ProTop For Developers

[Sub-title]

[Tom Bascom, White Star Software]

November 10, 11:00, 45 minutes

Abstract: "Isn't ProTop a DBA tool? I'm an ABL developer, why would I care about it?" Yes, ProTop is an invaluable tool for OpenEdge DBAs. But they are not the only users we consider as we enhance the product. We also consider the needs and use-cases of ABL developers and we add many features specifically for developers. There are two primary developer use cases that are enhanced by using ProTop: performing root-cause analysis of issues in deployed applications, and measuring the data-access patterns and network efficiency of applications in development to ensure sub-optimal code is not deployed in the first place. Come to this session to learn how integrating ProTop into your development and troubleshooting workflow can benefit your company. We will show you how ProTop can give you the proof that your data-access and index selection are what you expect them to be, and ProTop can help you quickly pinpoint root cause and reduce mean time to resolution.

White Star
software

ProTop for Developers!

Tom Bascom, White Star Software
tom@wss.com

White Star
software

Who Is White Star Software?

- The world's oldest independent consulting organization focused on Progress OpenEdge – since 1987, our breadth of experience is unmatched:
  - From very small and chaotic to the largest and most demanding customers
  - Databases and application environments of all descriptions
  - Frequent guest speakers at Progress user conferences around the world

White Star
software

A Few Words about the Speaker

- Tom Bascom: Progress user & roaming DBA since 1987
- Partner: White Star Software, LLC
  - Expert consulting services related to all aspects of Progress and OpenEdge.
  - Remote database management service for OpenEdge.
  - Author of: protop
  - Simplifying the job of managing and monitoring the world's best business applications.
  - **tom@wss.com**

White Star software

ProTop Is Not Just For DBAs!

- ProTop can also be very valuable in your development and test environments!
- Many ProTop features are specifically designed to be helpful to developers.
- Developers can even use ProTop to defend themselves from cruel and heartless DBAs

White Star
software

Agenda – ProTop For Developers

- Programmer Mode
- Where is the problem? (CSC, proGetStack)
- Table and Index Activity
  - For a specific user or connection, “U”
  - For users of specific tables; “8”, or indexes; “9”
- Active Transactions, Blocked Sessions
- How much time did that really take?
- What’s going on with Temp Tables?
- User Table Statistics

White Star software

Programmer Mode

White Star
software

ProTop State, Default Mode

ProTop Friendly Name
Sampling Mode
Sample Type
Time Mode
Data Method
Data Rows
Data Bytes
Fetch Time

DB Logical Name
My Usr#
My Logical Reads
My OS Reads
Programmer Mode

```
wssdba@: /home/wssdba/pt3
xus61t2 Auto Interval Rate DS 394 28721 1.964    ProTop
xus61t2 213 18310 0
  p
         Hit%       99.97          Commits:      620
   Log Reads:     1681019            Undos:        0
  Log Writes:         263
    OS Reads:         493     Lock Tbl HWM:   570093
   OS Writes:          13     Curr # Locks:       11
 LogRd/LogWr:     6383.66         Lock Tbl%    0.00%
 LogRd/RecRd:        1.39         Rec Lk/s:      713
```

White Star software

Programmer Mode

- Changes from the ProTop default of “rate” metrics to providing the raw count “on demand” rather than at automatic intervals.
- Sample time is no longer considered.
- So instead of seeing 12345 record reads per second with the sample automatically refreshing every 10 seconds you see that there were 98765 record reads in the sample period that you choose.
- Enable with control-p.

White Star
software

ProTop State, Programmer Mode

ProTop Friendly Name
Sampling Mode
Sample Type
Time Mode
Data Method
Data Rows
Data Bytes
Fetch Time

DB Logical Name
My Usr#
My Logical Reads
My OS Reads
Programmer Mode

```
wssdba@: /home/wssdba/pt3
xus61t2 OnDemand Interval Summary DS 387 27088 1ProTop
xus61t2 213 18065 0 programmerMode
  p^f^p p
          Hit%      100.00          Commits:      861
   Log Reads:    15719177            Undos:        0
  Log Writes:         292
   OS Reads:           38     Lock Tbl HWM:   570093
   OS Writes:         126     Curr # Locks:       11
 LogRd/LogWr:    53832.80         Lock Tbl%    0.00%
 LogRd/RecRd:        1.60         Rec Lk/s:    1052
```

Programmer Mode - Usage

- Prepare your test scenario (in a different window from ProTop).
- In ProTop, select the USR (#) or ProcessId (P) to monitor, then select “U” for User Information.
- Initialize the counters with a <spacebar> command.
- Run your test scenario (in the other window).
- Come back to ProTop and get the results with another <spacebar>.

