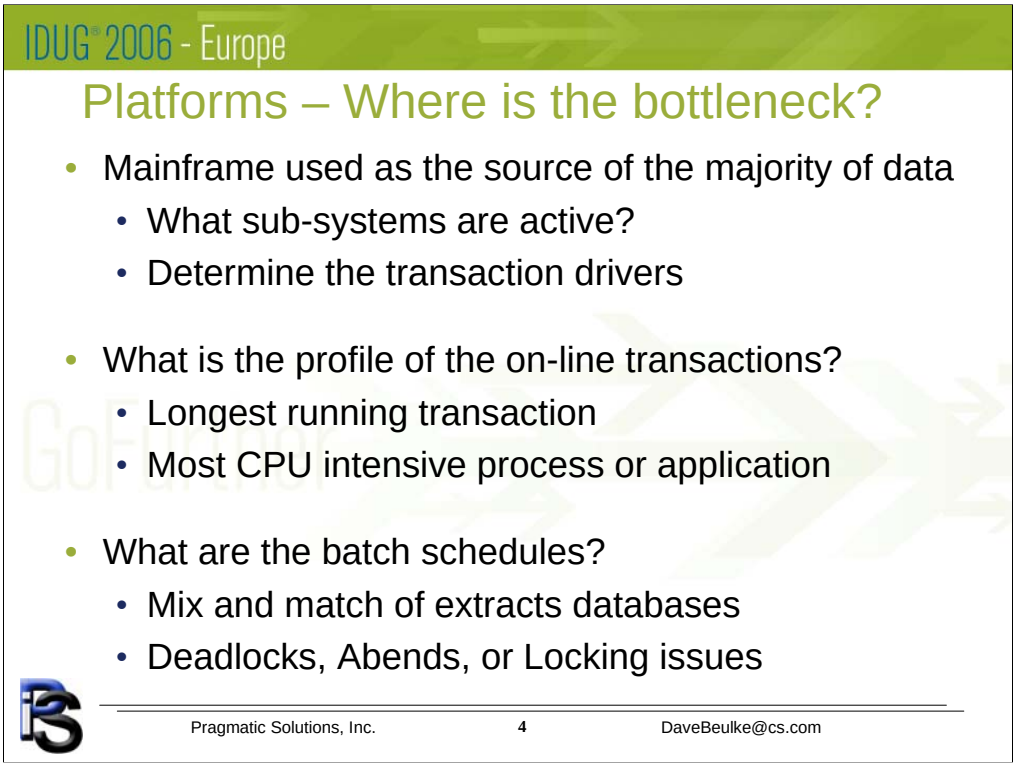
He is also President of Pragmatic Solutions, Inc. a firm he founded in 1994 and began consulting full time in 1998. He performs data management architecture, system, application and performance reviews for an average of 2-8 consulting clients and 1-6 training assignments annually.

DaveBeulke@cs.com (703) 798-3283

This presentation will discuss the research, process and impact of doing a DB2 performance review of your environments. An overall system, architectural, database and performance tuning review can dramatically reduce costs and improve application availability. By analyzing many processing performance conditions and detailing recommendations your can quickly improve database application performance, throughput and response time.

This presentation will detail activities that have been done at several clients where we evaluated and improved their production environment. This presentation details the improvements made in system configurations, database designs for mainframe, UNIX and windows platforms running data warehouse, OLTP, Web and a variety of applications. All of these techniques can be used in reducing daily CPU consumption and I/O processing within your company.

DaveBeulke@cs.com (703) 798-3283

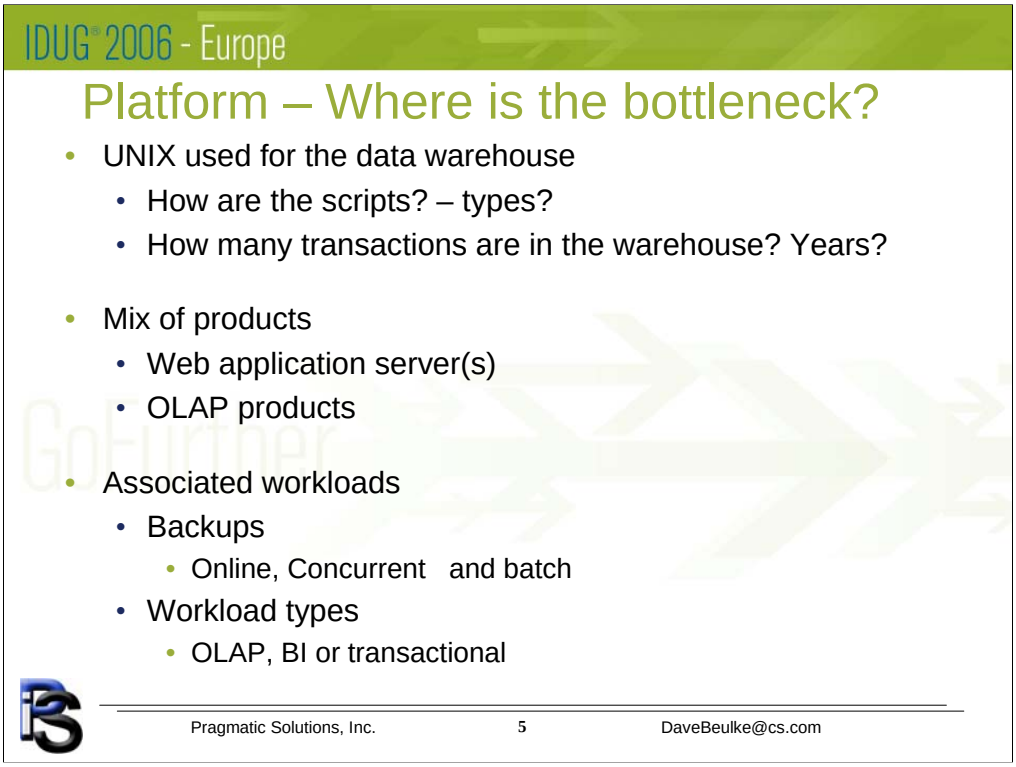


Today's computing environments are integrated and problems in one platform can cascade across other platforms ruining performance of the entire complex. This is especially true as the mainframe, UNIX and windows environments are passing data back and forth dependent on each other for inputs and outputs to get their work completed. Understanding where these transactions come from and what business processes or reasons drive them is vital to first realizing and then improving their performance.

The transaction drivers within your environment need to the evaluated to determine the most costly or longest running transactions and where their dependencies occur to your other platforms and systems. Determining the most CPU intensive processes or applications in your systems can help you identify opportunities for performance improvements and computing cost reductions.

These transaction drivers or CPU intensive processes can exist anywhere in your on-line or batch systems. The mix and match of databases, CICS, web or batch programs sometimes causes contention, deadlocks or other situations that can bring processing to a standstill. Knowing the workload scheduling is vital to determine the correct parameter settings for your system and its run-time requirements.

