GiAPA's "Tripple-A", Automatic Application Analysis, Includes All Jobs

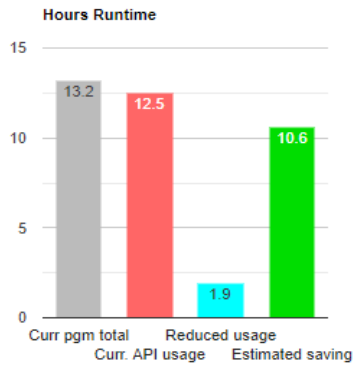
GiAPA by iPerformance

Program Optimization Hint

System: MAINSERV
781X22C LPAR 021

40.8 hours of data collected starting 2022-11-04 at 00:43

Program used	GOLCBZG/W9DOWYL	Validate transaction files received
Statement number	8	
GiAPA detected	API QUSROBJD was the active program in 3008 HotSpots (QUSROBJD = API: RTV OBJ DESCRIPTION)	
Job and user	NTRDCPN RDCPNDAXBV O66AKYRBH RDCPNDAXBV O20EBT RDCPNDAXBV	
Estimated saving	85 % of QUSROBJD = 639 minutes run time	
Effort required	Probably between 3 and 6 hours programmer time (excl. test)	



Technical explanation

APIs and CL commands provide many utility functions. Some are fairly resource intensive and repeated calls should be avoided.

Tips on how to optimize the performance

It is quite uncommon to see this API or CPP appear as the active program in several GiAPA HotSpots. We recommend investigating whether the call to this function is placed within a loop and accordingly executed e.g. once per record instead of only once in the beginning of the job. Many APIs/CPPs are fairly resource intensive since collection of the requested information requires accessing many objects. If called frequently it might improve performance to test if the API/CPP call parameters are unchanged compared to the previous call, and if so simply reuse the result. Another option is storing the parameters and results in an array so repeated calls can be replaced by a binary table look-up.

File Access Analyzed Across All Jobs Reveals Any Optimization Potential

GiAPA by iPerformance

File Access Optimization Hint

System: MAINSERV
781X22C LPAR 021

89.7 hours of data collected starting 2022-11-02 at 00:05

File accessed	B.DAFAA/LJGW#3	Product prices and descriptions table
Records in file	6,285	
GiAPA detected	652,689,227 repeated reads used for only 5,706 records	
Job and user	RPFAPIS KTPMBEDJ (3 jobs) RPFAPIS KFBVVBXBS (3 jobs) RPFAPIS EJPOW (3 jobs) (More job info shown by GiAPA Menu option 19, sel. 2)	
Estimated saving	171 minutes run time (mainly CPU time)	
Effort required	Probably between 3 and 5 programmer hours (excl. test)	

Technical explanation

Increase performance by avoiding reading the same records repeatedly. Keep such records in program work area or in a user index.

Tips on how to optimize the performance

Some tables/files with relatively few rows/records are used very heavily by several applications reading rows/records for each transactions processed. It is not uncommon to see records/rows being read more than a thousand times each by a job. Although the operating system automatically holds frequently used data in the main storage, quite some overhead remains connected with each access. Reading such records/rows into a program internal array and replacing the read with a binary lookup can provide significant CPU savings. An efficient alternativ is a user index, which also is a permanent object that can be saved. If only very few records are accessed, a simple test may be the solution: skip the reading if the record happens to be the last accessed.