White Star
software

```
xus61t2 Auto Interval Rate DS 378 27294 1.613   ProTop Version 318 commit 3016821 as of October 28, 2021 17:24
xus61t2 204 28227 0                                          /db/trax/xus61t2
 #
        Hit%      100.00                                                                                46d 23:14    Connections:     153
   Log Reads:    1456990                                                                                   04:27           -n %       5
  Log Writes:        205                                                                                                Brokers:       7
    OS Reads:          0    Lock                                                                        00:00:00    SQL Servers:       2
   OS Writes:         15    Curr                                                                               0    SQL Clients:       0
 LogRd/LogWr:    7115.55                                                                                        0    4gl Servers:      14
 LogRd/RecRd:       1.73                                                                                        0     4gl RemCnx:       1
   Rec Reads:     840911    Lk D                                                                               0     App Server:      44
   Idx Reads:     840845                                                                                           Web Speed:       0
 Rec Creates:         66                                                                                   1 of 12       Local:      39
 Idx Creates:       1230                                                                                     45949       Batch:      30
 Rec Updates:         43  Random                                                                                11  BIW/AIW/WDOG:   1 1 1
 Rec Deletes:          9   Sync                                                                                  0      AI Mgmt:       1
 Idx Deletes:         68                                                                                                 APWs:       2
 Idx Blk Spl:          1    User                                                                                 0    RPLA/RPLS:     0 1
   Rec Waits:          0       l                                                                                 0    Utilities:       0
 Resrc Waits:          1                                                                                      0.00
 Latch Waits:          0                                                                                                  TRX:       2
  Latch Reqs:    4818163                                                                                             Blocked Rec:       0
 Latch/logRd:       3.31                                                                                             Other Blkd:       0
```

DB Connection Number

Enter the DB Connection Number for the session of interest. This is the "Usr#" column found on many ProTop screens, in the DB .lg file, and in many Progress error messages.

Usr# __236

Client statement cache type: ? 0 = Off
1 = Single (Top of Stack)
2 = Full Call Stack

Most of the time you want "1". You typically do not need the full stack unless you are planning to look at the detailed User Info screen (the "U" command). The full stack option obviously takes more memory, uses more space in the client/server networking communications and is more likely to use the scratch disk (resulting in extraneous IO operations).

Table Activity

	Tbl#	Area#	BX	Table Name	RM Chain	#Records	Frag%	Scat	Churn	AvgRow	Create	Read v	Update	Delete	OS Read
>	376	114	B1	im-trans	234887	60469169	0.33%	1	0.01	215	0	485639	0	0	0
	758	174	B1	supplier	22	37158	0.00%	1	9.21	299	0	342256	0	0	0
	452	174	B1	location	31	477	0.00%	1	4.90	246	0	2336	0	0	0
	782	20	B1	s_crm-crm-field	59	206	0.00%	1	9.43	58	0	1943	0	0	0
	1031	223	B1	ReplTableXRef	2	1079	0.00%	1	0.75	77	0	815	0	0	0
	-1	6	B1	_File		1285	72.92%	5	0.51	392	0	660	0	0	0

Index Activity

	Idx#	Area#	BX	Index Name	Blocks	Util	Lvls	Idx Root	Note	Create	Read v	Split	Delete	BlkDl
>	743	115	B1	im-trans.link-recno	95254	67.90%	3	383		0	462464	0	0	0
	1688	175	B1	supplier.supplier	81	64.70%	2	196	PU	0	340012	0	0	0
	911	175	B1	location.location	1	56.70%	1	772	PU	0	5968	0	0	0
	1964	19	B1	wm-send.WMS-ID	43607	56.90%	3	13695	U	0	1344	0	0	0
	1530	105	B1	so-trans-d.so-trans-d	252116	68.70%	3	127	PU	0	802	0	0	0

User IO Activity

	Usr#	Name	PID	Flags	Blk Ac v	OS Rd	OS Wr	Hit%	Rec Lck	Lk HWM	CSC Age	Line#	Program Name
>	236	traxcron7	150466	S4B	972311	0	0	100.00%	0	0			
	334	traxcron8	108847	S4B	695347	0	0	100.00%	0	0			
	249	traxcrm	197169	S4B	10071	12	0	99.89%	1319	0			
	836	836	192378	SAB	5309	0	0	100.00%	0	0			

White Star software

```
xus61t2 OnDemand Interval Summary DS 3546 230595ProTop Version 318 commit 3016821 as of October 28, 2021 17:24
xus61t2 204 4 0 programmerMode                                  /db/trax/xus61t2
  #U^p
   Login Name: traxcron7  Login Time:Fri Nov  5 00:05:01 2021          -Bp Bufs:        0        BI Reads:        0      Logical Rd:   9128959
         Usr#:       236  Device/IP:                                   -Bp Used:        0      BI Writes:        0      Logical Wr:         0
   Connect Id:       237  Full Name: traxcron7                                             AI Reads:        0      Disk Reads:        14
          PID:    150466      Phone: ?                                   Server:        0      AI Writes:        0            Hit%       100
          TID:         0     E-Mail: ?                                 Serv PID:        0                                Num TRX:         3
                                                                       Serv TID:        0                              Curr Locks:         0
 Session Info: ABL SELF S4B Batch                                                                                        Lock HWM:         0
     TRX Info: --None--
```

Session Table Activity

	Tbl#	Area#	Table Name	RM Chain	#Records	Frag%	Scat	Churn	AvgRow	Create	Read v	Update	Delete	OS Read
>	376	114	im-trans	234887	60469169	0.33%	1	0.08	215	0	4551448	0	0	?
	849	22	wb_dept-user	59	10406	0.00%	1	0.04	38	0	440	0	0	?
	169	20	cost-factor	60	19	0.00%	1	22.00	46	0	418	0	0	?
	181	20	currency-rate	29	37586	0.00%	1	0.01	53	0	202	0	0	?
	452	174	location	31	477	0.00%	1	0.29	246	0	136	0	0	?
	601	22	setup-control-access	198	327414	0.00%	1	0.00	40	0	132	0	0	?
	536	18	prod-exp-loc	5345	14703949	5.17%	1	0.00	219	0	123	0	0	?