DaveBeulke@cs.com (703) 798-3283

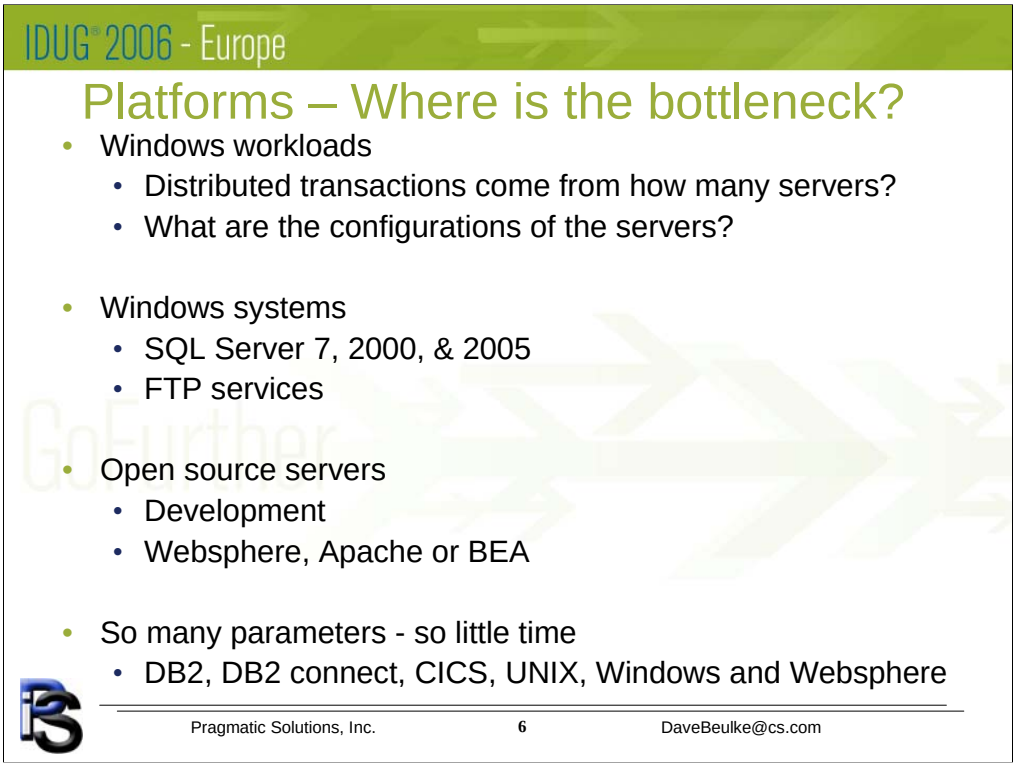


UNIX systems support a variety of applications these days especially large internet applications and large multi-terabyte data warehouse applications. These data warehouse applications typically have a large number of korn shell scripts doing a variety of activities. These scripts can be huge performance issues because they are interpreted by the operating system and offer very limit performance tuning options.

Also some databases are being stretched beyond their original capacity, For example one poorly performing system was originally design to handle 5 years of detail data. Its performance suffered tremendously because it had 8 years of data; 3 extra years of data causing the clustering, partitioning and indexes to become performance issues.

Sometimes performance problems are caused by the mix of processes or workload types. For example, the new disk technology allows concurrent database backups along with normal processing. Even though the new hardware components can support these activities they can become very CPU intensive steal precious resources from other critical processes.

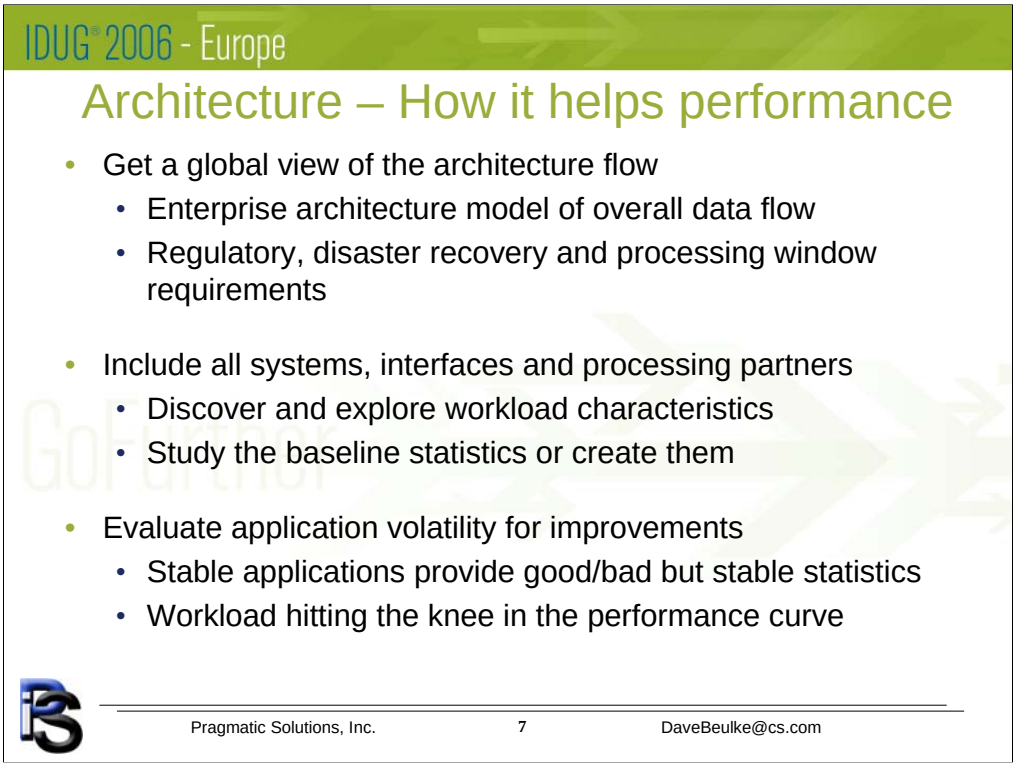
DaveBeulke@cs.com (703) 798-3283



Many platforms have various systems and transactions that can be originated from anywhere. With the internet exposure and customer service capabilities of many corporation web applications, servers from around the world could be feeding transactions to your systems.

Internal systems need to be specifically tuned to handle their transactions workloads, locking requirements, and database access. These many systems have a variety of different parameters, settings and considerations.

All of these parameters need to be documented and periodically reviewed so that their effects can be understood, realized and adjusted during performance problems or capacity planning reviews.