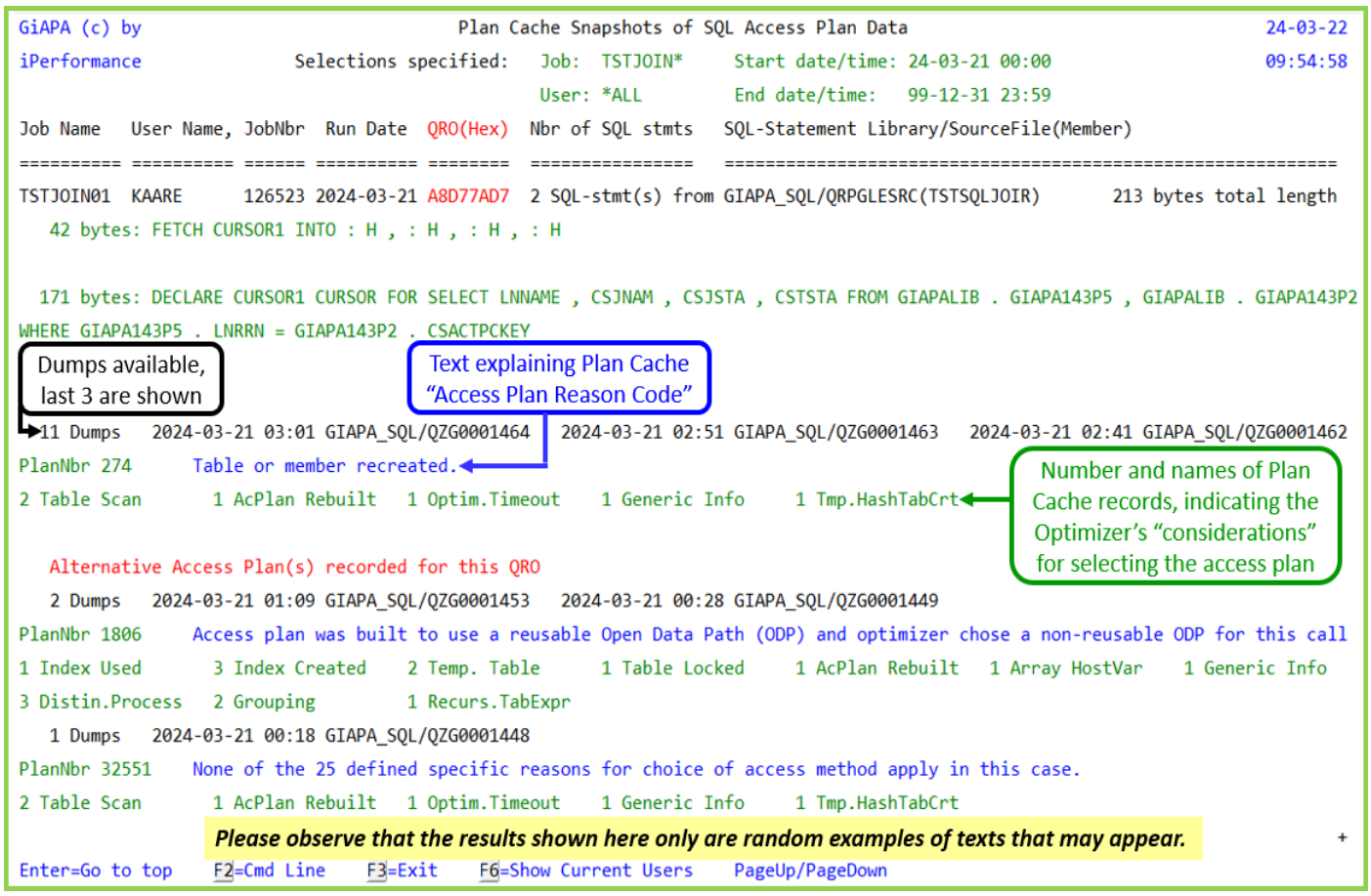
Totals for Estimated Savings of Optimization Potential Found

```
Potential Savings Found by Automatic Analysis:
   27 Improvements of program functions          2,435 Minutes
    8 Improvements of File Access Methods          277 Minutes
*** Total Potential Run Time Savings       45 Hours 12 Minutes
```