Session Index Activity

	Idx#	Area#	Index Name	Blocks	Util	Lvls	Idx Root	Note	Create	Read v	Split	Delete	BlkDl
>	743	115	im-trans.link-recno	95254	67.90%	3	383		0	4554928	0	0	0
	1816	23	wb_dept-user.wb_user	38	69.70%	2	13439		0	920	0	0	0
	1333	23	setup-dept-access.link-ref	1	66.30%	1	9727		0	836	0	0	0
	369	21	cost-factor.cost-factor	1	2.30%	1	10751	PU	0	462	0	0	0
	1330	23	setup-control-access.setup-control-access	648	63.90%	2	9535	PU	0	418	0	0	0
	372	21	cost-factor-loc.cost-factor-loc	1	8.30%	1	10879	PU	0	375	0	0	0

Session 4GL Call Stack

	Dep v	Line#	Program Name
>	4	1008	dp/runtab.p
	3	1932	wb/onebatch.p
	2	713	im/impelq.p
	1	1120	print-pelq im/impelq.p

User's Other Sessions

	USR ^	PID	Flags	Server	Device/IP Address	Login Time	Line#	Program Name
>								

White Star software

```
Clear Client Statement Cache

Clear the  session that you enabled the Client Statement Cache for?

This isn't strictly required but in most cases it is best to proactively

clean these up rather than leave the sessions enabled.

<Yes> <No>
```

White Star
software

Programmer Mode (^p)

programmer mode (^p)

In programmer mode, the sampling interval is changed from automatic to on demand and the displayed data is displayed in raw numbers rather than converted to a rate. This allows the programmer to run a program, refresh the screen, then see exactly how mnay table and index reads they did.

Non-Programmer Mode

The following displays in the upper left corner of the screen

```
s2k Auto Interval Rate DS 15 1899 0.048
s2k 48 17 0
```

In normal, non-programmer mode, ProTop automatically samples every 'x' seconds (default is 10). The results display rates per second: reads/sec, writes/sec, etc.

Programmer Mode

The following displays in the upper left corner of the screen

```
s2k OnDemand Interval Summary DS 15 1913 0.028
s2k 48 17 0 programmerMode
```

Insight into Programmer Mode

The goal of programmer mode is to zoom in on your test user to see how much database activity is generated by your program. Follow these steps:

Where Is The Problem?

The Client Statement Cache

White Star
software

Where Is The Problem?

- Knowing that you have unexpected activity is a good first step.
- Knowing what line# of which program is even better!

CSC vs ProGetStack

CSC

- Pro:
  - Can be easily and selectively enabled from the db server
  - Only need DBA privileges
- Con:
  - CSC only reports the line# of the last *database* activity
  - CSC is “forward looking”
  - Has an impact on client/server connections

ProGetStack

- Pro:
  - Not restricted to reporting database access line numbers
  - ProGetStack does not need to be enabled in advance
- Con:
  - Must be executed from wherever the client is running, not the db server
  - Need system admin privileges

White Star
software

Client Statement Cache

User IO Activity

	Usr#	Name	PID	Flags	Blk Ac v	OS Rd	OS Wr	Hit%	Rec Lck	Lk HWM	CSC Age	Line#	Program Name
>	270	henry	98890	S4	679475	6	0	100.00%	0	0	00:00:00	1080	so/soregcomus.p
	338	xpejpari	230846	S4	459138	27	0	99.99%	0	0	00:00:00	4417	im/value.p
	482	xusgrami	83160	S4	365853	0	0	100.00%	17	0	00:00:00	12573	im/shdebitg.p
	407	xuscfran	21652	S4	363334	0	0	100.00%	17	0	00:00:00	12573	im/shdebitg.p
	271	henry	99770	S4	158989	179	0	99.89%	0	0	00:00:00	1487	im/lstinvdetfisprd.p
	384	xuyangul	13486	S4	5174	0	0	100.00%	1	0	00:00:01	9004	ar/custlu.y

Session 4GL Call Stack

	Dep v	Line#	Program Name
>	6	906	wb/workbook.p
	5	3163	WBmain wb/workbook.p
	4	3075	RunFunction wb/workbook.p
	3	2981	wb/batchrun.p
	2	1932	wb/onebatch.p
	1	1080	so/soregcomus.p

White Star software

Client Statement Cache Caveat

```
Global Client Statement Cache

The client statement cache is a powerful feature that sometimes causes
problems. If you are not comfortable with the potential issues, please
do not enable it in Production.

Rather than globally enabling CSC, ProTop can enable or disable the
client statement cache for specific users by using the "#" command and
entering a usr#.

You can also use the menu at PROMON R&D, 1, 18 for fine-grained control
over individual sessions.

Global Client Statement Cache status: On     On, Off, ? = no change
```