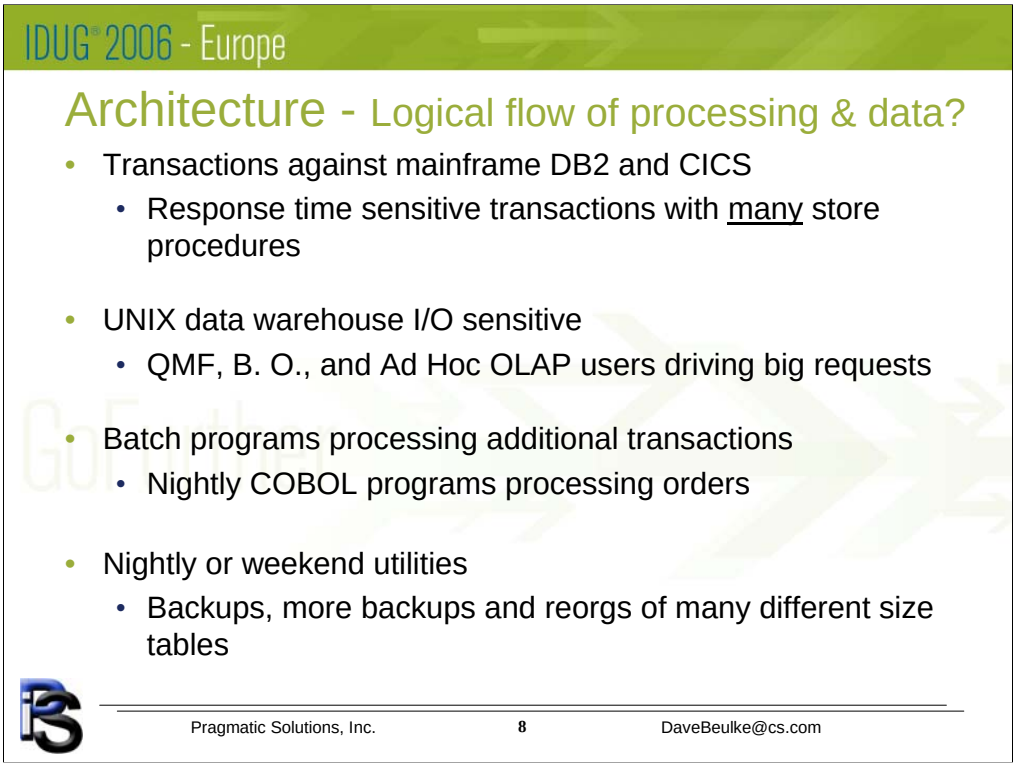


To begin a performance review evaluation of a company's environment, start by getting a global perspective of the architectural flow and data flow. Evaluate whether the platforms, systems and applications flow the data slowly across their environments or do is the complex constantly facing challenges such as missed processing windows, application errors or data recoveries. Analysis or these types of situations can often point to a management process problem or quality assurance testing scenarios. This situations are not only bad for performance but also bad for Sarbanes-Oxley or other regulatory requirements.

Analysis should include all aspects of the company computing environments and gather any processing or performance statistics available. These characteristics or statistics can give a starting point of understanding of the system. This information can also set the context of the system processing and give information on the characteristics of the system.

Also evaluate the volatility of the application source code itself. The volatility can also bring big swings in system performance and statistics. Documentation on all types of system settings, application source code, data models, and parameters changes should also be gathered. Performance can only be evaluated and improved in a stable environment and statistics are worthless if the system variables are not known. You can only improve what you can control, measure and adjust.

DaveBeulke@cs.com (703) 798-3283

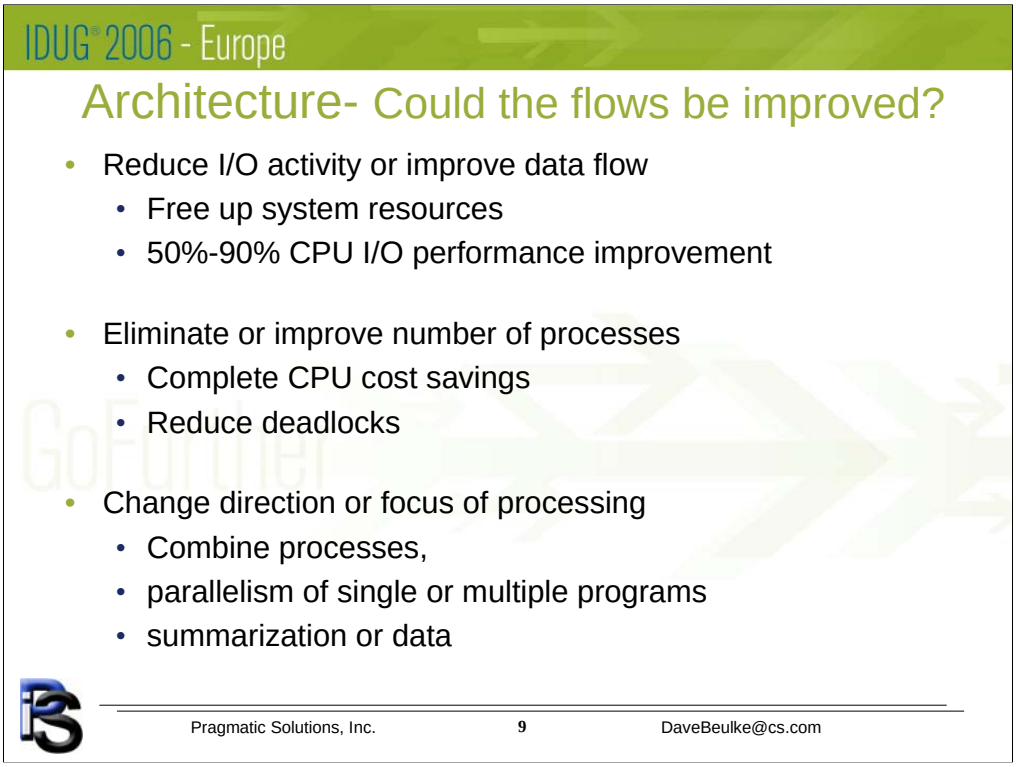


Begin to evaluate the architecture by categorizing the variety of workloads and transactions on the various platforms. Regardless of the type of applications work to understand their platforms, systems, transactions and application code considerations. This will help you understand which transactions or application are time or performance processing sensitive.

This information will also help you understand the application performance sensitivity threshold for CPU and I/O. Every application type is sensitive to either CPU or I/O as their major processing constraint and analysis should identify each system's requirements. For example most data warehousing databases have multiple terabytes of information supporting SQL workloads. In most cases these data warehouses are I/O constrained systems and may be able to benefit from more I/O paths or channels to parallelize their processing.

Next normal infrastructure capacities, such as nightly backups, database reorganizations, disaster recovery preparations also consume a large portion of the computing resources. Network communications and capacities should also be documented to understand any possible bottlenecks for distributing the workload to internet or remote users or communications with business to business computing partnerships.

