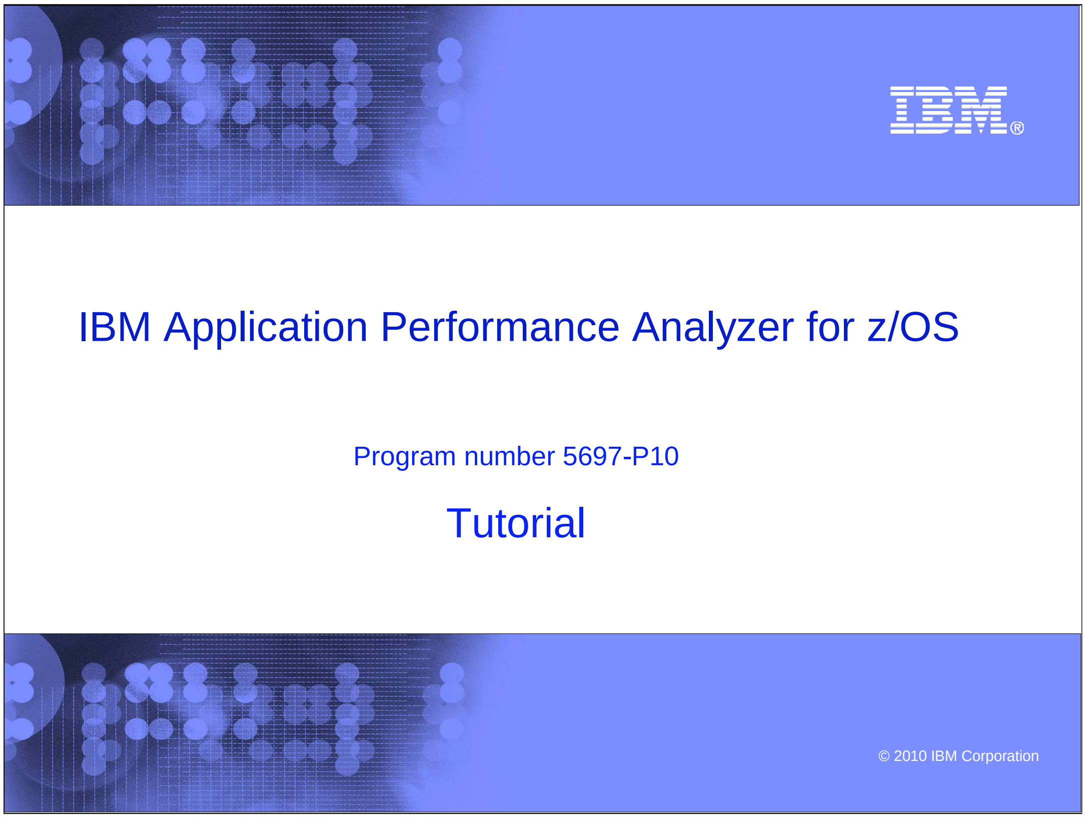
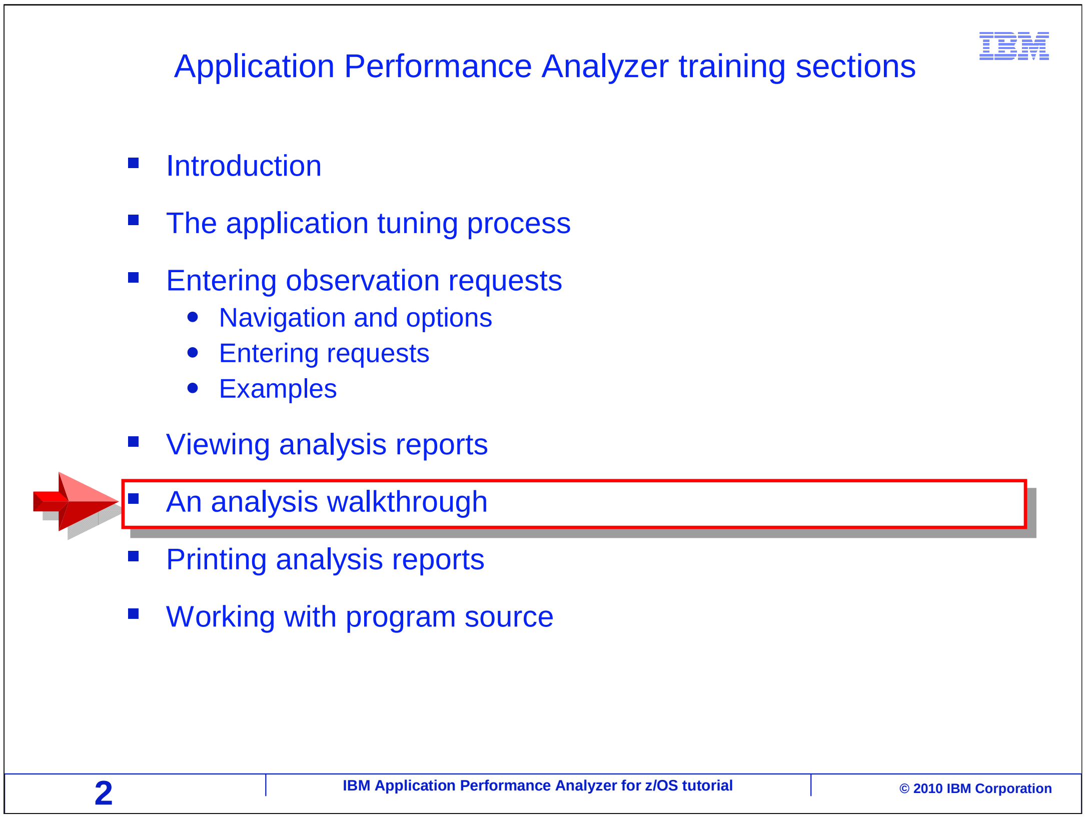


IBM Application Performance Analyzer for z/OS

Program number 5697-P10

Tutorial

This is the tutorial for IBM’s Application Performance Analyzer for z/OS®, one of the IBM zSeries® problem determination tools.



Application Performance Analyzer training sections

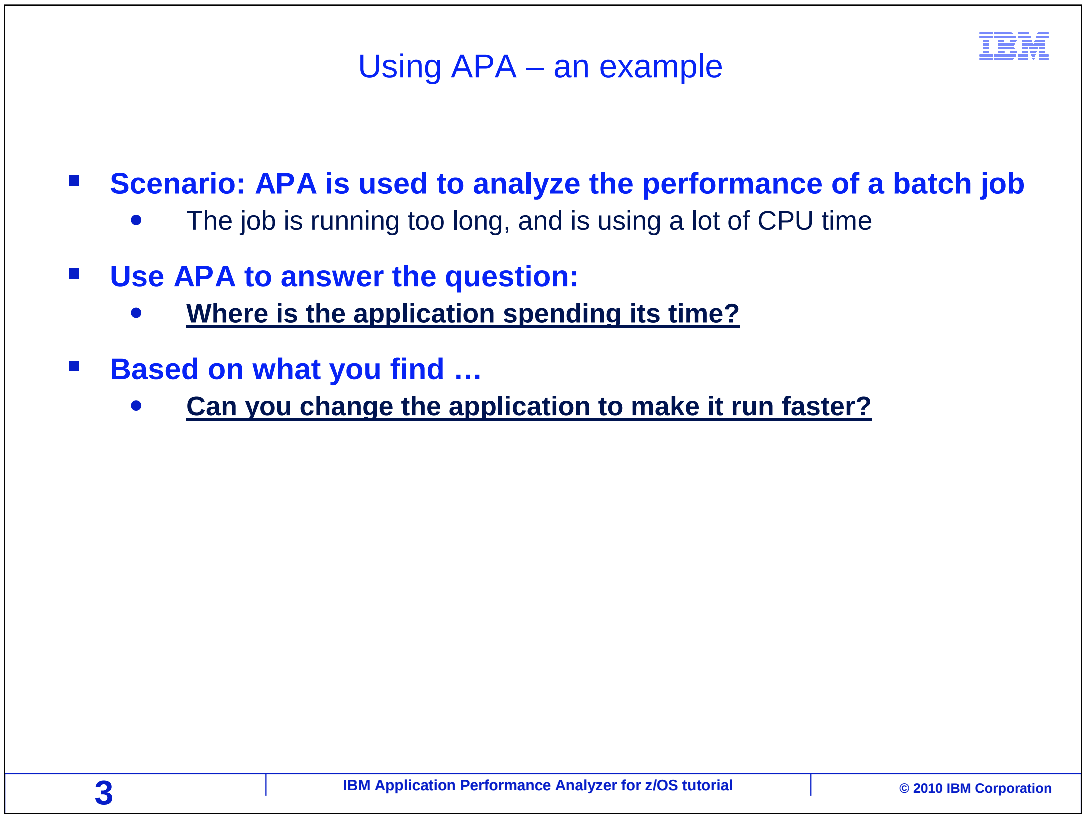
- Introduction
- The application tuning process
- Entering observation requests
  - Navigation and options
  - Entering requests
  - Examples
- Viewing analysis reports
- An analysis walkthrough
- Printing analysis reports
- Working with program source

2

IBM Application Performance Analyzer for z/OS tutorial

© 2010 IBM Corporation

In this section: “An analysis walkthrough”, there is a complete walkthrough of entering an observation request, monitoring an application, and analyzing its performance.

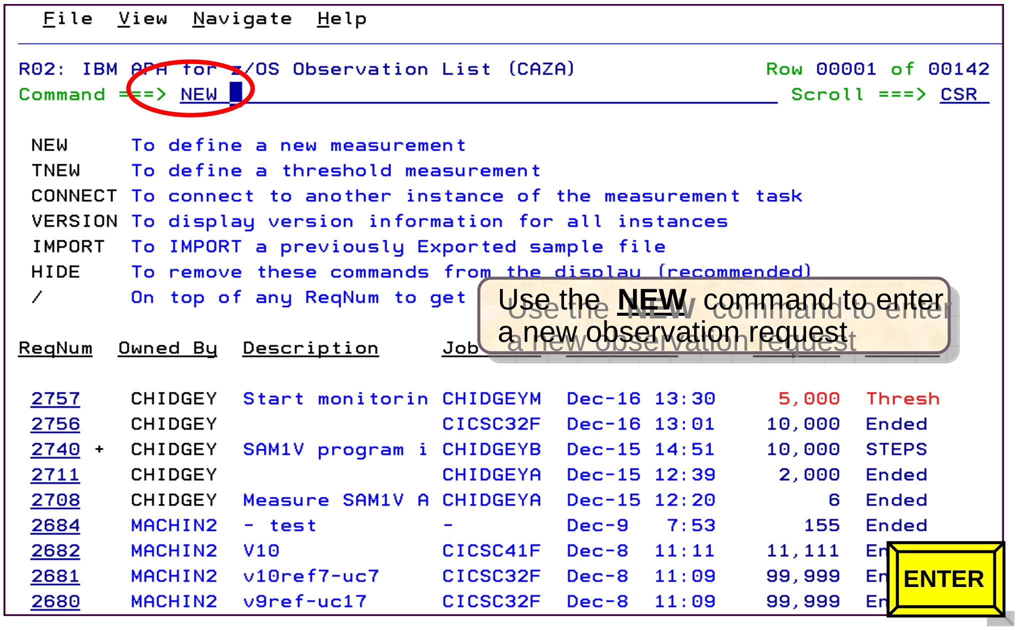


Using APA – an example

- **Scenario: APA is used to analyze the performance of a batch job**
  - The job is running too long, and is using a lot of CPU time
- **Use APA to answer the question:**
  - **Where is the application spending its time?**
- **Based on what you find …**
  - **Can you change the application to make it run faster?**

In this example, APA is used to analyze the performance of a batch job. The job in question is already running, and seems to be using an excessive amount of CPU time. You can use APA to answer the question: “where is the application spending its time?” And based on what you find, you can come up with ideas to answer the question “can you change the application to make it run faster?”

Enter a NEW observation request



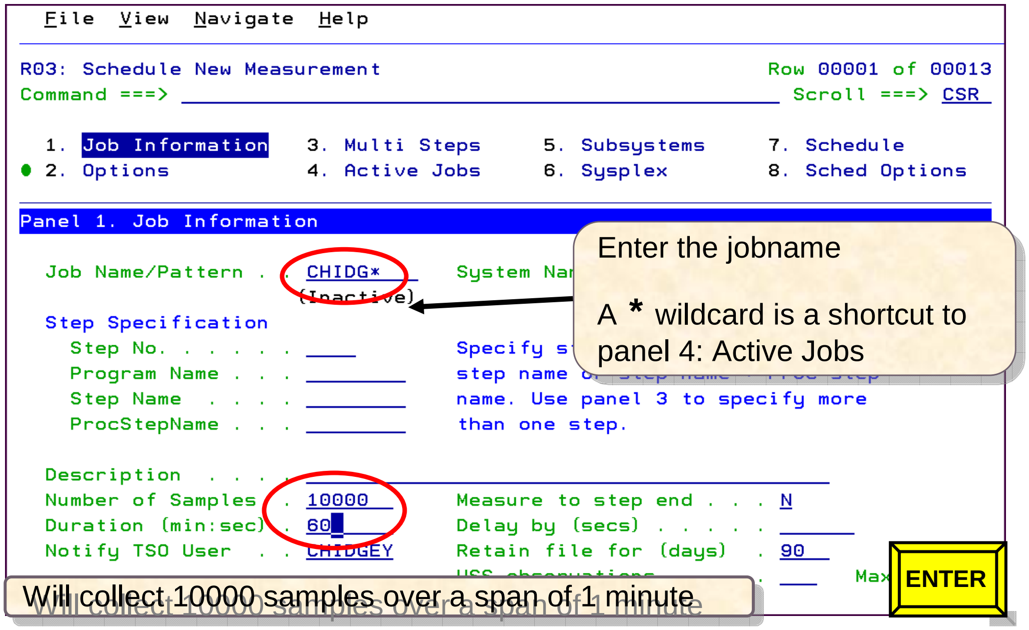
```
  File  View  Navigate  Help
R02: IBM APA for z/OS Observation List (CAZA)                Row 00001 of 00142
Command ===> NEW                                              Scroll ===> CSR

 NEW     To define a new measurement
 TNEW    To define a threshold measurement
 CONNECT To connect to another instance of the measurement task
 VERSION To display version information for all instances
 IMPORT  To IMPORT a previously Exported sample file
 HIDE    To remove these commands from the display (recommended)
 /       On top of any ReqNum to get

ReqNum  Owned By  Description     Job

 2757    CHIDGEY   Start monitorin CHIDGEYM  Dec-16 13:30    5,000  Thresh
 2756    CHIDGEY                   CICSC32F  Dec-16 13:01   10,000  Ended
 2740 +  CHIDGEY   SAM1V program i CHIDGEYB  Dec-15 14:51   10,000  STEPS
 2711    CHIDGEY                   CHIDGEYA  Dec-15 12:39    2,000  Ended
 2708    CHIDGEY   Measure SAM1V A CHIDGEYA  Dec-15 12:20        6  Ended
 2684    MACHIN2   - test          -         Dec-9   7:53      155  Ended
 2682    MACHIN2   V10             CICSC41F  Dec-8  11:11   11,111  En
 2681    MACHIN2   v10ref7-uc7     CICSC32F  Dec-8  11:09   99,999  En
 2680    MACHIN2   v9ref-uc17      CICSC32F  Dec-8  11:09   99,999  En
```

Use the **NEW** command to enter a new observation request

The job to monitor is already running, and you have noticed that it is running too long. You will enter a new observation request.

Just type "NEW” on the command line, and press Enter.

Specify request information



```
  File  View  Navigate  Help

R03: Schedule New Measurement                              Row 00001 of 00013
Command ===> ________________________________________________  Scroll ===> CSR

  1. Job Information    3. Multi Steps     5. Subsystems     7. Schedule
● 2. Options            4. Active Jobs     6. Sysplex        8. Sched Options

Panel 1. Job Information

  Job Name/Pattern  . . CHIDG*            System Nar
                        (Inactive)
  Step Specification
    Step No. . . . . .  ____              Specify s
    Program Name  . . . ________          step name o
    Step Name   . . . . ________          name. Use panel 3 to specify more
    ProcStepName  . . . ________          than one step.

  Description   . . . . __________________________________________
  Number of Samples   . 10000             Measure to step end . . . N
  Duration (min:sec)  . 60                Delay by (secs) . . . . . ______
  Notify TSO User   . . CHIDGEY           Retain file for (days)  . 90
                                          USS observations. . . . . ___    Max
```

Enter the jobname

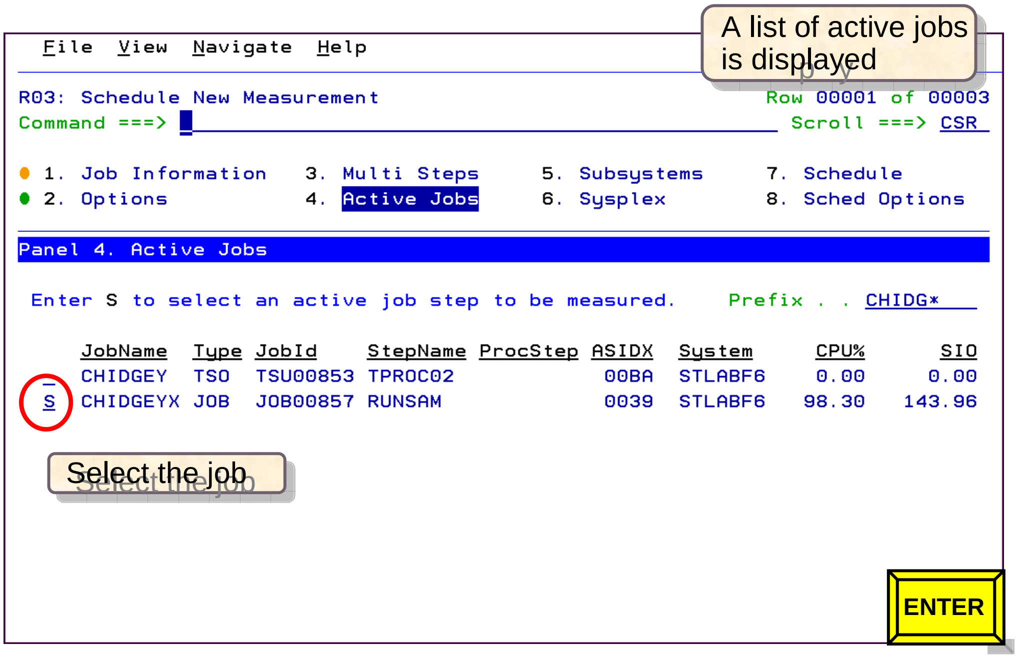
A * wildcard is a shortcut to panel 4: Active Jobs

Will collect 10000 samples over a span of 1 minute

Panel number one, “Job information”, is always displayed first. At a minimum, you will want to enter the job name, the number of samples, and the monitoring duration.

It is requested that APA should take 10,000 samples over a duration of 60 seconds. The job name is required. The first few characters of the job name followed by an * as a wildcard is entered. Remember that when you use an * wildcard in the job name, that is a shortcut to panel number four: Active jobs. Enter.

Select an active job step



```
  File  View  Navigate  Help
R03: Schedule New Measurement                                    Row 00001 of 00003
Command ===> _                                                     Scroll ===> CSR

● 1. Job Information    3. Multi Steps     5. Subsystems     7. Schedule
● 2. Options            4. Active Jobs     6. Sysplex        8. Sched Options

Panel 4. Active Jobs

 Enter S to select an active job step to be measured.     Prefix . . CHIDG*

    JobName  Type JobId    StepName ProcStep ASIDX System    CPU%      SIO
  _ CHIDGEY  TSO  TSU00853 TPROC02           00BA  STLABF6   0.00     0.00
  S CHIDGEYX JOB  JOB00857 RUNSAM            0039  STLABF6  98.30   143.96
```

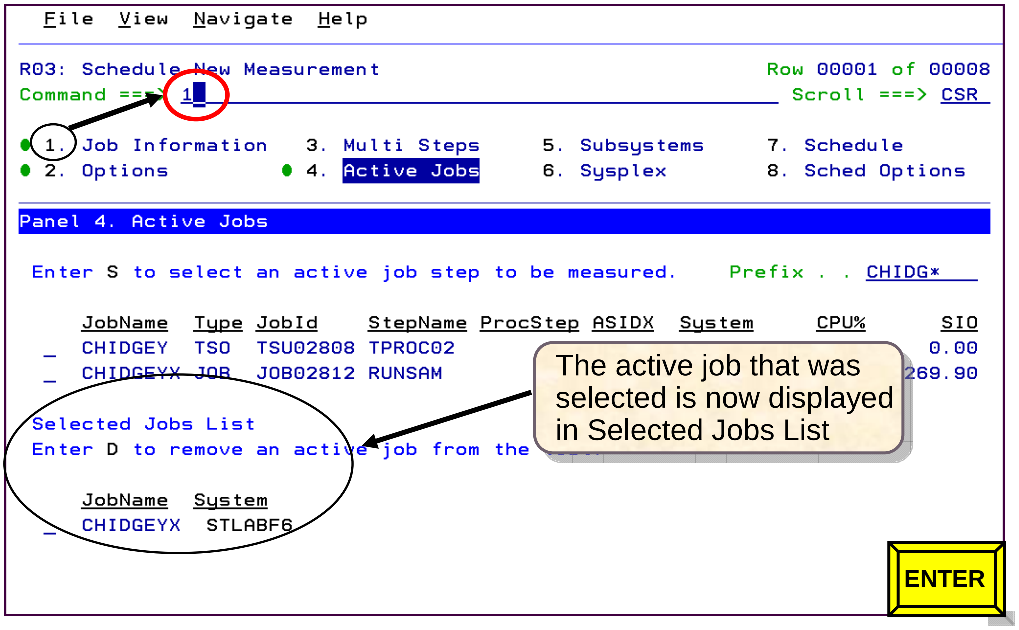
A list of active jobs is displayed

Select the job

Here is panel four, active jobs, where it displays a list of jobs running on the system that match the job name. Remember that when you select a job from panel four, APA will start monitoring the job immediately, rather than wait for the next time that it starts.

To select the job you want to monitor, use an “S” line command. Press Enter.