White Star software

Table and Index Activity

White Star
software

Setting The Stage For Table & Index Monitoring

```
Table and Index Range Information

                    -basetable:    -364                        -baseindex:  -1,679
                -tablerangesize:   1,600                    -indexrangesize:   4,250

            Highest Stats Table#:  1,235              Highest Stats Index#:   2,570
         Lowest Monitored Table#:   -364           Lowest Monitored Index#:  -1,679
        Highest Monitored Table#:  1,095          Highest Monitored Index#:   2,343

                          Application Tables and Indexes

      Actual Number of App Tables:  1,095       Actual Number of App Indexes:   2,200
               Minimum App Table#:      1                Minimum App  Index#:       8
               Maximum App Table#:  1,095                Maximum App  Index#:   2,343
           Unmonitored App Tables:      0            Unmonitored App Indexes:       0

                Excess Table Range:   140                 Excess Index Range:     228

           Minimal App -basetable:      1             Minimal App -baseindex:       8
      Minimal App -tablerangesize:  1,095        Minimal App -indexrangesize:   2,336

             Suggested -basetable:      1               Suggested -baseindex:       8
        Suggested -tablerangesize:  1,145          Suggested -indexrangesize:   2,385

                          System Tables and Indexes

                     Lowest Table#:   -364                      Lowest Index#:  -1,679
                    Highest Table#:  1,095                     Highest Index#:   2,343

     Suggested Complete -basetable:   -364       Suggested Complete -baseindex:  -1,679
Suggested Complete -tablerangesize:  1,509  Suggested Complete -indexrangesize:   4,072


* "System" tables and indexes include the meta-schema but do not count pseudo tables such as
  VSTs and SQL views as these do not have any CRUD statistics associated with them.

  When using "Complete" settings, "Excess Index Range" may seem high for databases with a
  small number of indexes. This is due to application and system index numbers overlapping.

  Suggested settings can also be found in:  /home/wssdba/pt3/tmp/xus61t2.range.pf
```

- By default OpenEdge only tracks the first 50 tables and indexes
- -tablerangesize and -indexrangesize need to be properly set
- “T” (upper case) will calculate the proper values
- System tables are surprisingly interesting
- OpenEdge does not track LOBs prior to OE12

White Star software

Per Session Table and Index Activity (Global)

Session Table Activity

	Tbl#	Area#	Table Name	RM Chain	#Records	Frag%	Scat	Churn	AvgRow	Create	Read v	Update	Delete	OS Read
>	376	114	im-trans	234887	60469169	0.33%	1	0.00	215	0	222171	0	0	?
	695	102	so-trans-log	479	480305157	0.01%	1	0.00	109	0	32	0	0	?
	338	18	gl-control	60	60	0.00%	1	0.08	111	0	5	0	0	?
	181	20	currency-rate	29	37586	0.00%	1	0.00	53	0	3	0	0	?
	638	124	so-pack-d	400791	59385144	4.62%	1	0.00	208	0	3	0	0	?
	523	18	prod-cp	18	10975013	0.01%	1	0.00	137	0	2	0	0	?
	526	20	prod-cp-price	16	11134130	0.31%	1	0.00	71	0	2	0	0	?

Session Index Activity

	Idx#	Area#	Index Name	Blocks	Util	Lvls	Idx Root	Note	Create	Read v	Split	Delete	BlkDl
>	743	115	im-trans.link-recno	95254	67.90%	3	383		0	222470	0	0	0
	742	115	im-trans.inquiry	111965	68.80%	3	319		0	98	0	0	0
	1573	103	so-trans-log.so-trans	286286	67.10%	4	255		0	34	0	0	0
	678	19	gl-control.gl-control	1	7.40%	1	4607	PU	0	5	0	0	0
	1232	23	prod-line-div.prod-live-div	1	53.30%	1	8959	PU	0	4	0	0	0
	772	149	im-trans-x.link-product	86134	70.60%	3	191		0	4	0	0	0

White Star software

Users of a Table or Index (“8” or “9”)

```
Table Name

  Track top users of:

userTblName: customer

  Enter a valid table name or number.  Use "" or ? to clear table tracking.
```

White Star
software

Users of a Specific Table or Index

Users of "customer" tblNum: 214 areaNum: 174 records: 246796

	Usr#	User Name	Churn	Create	Read v	Update	Delete	CSC	Age	Line#	Program Name
>	482	xusgrami	0.67	0	165007	0	0	00:00:00		14960	im/shdebitg.p
	348	xusmrami	0.64	0	157627	0	0	00:00:01		12573	im/shdebitg.p
	292	xuslviel	0.59	0	144473	0	0	00:00:00		12573	im/shdebitg.p
	371	xecjhalv	0.00	0	559	0	0	00:00:04		77628	so/ordereh.p
	824	xuspguti	0.00	0	425	0	0	00:00:01		3865	so/cinqoro.p
	932	xgtjajpu	0.00	0	228	0	0	00:00:10		7777	ar/custlu.y
	909	xecsayal	0.00	0	132	0	0	00:00:05		7777	ar/custlu.y
	2116		0.00	0	15	0	0	00:00:01		1910	getprafs4.p
	278		0.00	0	12	0	0	00:00:00		2246	ws.rpc.web.WebSessionRpcICMX