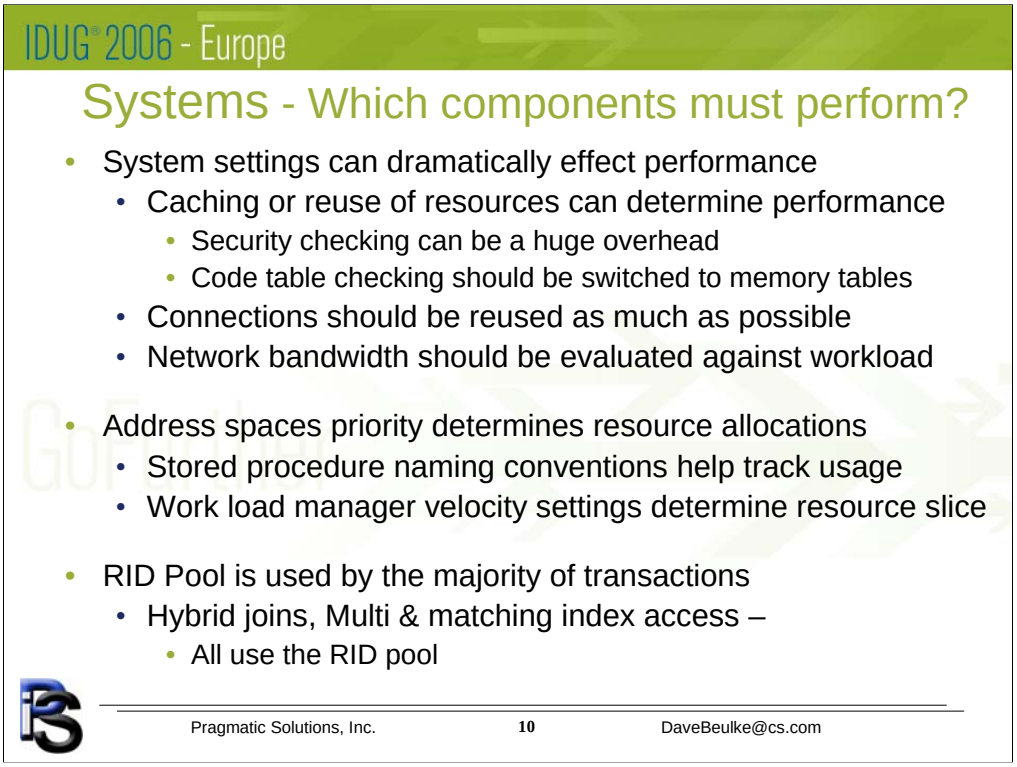
DaveBeulke@cs.com (703) 798-3283



To begin down the performance improvement path start by looking into the big opportunities in your processing to free up system resources. Understand the top ten worst performing programs that use the most I/O or CPU in your different applications. Sometimes these processes can be improve dramatically saving 50-90% of resources.

Also start by listing all the application processes within each platform, environment, system, or application. This process data list information should be combined with a typical processing I/O and CPU cost. This list can be a great way to identify duplicate processes or overlapping applications. These overlapping applications can sometimes lead to database contention and deadlocks further stressing CPU or I/O requirements.

The list of application processes in each environment can also be a starting point for evaluating whether any applications might be combined or run in parallel. Combining or parallelizing your application can dramatically reduce processing elapsed time but can focus or intensify CPU resource requirements into a small window. This situation then requires leveraging your processing schedule knowledge to squeeze the most CPU out of each processing hour.

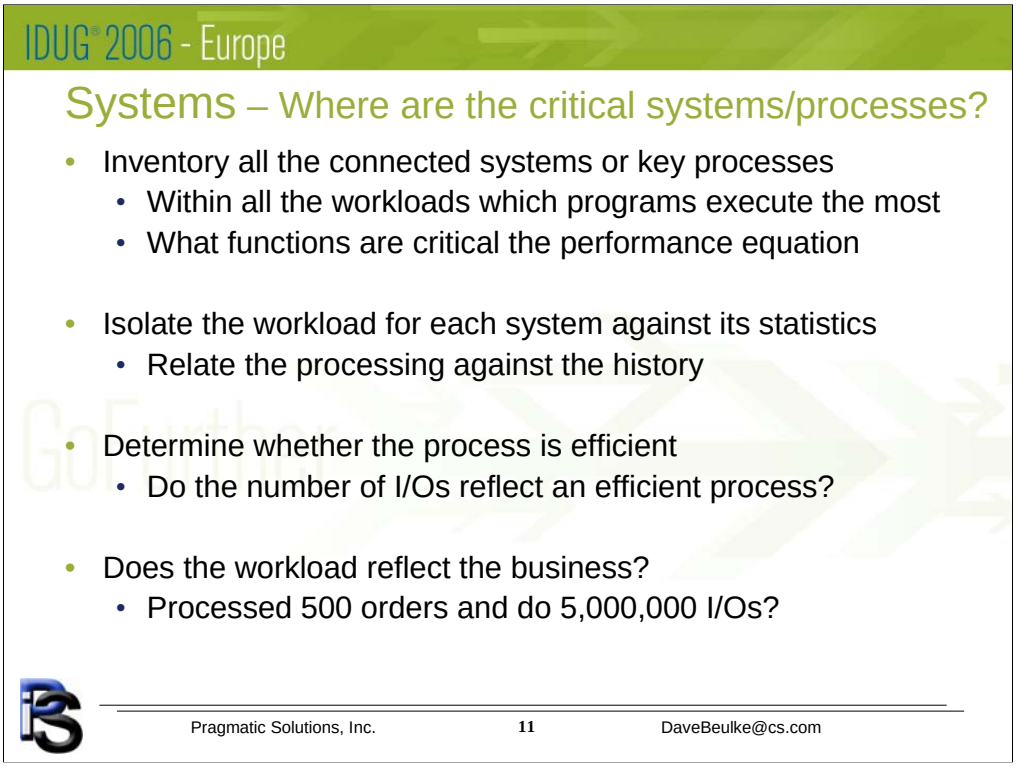


When evaluating performance of today's systems their parameter settings should be documented. With all the expanded memory and caching of data, security and network connections available memory allocations to resources can have a dramatic effect on process performance. Nothing is faster than memory and today's PCs have more memory than old mainframe computers of recent years.

Within all these platforms and systems different settings have given priority to different components and applications. These priorities need to be researched to understand how they will effect the different processes throughout the day. Regardless of the performance tuning efforts sometimes these priorities can be a major considerations for getting CPU or I/O resources.

Also these system settings can be effected by the workloads when they reach their capacity tipping point or knee of the performance curve. One example of a setting within DB2 that is sensitive to concurrent workloads is the Record Id Pool. This pool works with all the SQL that produces a list of rows to be retrieved and has a finite memory limit. If too many applications try to retrieve too many rows at the same time the limit is exceeded and query performance can suffer.