The “Selected Jobs List”; select “Job Information"



```
  File  View  Navigate  Help

R03: Schedule New Measurement                              Row 00001 of 00008
Command ===> 1                                                Scroll ===> CSR

● 1. Job Information    3. Multi Steps     5. Subsystems     7. Schedule
● 2. Options          ● 4. Active Jobs     6. Sysplex        8. Sched Options

Panel 4. Active Jobs

 Enter S to select an active job step to be measured.    Prefix . . CHIDG*

    JobName  Type JobId    StepName ProcStep ASIDX  System    CPU%      SIO
  _ CHIDGEY  TSO  TSU02808 TPROC02                                      0.00
  _ CHIDGEYX JOB  JOB02812 RUNSAM                                     269.90

 Selected Jobs List
 Enter D to remove an active job from the

    JobName  System
  _ CHIDGEYX  STLABF6
```

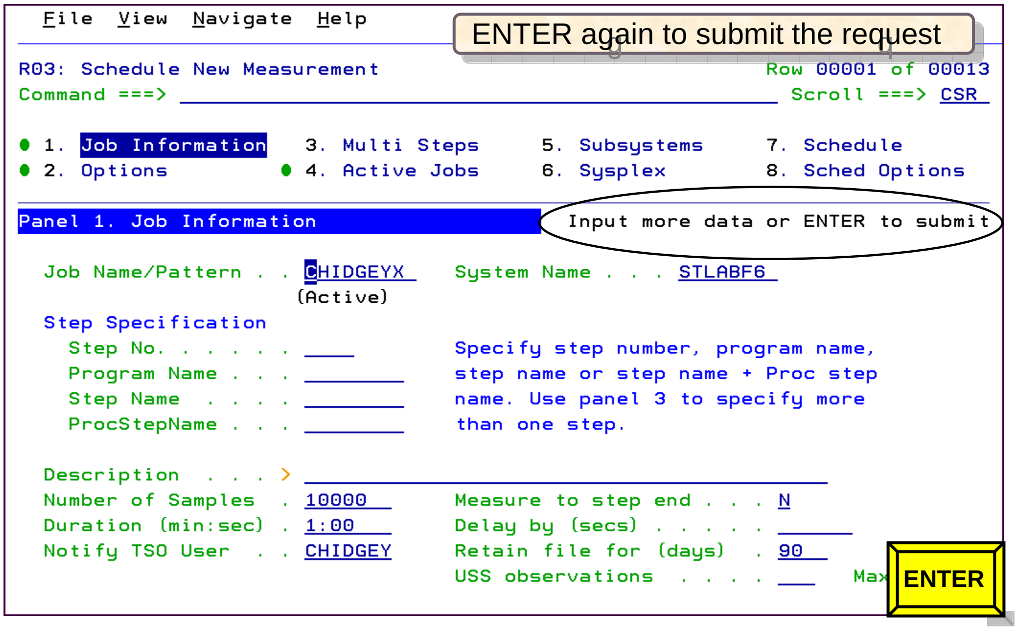
The active job that was selected is now displayed in Selected Jobs List

Notice that the active job you selected is now displayed in the lower half of the panel in Selected Jobs List.

Remember, from this panel, you can go directly to any of the other panels. Do not forget to go to panel two if you need to turn on any of the data extractors for CICS®, DB2®, IMS™, MQSeries®, or Java™. But in this case, the application does not need any of those extractors, so you will go directly to panel one, Job Information, to submit the request.

Enter one on the command line and press Enter.

Start the observation session



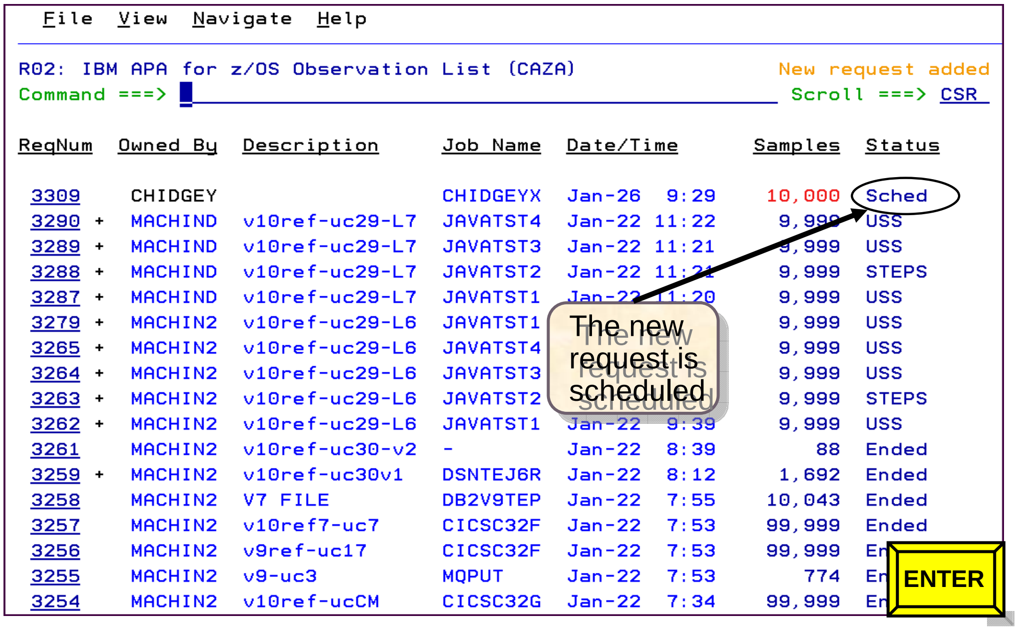
ENTER again to submit the request

```
   File  View  Navigate  Help
R03: Schedule New Measurement                              Row 00001 of 00013
Command ===> ________________________________________________ Scroll ===> CSR

● 1. Job Information     3. Multi Steps      5. Subsystems       7. Schedule
● 2. Options           ● 4. Active Jobs      6. Sysplex          8. Sched Options

Panel 1. Job Information                   Input more data or ENTER to submit

  Job Name/Pattern . . CHIDGEYX        System Name . . . STLABF6
                      (Active)
  Step Specification
    Step No. . . . . . ____            Specify step number, program name,
    Program Name . . . ________        step name or step name + Proc step
    Step Name  . . . . ________        name. Use panel 3 to specify more
    ProcStepName . . . ________        than one step.

  Description  . . . > __________________________________________
  Number of Samples  . 10000           Measure to step end . . . N
  Duration (min:sec) . 1:00            Delay by (secs) . . . . . ______
  Notify TSO User  . . CHIDGEY         Retain file for (days)  . 90
                                       USS observations  . . . . ___   Max
```

ENTER

When you get this message: "input more data or enter to submit", then APA is ready to accept the request. Press Enter to submit the request to APA.

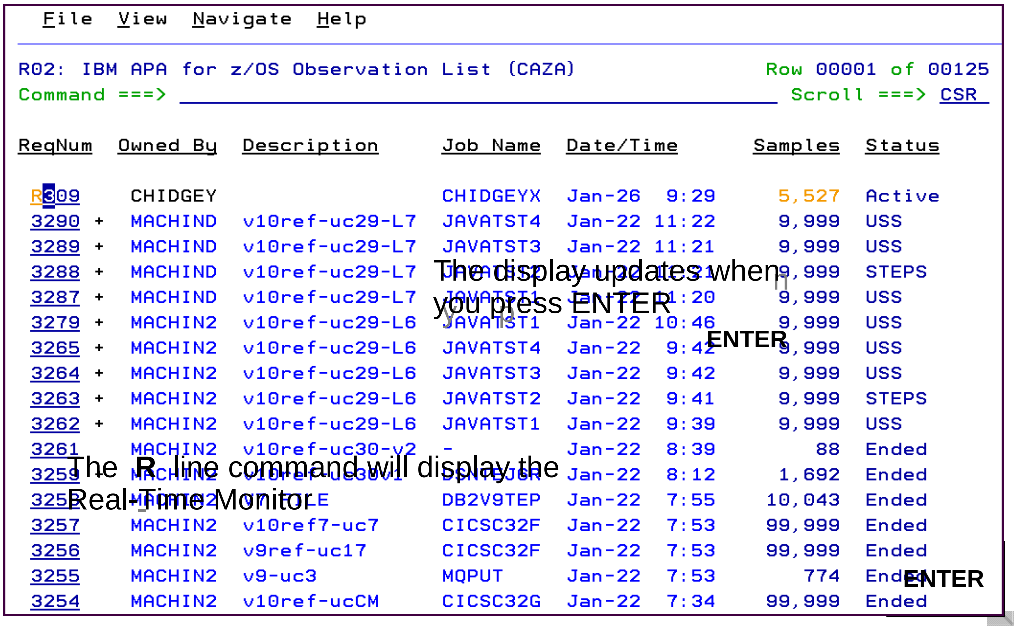
A new request was added



```
   File  View  Navigate  Help

R02: IBM APA for z/OS Observation List (CAZA)                  New request added
Command ===> _                                                  Scroll ===> CSR

ReqNum  Owned By  Description     Job Name  Date/Time      Samples  Status

 3309     CHIDGEY                 CHIDGEYX  Jan-26  9:29    10,000  Sched
 3290 +   MACHIND  v10ref-uc29-L7  JAVATST4  Jan-22 11:22     9,999  USS
 3289 +   MACHIND  v10ref-uc29-L7  JAVATST3  Jan-22 11:21     9,999  USS
 3288 +   MACHIND  v10ref-uc29-L7  JAVATST2  Jan-22 11:21     9,999  STEPS
 3287 +   MACHIND  v10ref-uc29-L7  JAVATST1  Jan-22 11:20     9,999  USS
 3279 +   MACHIN2  v10ref-uc29-L6  JAVATST1                   9,999  USS
 3265 +   MACHIN2  v10ref-uc29-L6  JAVATST4                   9,999  USS
 3264 +   MACHIN2  v10ref-uc29-L6  JAVATST3                   9,999  USS
 3263 +   MACHIN2  v10ref-uc29-L6  JAVATST2                   9,999  STEPS
 3262 +   MACHIN2  v10ref-uc29-L6  JAVATST1  Jan-22  9:39     9,999  USS
 3261     MACHIN2  v10ref-uc30-v2  -         Jan-22  8:39        88  Ended
 3259 +   MACHIN2  v10ref-uc30v1   DSNTEJ6R  Jan-22  8:12     1,692  Ended
 3258     MACHIN2  V7 FILE         DB2V9TEP  Jan-22  7:55    10,043  Ended
 3257     MACHIN2  v10ref7-uc7     CICSC32F  Jan-22  7:53    99,999  Ended
 3256     MACHIN2  v9ref-uc17      CICSC32F  Jan-22  7:53    99,999  En
 3255     MACHIN2  v9-uc3          MQPUT     Jan-22  7:53       774  En
 3254     MACHIN2  v10ref-ucCM     CICSC32G  Jan-22  7:34    99,999  En
```

The new request is displayed at the top of the observation list. Notice that its status is “scheduled”, but it will change to "active" immediately. Press Enter to refresh the list.

Status became active



```
  File  View  Navigate  Help

R02: IBM APA for z/OS Observation List (CAZA)                     Row 00001 of 00125
Command ===> ________________________________________________   Scroll ===> CSR

ReqNum  Owned By  Description     Job Name  Date/Time       Samples  Status

 R309     CHIDGEY                 CHIDGEYX  Jan-26  9:29      5,527  Active
 3290 +   MACHIND  v10ref-uc29-L7  JAVATST4  Jan-22 11:22      9,999  USS
 3289 +   MACHIND  v10ref-uc29-L7  JAVATST3  Jan-22 11:21      9,999  USS
 3288 +   MACHIND  v10ref-uc29-L7  JAVATST2  Jan-22 11:21      9,999  STEPS
 3287 +   MACHIND  v10ref-uc29-L7  JAVATST1  Jan-22 11:20      9,999  USS
 3279 +   MACHIN2  v10ref-uc29-L6  JAVATST1  Jan-22 10:46      9,999  USS
 3265 +   MACHIN2  v10ref-uc29-L6  JAVATST4  Jan-22  9:42      9,999  USS
 3264 +   MACHIN2  v10ref-uc29-L6  JAVATST3  Jan-22  9:42      9,999  USS
 3263 +   MACHIN2  v10ref-uc29-L6  JAVATST2  Jan-22  9:41      9,999  STEPS
 3262 +   MACHIN2  v10ref-uc29-L6  JAVATST1  Jan-22  9:39      9,999  USS
 3261     MACHIN2  v10ref-uc30-v2  -         Jan-22  8:39         88  Ended
 3259 +   MACHIN2  v10ref-uc30v1   DSNTEJ6R  Jan-22  8:12      1,692  Ended
 3258     MACHIN2  [illegible]     DB2V9TEP  Jan-22  7:55     10,043  Ended
 3257     MACHIN2  v10ref7-uc7     CICSC32F  Jan-22  7:53     99,999  Ended
 3256     MACHIN2  v9ref-uc17      CICSC32F  Jan-22  7:53     99,999  Ended
 3255     MACHIN2  v9-uc3          MQPUT     Jan-22  7:53        774  Ended
 3254     MACHIN2  v10ref-ucCM     CICSC32G  Jan-22  7:34     99,999  Ended
```

The display updates when you press ENTER

ENTER

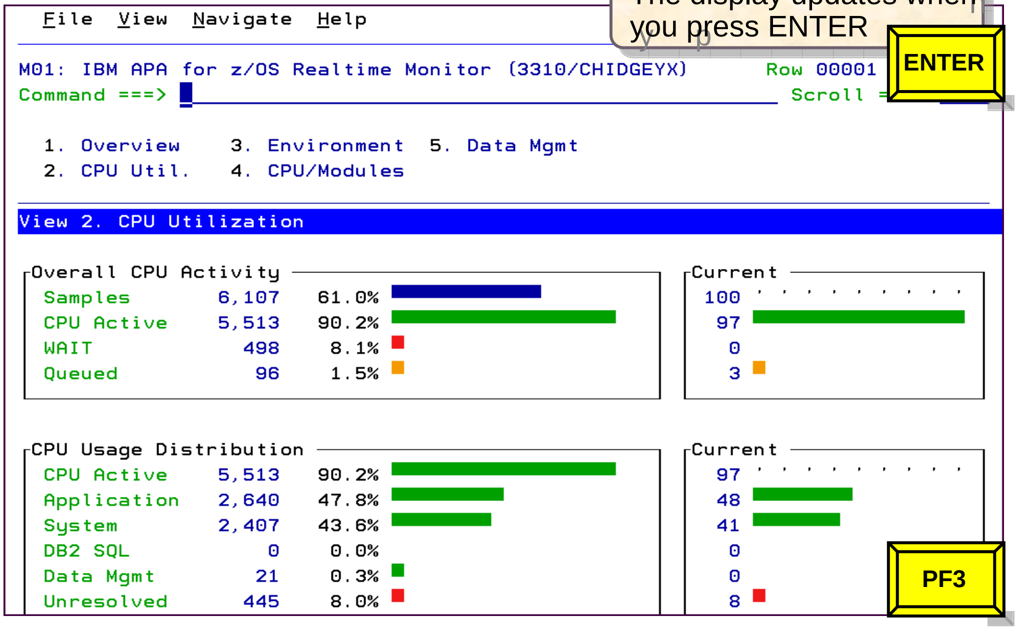
The **R** line command will display the Real-Time Monitor

ENTER

Now the status has changed to "active". That means that APA is now monitoring the application. It will monitor the application for 60 seconds, since that is what was requested. The samples column displays how many samples APA has taken so far. In this example, 10,000 samples were requested, and so far it has taken over 5,500 samples.

You can view real-time reports while APA is monitoring. Use the "R" line command to display the real-time monitor, and press Enter.

The Realtime Monitor during monitoring



```
  File  View  Navigate  Help

M01: IBM APA for z/OS Realtime Monitor (3310/CHIDGEYX)       Row 00001
Command ===> _                                                 Scroll =

  1. Overview    3. Environment  5. Data Mgmt
  2. CPU Util.   4. CPU/Modules

View 2. CPU Utilization

 Overall CPU Activity                                   Current
  Samples        6,107   61.0%                           100 ' ' ' ' ' ' ' ' '
  CPU Active     5,513   90.2%                            97
  WAIT             498    8.1%                             0
  Queued            96    1.5%                             3

 CPU Usage Distribution                                 Current
  CPU Active     5,513   90.2%                            97 ' ' ' ' ' ' ' ' '
  Application    2,640   47.8%                            48
  System         2,407   43.6%                            41
  DB2 SQL            0    0.0%                             0
  Data Mgmt         21    0.3%                             0
  Unresolved       445    8.0%                             8
```

The display updates when you press ENTER

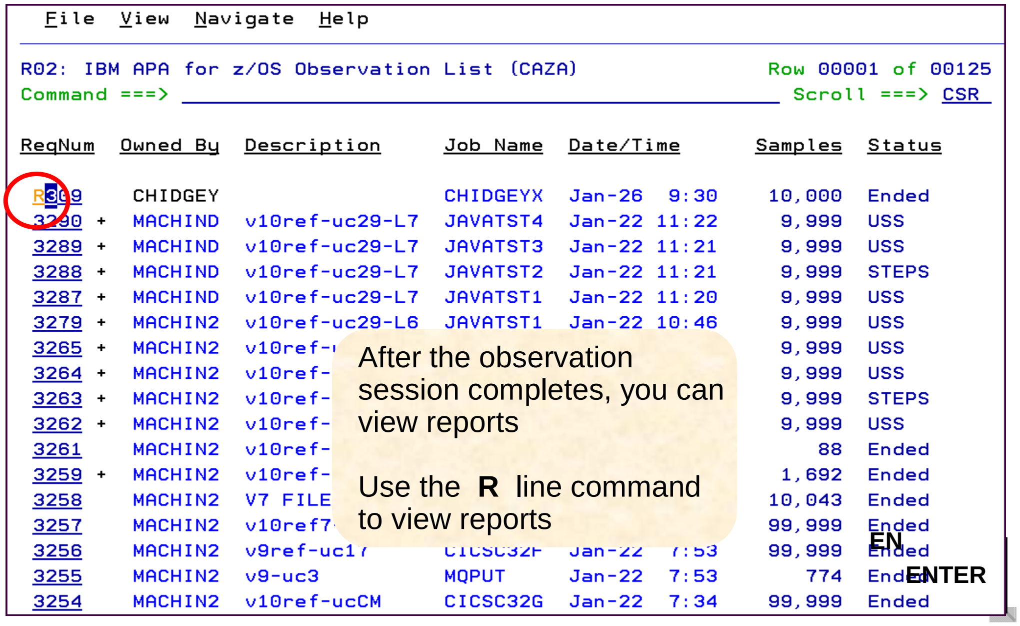
ENTER

PF3

The real-time monitor is displayed for the observation. A lot of information is available during real-time, although the real-time monitor does not provide nearly as much detail as the performance reports that you can view after the session is complete. On this screen, there is an overview of the application's performance characteristics, CPU utilization statistics, and information about the environment. There is also a list of modules that APA discovered during its sampling, and file and data management information. To refresh the display, press Enter.

If you want, you can continue to display the real-time report for as long as APA continues to monitor. Or you can exit at any time by pressing PF3.

View reports



```
  File  View  Navigate  Help

R02: IBM APA for z/OS Observation List (CAZA)                   Row 00001 of 00125
Command ===> ________________________________________________  Scroll ===> CSR

ReqNum  Owned By  Description     Job Name  Date/Time       Samples  Status

 R309     CHIDGEY                 CHIDGEYX  Jan-26  9:30     10,000  Ended
 3290  +  MACHIND  v10ref-uc29-L7 JAVATST4  Jan-22 11:22      9,999  USS
 3289  +  MACHIND  v10ref-uc29-L7 JAVATST3  Jan-22 11:21      9,999  USS
 3288  +  MACHIND  v10ref-uc29-L7 JAVATST2  Jan-22 11:21      9,999  STEPS
 3287  +  MACHIND  v10ref-uc29-L7 JAVATST1  Jan-22 11:20      9,999  USS
 3279  +  MACHIN2  v10ref-uc29-L6 JAVATST1  Jan-22 10:46      9,999  USS
 3265  +  MACHIN2  v10ref-                                    9,999  USS
 3264  +  MACHIN2  v10ref-                                    9,999  USS
 3263  +  MACHIN2  v10ref-                                    9,999  STEPS
 3262  +  MACHIN2  v10ref-                                    9,999  USS
 3261     MACHIN2  v10ref-                                       88  Ended
 3259  +  MACHIN2  v10ref-                                    1,692  Ended
 3258     MACHIN2  V7 FILE                                   10,043  Ended
 3257     MACHIN2  v10ref7                                   99,999  Ended
 3256     MACHIN2  v9ref-uc17     CICSC32F  Jan-22  7:53     99,999  Ended
 3255     MACHIN2  v9-uc3         MQPUT     Jan-22  7:53        774  Ended
 3254     MACHIN2  v10ref-ucCM    CICSC32G  Jan-22  7:34     99,999  Ended
```

After the observation session completes, you can view reports

Use the **R** line command to view reports

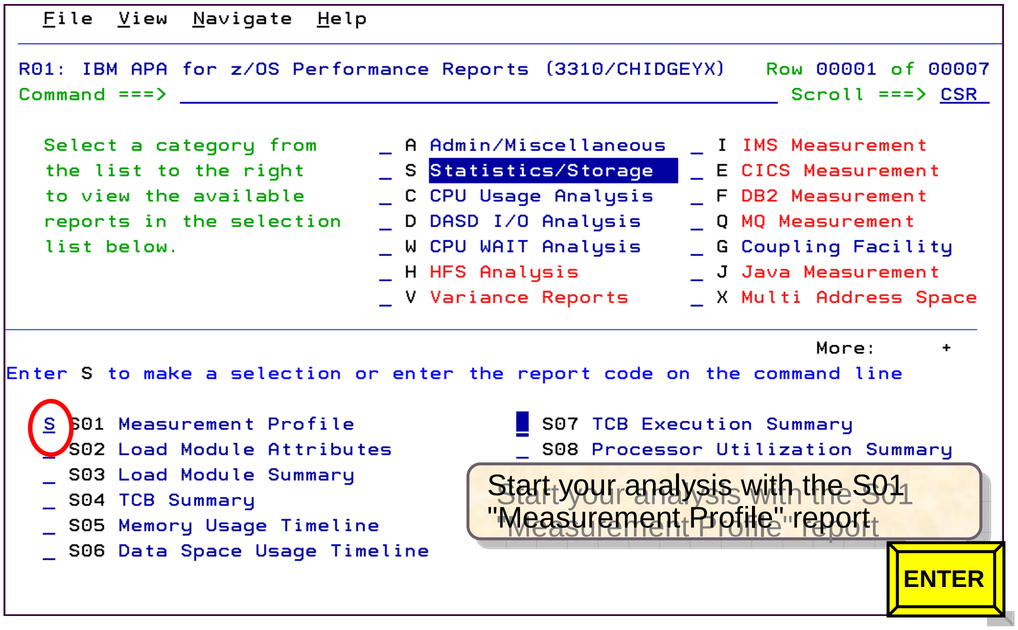
ENTER

In this example, a duration of 60 seconds was requested. When the monitoring duration elapsed, monitoring stopped and APA sent a notification that sampling was complete. Press Enter to clear the notification message.

APA writes the data that it collected to measurement data sets, and the measurement session is complete. Notice that the status changed to "ended".

After an observation session is complete, you can access the reports with an “R” line command. Understand that the “R” line command means “Real-time report” while a session is active, and “Reports” when a session is complete. Enter.