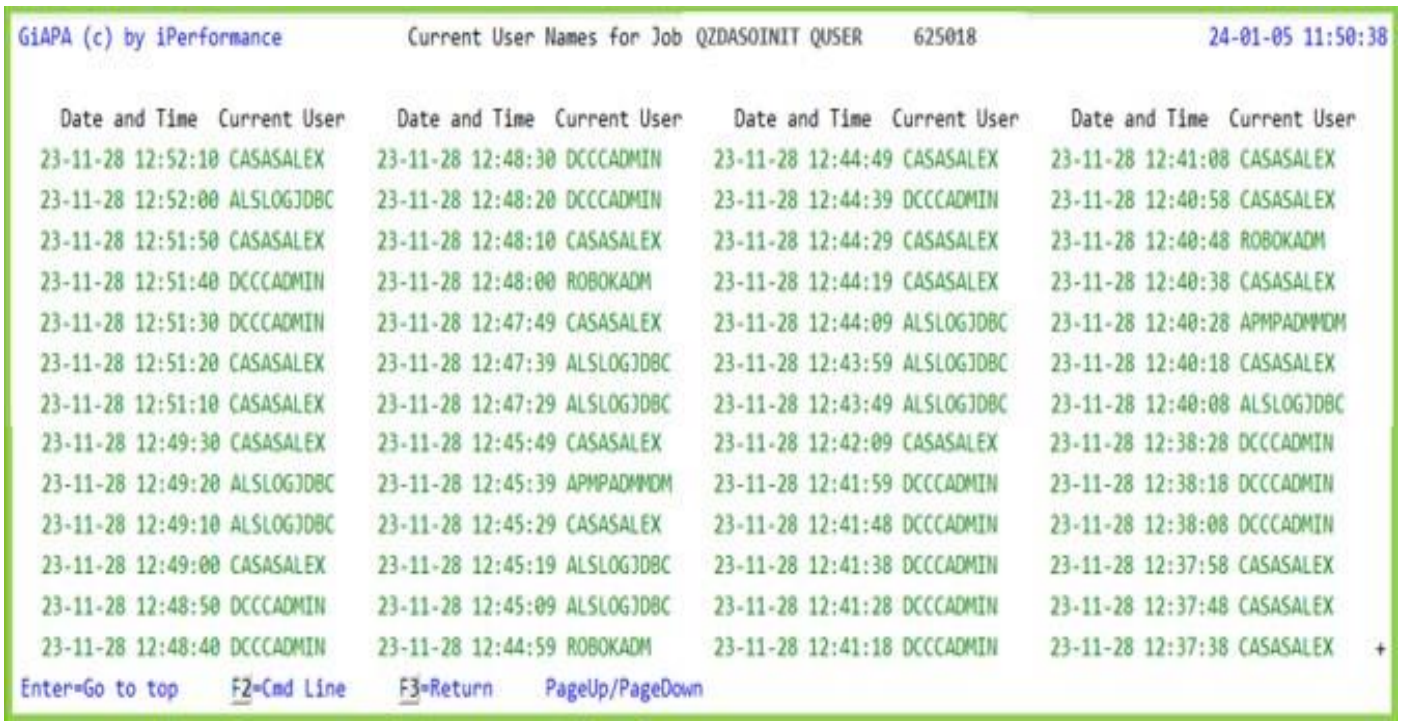
Data collection uses minimal disk space:

Data is compressed 92%

SQL Observer: Plan Cache Data Collected and Pre-Analyzed