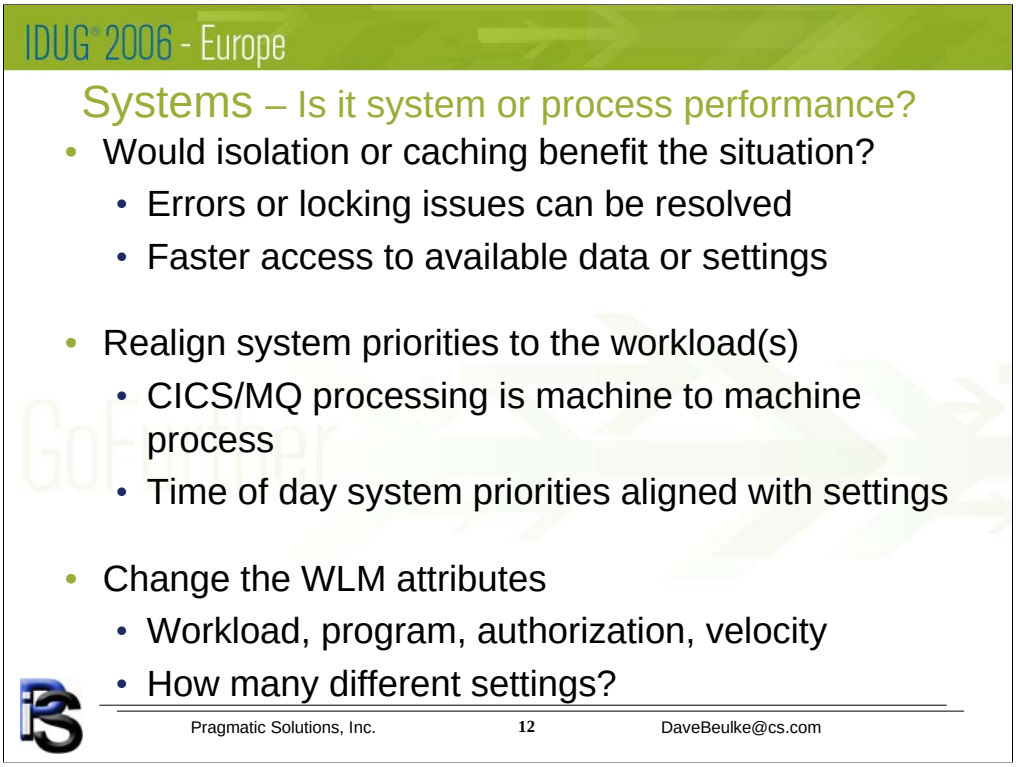
DaveBeulke@cs.com (703) 798-3283



Gathering a list of all the application processes, documentation should indicate which programs are critical or key to their processing windows. Documentation should be developed to determine the critical functions and processing average and peak resource requirements.

This critical performance and process information can be combined with its statistics to determine whether there is enough I/O and CPU capacity in the company complex. As workloads grow sometimes capacity requirements out strip computing even efficient computing environments.

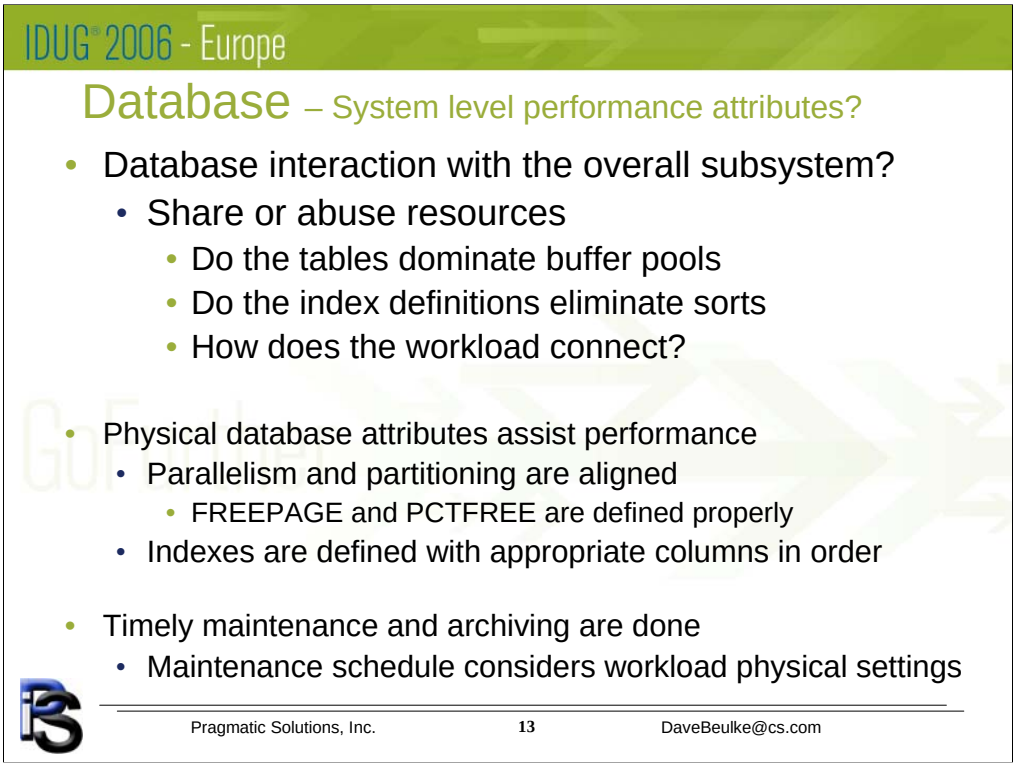
Also by comparing the performance statistics of the critical processes there should be some similarities between their processing factors such as number of order processed per hour or the amount of CPU needed. Comparing these critical processing performance figures can point to poorly written applications or architectural workflow issues that hinder the business's growth. These performance figures can help identify applications that have high CPU or I/O resource costs.



Next evaluate whether the application process could be more efficient. This analysis should focus on whether the application could benefit or is suffering from lack of I/O or CPU. Sometimes rescheduling the critical processes during a time frame when more of its required resources are available can have dramatic effects.

Another aspect of squeezing more performance out of existing resources is to realign system priorities to the businesses priorities. For example by evaluating the processing requirements during the time of the day settings can be changed to provide more data caching, CPU or I/O to different processes. These changes can be made through Work Load Manager and based on a wide variety of conditions, parameters and authorizations.