View the Measurement Profile report



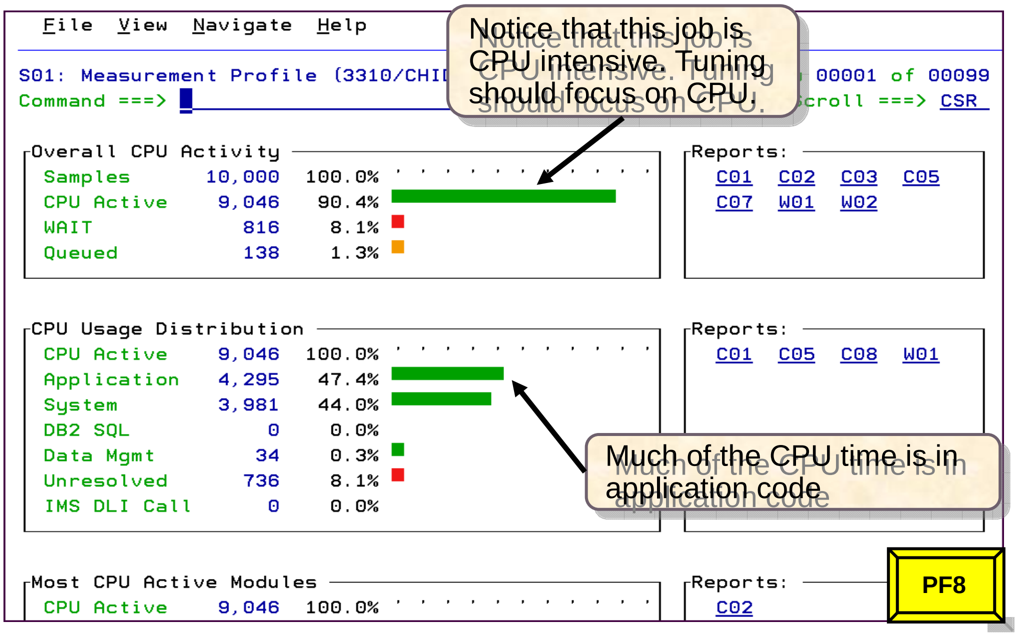
```
   File  View  Navigate  Help
-------------------------------------------------------------------------------
 R01: IBM APA for z/OS Performance Reports (3310/CHIDGEYX)    Row 00001 of 00007
 Command ===> ________________________________________________ Scroll ===> CSR

   Select a category from       _ A Admin/Miscellaneous   _ I IMS Measurement
   the list to the right        _ S Statistics/Storage    _ E CICS Measurement
   to view the available        _ C CPU Usage Analysis    _ F DB2 Measurement
   reports in the selection     _ D DASD I/O Analysis     _ Q MQ Measurement
   list below.                  _ W CPU WAIT Analysis     _ G Coupling Facility
                                _ H HFS Analysis          _ J Java Measurement
                                _ V Variance Reports      _ X Multi Address Space
-------------------------------------------------------------------------------
                                                                  More:     +
Enter S to make a selection or enter the report code on the command line

   S S01 Measurement Profile             _ S07 TCB Execution Summary
   _ S02 Load Module Attributes          _ S08 Processor Utilization Summary
   _ S03 Load Module Summary
   _ S04 TCB Summary
   _ S05 Memory Usage Timeline
   _ S06 Data Space Usage Timeline
```

Start your analysis with the S01 "Measurement Profile" report

As a general recommendation, always begin your analysis with the S01 "measurement profile" report. Notice that right now, in the upper part of the panel, the “statistics/storage” category is selected. Because of that, the statistics and storage reports are shown on the lower part of the panel.

Here, the S01 report is selected. Enter.

Measurement Profile report (S01)



```
  File  View  Navigate  Help

S01: Measurement Profile (3310/CHII                        00001 of 00099
Command ===> _                                           croll ===> CSR

 Overall CPU Activity                                      Reports:
   Samples       10,000  100.0%                              C01  C02  C03  C05
   CPU Active     9,046   90.4%                              C07  W01  W02
   WAIT             816    8.1%
   Queued           138    1.3%

 CPU Usage Distribution                                    Reports:
   CPU Active      9,046  100.0%                             C01  C05  C08  W01
   Application     4,295   47.4%
   System          3,981   44.0%
   DB2 SQL             0    0.0%
   Data Mgmt          34    0.3%
   Unresolved        736    8.1%
   IMS DLI Call        0    0.0%

 Most CPU Active Modules                                   Reports:
   CPU Active      9,046  100.0%                             C02
```

Notice that this job is CPU intensive. Tuning should focus on CPU.

Much of the CPU time is in application code

PF8

The S01 measurement profile report shows you overall statistics and application activity. Typically, the first thing you will want to look at is the "overall CPU activity" section. It shows you the percentage of time the application spent in each of the three states: CPU active, wait, or queued.

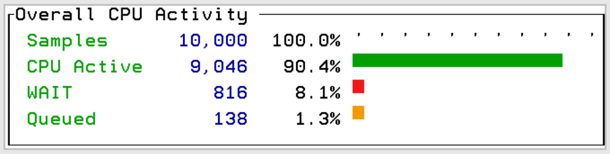
In this example, notice that the job is CPU intensive. Most of the non-queued time is spent in the CPU active state. In this case, about 90% of the time. Also notice the next section, CPU usage distribution. It shows you how that 90% is comprised. In this case, over 47% of the CPU time was spent executing application programs. That is, of the 90% CPU active time, 47% of it was spent in applications. Just so you know, that is a relatively high number.

Most healthy applications spend a much larger percentage of their time executing system code, that is, system level programs that are performing activities at the request of application programs. The fact that the application percentage is so high can be an indication of a loop. Scroll forward in the report with a PF8.

Why use the Measurement Profile report (S01)?

- Start your analysis with the S01 report
- This report tells you whether most of the time was spent in CPU or WAIT (or a lot of both)
  - Based on this information, you can focus your tuning effort on CPU or WAIT

Excerpt of the APA "Measurement Profile" report

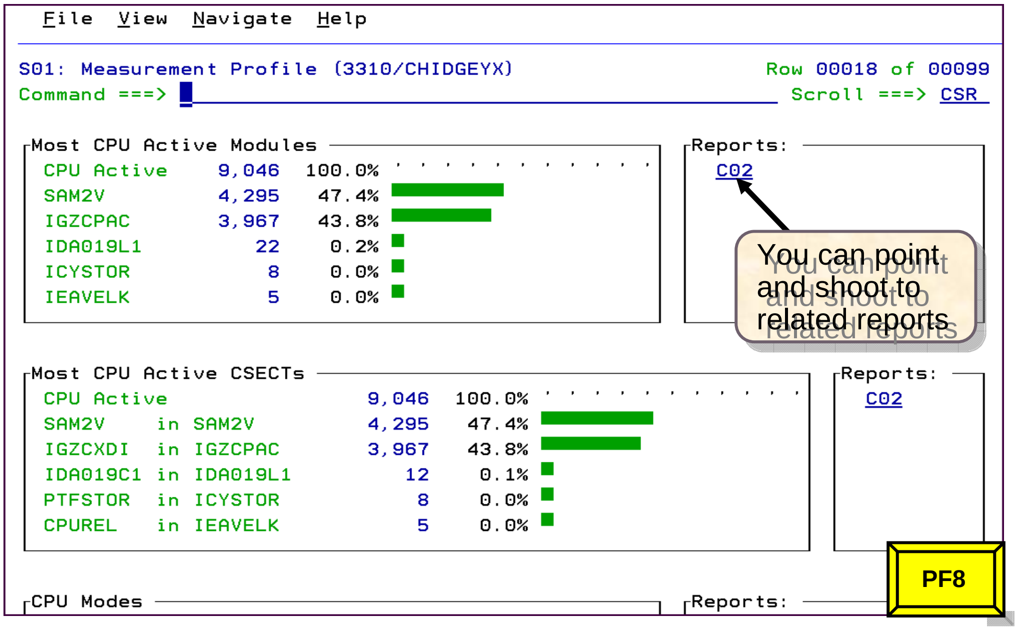


```
Overall CPU Activity
  Samples       10,000   100.0%
  CPU Active     9,046    90.4%
  WAIT             816     8.1%
  Queued           138     1.3%
```

- The S01 report also provides information about the "Quality" of the measurement and APA overhead

Remember, you will typically want to start your analysis with the S01 report. That is because this report tells you whether most of the time was spent in CPU or wait, and based on that, you can focus your efforts on the right thing. The trick to doing performance analysis is to identify where the application is spending most of its time, and then understanding why. So understanding whether an application spends more time in a CPU or wait state tells you how to begin your research.

Measurement Profile report (S01) continued…



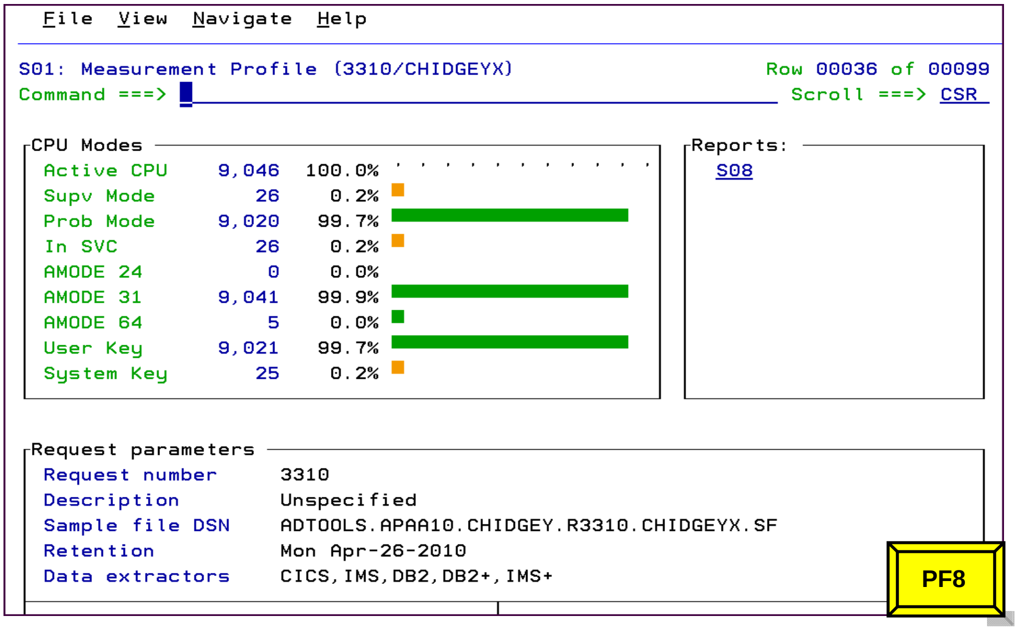
```
   File  View  Navigate  Help

S01: Measurement Profile (3310/CHIDGEYX)                        Row 00018 of 00099
Command ===>                                                    Scroll ===> CSR

 Most CPU Active Modules                                       Reports:
  CPU Active      9,046   100.0%                                 C02
  SAM2V           4,295    47.4%
  IGZCPAC         3,967    43.8%
  IDA019L1           22     0.2%
  ICYSTOR             8     0.0%
  IEAVELK             5     0.0%

 Most CPU Active CSECTs                                                   Reports:
  CPU Active                        9,046   100.0%                          C02
  SAM2V     in SAM2V                4,295    47.4%
  IGZCXDI   in IGZCPAC              3,967    43.8%
  IDA019C1  in IDA019L1                12     0.1%
  PTFSTOR   in ICYSTOR                  8     0.0%
  CPUREL    in IEAVELK                  5     0.0%

 CPU Modes                                                     Reports:
```

You can point and shoot to related reports

PF8

Back to the S01 report. The next section shows a breakdown of CPU activity by CSECTs. Remember that 90% of the CPU activity was spent executing application instructions. This section shows what percentage of that 90% CPU activity was used by individual CSECTs. In this example, 47% of the 90% CPU activity was spent in CSECT SAM2V which is the application. So your tuning efforts might want to focus in on that module. Scroll forward again. PF8.

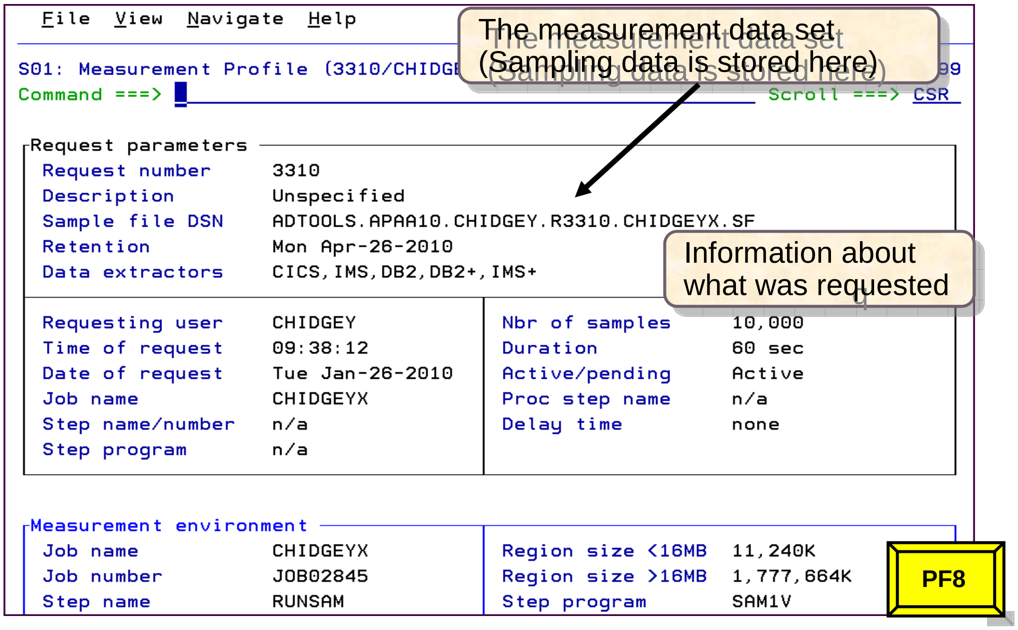
Measurement Profile report (S01) continued…



```
   File  View  Navigate  Help

S01: Measurement Profile (3310/CHIDGEYX)                     Row 00036 of 00099
Command ===> _                                                 Scroll ===> CSR

 CPU Modes                                                  Reports:
   Active CPU    9,046  100.0%                                S08
   Supv Mode        26    0.2%
   Prob Mode     9,020   99.7%
   In SVC           26    0.2%
   AMODE 24          0    0.0%
   AMODE 31      9,041   99.9%
   AMODE 64          5    0.0%
   User Key      9,021   99.7%
   System Key       25    0.2%

 Request parameters
   Request number      3310
   Description         Unspecified
   Sample file DSN     ADTOOLS.APAA10.CHIDGEY.R3310.CHIDGEYX.SF
   Retention           Mon Apr-26-2010
   Data extractors     CICS,IMS,DB2,DB2+,IMS+
```

The next section is CPU modes. There are several CPU modes and addressing modes that an application can be running in at any given point in time, and this section shows you the percentages. Most applications spend the bulk of their time in problem mode, which is the case in this example. Supervisor mode is typically time spent in a special mode required by certain system-level functions. Notice that on the right side of the report, there are some point and shoot fields for other reports. These are reports you can access if you want to display more detailed, related information. PF8 again to scroll forward.

Measurement Profile report (S01) continued…



```
  File  View  Navigate  Help

S01: Measurement Profile (3310/CHIDGE                              99
Command ===>                                          Scroll ===> CSR

Request parameters
  Request number        3310
  Description           Unspecified
  Sample file DSN       ADTOOLS.APAA10.CHIDGEY.R3310.CHIDGEYX.SF
  Retention             Mon Apr-26-2010
  Data extractors       CICS,IMS,DB2,DB2+,IMS+

  Requesting user       CHIDGEY              Nbr of samples      10,000
  Time of request       09:38:12             Duration            60 sec
  Date of request       Tue Jan-26-2010      Active/pending      Active
  Job name              CHIDGEYX             Proc step name      n/a
  Step name/number      n/a                  Delay time          none
  Step program          n/a

Measurement environment
  Job name              CHIDGEYX             Region size <16MB   11,240K
  Job number            JOB02845             Region size >16MB   1,777,664K
  Step name             RUNSAM               Step program        SAM1V
```

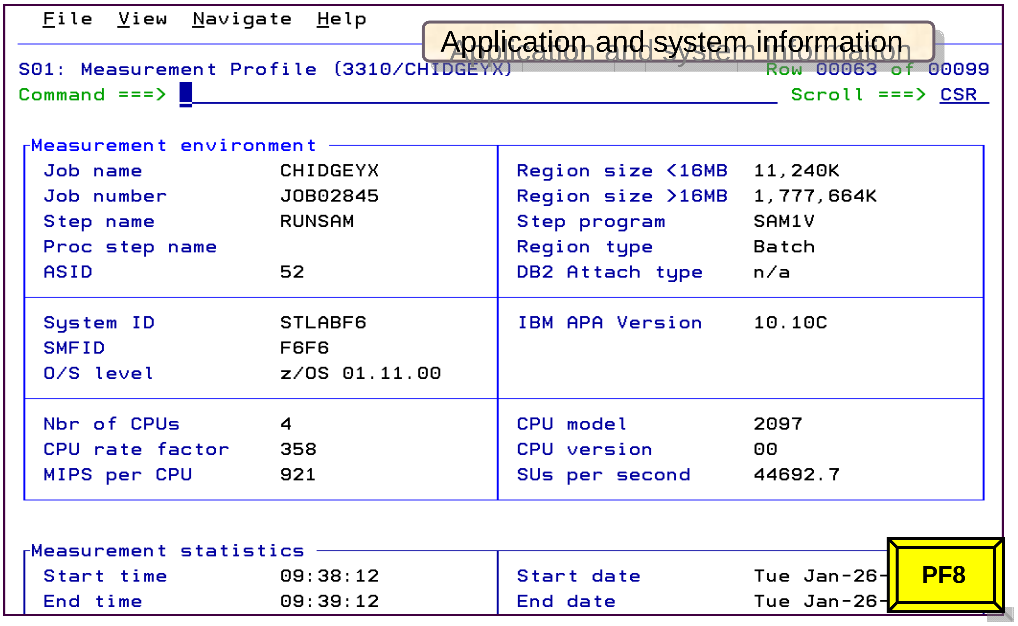
The measurement data set (Sampling data is stored here)

Information about what was requested

PF8

Next is the "request parameters" section. This section of the report displays the parameters specified to APA when the observation request was entered. It can be handy to have this documented, so that you can know, for example, the name of the job, number of samples, and duration requested. PF8 to scroll forward.

Measurement Profile report (S01) continued…



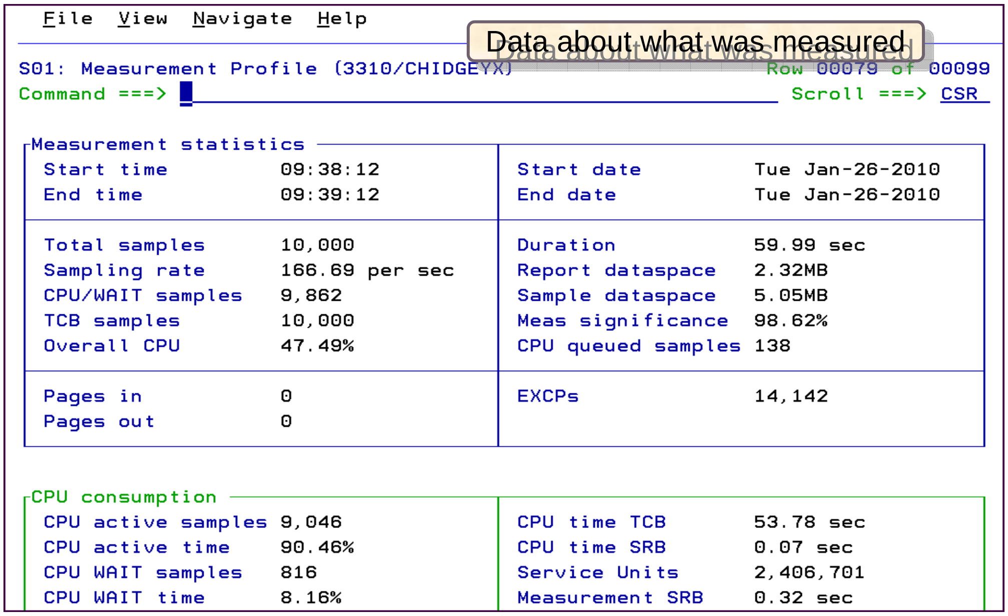
Application and system information

```
  File  View  Navigate  Help
S01: Measurement Profile (3310/CHIDGEYX)                      Row 00063 of 00099
Command ===> _                                                Scroll ===> CSR

 Measurement environment
   Job name            CHIDGEYX          Region size <16MB   11,240K
   Job number          JOB02845          Region size >16MB   1,777,664K
   Step name           RUNSAM            Step program        SAM1V
   Proc step name                        Region type         Batch
   ASID                52                DB2 Attach type     n/a

   System ID           STLABF6           IBM APA Version     10.10C
   SMFID               F6F6
   O/S level           z/OS 01.11.00

   Nbr of CPUs         4                 CPU model           2097
   CPU rate factor     358               CPU version         00
   MIPS per CPU        921               SUs per second      44692.7

 Measurement statistics
   Start time          09:38:12          Start date          Tue Jan-26-
   End time            09:39:12          End date            Tue Jan-26-
```

PF8

That brings you to the “measurement environment" section. This shows information about the job that was monitored and the system it was running on. This section of the report displays the job name, job number, step name, and region size, in addition to the z/OS system ID, and information about the processor. PF8.

Measurement Profile report (S01) continued…



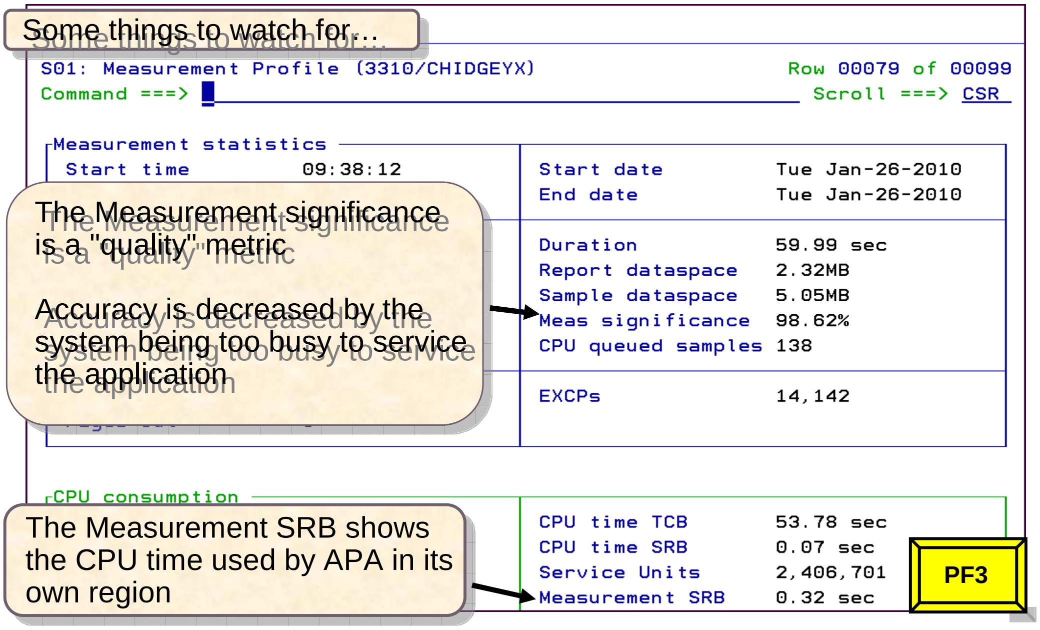
Data about what was measured

```
   File  View  Navigate  Help
 ------------------------------------------------------------------------------
 S01: Measurement Profile (3310/CHIDGEYX)                      Row 00079 of 00099
 Command ===> _                                                Scroll ===> CSR

 ┌Measurement statistics ───────────────────────────────────────────────────────┐
 │ Start time          09:38:12          │ Start date          Tue Jan-26-2010  │
 │ End time            09:39:12          │ End date            Tue Jan-26-2010  │
 │───────────────────────────────────────│──────────────────────────────────────│
 │ Total samples       10,000            │ Duration            59.99 sec        │
 │ Sampling rate       166.69 per sec    │ Report dataspace    2.32MB           │
 │ CPU/WAIT samples    9,862             │ Sample dataspace    5.05MB           │
 │ TCB samples         10,000            │ Meas significance   98.62%           │
 │ Overall CPU         47.49%            │ CPU queued samples  138              │
 │───────────────────────────────────────│──────────────────────────────────────│
 │ Pages in            0                 │ EXCPs               14,142           │
 │ Pages out           0                 │                                      │
 └───────────────────────────────────────┴──────────────────────────────────────┘

 ┌CPU consumption ──────────────────────────────────────────────────────────────┐
 │ CPU active samples  9,046             │ CPU time TCB        53.78 sec        │
 │ CPU active time     90.46%            │ CPU time SRB        0.07 sec         │
 │ CPU WAIT samples    816               │ Service Units       2,406,701        │
 │ CPU WAIT time       8.16%             │ Measurement SRB     0.32 sec         │
```

20 | IBM Application Performance Analyzer for z/OS tutorial | © 2010 IBM Corporation

Next is the measurement statistics section. One of the earlier sections was "request parameters", which contained information about what was requested. This section shows information about what was actually collected. It shows the start and end time of APA’s monitoring, how many total samples were taken, the sampling rate, and the actual measurement duration. It is possible that the requested duration and number of samples can be different from the actual. For example, say that you requested a monitoring duration of 60 seconds, but then the job only ran for 30 seconds. That will cause the total duration to be only 30 seconds, and only about half of the requested number of samples will have been taken. And that is the type of information reported here in this section.