for User Selected Jobs



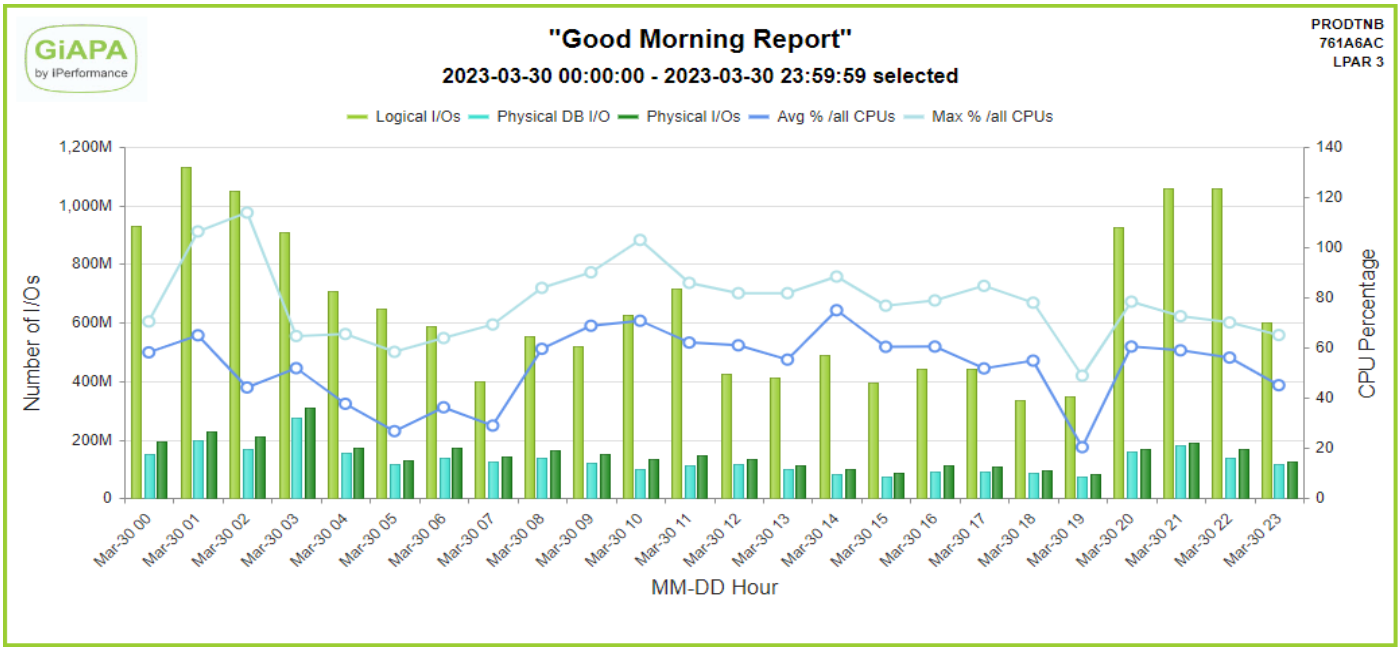
✔ **Shows Location of Snapshots Needed for Analysis Using IBM's SQL Performance Center**