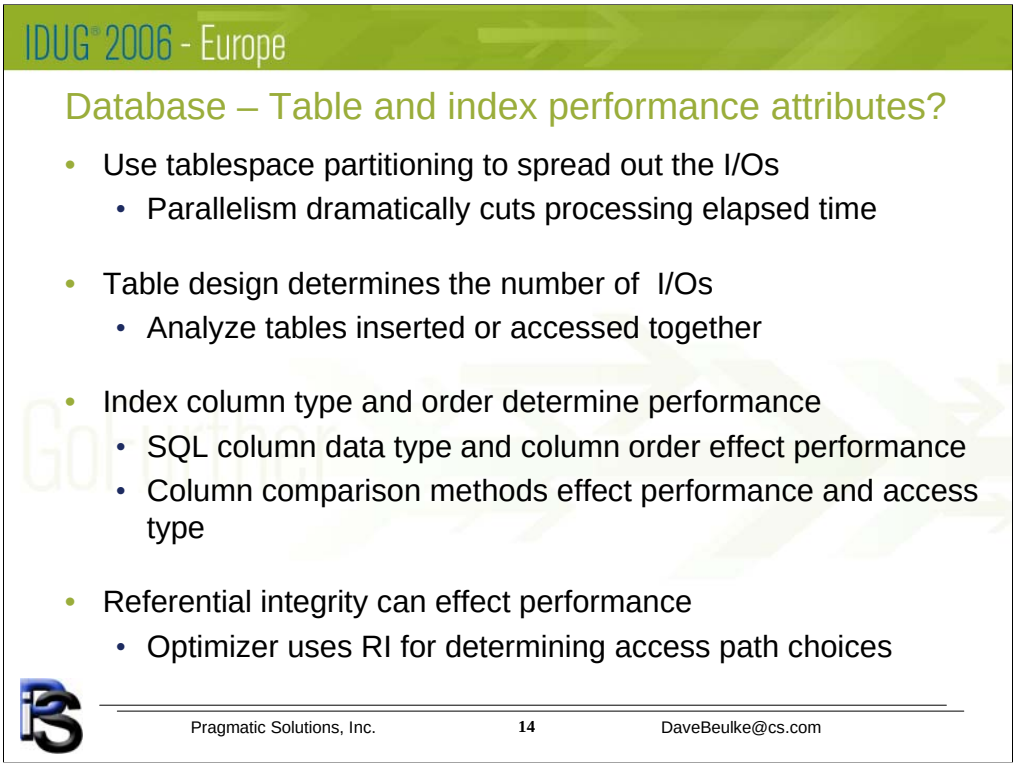
DaveBeulke@cs.com (703) 798-3283



After the different critical processes have been identified, their database requirements should be evaluated. These database table requirements should be evaluated from a number of different angles such as number of selects, inserts, updates or deletes and many others that will be detailed in the coming chapters.

Each of the database objects and tables should also be evaluated to determine their optimum physical characteristics and settings. These characteristics should be matched to their processing requirements and workload considerations for concurrency, parallelism and performance.

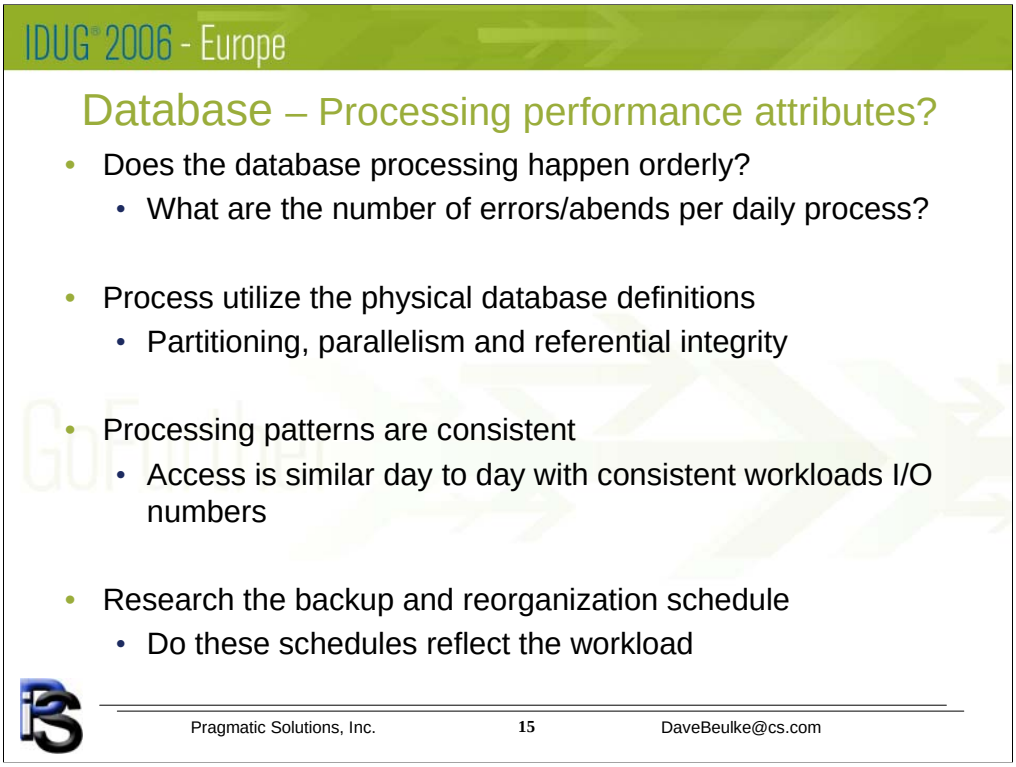
Analysis should also evaluate each database object's activity against its maintenance schedule. This analysis should guarantee that enough space is available for efficient processing regardless of the workload or number of database rows within the tables.



Tables and their associated tablespace should be analyzed for concurrency and spreading out the workload across many physical devices. Spreading the workload out will help parallelize the workload and cache as much of the data in devices as possible.

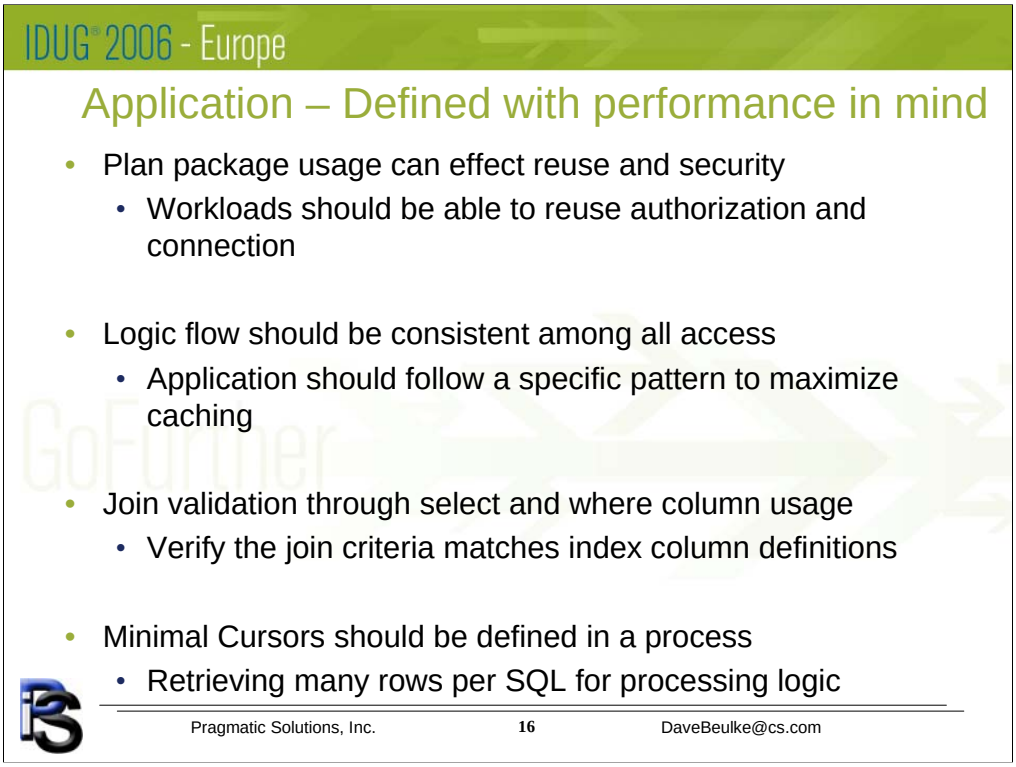
The database logical design, normalization process and physical design combines and merges logical entities together. This analysis should be done along with the knowledge of the access conventions of the workload to minimize I/Os by combining like items.