Measurement Profile report (S01) continued…



Some things to watch for…

```
S01: Measurement Profile (3310/CHIDGEYX)                 Row 00079 of 00099
Command ===>                                                 Scroll ===> CSR

Measurement statistics
 Start time          09:38:12      Start date         Tue Jan-26-2010
                                   End date           Tue Jan-26-2010

                                   Duration           59.99 sec
                                   Report dataspace   2.32MB
                                   Sample dataspace   5.05MB
                                   Meas significance  98.62%
                                   CPU queued samples 138

                                   EXCPs              14,142

CPU consumption
                                   CPU time TCB       53.78 sec
                                   CPU time SRB       0.07 sec
                                   Service Units      2,406,701
                                   Measurement SRB    0.32 sec
```

The Measurement significance is a "quality" metric

Accuracy is decreased by the system being too busy to service the application

The Measurement SRB shows the CPU time used by APA in its own region

PF3

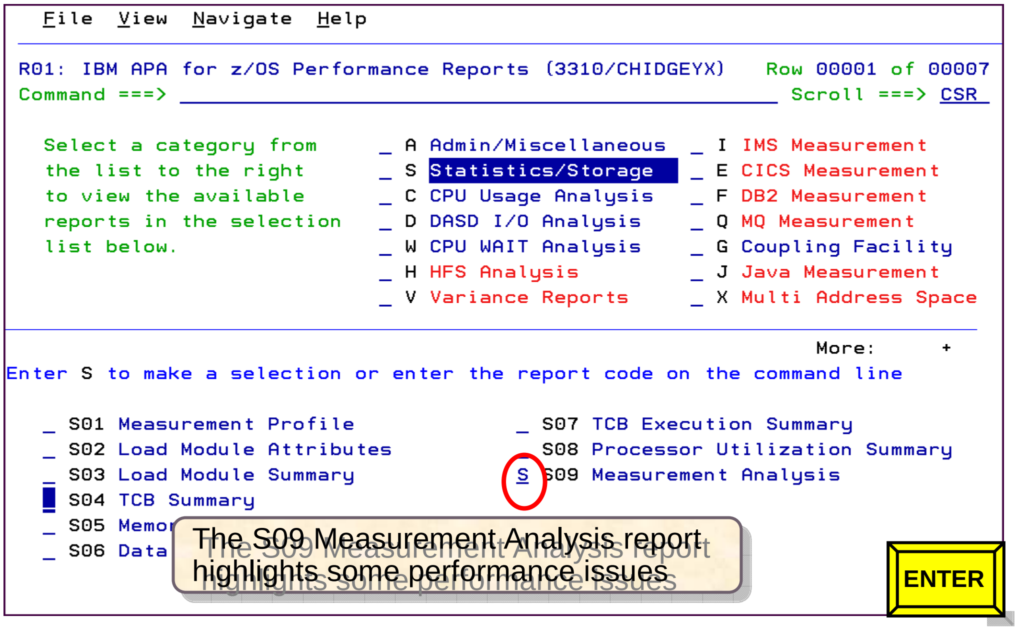
Here are a couple of things to watch for in the measurement profile report. The measurement significance is a quality metric. The accuracy of APA's reports is decreased by the system being too busy to service the application.

In a worst-case scenario, the system can be so busy processing other tasks that the application is in a queued state 100% of the time. Therefore, no CPU or wait time is incurred. Obviously, if that happens, there is not any meaningful information in APA's reports.

The measurement significance shows the percentage of time that the application was not in the queued state. The higher the number, the better, although sometimes when you are running on an extremely busy system, the number might always be somewhat low and there might not be anything you can do about that.

It is also good to know about another statistic called the measurement SRB. It shows APA's "overhead", and it is CPU time used by APA in its own region to collect the performance data. Once you are done with this report, press PF3 to return to the report menu.

View the Measurement Analysis report



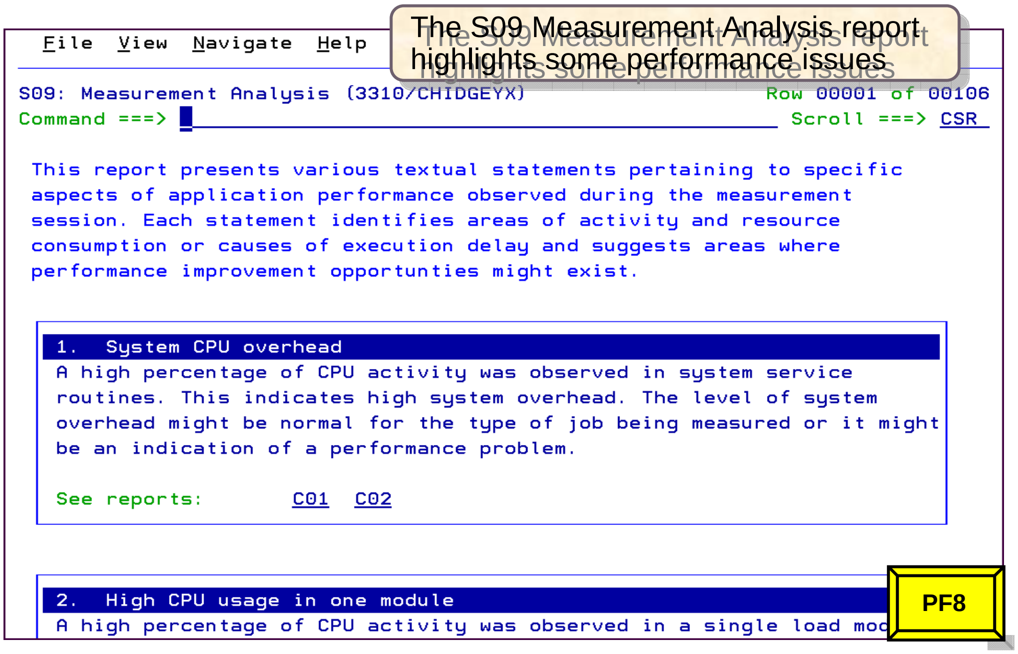
```
   File  View  Navigate  Help
 ------------------------------------------------------------------------------
 R01: IBM APA for z/OS Performance Reports (3310/CHIDGEYX)    Row 00001 of 00007
 Command ===> ______________________________________________     Scroll ===> CSR

   Select a category from      _ A Admin/Miscellaneous  _ I IMS Measurement
   the list to the right       _ S Statistics/Storage   _ E CICS Measurement
   to view the available       _ C CPU Usage Analysis   _ F DB2 Measurement
   reports in the selection    _ D DASD I/O Analysis    _ Q MQ Measurement
   list below.                 _ W CPU WAIT Analysis    _ G Coupling Facility
                               _ H HFS Analysis         _ J Java Measurement
                               _ V Variance Reports     _ X Multi Address Space
 ------------------------------------------------------------------------------
                                                                 More:     +
 Enter S to make a selection or enter the report code on the command line

   _ S01 Measurement Profile                 _ S07 TCB Execution Summary
   _ S02 Load Module Attributes              _ S08 Processor Utilization Summary
   _ S03 Load Module Summary                 S S09 Measurement Analysis
   _ S04 TCB Summary
   _ S05 Memor
   _ S06 Data
```

The S09 Measurement Analysis report highlights some performance issues

The S09 Measurement Analysis report highlights some performance issues

What information is available in the “S09 Measurement Analysis” reports?

Measurement Analysis report (S09)



The S09 Measurement Analysis report highlights some performance issues

```
 File  View  Navigate  Help
S09: Measurement Analysis (3310/CHIDGEYX)                        Row 00001 of 00106
Command ===>                                                       Scroll ===> CSR

 This report presents various textual statements pertaining to specific
 aspects of application performance observed during the measurement
 session. Each statement identifies areas of activity and resource
 consumption or causes of execution delay and suggests areas where
 performance improvement opportunties might exist.

   1.  System CPU overhead
   A high percentage of CPU activity was observed in system service
   routines. This indicates high system overhead. The level of system
   overhead might be normal for the type of job being measured or it might
   be an indication of a performance problem.

   See reports:       C01  C02


   2.  High CPU usage in one module
   A high percentage of CPU activity was observed in a single load mod
```

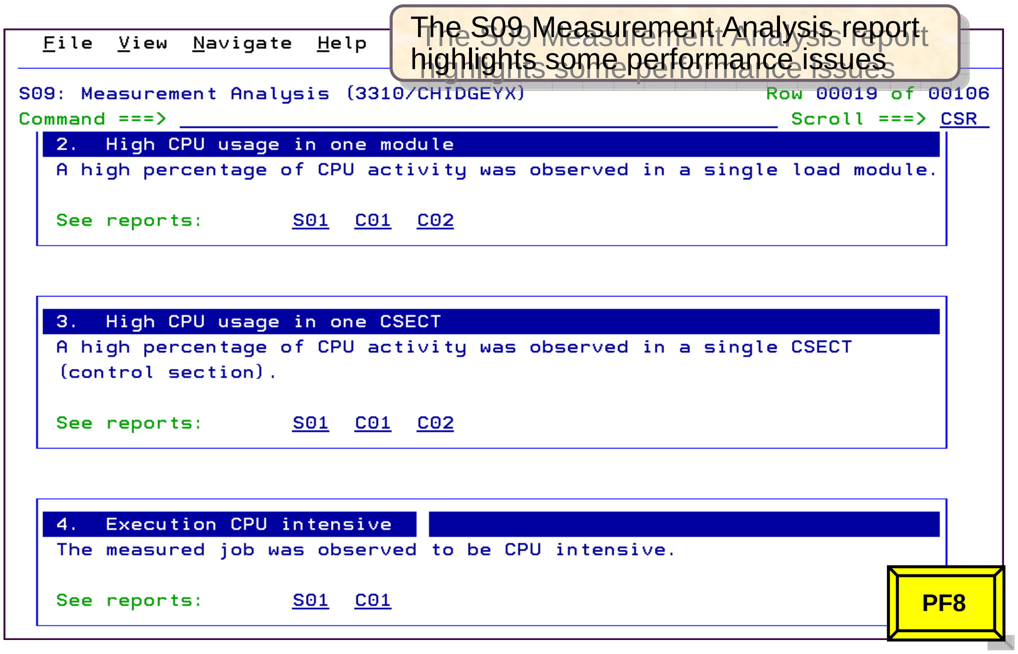
PF8

S09 presents statements, each representing an observation made about some aspect of the measured job. The purpose of each of these observations is to provide a synoptic analysis of an area of resource usage and, in some cases, suggest where some performance improvement opportunities might exist.

In this example, the focus is on CPU consumption.

Analyze these observations in the context of how you expect the measured job to perform. Some of the statements in this report might draw your attention to aspects of resource consumption that is perfectly normal for the job. For example, high CPU consumption might be noted in a certain module in a situation where you actually expect high CPU usage in that module. To display more scroll forward with PF8.

Measurement Analysis report (S09) continued…

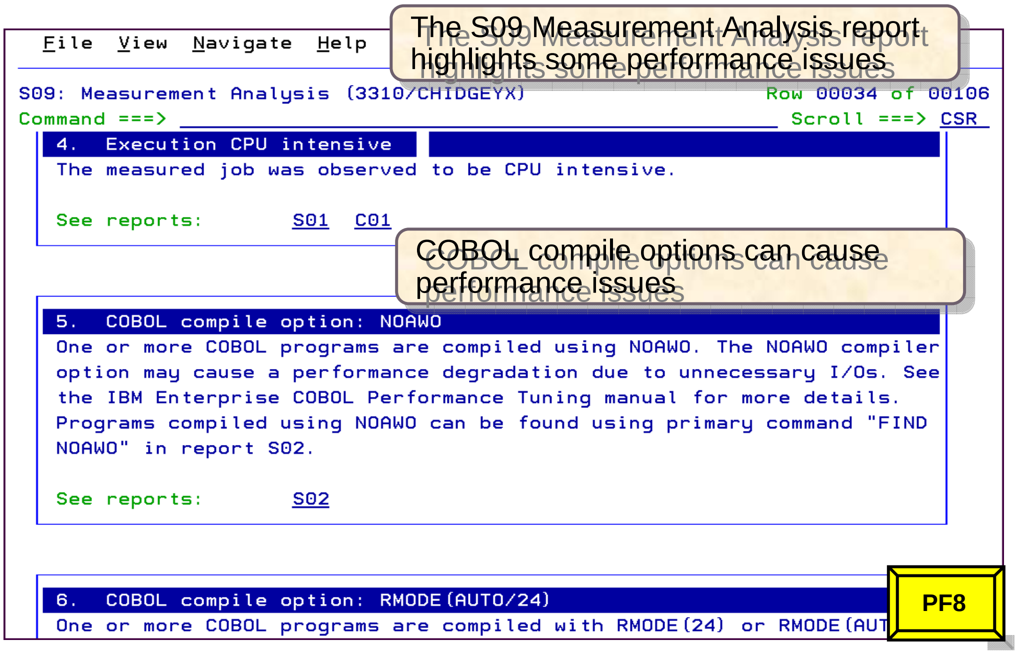


The S09 Measurement Analysis report highlights some performance issues

```
S09: Measurement Analysis (3310/CHIDGEYX)                      Row 00019 of 00106
Command ===> ______________________________________________  Scroll ===> CSR

  2.  High CPU usage in one module
  A high percentage of CPU activity was observed in a single load module.

  See reports:        S01  C01  C02

  3.  High CPU usage in one CSECT
  A high percentage of CPU activity was observed in a single CSECT
  (control section).

  See reports:        S01  C01  C02

  4.  Execution CPU intensive
  The measured job was observed to be CPU intensive.

  See reports:        S01  C01
```

In this example, the Measurement Analysis has indicated that there is High CPU consumption in a single load module. Message three narrowed it down to a single CSECT. This report also suggests specific reports where you will find detailed related information. Press PF8 to go down.

Measurement Analysis report (S09) continued…



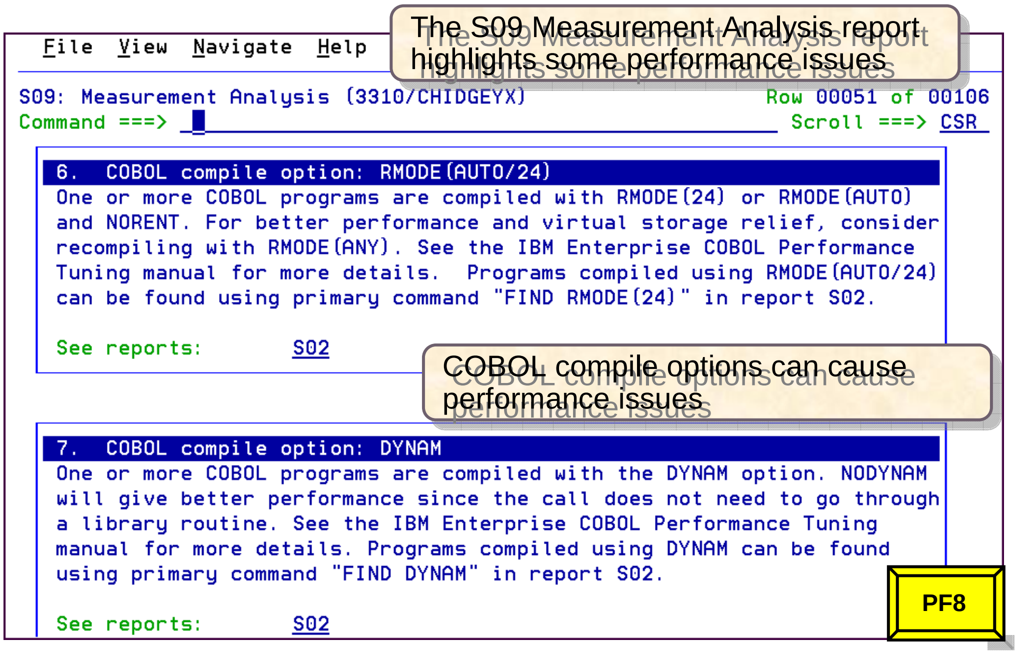
```
 File  View  Navigate  Help
S09: Measurement Analysis (3310/CHIDGEYX)                          Row 00034 of 00106
Command ===> ______________________________________________________  Scroll ===> CSR
   4.  Execution CPU intensive
   The measured job was observed to be CPU intensive.

   See reports:        S01  C01


   5.  COBOL compile option: NOAWO
   One or more COBOL programs are compiled using NOAWO. The NOAWO compiler
   option may cause a performance degradation due to unnecessary I/Os. See
   the IBM Enterprise COBOL Performance Tuning manual for more details.
   Programs compiled using NOAWO can be found using primary command "FIND
   NOAWO" in report S02.

   See reports:        S02


   6.  COBOL compile option: RMODE(AUTO/24)
   One or more COBOL programs are compiled with RMODE(24) or RMODE(AUT
```

The S09 Measurement Analysis report highlights some performance issues

COBOL compile options can cause performance issues

This report will also indicate whether the job is CPU intensive or wait intensive. In this example, the report indicated that the job is CPU intensive as identified in message four. Also, the S09 report will highlight any possible performance issues. Sometimes certain compile options can cause these performance issues to occur. The S09 report will highlight these compile options and provide information regarding their impact. Press PF8 to scroll through the rest of this report.

Measurement Analysis report (S09) continued…



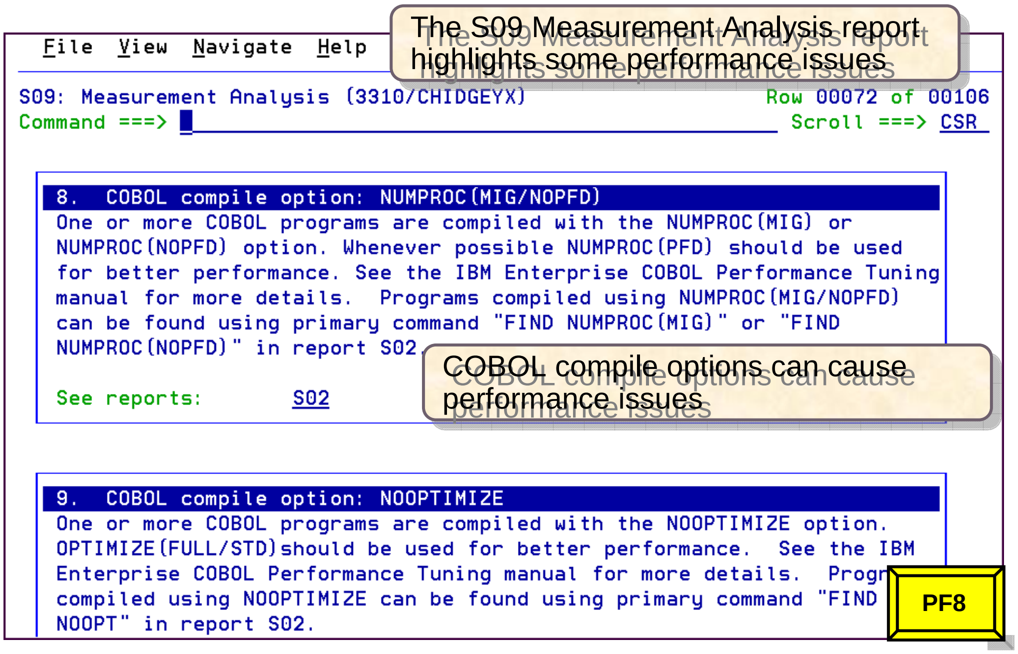
```
  File  View  Navigate  Help

S09: Measurement Analysis (3310/CHIDGEYX)                     Row 00051 of 00106
Command ===> _                                                 Scroll ===> CSR

   6.  COBOL compile option: RMODE(AUTO/24)
   One or more COBOL programs are compiled with RMODE(24) or RMODE(AUTO)
   and NORENT. For better performance and virtual storage relief, consider
   recompiling with RMODE(ANY). See the IBM Enterprise COBOL Performance
   Tuning manual for more details.  Programs compiled using RMODE(AUTO/24)
   can be found using primary command "FIND RMODE(24)" in report S02.

   See reports:        S02

   7.  COBOL compile option: DYNAM
   One or more COBOL programs are compiled with the DYNAM option. NODYNAM
   will give better performance since the call does not need to go through
   a library routine. See the IBM Enterprise COBOL Performance Tuning
   manual for more details. Programs compiled using DYNAM can be found
   using primary command "FIND DYNAM" in report S02.

   See reports:        S02
```

The S09 Measurement Analysis report highlights some performance issues

COBOL compile options can cause performance issues

Here are additional compile options that can affect the performance of the application. PF8 to scroll down.