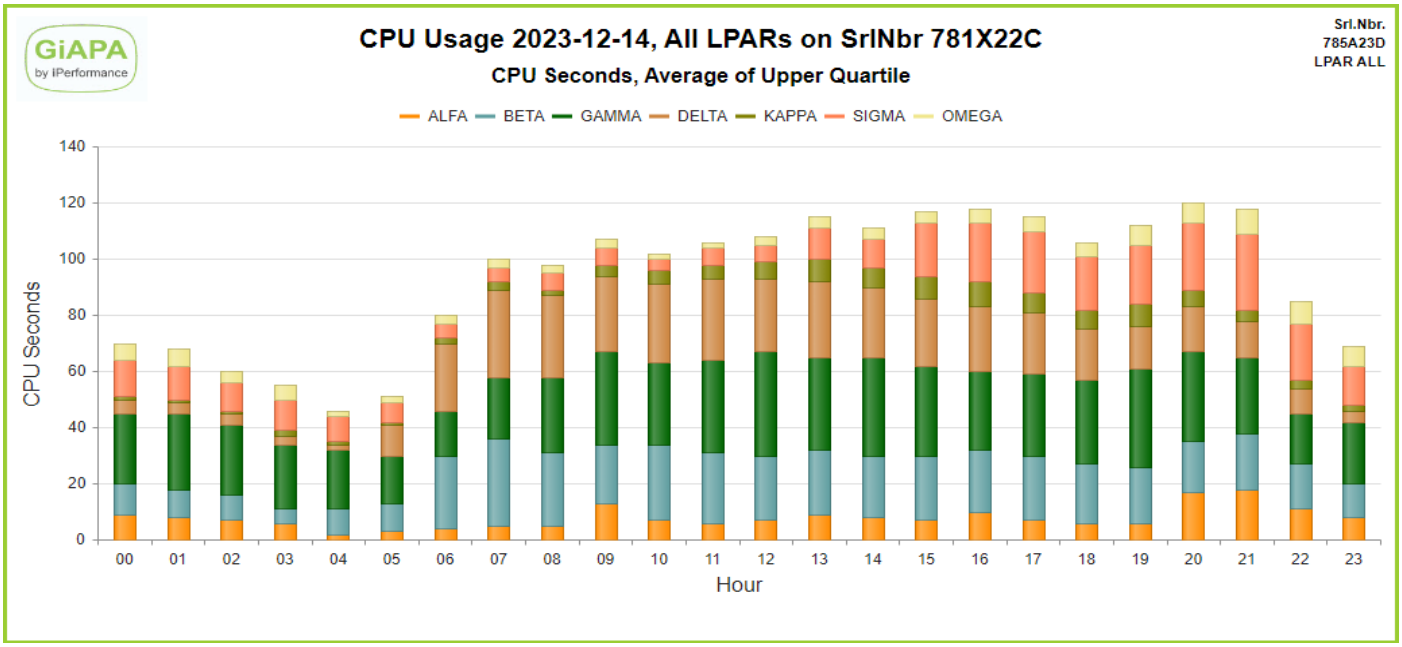


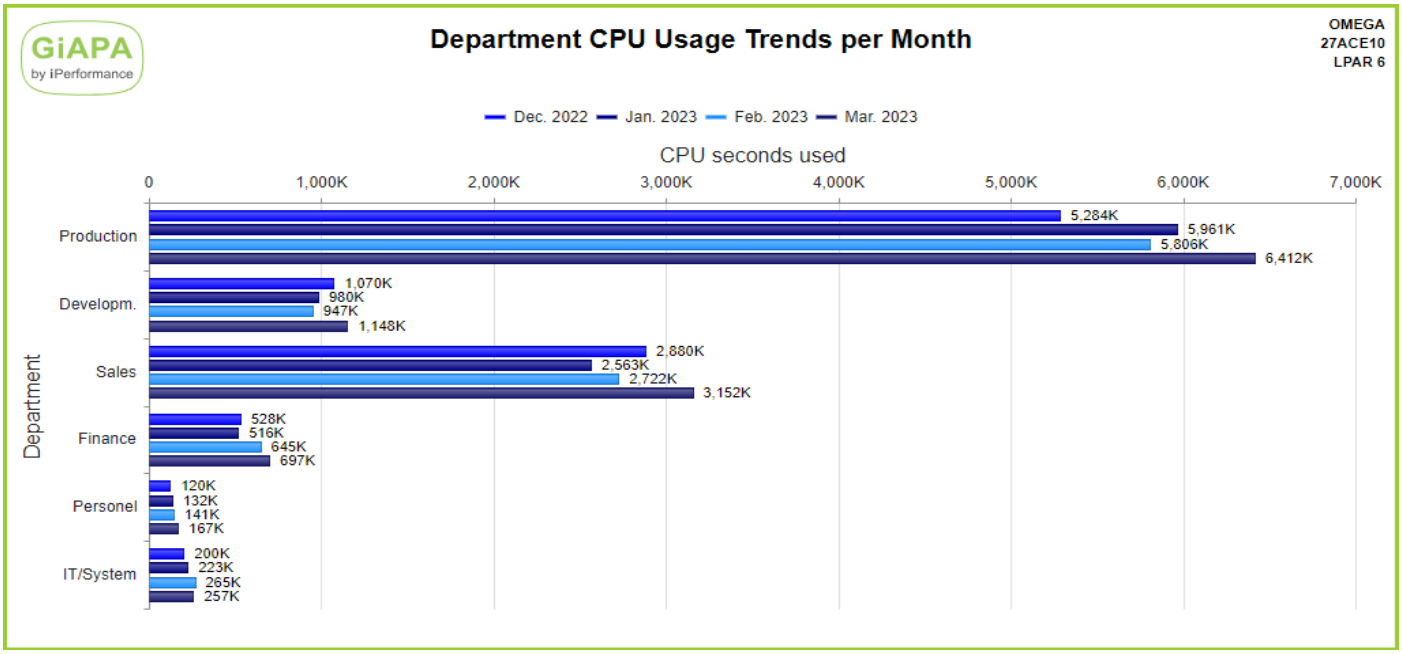
✔ **Current User Names Are Valuable Information When Analyzing Data Base Host Server Jobs**

GiAPA has <u>much</u> more to offer - please visit www.giapa.com to see [five-minute video](), [technical presentation](), [references](), and [Free Trial]()