Users of "customer.customer" PU idxNum: 454 areaNum: 175 idxRoot: 388

	Usr#	User Name	Idx Root	Create	Read v	Split	Delete	BlkDl	CSC	Age	Line#	Program Name
>	482	xusgrami	388	0	166566	0	0	0	00:00:01		12573	im/shdebitg.p
	348	xusmrami	388	0	158672	0	0	0	00:00:02		12573	im/shdebitg.p
	292	xuslviel	388	0	144919	0	0	0	00:00:01		12573	im/shdebitg.p
	824	xuspguti	388	0	400	0	0	0	00:00:03		3865	so/cinqoro.p
	2116		388	0	32	0	0	0	00:00:01		645	ws.rpc.im.ProdExpRpcICMX
	278		388	0	12	0	0	0	00:00:01		1930	getprafs4.p
	239		388	0	12	0	0	0	00:00:01		1117	ws.rpc.wm.WmPickRpcICMX
	2118		388	0	12	0	0	0	00:00:01		89	getDeliveryInfo.p
	1897		388	0	11	0	0	0	00:00:01		2246	ws.rpc.web.WebSessionRpcICMX

White Star software

Active Transactions, Blocked Sessions

White Star
software

Active Transactions, Blocked Sessions

Blocked Sessions

	Usr	Name	PID	Flags	Durati v	Wait	Resrc Id	Table	Blocker-Usr#:Device:PID	Blocker-StatementCache	WaitList
>	220	xcle1wt3	100051	S4B*	00:00:00	REC XQH	1966077448	wm-pick	217:batch:52762	422	acct/getoparn.p

Active Transactions

	Usr#	Name	PID	Flags	Device	TRX#	BIClstr	Stat	Durati v	Idle	Wait	Resource (dbkey)
>	474	xclnazoc	81701	S4 *	/dev/pts/212	299181305	296347	ACTV	00:14:29	00:01:12	-- 0	7777 ar/custlu.y
	442	xclnriva	50669	S4 *	/dev/pts/176	299694668	296349	ACTV	00:10:05	00:10:03	-- 0	112538 so/orderle2.p
	590	xjmdclar	136616	S4 *	/dev/pts/326	299755582	296352	ACTV	00:08:54	00:08:51	-- 0	28981 im/synlocq.y
	226	xuyctaba	253062	S4 *	/dev/pts/6	299820602	296352	ACTV	00:07:27	00:00:00	-- 0	14026 po/rwserste.p
	709	xpejgalv	236479	S4 *	/dev/pts/447	300008646	296356	ACTV	00:03:02	00:03:04	-- 0	13453 so/orders.p
	367	xmxmanto	8348	S4 *	/dev/pts/107	300040923	296356	ACTV	00:02:12	00:00:00	-- 0	26678 gl/journede.p
	655	xuslcast	218554	S4 *	/dev/pts/391	300054545	296356	ACTV	00:01:25	00:01:01	-- 0	259 wb/message.p
	719	xjmaclar	240009	S4 *	/dev/pts/457	300078446	296356	ACTV	00:01:17	00:00:51	-- 0	92384 im/synlocq.y
	508	xjmkatho	240357	S4 *	/dev/pts/247	300340135	296356	ACTV	00:00:04	00:00:00	-- 0	112557 so/orderle2.p

White Star software

Caveat Regarding _LOCK

- Access is much faster in 11.4+
- But it is still slow if -L is very large!
- And many production databases run with very, very large -L values.
- Embedding code in applications to find out who has a record lock is not always good idea.

How Much Time Did That Really Take?

The Code Profiler

White Star
software

The Code Profiler Is Awesome!

- As a developer you may already be familiar with it from PDSOE
- You can also programmatically embed an ad-hoc profiling capability in your application:

```
profiler:enabled     = yes.
profiler:description = “helpful description”.
profiler:profiling   = yes.
profiler:file-name   = “profiler.prf”.
/* do stuff */
profiler:enabled     = no.
profiler:profiling   = no.
profiler:write-data().
```

- Sample code is in protop-src.zip, lib/zprof*

White Star software

Embedding The Profiler In Your Application

```
Are You Sure?

The Profiler capability is used to track down performance issues
within the ProTop client.  It is very unusual for an end-user to
need to run this for that purpose.

Aside from debugging ProTop this code is also a useful example of
embedding the profiler within an application.  The source can be
found in lib/zprof*.p

It is fine to run this code in order to get a feel for how useful
embedded profiling can be (IMHO it is *VERY* useful).

But be aware that profiling can very quickly create very large
temp files (gigabytes in minutes) so do not run this just for
giggles and do not leave it running unattended.

<Yes> <No>
```

White Star
software

Embedding The Profiler In Your Application

```
Profiler Enabled

The Profiler is now enabled.  Press "y"
when you are ready to view the results.

<OK>
```

White Star
software

Embedding The Profiler In Your Application

Profiler: Top 20 Results

```
Description: ProTop3 Execution Profile [00:01:15]
Session Total Execution Time   00:00:19
Line 0 = initialization, line -1 = cleanup
```