Measurement Analysis report (S09) continued…



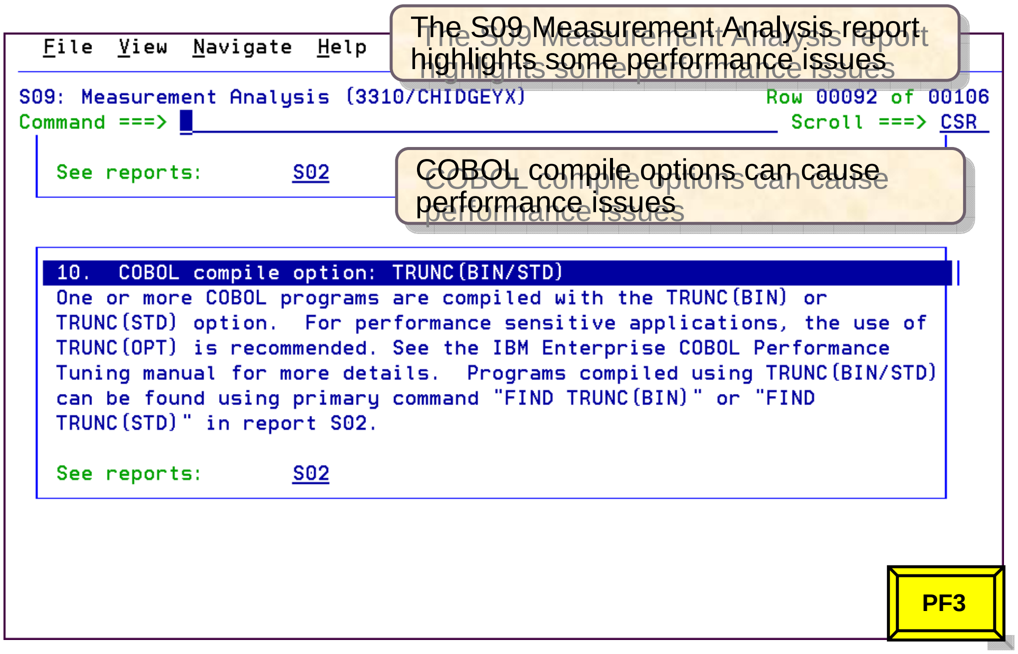
```
  File  View  Navigate  Help

S09: Measurement Analysis (3310/CHIDGEYX)                          Row 00072 of 00106
Command ===> _                                                       Scroll ===> CSR

   8.  COBOL compile option: NUMPROC(MIG/NOPFD)
   One or more COBOL programs are compiled with the NUMPROC(MIG) or
   NUMPROC(NOPFD) option. Whenever possible NUMPROC(PFD) should be used
   for better performance. See the IBM Enterprise COBOL Performance Tuning
   manual for more details.  Programs compiled using NUMPROC(MIG/NOPFD)
   can be found using primary command "FIND NUMPROC(MIG)" or "FIND
   NUMPROC(NOPFD)" in report S02.

   See reports:       S02


   9.  COBOL compile option: NOOPTIMIZE
   One or more COBOL programs are compiled with the NOOPTIMIZE option.
   OPTIMIZE(FULL/STD)should be used for better performance.  See the IBM
   Enterprise COBOL Performance Tuning manual for more details.  Progr
   compiled using NOOPTIMIZE can be found using primary command "FIND
   NOOPT" in report S02.
```

The S09 Measurement Analysis report highlights some performance issues

COBOL compile options can cause performance issues

More compile options are highlighted. PF8 to scroll down.

Measurement Analysis report (S09) continued…



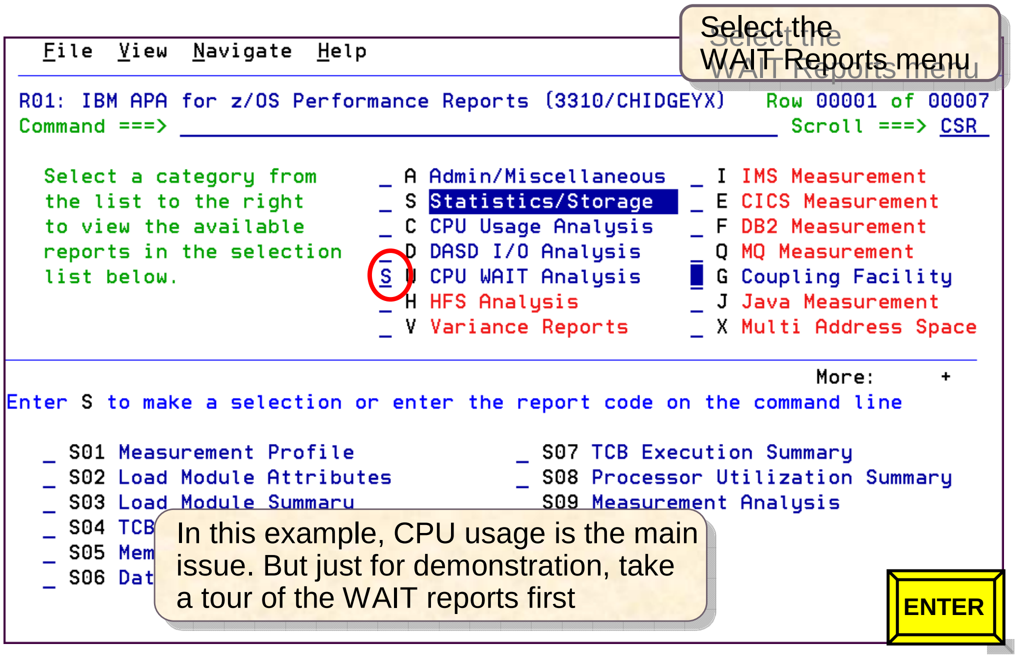
```
 File  View  Navigate  Help
S09: Measurement Analysis (3310/CHIDGEYX)                          Row 00092 of 00106
Command ===> _                                                       Scroll ===> CSR

   See reports:        S02

   10.  COBOL compile option: TRUNC(BIN/STD)
   One or more COBOL programs are compiled with the TRUNC(BIN) or
   TRUNC(STD) option.  For performance sensitive applications, the use of
   TRUNC(OPT) is recommended. See the IBM Enterprise COBOL Performance
   Tuning manual for more details.  Programs compiled using TRUNC(BIN/STD)
   can be found using primary command "FIND TRUNC(BIN)" or "FIND
   TRUNC(STD)" in report S02.

   See reports:        S02
```

The S09 Measurement Analysis report highlights some performance issues

COBOL compile options can cause performance issues

Now you are at the end of the report. Using the recommendations and focusing on the modules indicated in this report could help you significantly improve your applications performance. PF3 to exit the report.

View WAIT reports menu



Select the WAIT Reports menu

```
  File  View  Navigate  Help
R01: IBM APA for z/OS Performance Reports (3310/CHIDGEYX)   Row 00001 of 00007
Command ===> ________________________________________________ Scroll ===> CSR

  Select a category from      _ A Admin/Miscellaneous   _ I IMS Measurement
  the list to the right       _ S Statistics/Storage    _ E CICS Measurement
  to view the available       _ C CPU Usage Analysis    _ F DB2 Measurement
  reports in the selection    _ D DASD I/O Analysis     _ Q MQ Measurement
  list below.                 S U CPU WAIT Analysis     _ G Coupling Facility
                              _ H HFS Analysis          _ J Java Measurement
                              _ V Variance Reports      _ X Multi Address Space

                                                                More:     +
Enter S to make a selection or enter the report code on the command line

   _ S01 Measurement Profile                  _ S07 TCB Execution Summary
   _ S02 Load Module Attributes               _ S08 Processor Utilization Summary
   _ S03 Load Module Summary                    S09 Measurement Analysis
   _ S04 TCB
   _ S05 Mem
   _ S06 Dat
```

In this example, CPU usage is the main issue. But just for demonstration, take a tour of the WAIT reports first

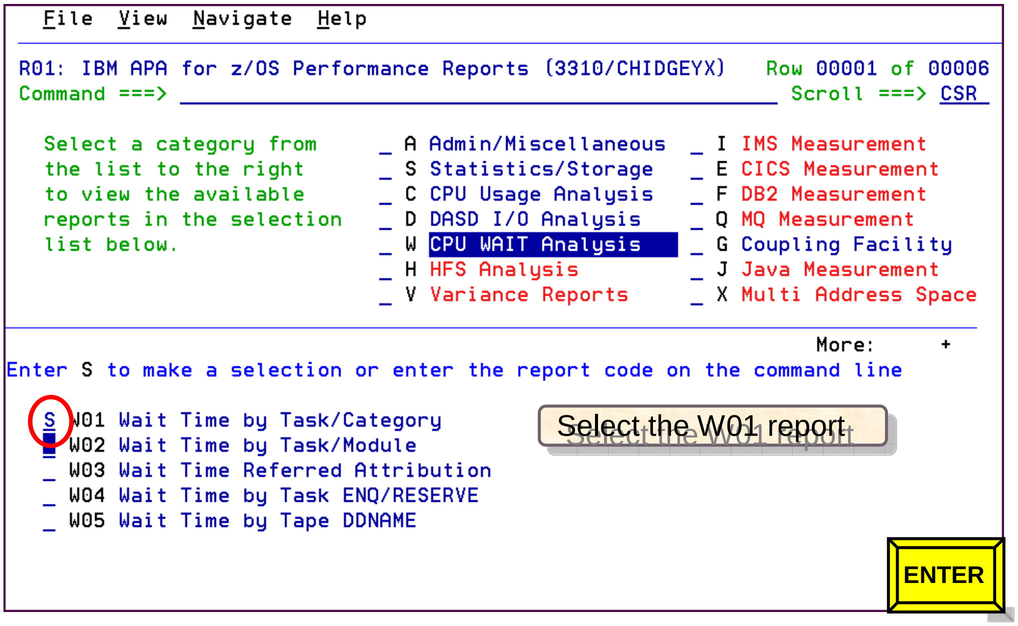
ENTER

29 | IBM Application Performance Analyzer for z/OS tutorial | © 2010 IBM Corporation

In this example, CPU usage is the main issue, and that is where the focus should be in your performance analysis efforts. But just for demonstration, you are guided through the information that is available in the “wait” reports.

To select the “wait” reports category, place an ”S” next to it and Press Enter.

View the Wait Time by Task/Category report



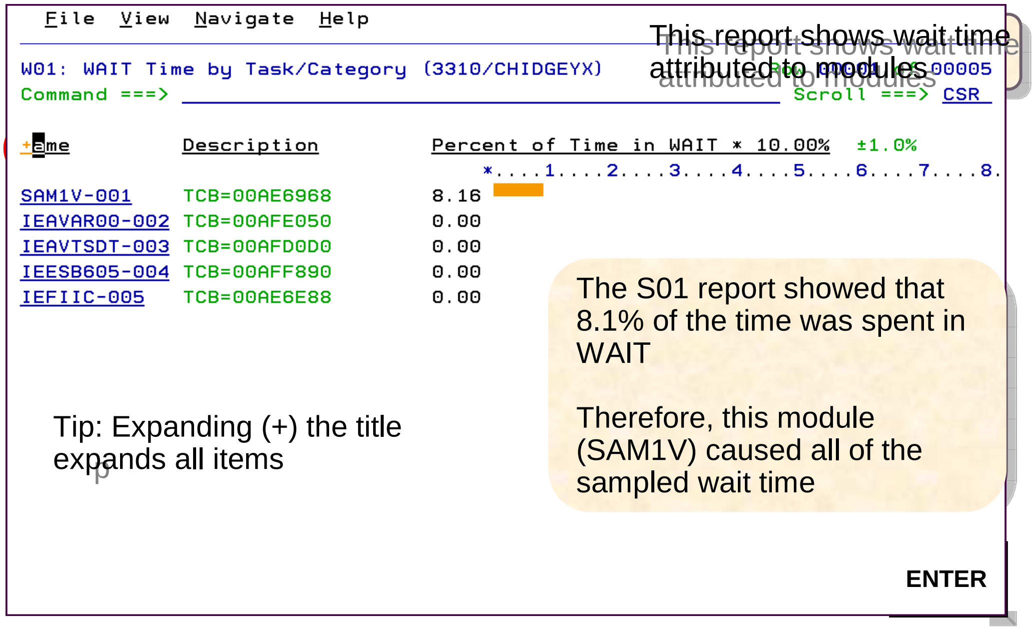
```
  File  View  Navigate  Help

R01: IBM APA for z/OS Performance Reports (3310/CHIDGEYX)    Row 00001 of 00006
Command ===> ________________________________________________  Scroll ===> CSR

  Select a category from       _ A Admin/Miscellaneous   _ I IMS Measurement
  the list to the right        _ S Statistics/Storage    _ E CICS Measurement
  to view the available        _ C CPU Usage Analysis    _ F DB2 Measurement
  reports in the selection     _ D DASD I/O Analysis     _ Q MQ Measurement
  list below.                  _ W CPU WAIT Analysis     _ G Coupling Facility
                               _ H HFS Analysis          _ J Java Measurement
                               _ V Variance Reports      _ X Multi Address Space

                                                                   More:     +
Enter S to make a selection or enter the report code on the command line

   S W01 Wait Time by Task/Category
   _ W02 Wait Time by Task/Module
   _ W03 Wait Time Referred Attribution
   _ W04 Wait Time by Task ENQ/RESERVE
   _ W05 Wait Time by Tape DDNAME
```

Select the W01 report

And now the ‘wait’ reports are shown. In this example the ‘W01 wait time by task/category’ report is selected.

You selected it with an “S” and Press Enter.

W01: Wait Time by Task/Category report



```
  File  View  Navigate  Help
W01: WAIT Time by Task/Category (3310/CHIDGEYX)           Row 00001 of 00005
Command ===> ____________________________________________ Scroll ===> CSR

+ame           Description           Percent of Time in WAIT * 10.00%  ±1.0%
                                        *....1....2....3....4....5....6....7....8.
SAM1V-001      TCB=00AE6968          8.16
IEAVAR00-002   TCB=00AFE050          0.00
IEAVTSDT-003   TCB=00AFD0D0          0.00
IEESB605-004   TCB=00AFF890          0.00
IEFIIC-005     TCB=00AE6E88          0.00
```

This report shows wait time attributed to modules

This report shows wait time attributed to modules

The S01 report showed that 8.1% of the time was spent in WAIT

Therefore, this module (SAM1V) caused all of the sampled wait time

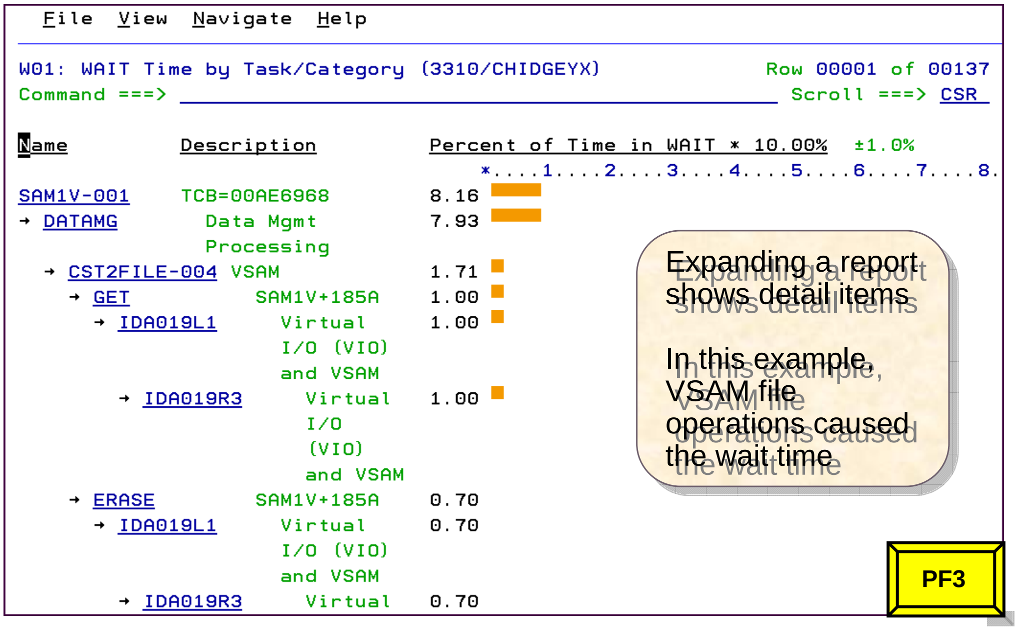
Tip: Expanding (+) the title expands all items

ENTER

This report shows wait time that is attributed to modules. On the left is a list of program modules, and on the right the percentage of wait time caused by each one is shown. You might recall that the S01 report showed that a total of 8.1% of the time was spent in wait. In this report, that number is broken down by program. Notice that one module, SAM1V, caused all of the sampled wait time.

To expand all of the items, use a "+" line command in the column heading, and press Enter.

W01: Wait Time by Task/Category report expanded



```
  File  View  Navigate  Help

W01: WAIT Time by Task/Category (3310/CHIDGEYX)             Row 00001 of 00137
Command ===> _______________________________________________  Scroll ===> CSR

Name         Description         Percent of Time in WAIT * 10.00%  ±1.0%
                                     *....1....2....3....4....5....6....7....8.
SAM1V-001    TCB=00AE6968        8.16 ████
→ DATAMG       Data Mgmt         7.93 ████
               Processing
  → CST2FILE-004 VSAM            1.71 █
    → GET              SAM1V+185A    1.00 █
      → IDA019L1       Virtual       1.00 █
                     I/O (VIO)
                     and VSAM
        → IDA019R3       Virtual     1.00 █
                       I/O
                       (VIO)
                       and VSAM
    → ERASE            SAM1V+185A    0.70
      → IDA019L1       Virtual       0.70
                     I/O (VIO)
                     and VSAM
        → IDA019R3       Virtual     0.70
```

Expanding a report shows detail items

In this example, VSAM file operations caused the wait time

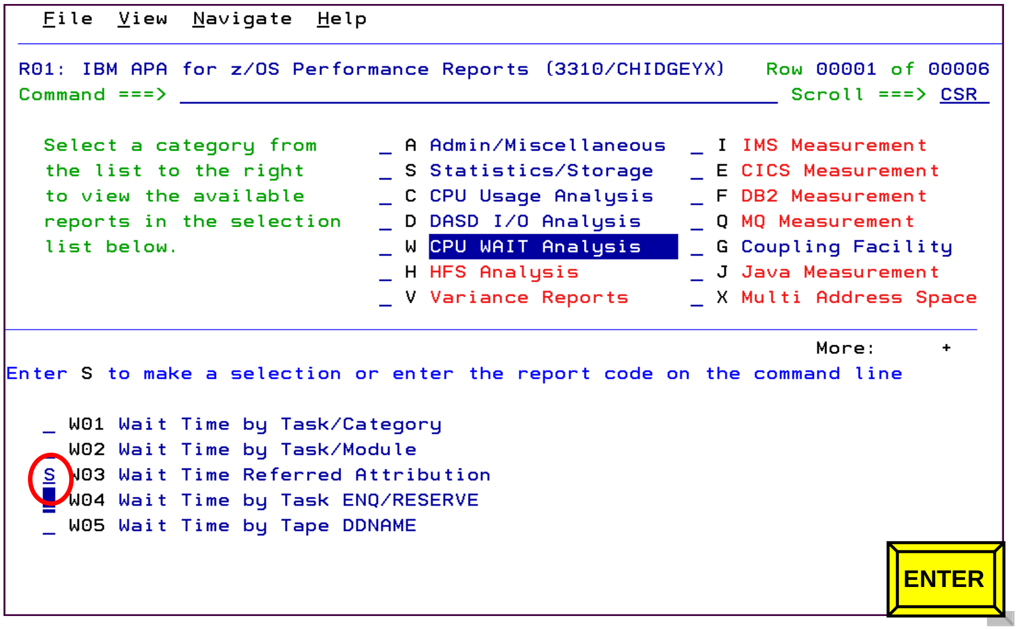
PF3

Now all of the items have been fully expanded. And notice that you can identify the individual operations in the program that caused wait time. Right under the program name is a line item for data management processing. Notice that almost all of the wait time for this program was caused by data management processing. About 7.93%.

And it is broken down further. Under data management, it shows a DD name: CST2FILE. It looks like most of the wait time was spent processing this file. Below that is a line item for GET. Notice that about 1% of the time was spent waiting because of the GET. And further down is an “ERASE” which caused .7% also..

That is good information, since it shows you exactly which programs are causing the most wait time, and what operations the program is doing to cause that wait time. In a moment, you will learn how to drill down into the program source to access the actual source statements that caused this wait time. PF3 to return.

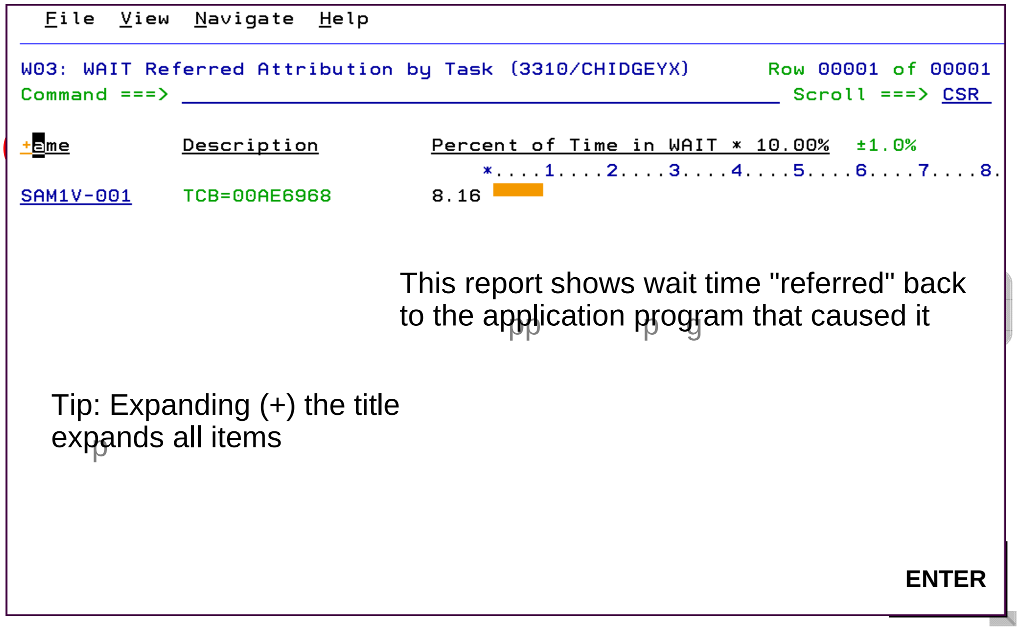
View the W03 report



```
  File  View  Navigate  Help

R01: IBM APA for z/OS Performance Reports (3310/CHIDGEYX)    Row 00001 of 00006
Command ===> ______________________________________________    Scroll ===> CSR

  Select a category from     _ A Admin/Miscellaneous  _ I IMS Measurement
  the list to the right      _ S Statistics/Storage   _ E CICS Measurement
  to view the available      _ C CPU Usage Analysis   _ F DB2 Measurement
  reports in the selection   _ D DASD I/O Analysis    _ Q MQ Measurement
  list below.                _ W CPU WAIT Analysis    _ G Coupling Facility
                             _ H HFS Analysis         _ J Java Measurement
                             _ V Variance Reports     _ X Multi Address Space

                                                                  More:     +
Enter S to make a selection or enter the report code on the command line

   _ W01 Wait Time by Task/Category
     W02 Wait Time by Task/Module
   S W03 Wait Time Referred Attribution
   _ W04 Wait Time by Task ENQ/RESERVE
   _ W05 Wait Time by Tape DDNAME
```

Here the ‘W03 wait time referred attribution’ report is selected.