Index definitions should be analyzed and designed to combine referential integrity and access support within the same object. This cuts the number of extra indexes, minimizes their insert or update overhead and focuses the processing to support the data relationships.



The processing within the database should flow in the same direction inside the database. For example all processing might first get the customer data, then their orders and then their order details such as payment and shipping. This order should be followed by other processing to minimize deadlocks. This flows helps everyone get and release their data and database locks consistently and maximizes caching within the DB2 buffer pools.

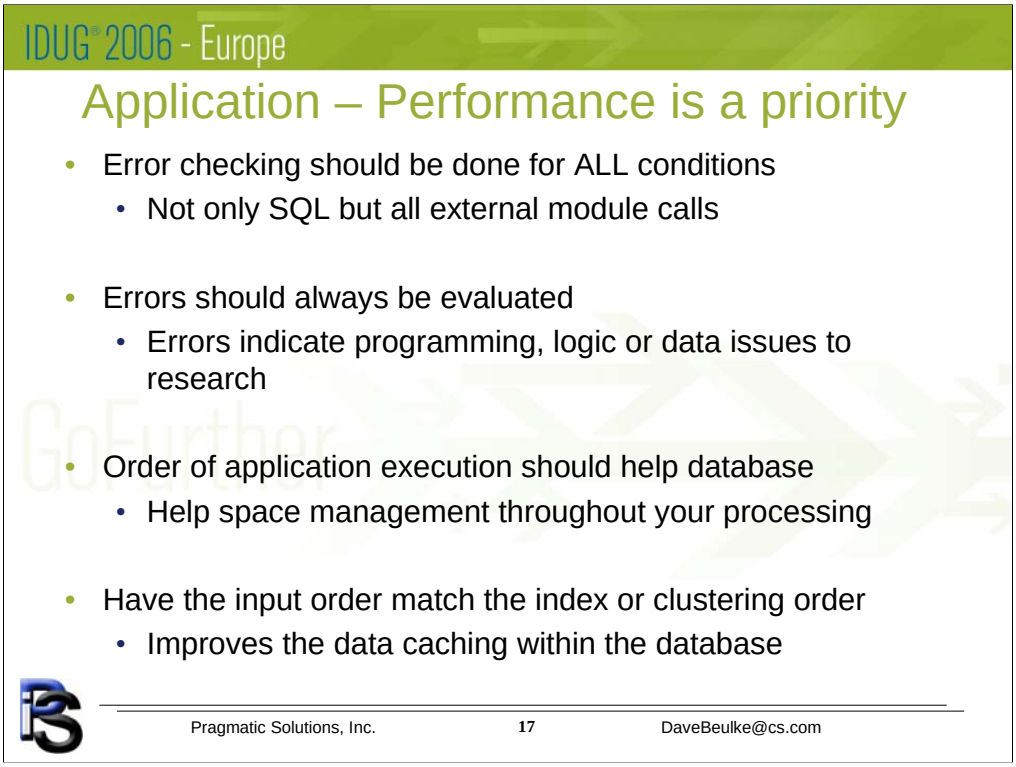
The database tables should each have their insert, update, delete and select activity estimated and documented. This data allows the database administrators to setup enough allocated space for the efficient activity. By having enough space the database is able to find space for data inserts and maintain clustering order for efficient select access.



Application should be reflective of the business flows and their users given only the authority that allows them to accomplish their jobs. Security is a major portion of every application and grouping users and their authorizations can help speed up access to resources and data.

Logic flows and pattern analysis of the application flow should show a specific consistent path for the majority of the transactions. These transaction paths and patterns should be analyzed to determine how much resources are necessary and what caching opportunities could improve performance.

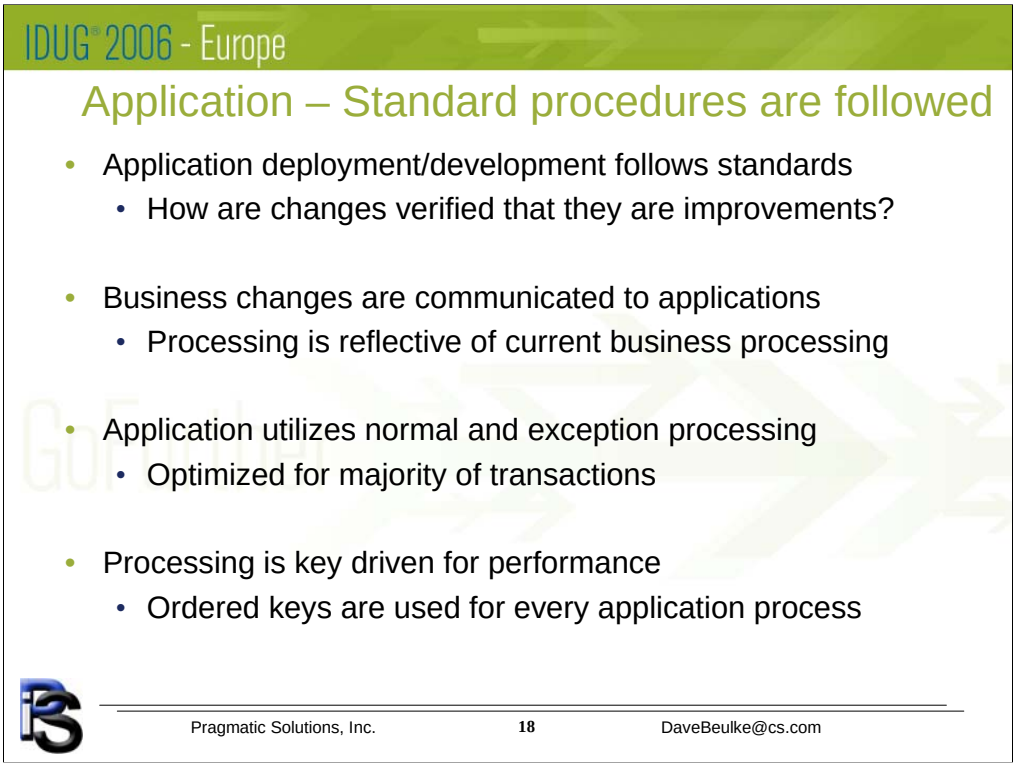
The DB2 Engine is faster than any application program or process. DB2 Joins of multiple tables can compare and retrieve data faster and should be used instead of executing SQL against single tables.



Errors evaluation processing should be evaluated after all SQL statements. This is very important since some errors conditions can cause run away processing.