Top 20 Lines: Total Execution Time

Program/Class	Line	Time	Avg Time	Calls	Internal Procedure/Method
dc/dashboard.p	4163	13.531276	1.503475	9	mon-update
dc/dashboard.p	3878	2.705268	0.000000	9000009	mon-update
dc/dashboard.p	0	1.536005	0.170667	9	mon-update
lib/trax.p	0	0.417043	0.046338	9	userMon
dc/dashboard.p	3879	0.342437	0.000000	9000000	mon-update
lib/trax.p	74	0.340062	0.000020	16951	userMon
lib/zippy.p	0	0.237650	0.026406	9	zippy
ssg/sausage00.p	1989	0.151137	0.000002	77724	scanDataSet
ssg/sausage00.p	0	0.148574	0.016508	9	scanDataSet
ssg/sausage00.p	2026	0.127156	0.000002	70885	scanDataSet
dc/uio.p	1277	0.119173	0.000022	5410	age_xstat
lib/xrange.p	0	0.103272	0.103272	1	getRangeData
dc/uio.p	0	0.101518	0.020304	5	mon-update
ssg/sausage00.p	2025	0.094634	0.000001	72253	scanDataSet
ssg/sausage00.p	1976	0.092393	0.000001	83134	scanDataSet
lib/vstlib.p	1393	0.090177	0.003340	27	isBackupRunning
ssg/sausage00.p	2000	0.088533	0.000001	77724	scanDataSet
ssg/sausage00.p	2003	0.082407	0.000001	77724	scanDataSet
ssg/sausage00.p	1984	0.077652	0.000001	77724	scanDataSet
ssg/sausage00.p	1987	0.068471	0.000001	77724	scanDataSet

White Star software

Profiler Caveats

- The Profiler Creates VERY Large Temp-Files!
- You must exit the profiled code cleanly, if an error occurs you will not get any useable data.
- You need the DEBUG-LIST files that match the r-code being profiled.
- Code that contains multiple statements on a single line can hide from the profiler.
- Profiling can sometimes have a noticeable impact on runtime.
- Documentation is “light”.

White Star
software

What’s Going On With Temp-Tables?

Temp Table “VSTs”

White Star
software

Temp Table Statistics

- Temp-tables and ProDataSets are vital components of modern applications
- Programmers have very little insight into how the temp tables in their code are behaving
- Temp Table Statistics were introduced in OE11

White Star
software

Aggregate Temp-Table Info

```
Temp-Table Info

/home/pt3dev/tmp/DBI-9950762889q2d8B

1048576 DBI File Size              84 current temp-tables
   1KB TT DB Block Size             5 archived
1288 TT DB Total Blocks           125 peak
 193 TT DB Empty Blocks           275 tt indexes
   2 TT DB Free Blocks           1669 total current records
   0 TT DB RM Free Blocks      109831 total current bytes

99.53% tt hit ratio

3225 tt rec create
34660 tt rec read
3032 tt rec update
85 tt rec delete
96186 tt rec log rd
453 tt rec os rd
1046 tt rec os wr
5376 tt TRX
64 tt Undos

<OK> <Help>
```

Detailed Temp-Table Info

TT Name	Procedure Name	Bytes	Records	Create	Read	Update	Del	OSRd
tt_tbl	protop.p	5863	184	184	17145	9	3	
tt_tbl.xid-idx	protop.p			185	17801			
tt_idx	protop.p	10650	201	201	416	32	4	
tt_idx.xid-idx	protop.p			202	744			
tt_screenElement	lib/dynscreen.p	34254	408	408	825	165	34	
tt_screenElement.scrFrame	lib/dynscreen.p			418	1409			
tt_screenElement.elNm_frNm_elH	lib/dynscreen.p			418	407			
tt_browseColumnList	lib/dynscreen.p	2701	65	65	989	37	4	
tt_browseColumnList.brwCol	lib/dynscreen.p			65	468			
tt_browseColumnList.brwHdl	lib/dynscreen.p			102	734			

White Star
software

Progress.Database.TempTableInfo

- ArchiveIndexStatistics
- ArchiveTableStatistics
- TempTableCount
- TempTablePeak

- GetTableInfoByPosition()

- GetTableInfoByID()
- GetTableStatHistoryHandle()

- GetIndexInfoByID()
- GetIndexStatHistoryHandle()

- GetVSTHandle()

"Id" is what you need to link things together!

- -ttbaseindex 1
- -ttbasetable 1
- -ttindexrangesize 1000
- -tttablerangesize 1000

White Star software

Enabling TT Data Collection

```
&IF DECIMAL(SUBSTRING(PROVERSION,1,INDEX(PROVERSION,".") + 1)) > 11.0
&THEN

if os-getenv( "TTDEBUG" ) = "yes" then
  do:
    Progress.Database.TempTableInfo:ArchiveTableStatistics = true no-error.
    Progress.Database.TempTableInfo:ArchiveIndexStatistics = true no-error.
  end.

&ENDIF
```

** Cannot set Progress.Database.TempTableInfo:ArchiveTableStatistics (15247)

(means that you forgot to set –ttrangesize etc…)

White Star software

Sample Code

```
/* lib/ttinfo.p
 *
 * show some useful information about this session's temp-tables
 * temp-table info requires OpenEdge 11 or higher
 *
 *      # these define the temp-table stats collection for oe11 clients
 *      # older clients should ignore these parameters (but we comment them out
anyway).
 *
 *      -ttbaseindex 1
 *      -ttbasetable 1
 *      -ttindexrangesize 1000    # 1000 is a guess at the maximum number of TT
indexes used
 *      -tttablerangesize 1000
 *      -tmpbsize 1                    #  32 rows per block
 *      -tmpbsize 4                    # 256 rows per block
 *      -tmpbsize 8                    # 256 rows per block
 *
 * also of interest:  http://knowledgebase.progress.com/articles/Article/P95826
```

User Table Stats

(and Index Statistics Too)

White Star
software

User Table and Index Statistics

- Aggregate Table and Index stats were introduced in Progress v8.3
- That was such a great feature that user level stats were introduced in OE 10.1B!
- Now you can see how much of your database activity is from a given user.
- This is run-time behavior – not static, compile time analysis of index selection; IOW, what **really** happens vs what “should” happen.