W03: Wait Referred Attribution by Task report



```
  File  View  Navigate  Help

W03: WAIT Referred Attribution by Task (3310/CHIDGEYX)       Row 00001 of 00001
Command ===> ________________________________________________  Scroll ===> CSR

+ame         Description         Percent of Time in WAIT * 10.00%  ±1.0%
                                     *....1....2....3....4....5....6....7....8.
SAM1V-001    TCB=00AE6968        8.16
```

This report shows wait time "referred" back to the application program that caused it

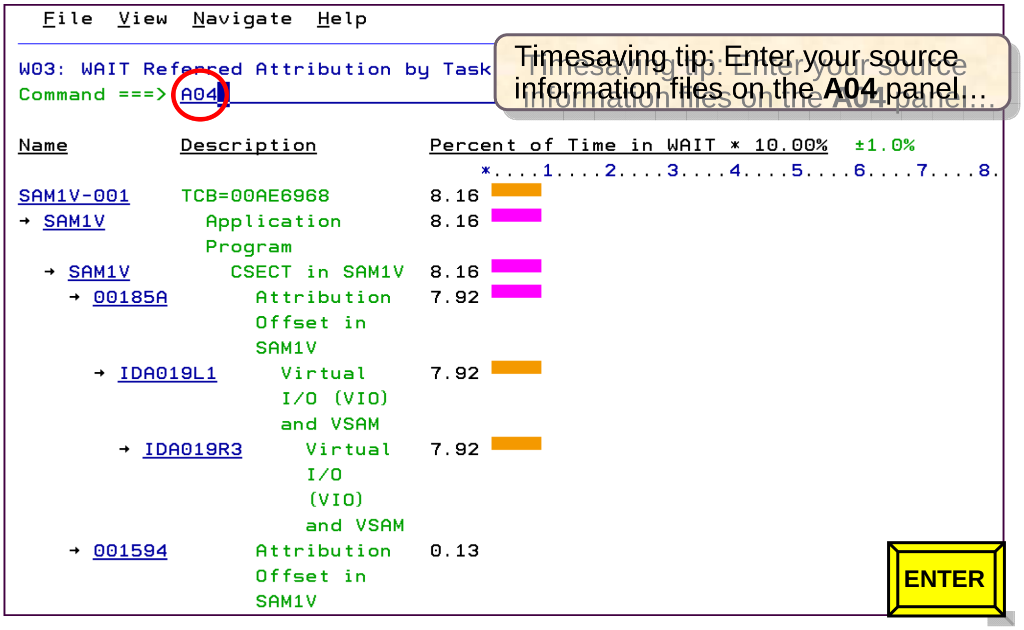
Tip: Expanding (+) the title expands all items

ENTER

This report shows you how wait time is broken down by tasks and programs. It is the same amount of wait time that was indicated in the W01 report, but sorted a different way.

To expand the report to its deepest level, type a “+” line command in the column heading. Press Enter.

W03 Report: Expanded



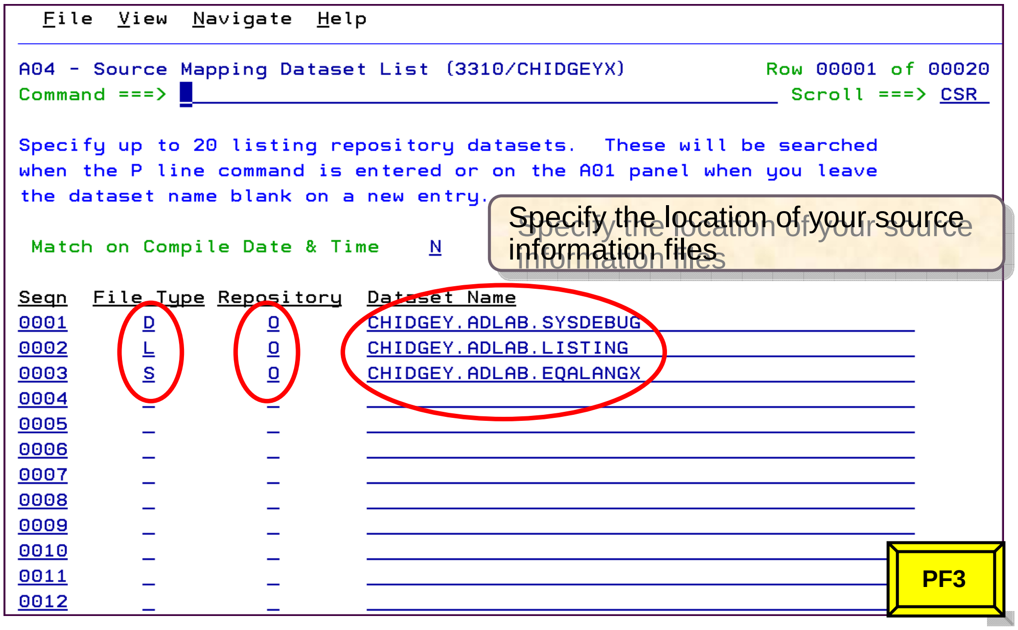
```
 File  View  Navigate  Help

W03: WAIT Referred Attribution by Task
Command ===> A04

Name          Description         Percent of Time in WAIT * 10.00%  ±1.0%
                                      *....1....2....3....4....5....6....7....8.
SAM1V-001     TCB=00AE6968        8.16
→ SAM1V         Application       8.16
                Program
  → SAM1V         CSECT in SAM1V  8.16
    → 00185A        Attribution   7.92
                    Offset in
                    SAM1V
      → IDA019L1      Virtual     7.92
                      I/O (VIO)
                      and VSAM
        → IDA019R3      Virtual   7.92
                        I/O
                        (VIO)
                        and VSAM
    → 001594        Attribution   0.13
                    Offset in
                    SAM1V
```

Timesaving tip: Enter your source information files on the **A04** panel…

The first line shows the task called SAM1V-1. Under that task, is the program, SAM1V. The task was responsible for all of the wait time, 8.16%, and module SAM1V within the task was responsible for all of it. Within the Sam1V module is a CSECT named SAM1V. When a program CSECT is shown, you can drill down into program source code.

But before you do that, you should tell APA where to find your source information. You do that with the “A04” primary command. Just type “A04” on the command line and press Enter.

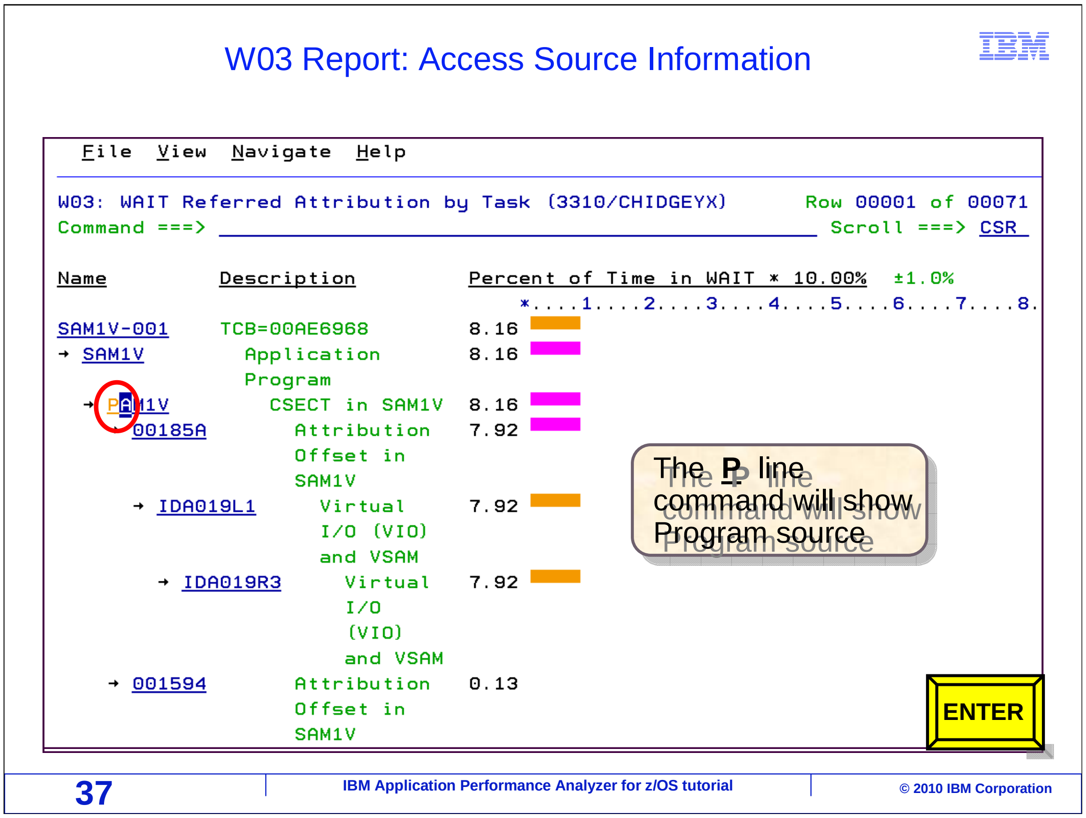
A04: Source Mapping Dataset List



```
  File  View  Navigate  Help

A04 - Source Mapping Dataset List (3310/CHIDGEYX)            Row 00001 of 00020
Command ===> _                                                  Scroll ===> CSR

Specify up to 20 listing repository datasets.  These will be searched
when the P line command is entered or on the A01 panel when you leave
the dataset name blank on a new entry.

 Match on Compile Date & Time    N

Seqn   File Type Repository  Dataset Name
0001       D         O       CHIDGEY.ADLAB.SYSDEBUG
0002       L         O       CHIDGEY.ADLAB.LISTING
0003       S         O       CHIDGEY.ADLAB.EQALANGX
0004       _         _
0005       _         _
0006       _         _
0007       _         _
0008       _         _
0009       _         _
0010       _         _
0011       _         _
0012       _         _
```

Specify the location of your source information files

This will bring up the A04 Source Mapping Dataset List panel. Use this panel to specify the location of your source information files. If you do this,   APA will automatically search the list for your source. In this example, the libraries that contain source information is specified. You can learn more about source information files in the “Working with Program Source” section of this tutorial. PF3 back.



W03 Report: Access Source Information

```
  File  View  Navigate  Help

W03: WAIT Referred Attribution by Task (3310/CHIDGEYX)      Row 00001 of 00071
Command ===> ________________________________________________  Scroll ===> CSR

Name         Description         Percent of Time in WAIT * 10.00%  ±1.0%
                                     *....1....2....3....4....5....6....7....8.
SAM1V-001    TCB=00AE6968        8.16 ████
→ SAM1V        Application       8.16 ████
               Program
  → PAM1V        CSECT in SAM1V  8.16 ████
    → 00185A       Attribution   7.92 ████
                   Offset in
                   SAM1V
      → IDA019L1     Virtual     7.92 ████
                     I/O (VIO)
                     and VSAM
        → IDA019R3     Virtual   7.92 ████
                       I/O
                       (VIO)
                       and VSAM
    → 001594       Attribution   0.13
                   Offset in
                   SAM1V
```

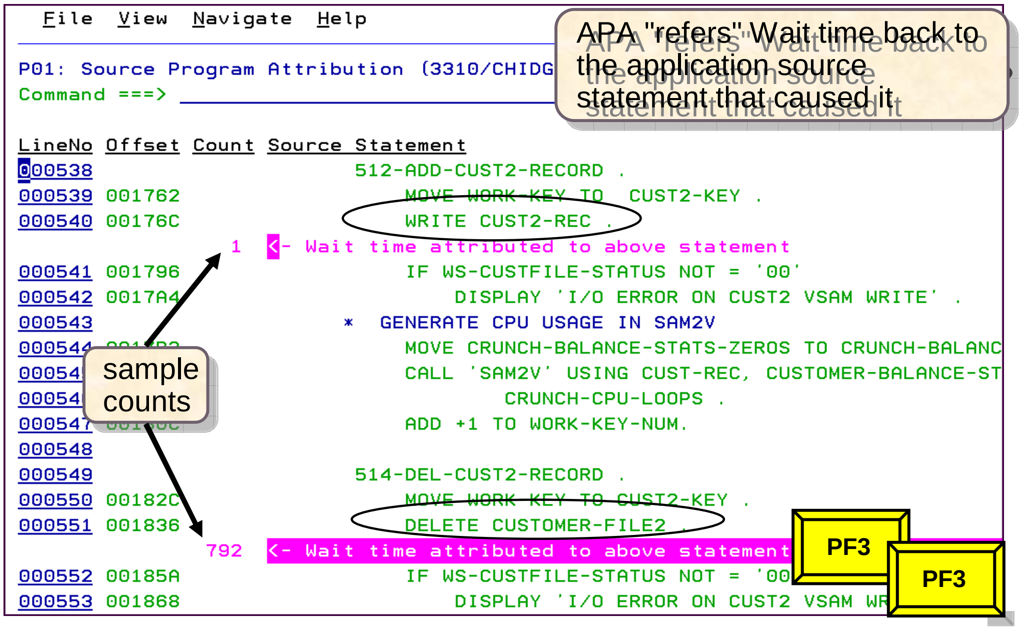
The **P** line command will show Program source

ENTER

Remember that when a program CSECT is shown, you can drill down into program source code.

You do that with the “P” (for program) line command. Here, “P” is typed on the entry for CSECT SAM1V to access the source, and Enter is pressed.

Source information is displayed



```
  File  View  Navigate  Help

P01: Source Program Attribution (3310/CHIDG
Command ===> ____________________________

LineNo Offset Count Source Statement
000538                     512-ADD-CUST2-RECORD .
000539 001762                  MOVE WORK-KEY TO  CUST2-KEY .
000540 00176C                  WRITE CUST2-REC .
                    1  <- Wait time attributed to above statement
000541 001796                  IF WS-CUSTFILE-STATUS NOT = '00'
000542 0017A4                      DISPLAY 'I/O ERROR ON CUST2 VSAM WRITE' .
000543                       *  GENERATE CPU USAGE IN SAM2V
000544                         MOVE CRUNCH-BALANCE-STATS-ZEROS TO CRUNCH-BALANC
00054                          CALL 'SAM2V' USING CUST-REC, CUSTOMER-BALANCE-ST
00054                                  CRUNCH-CPU-LOOPS .
000547                         ADD +1 TO WORK-KEY-NUM.
000548
000549                     514-DEL-CUST2-RECORD .
000550 00182C                  MOVE WORK-KEY TO CUST2-KEY .
000551 001836                  DELETE CUSTOMER-FILE2 .
                  792  <- Wait time attributed to above statement
000552 00185A                  IF WS-CUSTFILE-STATUS NOT = '00
000553 001868                      DISPLAY 'I/O ERROR ON CUST2 VSAM WR
```

APA "refers" Wait time back to the application source statement that caused it

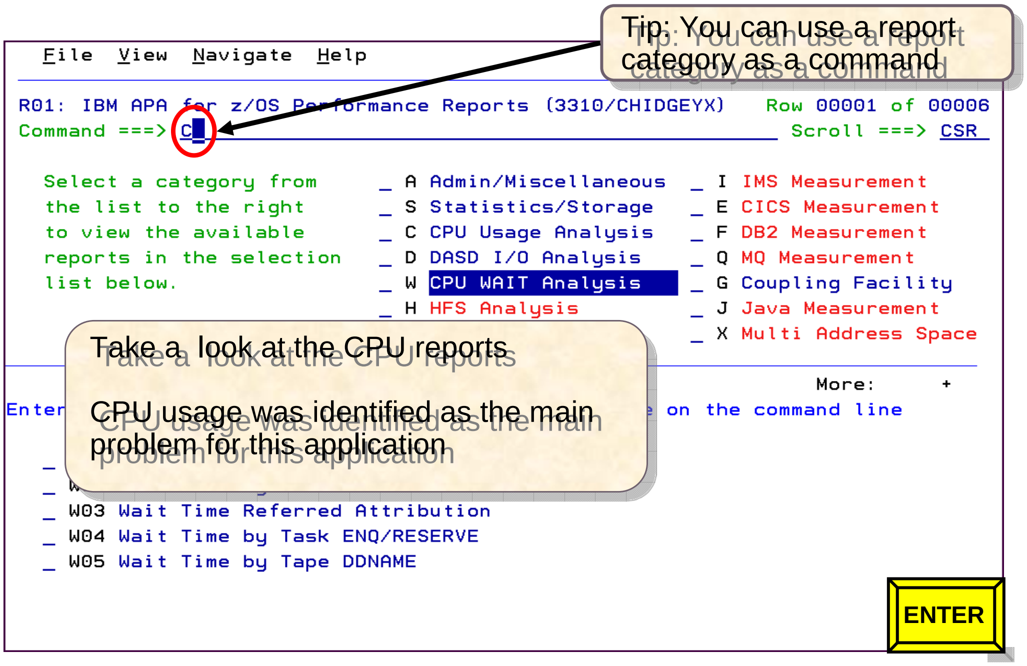
sample counts

Now, APA has drilled down into the program source information using data from the source mapping file. This is a very important feature of APA. In this report, it refers wait time back to the application's source statements that caused it. This is very important, because when it comes to wait time, there is typically some system-level program that executes the code that actually causes the wait. But the system-level programs are running because they are doing something that was requested by the application.

When APA samples an application in a wait state, it traverses the call chain back to the application program. Because of this capability, you can identify exactly which of the programs statements caused the wait time, and this is extremely valuable information. In this example, there are two program statements that caused wait time. A ‘write’ statement, and a ‘delete’ statement. Notice the sample counts. APA collected one sample in wait state because of the ‘write’ statement. That is very insignificant because of the low sample count. However, it collected 792 samples in wait state because of the ‘delete’ statement.

If you want to tune this application to reduce wait time, you should probably focus on the delete statement, since most of the wait time is attributed to that statement. In this example, PF3 is hit a couple of times to get back to the report menu.

Select the CPU reports category



```
 File  View  Navigate  Help
R01: IBM APA for z/OS Performance Reports (3310/CHIDGEYX)    Row 00001 of 00006
Command ===> C                                               Scroll ===> CSR

  Select a category from      _ A Admin/Miscellaneous   _ I IMS Measurement
  the list to the right       _ S Statistics/Storage    _ E CICS Measurement
  to view the available       _ C CPU Usage Analysis    _ F DB2 Measurement
  reports in the selection    _ D DASD I/O Analysis     _ Q MQ Measurement
  list below.                 _ W CPU WAIT Analysis     _ G Coupling Facility
                              _ H HFS Analysis          _ J Java Measurement
                                                        _ X Multi Address Space

                                                             More:     +
Enter                                           on the command line

   _
   _ W
   _ W03 Wait Time Referred Attribution
   _ W04 Wait Time by Task ENQ/RESERVE
   _ W05 Wait Time by Tape DDNAME
```

Tip: You can use a report category as a command

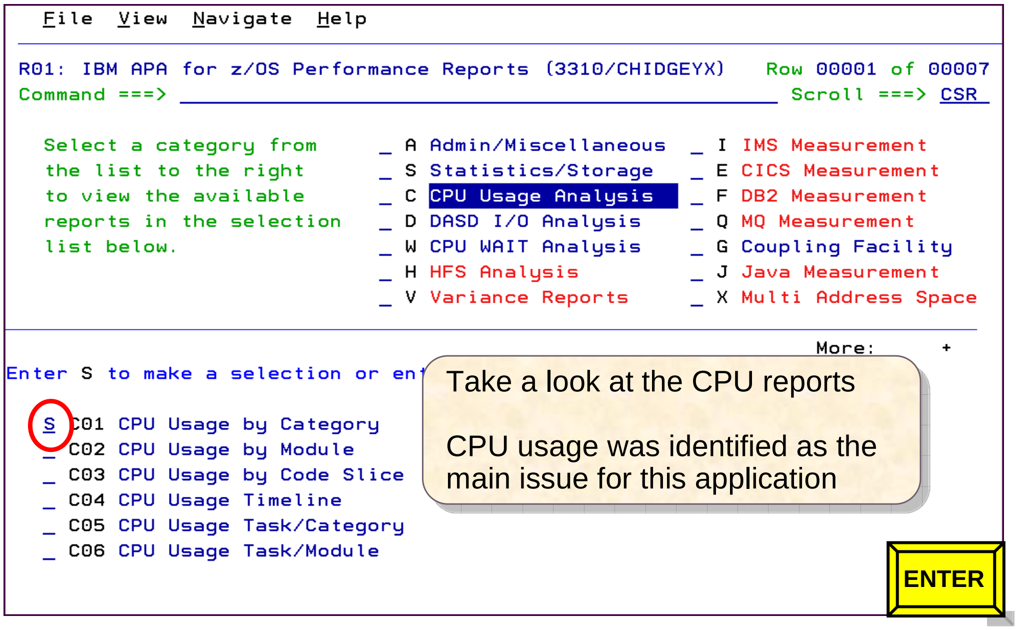
Take a look at the CPU reports

CPU usage was identified as the main problem for this application

At this point, you have explored a couple of the more important reports that can be used to find out where an application spent its ‘wait’ time. Next, you will explore some of the CPU reports. Remember that this application spent about 88% of its time in CPU, so CPU is the main performance issue in this example. You can switch over to the CPU reports by typing “S” next to the CPU category. Or, here is another way to select a category.

Just type the first letter of a report on the command line. For instance, type “C” (for CPU) on the command line, and press Enter.

Select the CPU Usage By Category report



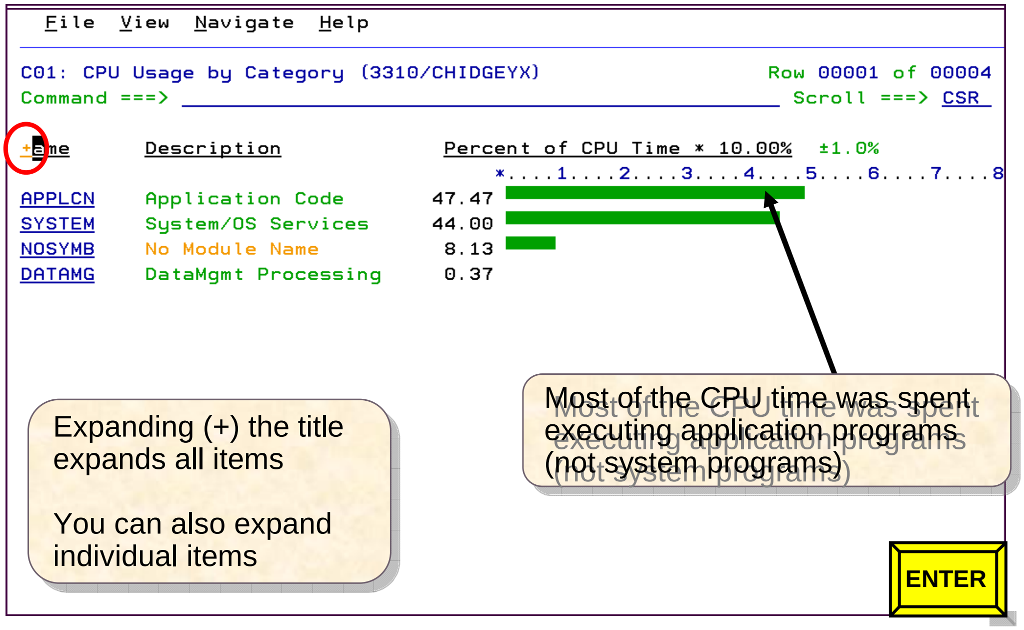
```
  File  View  Navigate  Help

R01: IBM APA for z/OS Performance Reports (3310/CHIDGEYX)    Row 00001 of 00007
Command ===> ________________________________________________  Scroll ===> CSR

  Select a category from     _ A Admin/Miscellaneous    _ I IMS Measurement
  the list to the right      _ S Statistics/Storage     _ E CICS Measurement
  to view the available      _ C CPU Usage Analysis     _ F DB2 Measurement
  reports in the selection   _ D DASD I/O Analysis      _ Q MQ Measurement
  list below.                _ W CPU WAIT Analysis      _ G Coupling Facility
                             _ H HFS Analysis           _ J Java Measurement
                             _ V Variance Reports       _ X Multi Address Space

                                                                    More:     +
Enter S to make a selection or ent

  S C01 CPU Usage by Category
  _ C02 CPU Usage by Module
  _ C03 CPU Usage by Code Slice
  _ C04 CPU Usage Timeline
  _ C05 CPU Usage Task/Category
  _ C06 CPU Usage Task/Module
```

Take a look at the CPU reports

CPU usage was identified as the main issue for this application

That shows the CPU reports. You can start the CPU research by examining the ‘C01: CPU Usage by Category’ report.