For example, a java Websphere program was running on application servers sending transactions across the network. Error conditions on the server that were not checked caused the application to send the transaction to the mainframe DB2 system without filled in host variables.

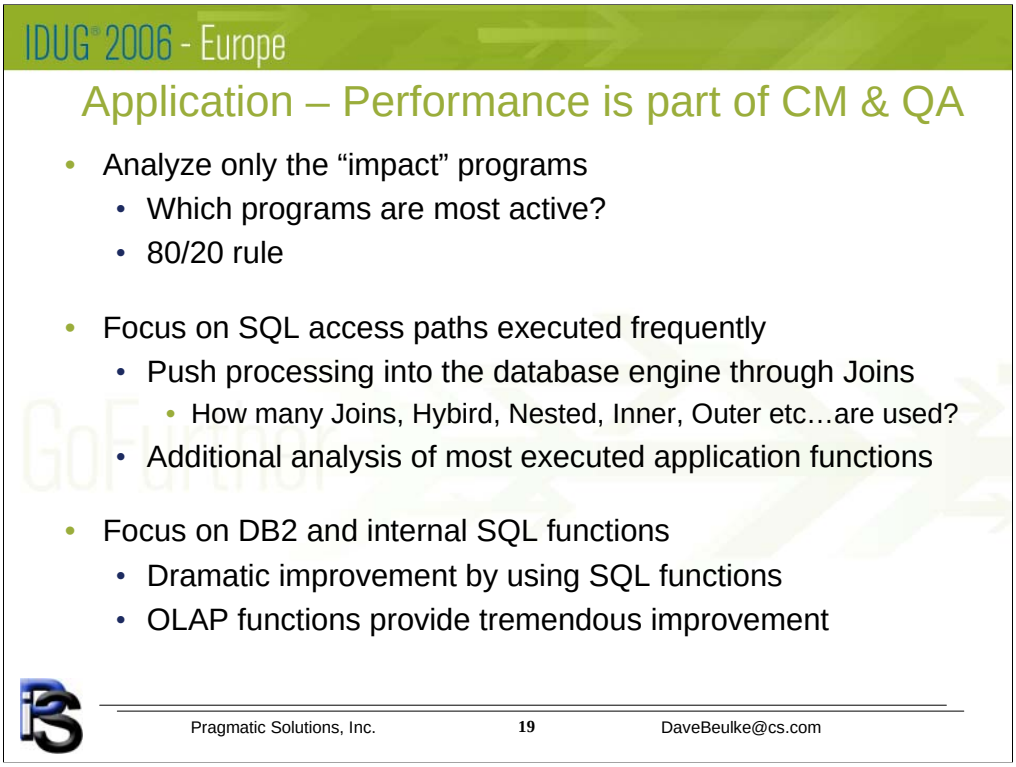
DaveBeulke@cs.com (703) 798-3283



Company development needs to have standard code and SQL Explain reviews to verify processing efficiency. These processes need to be leveraged consistently and constantly to verify that SQL access paths do not change when code changes or recompiles are done.

Program and Explain reviews need also to verify index usage and the type and number of sorts processed. Index verification should also include noting how many index columns are being used and what type of access method is being utilized.

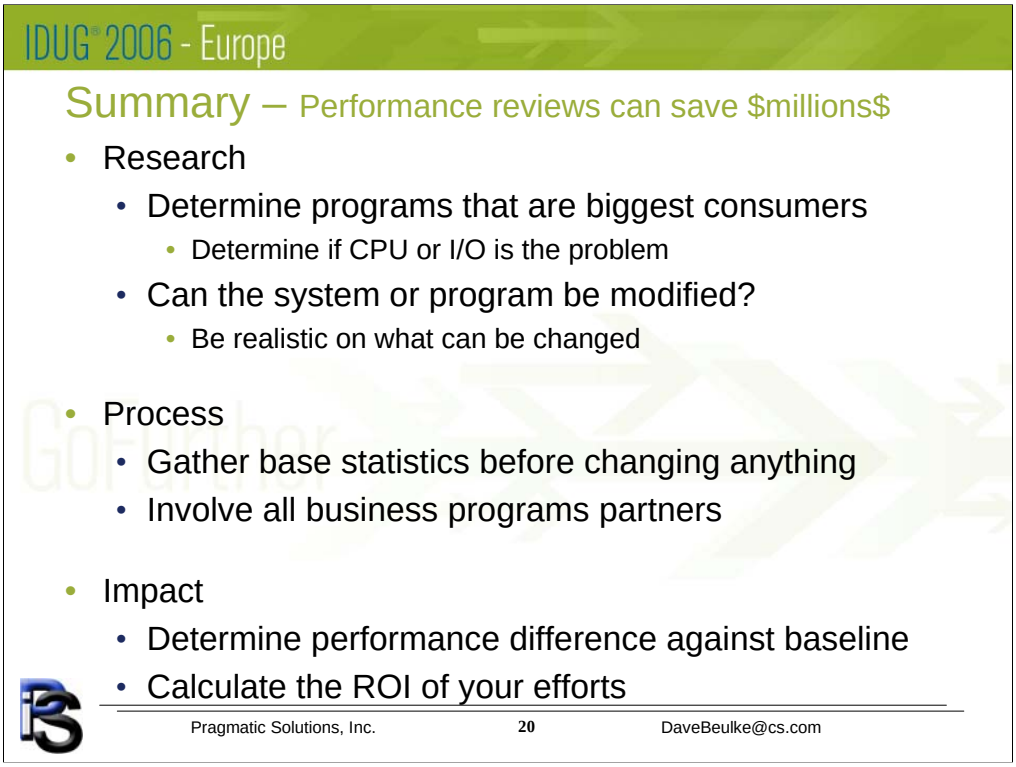
DaveBeulke@cs.com (703) 798-3283



When doing a performance review, begin by understanding the most executed programs and the most expensive processes within your environment. This will help you concentrate your tuning efforts on programs that can make a bigger impact on the overall performance within the environment.

SQL paths are a major consideration in overall performance. Analyze the access paths of each SQL statement through the Explain process. Any tablespace scans or indexes scans should be evaluated to determine how they can be improved.

Also using DB2 built-in SQL functions can dramatically help performance since no application is faster than the DB2 engine.



Performance reviews can save millions of dollars for companies.

•Tuned a DB2 z/OS system buffer pools, utilities usage and SQL paths and saved the company over $8.6 million dollars in CPU charges in the first year

•Improved the database design, adding MQTs and adjusted SQL to reduce a company's CPU charges by over 27% and saved over 750 million I/Os per day on their system.

•Adjusted buffer pools, altered their database design and added indexes to another z/OS system reducing their CPU and I/O by over 17%.

All of these and many other situations saved the company immediate bottom line dollars by reducing their CPU requirements. These performance review processes work and can help your company save bottom line IT costs.

DaveBeulke@cs.com (703) 798-3283

IDUG® 2006 - Europe

Session F16

How to do your own performance review

David Beulke

Pragmatic Solutions, Inc.

DaveBeulke@cs.com

703 798-3283