White Star
software

Gathering User Table & Index Statistics

```
run lib/usertablestats.p persistent.

for each dictdb.order no-lock:
end.

{lib/userstats.i}

run getUStats (
  output table tt_usrTblInfo by-reference,
  output table tt_usrIdxInfo by-reference
).

for each tt_usrTblInfo by tt_usrTblInfo.tblRd descending:
  display tblName tblRd tblCr tblUp tblDl with 5 down.
end.

for each tt_usrIdxInfo by tt_usrIdxInfo.idxRd descending:
  display idxName idxRd idxCr idxDl with 5 down.
end.
```

White Star software

Top 5 User Tables & Indexes

Top 5 Tables Used by My Session

tblName	tblRd	tblCr	tblUp	tblDl
Order-Line	2,619	0	0	0
Customer	332	0	0	0
Order	207	0	0	0
Item	165	0	0	0
Salesrep	9	0	0	0

Top 5 Indexes Used by My Session

idxName	idxRd	idxCr	idxDl
Order-Line.order-line	2,622	0	0
_Field._Field-Name	2,191	0	0
_Index._File/Index	873	0	0
_File._File-Name	577	0	0

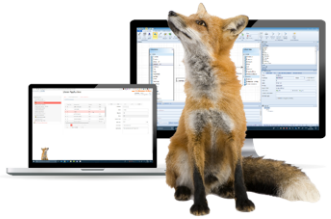
White Star software

More Sample Code

```
/* lib/utblstats.p
 *
 * example test harness for using lib/
usertablestats.p
 *
 * mpro /db/db/s2k -p lib/utblstats.p
 *
 */
```

White Star
software

The SmartComponent Library



Query Information

FOR EACH Customer WHERE Customer.City = "Boston" BY Customer.CustNum INDEXED-REPOSITION

Ok

User Table and Index Statistics

Close

User Table Statistics

Drag a column header here to group by that column.

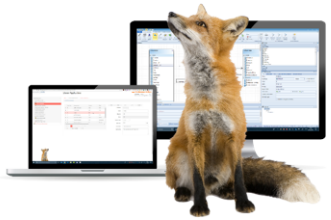
Database Name	Table Name	Reads	Creates	Updates	Deletes
icfdb	_File	2	0	0	0
SmartDB	SmartAttachment	1	0	0	0
SmartDB	SmartTable	1	0	0	0
SmartDB	_File	2	0	0	0
sports2000	Customer	201121	0	0	0
sports2000	Salesrep	80	0	0	0
sports2000	_Field	3	0	0	0
sports2000	_File	2	0	0	0

User Index Statistics

Drag a column header here to group by that column.

Database Name	Index Name	Reads	Creates	Deletes
SmartDB	SmartAttachment.AttachmentRole	80	0	0
SmartDB	SmartAttachment.CommentGUID	80	0	0
SmartDB	SmartTable.DatabaseTable	1	0	0
sports2000	Customer.CustNum	201124	0	0
sports2000	Salesrep.SalesRep	160	0	0