C01: CPU Usage by Category report



```
   File  View  Navigate  Help

C01: CPU Usage by Category (3310/CHIDGEYX)                    Row 00001 of 00004
Command ===> ______________________________________________     Scroll ===> CSR 

+ame      Description          Percent of CPU Time * 10.00%  ±1.0%
                                   *....1....2....3....4....5....6....7....8
APPLCN    Application Code      47.47 ========================
SYSTEM    System/OS Services    44.00 ======================
NOSYMB    No Module Name         8.13 ====
DATAMG    DataMgmt Processing    0.37
```

Expanding (+) the title expands all items

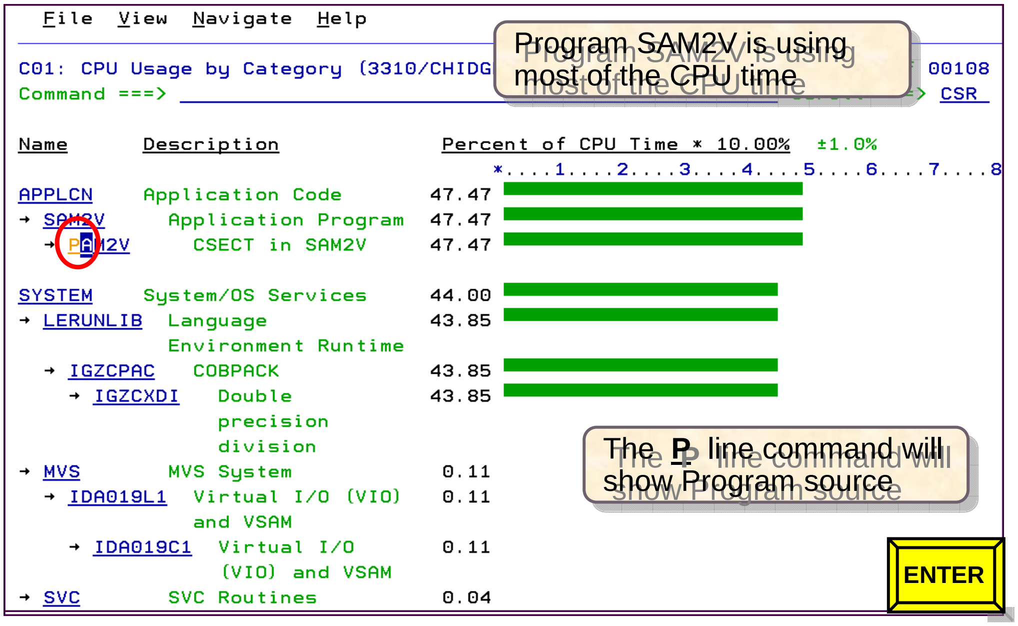
You can also expand individual items

Most of the CPU time was spent executing application programs (not system programs)

This report breaks CPU time down into categories. In this example, you know that nearly all of the CPU time was spent executing application programs, since most of the time is attributed to the application category. There are four categories listed on this report:: application, system services, no module name and data management. There are other categories, but only these four were sampled in this application. It is interesting that such a large percentage of the CPU time was spent in the application category. That means that most of the CPU time was spent actually executing the instructions within the application programs. It is typical to have more time spent in other categories. For example, a lot of CPU time might be spent performing data management operations. The fact that so much time was spent in application code is an indicator that this application might have been in a loop.

To expand all of the entries, type a “+” line command in the column heading. Press Enter.

C01: CPU Usage by Category report expanded



```
C01: CPU Usage by Category (3310/CHIDG                        00108
Command ===>                                               ===> CSR

Name      Description                Percent of CPU Time * 10.00%  ±1.0%
                                         *....1....2....3....4....5....6....7....8
APPLCN    Application Code         47.47
→ SAM2V     Application Program    47.47
  → PAM2V     CSECT in SAM2V       47.47

SYSTEM    System/OS Services       44.00
→ LERUNLIB  Language               43.85
            Environment Runtime
  → IGZCPAC   COBPACK              43.85
    → IGZCXDI   Double             43.85
                precision
                division
→ MVS       MVS System              0.11
  → IDA019L1  Virtual I/O (VIO)     0.11
              and VSAM
    → IDA019C1  Virtual I/O         0.11
                (VIO) and VSAM
→ SVC       SVC Routines            0.04
```

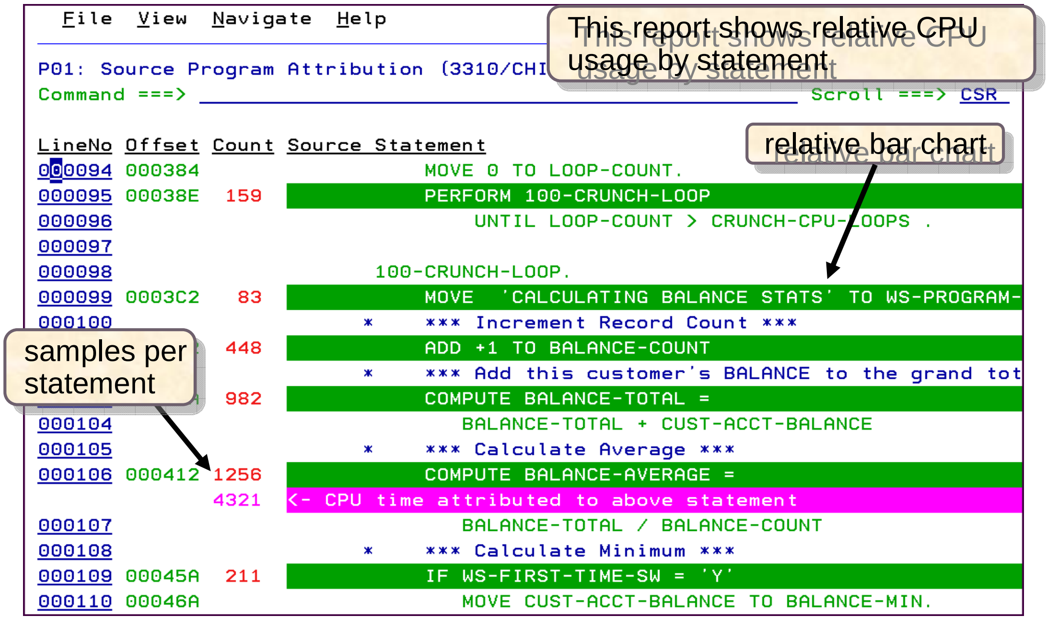
Program SAM2V is using most of the CPU time

The **P** line command will show Program source

With everything expanded, you can identify which application programs used all that CPU time. In this case, all of the CPU time was used by just one program: SAM2V. You can go straight to the source code to identify which statements caused the most CPU time.

Here, a “P” (for program) line command is typed next to the CSECT entry for the program and enter is pressed.

Source information is displayed



```
  File  View  Navigate  Help

P01: Source Program Attribution (3310/CHI
Command ===> ________________________________________________ Scroll ===> CSR

LineNo Offset Count Source Statement
000094 000384                  MOVE 0 TO LOOP-COUNT.
000095 00038E   159            PERFORM 100-CRUNCH-LOOP
000096                             UNTIL LOOP-COUNT > CRUNCH-CPU-LOOPS .
000097
000098                     100-CRUNCH-LOOP.
000099 0003C2    83            MOVE  'CALCULATING BALANCE STATS' TO WS-PROGRAM-
000100                   *     *** Increment Record Count ***
                448            ADD +1 TO BALANCE-COUNT
                         *     *** Add this customer's BALANCE to the grand tot
                982            COMPUTE BALANCE-TOTAL =
000104                            BALANCE-TOTAL + CUST-ACCT-BALANCE
000105                   *     *** Calculate Average ***
000106 000412  1256            COMPUTE BALANCE-AVERAGE =
               4321    <- CPU time attributed to above statement
000107                            BALANCE-TOTAL / BALANCE-COUNT
000108                   *     *** Calculate Minimum ***
000109 00045A   211            IF WS-FIRST-TIME-SW = 'Y'
000110 00046A                     MOVE CUST-ACCT-BALANCE TO BALANCE-MIN.
```

This report shows relative CPU usage by statement

relative bar chart

samples per statement

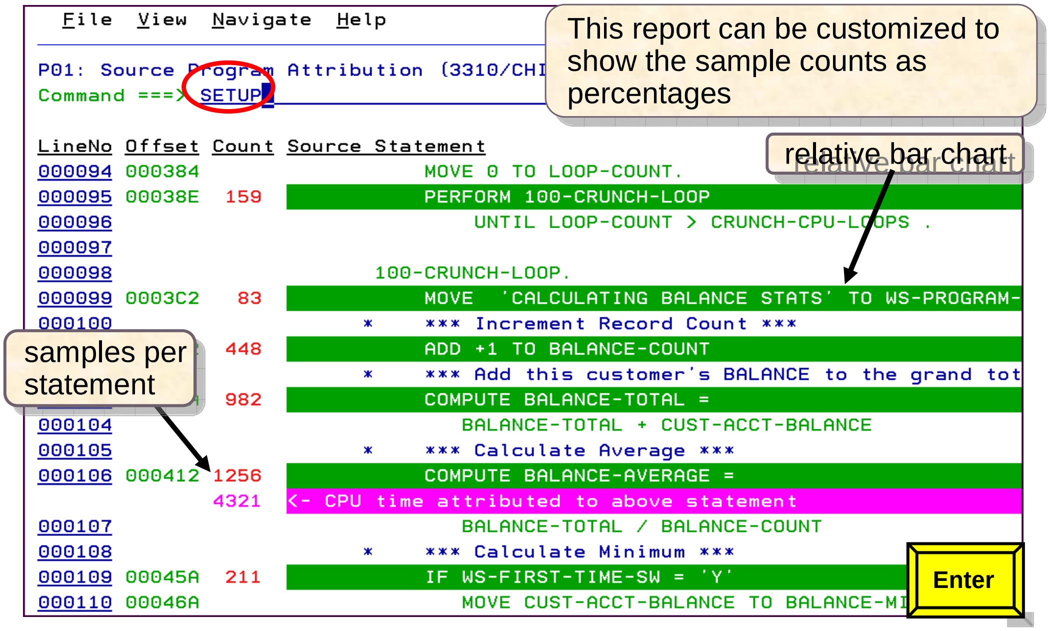
Here are the program statements that used the most CPU time. There is a bar chart overlaid on top of the program, to give you visual clues as to which statements were the heavy hitters. The bar chart is helpful, but it is more important to reference the sample counts, or the samples per statement. These show the relative impact of each statement. Notice that most of the CPU time was spent in the same area, and two of the statements (both COMPUTE) have the bulk of the responsibility.

But there are statements that did NOT show any samples. If a statement does not have a sample count, does that mean that it did not run at all? The answer is… maybe. Remember that APA is taking samples at fixed intervals, it is not monitoring every single instruction in the program.

As a result, if a program statement does not use much resource, there is a good chance that it is not running when APA takes a sample, and it will not have a sample count. However, that is OK. Because if a statement did not use much resource, when it comes to performance analysis, it really does not matter if it ran or not.

The idea is to identify the statements that use the most resource. And statistically speaking, those are the statements with the highest sample counts. The same thing can hold true for programs. Subroutines that run very quickly might not be caught in the act of running by APA. Again, that is OK, because those programs did not use much CPU or wait time. .

Use the setup command



```
  File  View  Navigate  Help

P01: Source Program Attribution (3310/CHI
Command ===> SETUP

LineNo Offset Count Source Statement
000094 000384              MOVE 0 TO LOOP-COUNT.
000095 00038E   159        PERFORM 100-CRUNCH-LOOP
000096                         UNTIL LOOP-COUNT > CRUNCH-CPU-LOOPS .
000097
000098                     100-CRUNCH-LOOP.
000099 0003C2    83        MOVE  'CALCULATING BALANCE STATS' TO WS-PROGRAM-
000100                 *     *** Increment Record Count ***
                448        ADD +1 TO BALANCE-COUNT
                       *     *** Add this customer's BALANCE to the grand tot
                982        COMPUTE BALANCE-TOTAL =
000104                        BALANCE-TOTAL + CUST-ACCT-BALANCE
000105                 *     *** Calculate Average ***
000106 000412  1256        COMPUTE BALANCE-AVERAGE =
               4321  <- CPU time attributed to above statement
000107                        BALANCE-TOTAL / BALANCE-COUNT
000108                 *     *** Calculate Minimum ***
000109 00045A   211        IF WS-FIRST-TIME-SW = 'Y'
000110 00046A                 MOVE CUST-ACCT-BALANCE TO BALANCE-MI
```

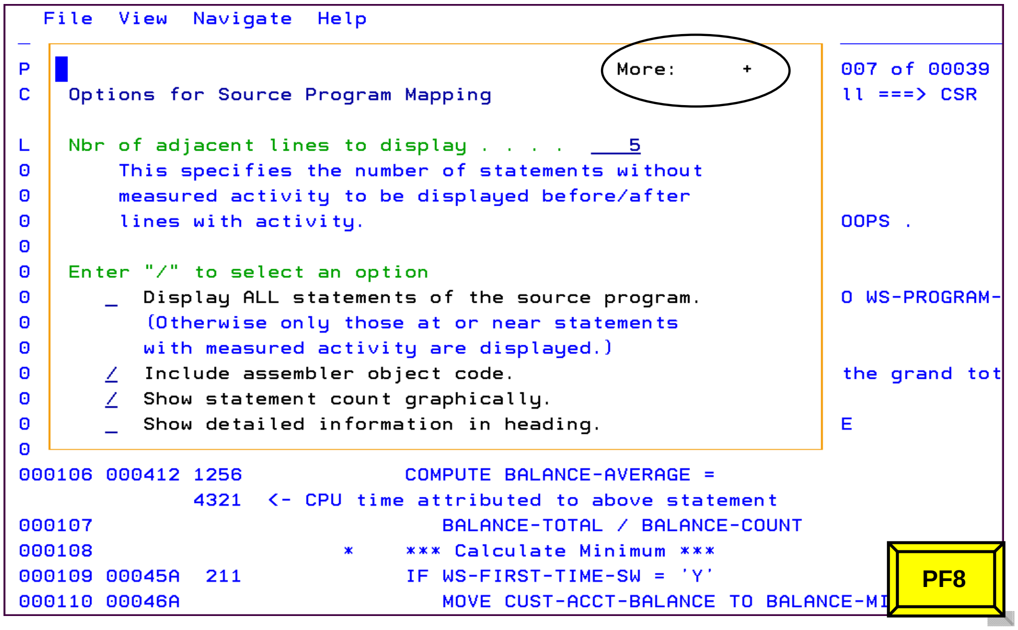
This report can be customized to show the sample counts as percentages

relative bar chart

samples per statement

Remember that many of the reports can be customized. In this example the setup primary command is used to customize this report. Enter.

Customizing the Source Program Mapping

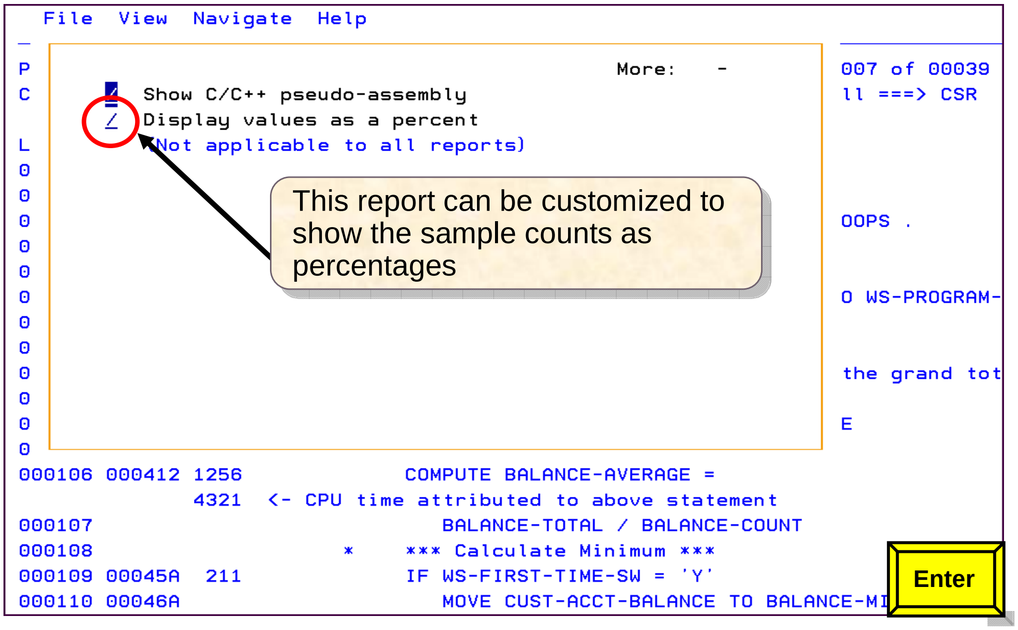


```
  File  View  Navigate  Help
 _
 P                                              More:     +     007 of 00039
 C   Options for Source Program Mapping                         ll ===> CSR

 L   Nbr of adjacent lines to display . . . .   ___5
 0       This specifies the number of statements without
 0       measured activity to be displayed before/after
 0       lines with activity.                                    OOPS .
 0
 0   Enter "/" to select an option
 0      _  Display ALL statements of the source program.        O WS-PROGRAM-
 0         (Otherwise only those at or near statements
 0         with measured activity are displayed.)
 0      /  Include assembler object code.                       the grand tot
 0      /  Show statement count graphically.
 0      _  Show detailed information in heading.                E
 0
000106 000412 1256             COMPUTE BALANCE-AVERAGE =
              4321  <- CPU time attributed to above statement
000107                            BALANCE-TOTAL / BALANCE-COUNT
000108                      *     *** Calculate Minimum ***
000109 00045A  211             IF WS-FIRST-TIME-SW = 'Y'
000110 00046A                     MOVE CUST-ACCT-BALANCE TO BALANCE-MI
```

What customizations can be done depends on the report or panel where the setup command was issued. In the Source program attribution report, several options are available for customization.

PF8 to see more options.

Customizing the Source Program Mapping

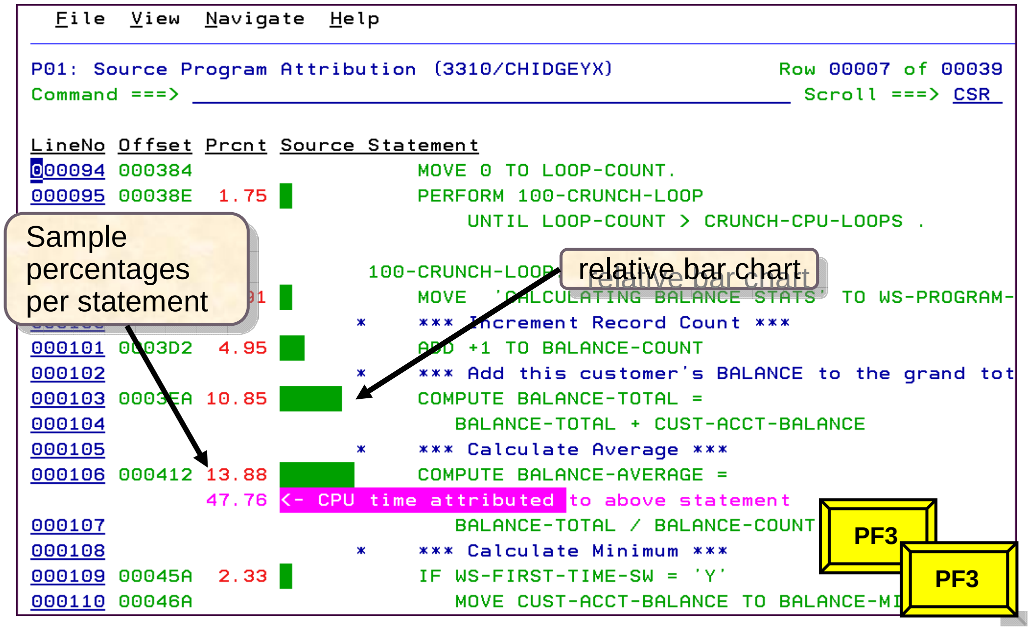


```
  File  View  Navigate  Help
_
P                                                               More:    -    007 of 00039
C       /  Show C/C++ pseudo-assembly                                         ll ===> CSR
        /  Display values as a percent
L          (Not applicable to all reports)
0
0
0                                                                             OOPS .
0
0
0                                                                             O WS-PROGRAM-
0
0
0                                                                             the grand tot
0
0                                                                             E
0
000106 000412 1256             COMPUTE BALANCE-AVERAGE =
              4321  <- CPU time attributed to above statement
000107                            BALANCE-TOTAL / BALANCE-COUNT
000108                      *    *** Calculate Minimum ***
000109 00045A  211             IF WS-FIRST-TIME-SW = 'Y'
000110 00046A                     MOVE CUST-ACCT-BALANCE TO BALANCE-MI
```

This report can be customized to show the sample counts as percentages

Here, a slash is placed by the option to display the sample values as percentages. Enter.