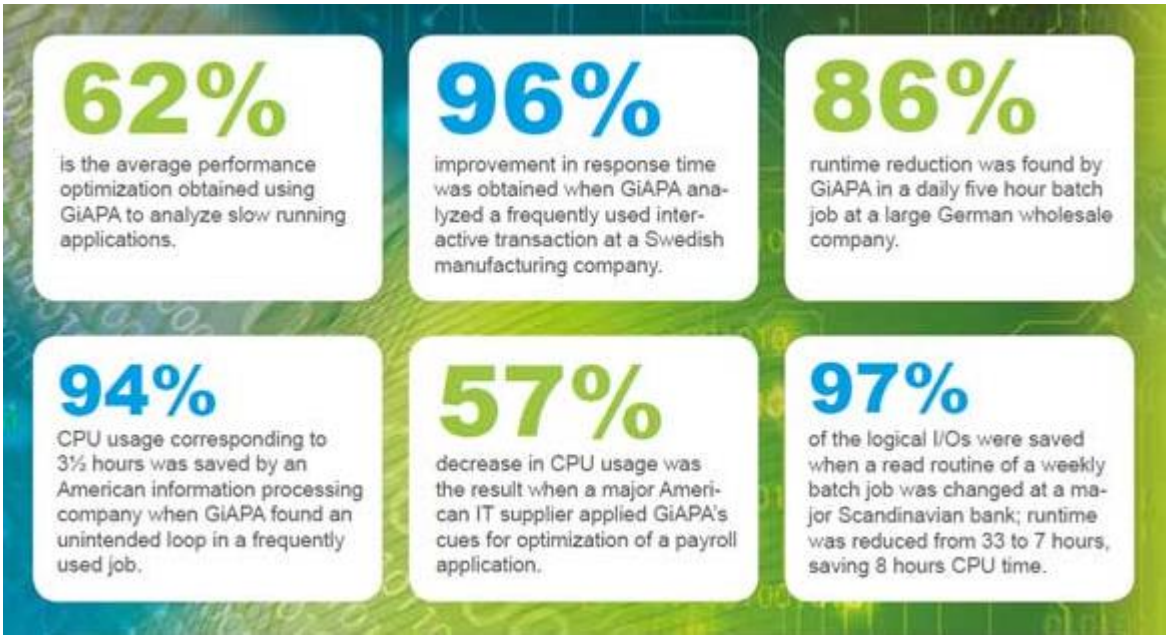
Standard or User Defined Graph Examples, Generated and Emailed by Batch Jobs







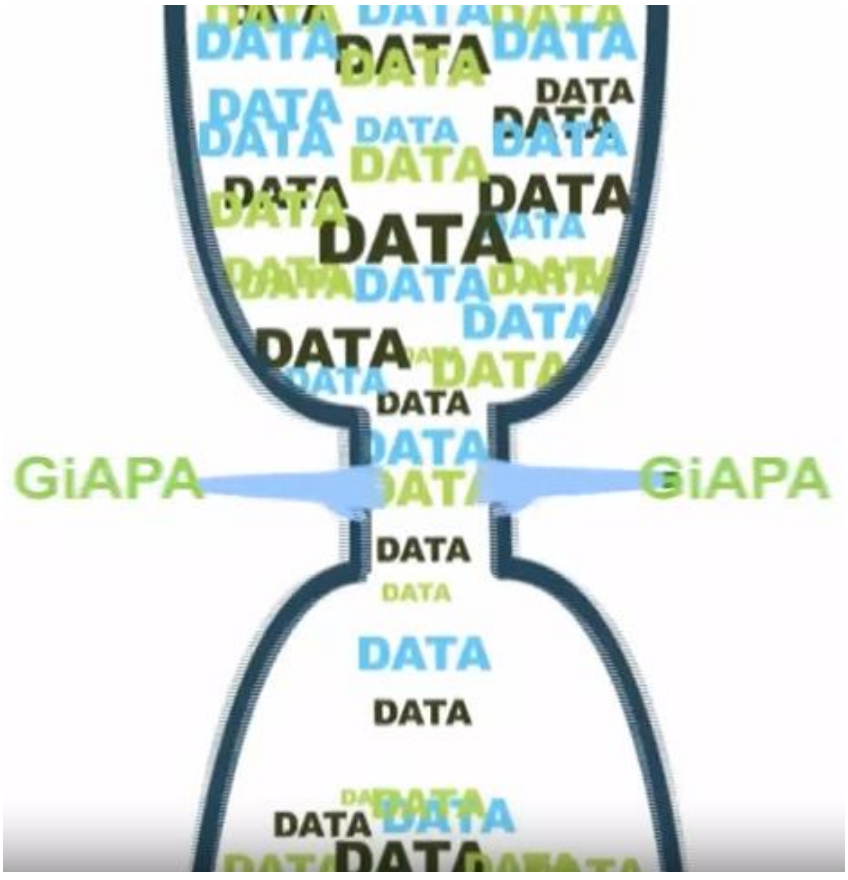
References: On https://www.giapa.com click on these fields - - -



- - - you will find a success story behind each of them!

GiAPA uses < 0.1 % CPU while collecting resource data for all jobs running – customers run GiAPA 24/7.

Analyzing all applications GiAPA will locate the bottlenecks and show modifications needed to improve performance.



GiAPA typically detects substantial saving potentials in applications believed to run efficiently, because programs producing the correct results within a reasonable time never were performance analyzed.