The SmartComponent Library



```
AS-4 ROOT:w:00000007 QRYINFO        Query Handle: 39188
AS-4 ROOT:w:00000007 QRYINFO        Times prepared: 1
AS-4 ROOT:w:00000007 QRYINFO        Time to prepare (ms): 0
AS-4 ROOT:w:00000007 QRYINFO        DB Blocks accessed to prepare:
AS-4 ROOT:w:00000007 QRYINFO         C:/Work/SmartComponents4NET/124_64/DB/Sports2000/sports2000 : 2
AS-4 ROOT:w:00000007 QRYINFO        Times opened: 1
AS-4 ROOT:w:00000007 QRYINFO        Used REPOSITION: N
AS-4 ROOT:w:00000007 QRYINFO        DB Blocks accessed:
AS-4 ROOT:w:00000007 QRYINFO         C:/Work/SmartComponents4NET/124_64/DB/Sports2000/sports2000 : 89700
AS-4 ROOT:w:00000007 QRYINFO        DB Reads:
AS-4 ROOT:w:00000007 QRYINFO         Table: C:/Work/SmartComponents4NET/124_64/DB/Sports2000/sports2000.Customer : 44625
AS-4 ROOT:w:00000007 QRYINFO         Index: Customer.CustNum : 44625
AS-4 ROOT:w:00000007 QRYINFO        C:/Work/SmartComponents4NET/124_64/DB/Sports2000/sports2000.Customer Table:
AS-4 ROOT:w:00000007 QRYINFO         4GL Records: 21
AS-4 ROOT:w:00000007 QRYINFO         Records from server: 21
AS-4 ROOT:w:00000007 QRYINFO          Useful: 21
AS-4 ROOT:w:00000007 QRYINFO          Failed: 0
AS-4 ROOT:w:00000007 QRYINFO         Select By Client: N
AS-4 ROOT:w:00000007 DataAccess     Next-Rowid for Table eCustomer: 0x0000000000010d62
AS-4 ROOT:w:00000007 DataAccess     Last-Batch for Table eCustomer: no
AS-4 ROOT:w:00000007 Activation     ### ServiceInterface:Deactivate()
AS-4 ROOT:w:00000007 Activation     ### Raising ServiceInterface:Deactivated event
AS-4 ROOT:w:00000007 APPL           ##########################################################################################
AS-4 ROOT:w:00000007 APPL           Table Name                                  Record Reads         Updates         Creates         Deletes
AS-4 ROOT:w:00000007 APPL           ------------------------------------------------------------------------------------------------------
AS-4 ROOT:w:00000007 APPL           SmartDB.SmartAttachment                                1               0               0               0
AS-4 ROOT:w:00000007 APPL           SmartDB.SmartSecurityAssignment                       21               0               0               0
AS-4 ROOT:w:00000007 APPL           SmartDB.SmartSecurityRealm                             1               0               0               0
AS-4 ROOT:w:00000007 APPL           SmartDB.SmartTable                                     1               0               0               0
AS-4 ROOT:w:00000007 APPL           SmartDB._Field                                         1               0               0               0
AS-4 ROOT:w:00000007 APPL           sports2000.Customer                               44,626               0               0               0
AS-4 ROOT:w:00000007 APPL           sports2000.Salesrep                                    1               0               0               0
AS-4 ROOT:w:00000007 APPL           sports2000._Field                                     72               0               0               0
AS-4 ROOT:w:00000007 APPL           sports2000._File                                      11               0               0               0
AS-4 ROOT:w:00000007 APPL           sports2000._Index                                      7               0               0               0
AS-4 ROOT:w:00000007 APPL           sports2000._Index-Field                                8               0               0               0
AS-4 ROOT:w:00000007 APPL           ------------------------------------------------------------------------------------------------------
AS-4 ROOT:w:00000007 APPL           Index Name                                    Index Reads                         Creates         Deletes
AS-4 ROOT:w:00000007 APPL           ------------------------------------------------------------------------------------------------------
AS-4 ROOT:w:00000007 APPL           SmartDB.SmartAttachment.AttachmentRole                20                               0               0
AS-4 ROOT:w:00000007 APPL           SmartDB.SmartAttachment.CommentGUID                   20                               0               0
AS-4 ROOT:w:00000007 APPL           SmartDB.SmartSecurityAssignment.Security              25                               0               0
AS-4 ROOT:w:00000007 APPL           SmartDB.SmartSecurityAssignment.Security              16                               0               0
AS-4 ROOT:w:00000007 APPL           SmartDB.SmartSecurityRealm.SecurityRealm               2                               0               0
AS-4 ROOT:w:00000007 APPL           SmartDB.SmartTable.DatabaseTable                       1                               0               0
AS-4 ROOT:w:00000007 APPL           sports2000.Customer.CustNum                       44,625                               0               0
AS-4 ROOT:w:00000007 APPL           sports2000._Field._Field-Position                     62                               0               0
AS-4 ROOT:w:00000007 APPL           sports2000._Index-Field._Index/Number                 15                               0               0
AS-4 ROOT:w:00000007 APPL           sports2000._Index._File/Index                         10                               0               0
AS-4 ROOT:w:00000007 APPL           ##########################################################################################
AS-4 ROOT:w:00000007 ContextDat     [StoreSessionContext] Creating new record in persistent store.
AS-4 ROOT:w:00000007 ServiceMan     Stopping Business Service: Consultingwerk.SmartComponentsDemo.OERA.Sports2000.CustomerBusinessEntity.
AS-4 ROOT:w:00000007 Activation     ### Raising ServiceInterface:AfterDeactivated event
AS-4 ROOT:w:00000007 SmartWebHa     ##################################################################################################################
AS-4 ROOT:w:00000007 SmartWebHa     ### End Web Handler Request: GET /Entities/Customers
AS-4 ROOT:w:00000007 SmartWebHa     ### Request runtime: 167 msecs, Response Content Length: 0 bytes
AS-4 ROOT:w:00000007 SmartWebHa     ##################################################################################################################
```

White Star software

Caution

- BLOB and CLOB field activity is misreported prior to OE12.2!
- It will be recorded as activity on tables that have the same “table Id” as the “LOB Id” (fixed in OE12.2).
- Memory use:

  (-n + -Mn + 1) * tablerangesize * 32

  (-n + -Mn + 1) * indexrangesize * 40

White Star software

More Stuff!!!

- Server info (s)
  - Client/server is becoming more popular, with PASOE containers, etc., so tuning remote clients is more important than ever. Are your client params giving you the best throughput and minimizing round trips? Use this DC to answer those questions quantitatively.
- Ctrl+r reports
  - Useful reports for both DBAs and devs; devs may be interested in the dictionary reports, index overlap, and redundant indexes
- Extensibility
  - Appmon lets you monitor application-specific metrics that are important to your business
- Pause (useful when sampling) so you can manually scrape or screenshot
- Easily e-mail a screenshot of ProTop data (@)
- Sequence viewer (/); definitions and current values
- For SQL developers: when did you last update your query-optimizer stats?
  - ProTop Configuration (c) will tell you
  - ProTop gives you Ctrl-u to create a SQL update stats script

White Star software

Questions?

Thank You!

White Star
software

protop

- Real time monitoring and detailed drill-down
- Historical trending, zoom in or out across years of data at will
- Insightful alerting – the information needed to act on alerts
- Routine “health checks”
- A single pane of glass dashboard

- It’s not just the database!
  - App servers
  - Pro2
  - CODE behaviors and profiling
  - User defined, application specific metrics

White Star
software