Samples shown as percentages



```
P01: Source Program Attribution (3310/CHIDGEYX)                 Row 00007 of 00039
Command ===> ______________________________________________     Scroll ===> CSR

LineNo Offset Prcnt Source Statement
000094 000384                      MOVE 0 TO LOOP-COUNT.
000095 00038E  1.75 █              PERFORM 100-CRUNCH-LOOP
                                       UNTIL LOOP-COUNT > CRUNCH-CPU-LOOPS .

                                100-CRUNCH-LOOP
                  91 █              MOVE  'CALCULATING BALANCE STATS' TO WS-PROGRAM-
                               *    *** Increment Record Count ***
000101 0003D2  4.95 ██             ADD +1 TO BALANCE-COUNT
000102                         *    *** Add this customer's BALANCE to the grand tot
000103 0003EA 10.85 █████          COMPUTE BALANCE-TOTAL =
000104                                 BALANCE-TOTAL + CUST-ACCT-BALANCE
000105                         *    *** Calculate Average ***
000106 000412 13.88 ██████         COMPUTE BALANCE-AVERAGE =
              47.76 <- CPU time attributed to above statement
000107                                 BALANCE-TOTAL / BALANCE-COUNT
000108                         *    *** Calculate Minimum ***
000109 00045A  2.33 █              IF WS-FIRST-TIME-SW = 'Y'
000110 00046A                          MOVE CUST-ACCT-BALANCE TO BALANCE-MI
```

Sample percentages per statement

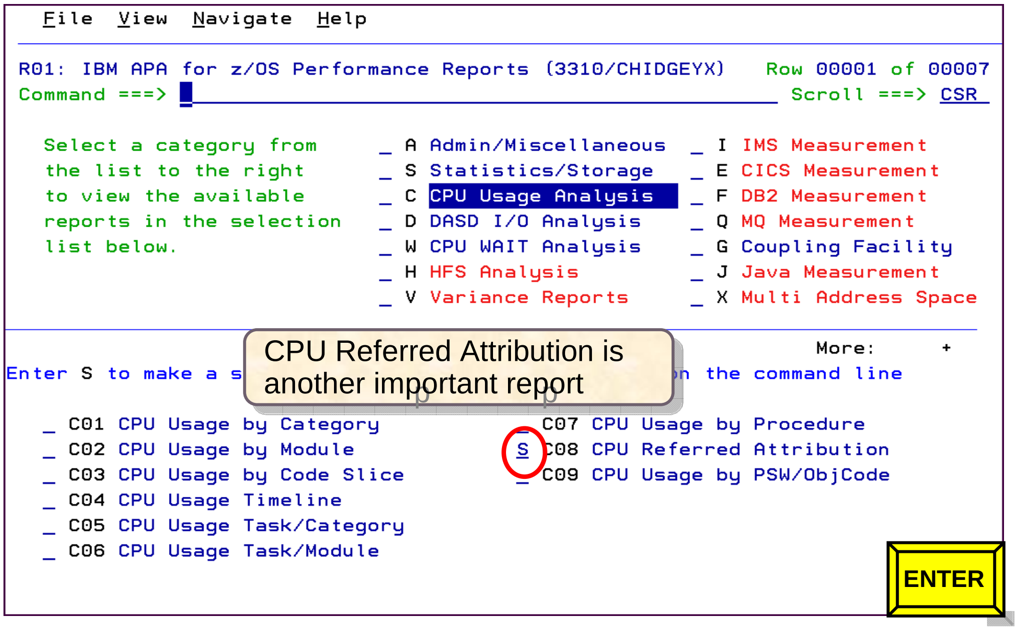
relative bar chart

PF3

PF3

Now, instead of showing the number of samples taken while a statement was executing. This report shows the samples as percentages of the total sample count taken while the application was executing instructions. Now press PF3 a couple of times to exit this report and get back to the report menu..

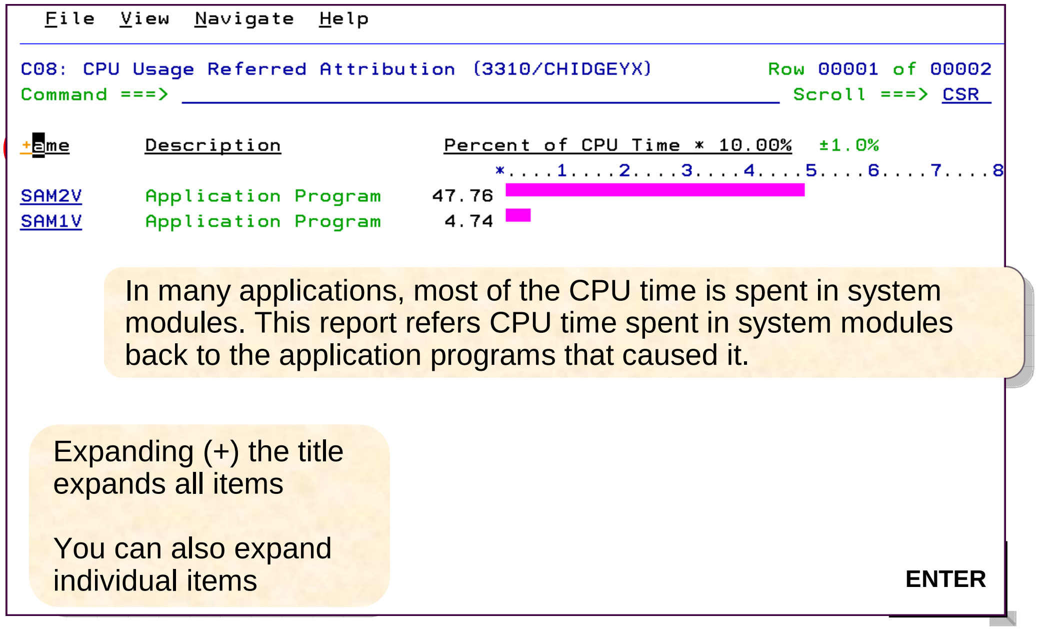
Select the CPU Referred Attribution report



```
   File  View  Navigate  Help
 ---------------------------------------------------------------------------
 R01: IBM APA for z/OS Performance Reports (3310/CHIDGEYX)    Row 00001 of 00007
 Command ===> _                                                 Scroll ===> CSR

   Select a category from     _ A Admin/Miscellaneous  _ I IMS Measurement
   the list to the right      _ S Statistics/Storage   _ E CICS Measurement
   to view the available      _ C CPU Usage Analysis   _ F DB2 Measurement
   reports in the selection   _ D DASD I/O Analysis    _ Q MQ Measurement
   list below.                _ W CPU WAIT Analysis    _ G Coupling Facility
                              _ H HFS Analysis         _ J Java Measurement
                              _ V Variance Reports     _ X Multi Address Space
 ---------------------------------------------------------------------------
                                                               More:     +
Enter S to make a s                                 n the command line

   _ C01 CPU Usage by Category                 _ C07 CPU Usage by Procedure
   _ C02 CPU Usage by Module                   S C08 CPU Referred Attribution
   _ C03 CPU Usage by Code Slice               _ C09 CPU Usage by PSW/ObjCode
   _ C04 CPU Usage Timeline
   _ C05 CPU Usage Task/Category
   _ C06 CPU Usage Task/Module
```

There is another report that is very important when you are researching CPU time in an application. It is the ‘C08 CPU referred attribution’ report. You can select it from the menu with an “S” line command. Enter.

C08: CPU Usage Referred Attribution report



```
  File  View  Navigate  Help

C08: CPU Usage Referred Attribution (3310/CHIDGEYX)         Row 00001 of 00002
Command ===> ________________________________________________  Scroll ===> CSR

+ame      Description          Percent of CPU Time * 10.00%  ±1.0%
                                   *....1....2....3....4....5....6....7....8
SAM2V     Application Program   47.76 
SAM1V     Application Program    4.74 
```

In many applications, most of the CPU time is spent in system modules. This report refers CPU time spent in system modules back to the application programs that caused it.

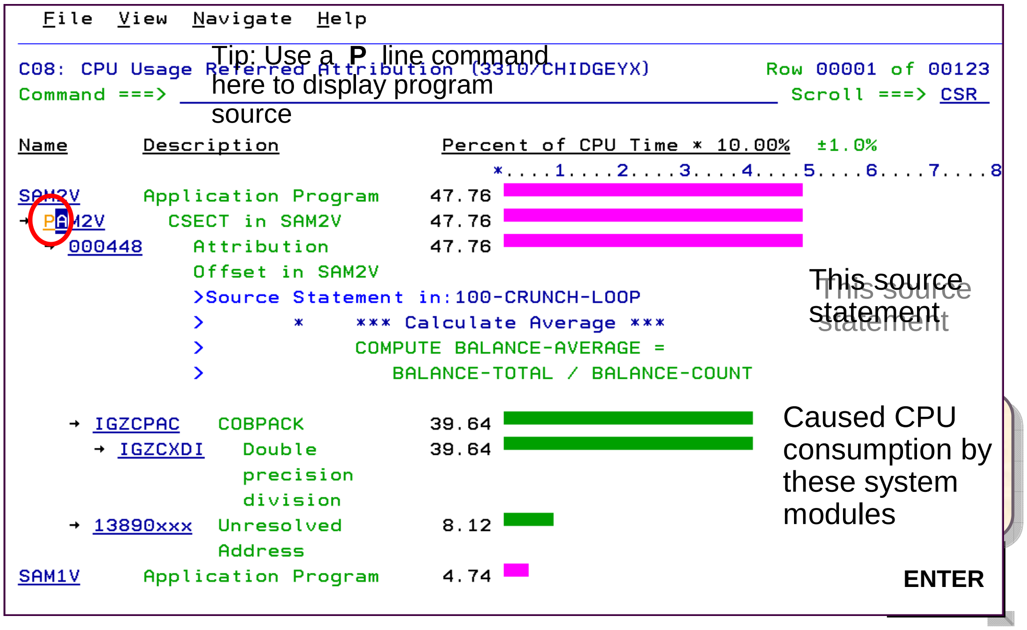
Expanding (+) the title expands all items

You can also expand individual items

ENTER

This report shows you how CPU time spent in system modules is referred back to the application program or programs that caused them to run. In many applications, most of the CPU time is spent in system modules, so this can be a very important report. Here, a “+” line command is entered in the column heading to expand all entries. Enter.

C08: CPU Usage Referred Attribution report expanded



```
  File  View  Navigate  Help

C08: CPU Usage Referred Attribution (3310/CHIDGEYX)          Row 00001 of 00123
Command ===> ________________________________________________  Scroll ===> CSR

Name      Description              Percent of CPU Time * 10.00%  ±1.0%
                                       *....1....2....3....4....5....6....7....8
SAM2V     Application Program     47.76 ========================
→ PAM2V     CSECT in SAM2V        47.76 ========================
  → 000448    Attribution         47.76 ========================
              Offset in SAM2V
              >Source Statement in:100-CRUNCH-LOOP
              >       *     *** Calculate Average ***
              >             COMPUTE BALANCE-AVERAGE =
              >                BALANCE-TOTAL / BALANCE-COUNT

    → IGZCPAC   COBPACK           39.64 ====================
      → IGZCXDI   Double          39.64 ====================
                  precision
                  division
    → 13890xxx  Unresolved         8.12 ====
                Address
SAM1V     Application Program      4.74 ==
```

Tip: Use a **P** line command here to display program source

This source statement

Caused CPU consumption by these system modules

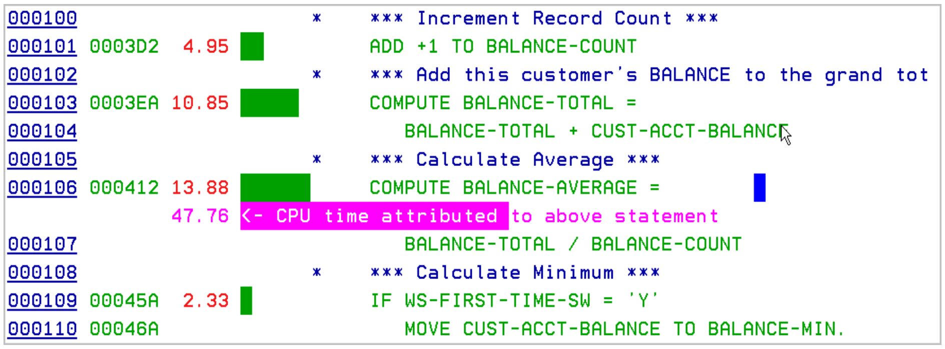
ENTER

Now all of the entries are expanded. Notice that there are system level programs listed under program SAM2V. These are all programs that are run due to service requests made by SAM2V. This is extremely important information, because it tells you exactly which statement caused the lower-level system routines to run. In this example, the 'Compute' caused an excessive amount of CPU time to be used by system-level routines.

If you use a "P" (for program) line command next to the CSECT entry for the application program , you can drill down into the program source. Hit Enter.

Tuning the application

- At this point, there is enough information to start tuning the application
  - The application spent most of its time in CPU Active
  - Most of the CPU time was spent executing instructions in application program SAM2V or system modules invoked by SAM2V
  - These source statements below caused most of the CPU time



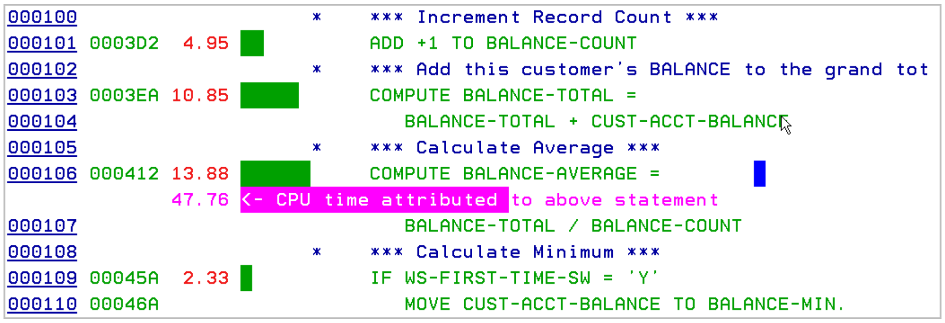
```
000100                     *    *** Increment Record Count ***
000101 0003D2   4.95            ADD +1 TO BALANCE-COUNT
000102                     *    *** Add this customer's BALANCE to the grand tot
000103 0003EA  10.85            COMPUTE BALANCE-TOTAL =
000104                             BALANCE-TOTAL + CUST-ACCT-BALANCE
000105                     *    *** Calculate Average ***
000106 000412  13.88            COMPUTE BALANCE-AVERAGE =
               47.76 <- CPU time attributed to above statement
000107                             BALANCE-TOTAL / BALANCE-COUNT
000108                     *    *** Calculate Minimum ***
000109 00045A   2.33            IF WS-FIRST-TIME-SW = 'Y'
000110 00046A                      MOVE CUST-ACCT-BALANCE TO BALANCE-MIN.
```

At this point, there is enough information to start tuning the application. You have learned that this application spent most of its time in the CPU active state. Most, in fact, nearly all, of the CPU time was spent executing instructions in an application program called SAM2V. These source statements caused most of the CPU time. Statements 101 through 110 are the primary culprits.

To recap how you can use APA to analyze an application's performance. Remember that the first order of business is to identify whether most of the time is being spent in CPU or wait state. After that, you know whether to focus your efforts on researching CPU or wait. Then find the program or programs that are causing most of the time, and if possible, drill down into the application source code to determine which statements are responsible.

Tuning the application

- What can be improved in this application?
- Statements 101 through 110 use a lot of CPU because they are inside an iterative loop, and do not need to be.
  - Improvement: move them outside the loop
- Many of the numeric work variables are in character format, such as BALANCE-COUNT and BALANCE-TOTAL
  - Improvement: define them as packed or binary numeric



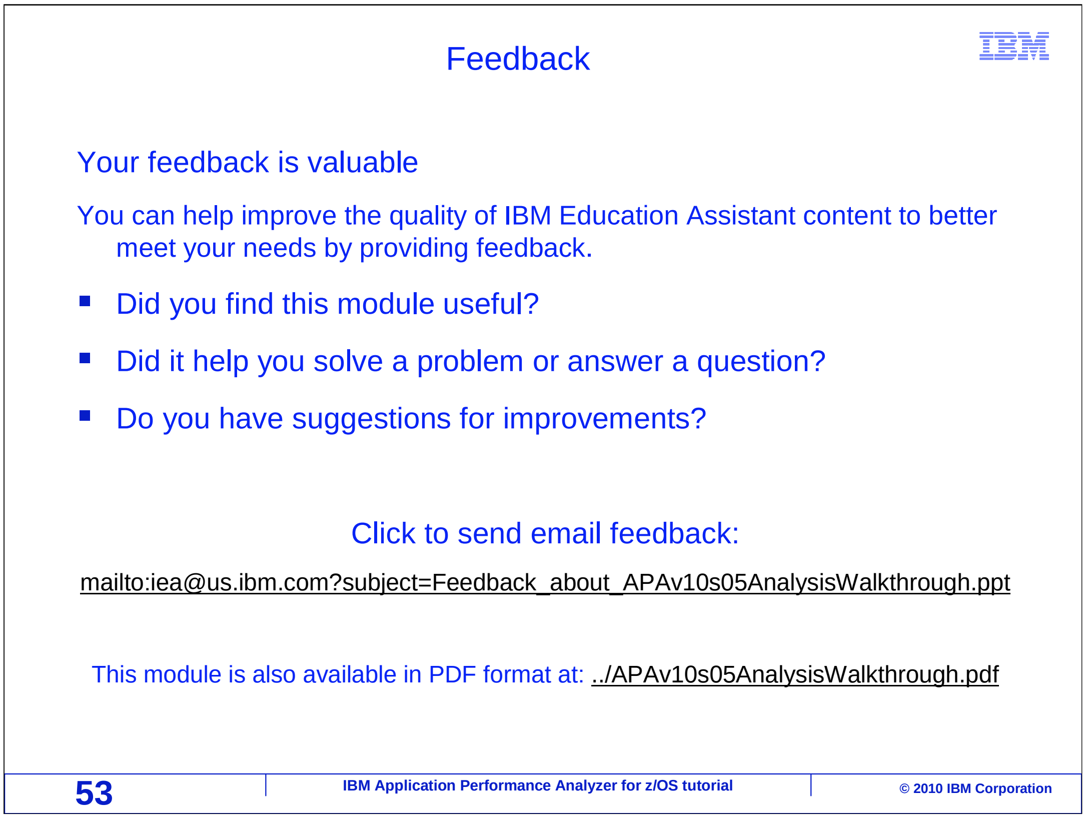
```
000100                      *    *** Increment Record Count ***
000101 0003D2  4.95              ADD +1 TO BALANCE-COUNT
000102                      *    *** Add this customer's BALANCE to the grand tot
000103 0003EA 10.85              COMPUTE BALANCE-TOTAL =
000104                              BALANCE-TOTAL + CUST-ACCT-BALANCE
000105                      *    *** Calculate Average ***
000106 000412 13.88              COMPUTE BALANCE-AVERAGE =
              47.76 <- CPU time attributed to above statement
000107                              BALANCE-TOTAL / BALANCE-COUNT
000108                      *    *** Calculate Minimum ***
000109 00045A  2.33              IF WS-FIRST-TIME-SW = 'Y'
000110 00046A                       MOVE CUST-ACCT-BALANCE TO BALANCE-MIN.
```

Using this example, what can be improved in this application? Statements 101 through 110 used a lot of CPU time because they are inside an iterative loop, and they do not need to be. An improvement can be to move them outside the loop. Also, a lot of the numeric work variables are defined in character format. That causes the program to convert them into a numeric format every time they are compared or used in a computation. It is more efficient to define them, for example, as packed decimal or binary. Just so you know, making those changes to this example program reduced the CPU time by more than half.

At this point, you have learned how you can use APA to research performance problems in your applications. There are a lot of reports available, and so far you have only gone through a few of the most important ones. However, now you should be able to navigate to reports, and research your applications using which ever reports is helpful based on your applications.

-

That is the end of this section, a walkthrough of a performance analysis.



Feedback

Your feedback is valuable

You can help improve the quality of IBM Education Assistant content to better meet your needs by providing feedback.

- Did you find this module useful?
- Did it help you solve a problem or answer a question?
- Do you have suggestions for improvements?

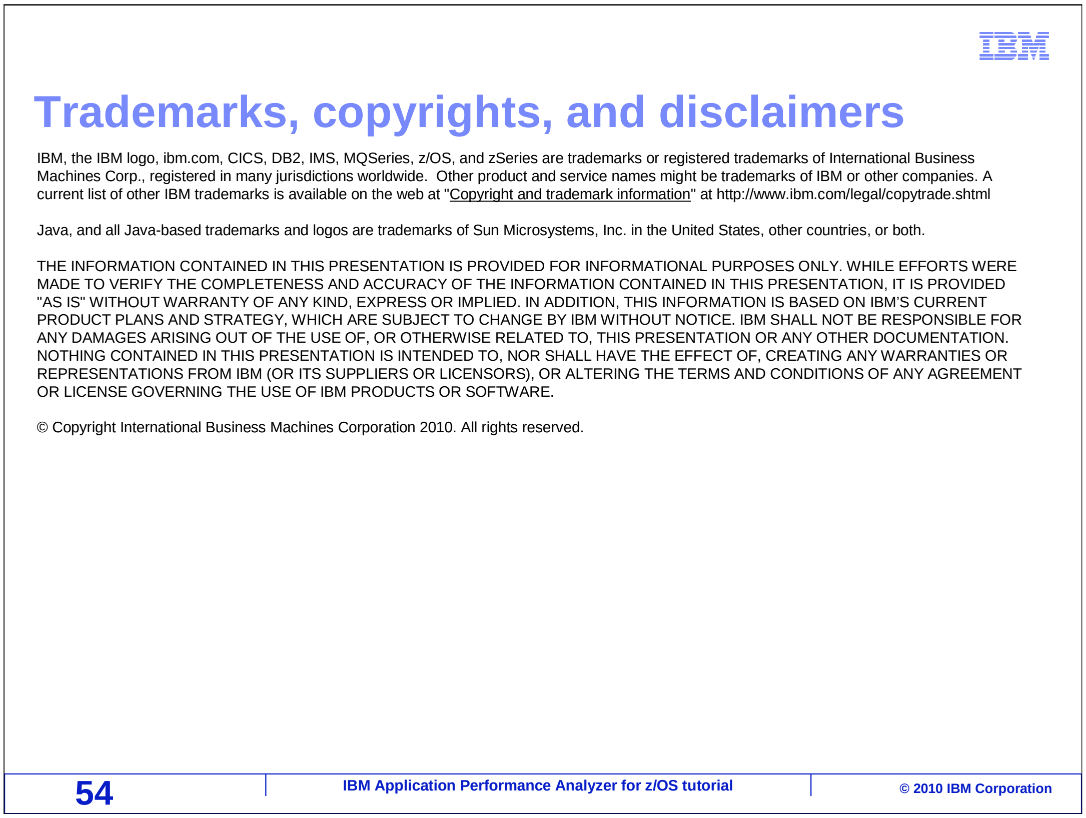
Click to send email feedback:

mailto:iea@us.ibm.com?subject=Feedback_about_APAv10s05AnalysisWalkthrough.ppt

This module is also available in PDF format at: ../APAv10s05AnalysisWalkthrough.pdf

53 IBM Application Performance Analyzer for z/OS tutorial © 2010 IBM Corporation

You can help improve the quality of IBM Education Assistant content by providing feedback.



Trademarks, copyrights, and disclaimers

IBM, the IBM logo, ibm.com, CICS, DB2, IMS, MQSeries, z/OS, and zSeries are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of other IBM trademarks is available on the web at "Copyright and trademark information" at http://www.ibm.com/legal/copytrade.shtml

Java, and all Java-based trademarks and logos are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

THE INFORMATION CONTAINED IN THIS PRESENTATION IS PROVIDED FOR INFORMATIONAL PURPOSES ONLY. WHILE EFFORTS WERE MADE TO VERIFY THE COMPLETENESS AND ACCURACY OF THE INFORMATION CONTAINED IN THIS PRESENTATION, IT IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED. IN ADDITION, THIS INFORMATION IS BASED ON IBM'S CURRENT PRODUCT PLANS AND STRATEGY, WHICH ARE SUBJECT TO CHANGE BY IBM WITHOUT NOTICE. IBM SHALL NOT BE RESPONSIBLE FOR ANY DAMAGES ARISING OUT OF THE USE OF, OR OTHERWISE RELATED TO, THIS PRESENTATION OR ANY OTHER DOCUMENTATION. NOTHING CONTAINED IN THIS PRESENTATION IS INTENDED TO, NOR SHALL HAVE THE EFFECT OF, CREATING ANY WARRANTIES OR REPRESENTATIONS FROM IBM (OR ITS SUPPLIERS OR LICENSORS), OR ALTERING THE TERMS AND CONDITIONS OF ANY AGREEMENT OR LICENSE GOVERNING THE USE OF IBM PRODUCTS OR SOFTWARE.