Exploring the ONSTAT Utility

by Lester Knutsen

Introduction

Provided with Informix Dynamic Server is a set of powerful command-line utilities that enable the monitoring, tuning, and configuring of the database server. This article focuses on one of these utilities, `ONSTAT`, and presents ways to use this utility for server performance optimization.

ONSTAT

`ONSTAT` is the command-line utility which is most widely used. `ONSTAT` reads OnLine's shared memory structures and provides much useful information about the state of a server. The utility does not place locks on the shared memory structure and uses very little overhead, so it can be used at any time. It is important to keep in mind that the information is current at the time the command is issued, and that data can change after issuing the command.

More options are available for `ONSTAT` than for any other Informix utility. Many options are debugging parameters, which are not well-documented and easily understood by the average database administrator (DBA). `ONSTAT` also provides many useful options for managing the server. This article will focus on the more useful of these options. For a complete list of the syntax and options, refer to Table 1. In Informix Dynamic Server, the `ONSTAT` command has been greatly enhanced with a new set of monitoring and debugging options. These options begin with `-g` and are listed in Table 2.

```
onstat [–abcdfghklmpstuxzBCDFRX][–I][–r seconds][–o file][infile]
```

–a	Print all information
–b	Print buffers
–c	Print configuration file
–d	Print DBspaces and chunks
–f	Print dataskip status
–g	New Monitoring subcommands (default: all). See Figure 2 for all options
–i	Interactive mode
–k	Print locks
–l	Print logging
–m	Print message log
–p	Print profile
–s	Print latches
–t	Print TBLspaces
–u	Print user threads
–x	Print transactions
–z	Zero profile counts
–B	Print all buffers
–C	Print btree cleaner requests
–D	Print DBspaces and detailed chunk stats
–F	Print page flushers
–R	Print LRU queues
–X	Print entire list of sharers and waiters for buffers
–r	Repeat options every n seconds (default: 5)
–o	Put shared memory into specified file (default: onstat.out)
infile	Use infile to obtain shared memory information
–	Displays OnLine mode

Table 1: **ONSTAT** *syntax and options.*

```
onstat –g [options from list below]

   all    Print all MT information
   ath    Print all threads
   wai    Print waiting threads
   act    Print active threads
   rea    Print ready threads
   sle    Print all sleeping threads
   spi    Print spin locks with long spins
   sch    Print VP scheduler statistics
   lmx    Print all locked mutexes
   wmx    Print all mutexes with waiters
   con    Print conditions with waiters
   stk    <tid> Dump the stack of a specified thread
   glo    Print MT global information
   mem    <pool name|session id> Print pool statistics
   seg    Print memory segment statistics
   rbm    Print block map for resident segment
   nbm    Print block map for non-resident segments
   afr    <pool name|session id> Print allocated pool fragments
   ffr    <pool name|session id> Print free pool fragments
   ufr    <pool name|session id> Print pool usage breakdown
   iov    Print disk IO statistics by vp
   iof    Print disk IO statistics by chunk/file
   ioq    Print disk IO statistics by queue
   iog    Print AIO global information
   iob    Print big buffer usage by IO VP class
   ppf    [<partition number>|0]  Print partition profiles
   tpf    [<tid>|0] Print thread profiles
   ntu    Print net user thread profile information
   ntt    Print net user thread access times
   ntm    Print net message information
   ntd    Print net dispatch information
   nss    <session id> Print net shared memory status
   nsc    <client id> Print net shared memory status
   nsd    Print net shared memory data
   sts    Print max and current stack sizes
   dic    Print dictionary cache information
   qst    Print queue statistics
   wst    Print thread wait statistics
   ses    <session id> Print session information
   sql    <session id> Print sql information
   dri    Print data replication information
   pos    Print /INFORMIXDIR/etc/.infos.DBSERVERNAME file
   mgm    Print mgm resource manager information
   ddr    Print DDR log post processing information
```

Table 2: Options for the* `onstat –g` *syntax.

ONSTAT commands are too numerous to cover in one article. Instead, this article will focus on the key ONSTAT commands, which are most useful to a DBA in monitoring a server.

Current Status of the Server: `onstat -`

The `onstat -` command prints out a one-line message indicating the current status of a server. Figure 1 provides a sample output:

```
lester@merlin >onstat -

INFORMIX-OnLine Version 7.23.UC1 -- On-Line --
Up 7 days 11:54:44 -- 10656 Kbytes
```

Figure 1: The `onstat -` command provides current status.

The `onstat -` command displays the version number of Informix Dynamic Server, the mode that the server is in, how long it has been up and running, and how much memory it is using. If the server was down, an error message may result which indicates that "shared memory is not initialized," as in Figure 2.

```
lester@merlin >onstat -

shared memory not initialized for INFORMIXSERVER 'merlindb713'
lester@merlin >
```

Figure 2: Current status when the server is down.

Database Server Profile: `onstat -p`

The `-p` option displays the basic I/O and performance profile of a system. Figure 3 provides a sample output. These statistics are current as of the last time that the server was rebooted, or the time at which the statistics were last reset with the `onstat -z` option.

```
lester@merlin >onstat -p

INFORMIX-OnLine Version 7.23.UC1 -- On-Line -- Up 7 days 12:03:37 --
10656 Kbytes

Profile
dskreads pagreads bufreads %cached dskwrits pagwrits bufwrits %cached
215537   2098656  5208789  95.86   178116   179527   2883605  93.82

isamtot  open     start    read    write    rewrite  delete commit rollbk
6955097  2172     2858     2312158 2305564  129      43     22238  0

ovlock   ovuserthread      ovbuff  usercpu  syscpu   numckpts flushes
0        0                 168     6625.92  722.70   35       4320

bufwaits lokwaits lockreqs deadlks dltouts  ckpwaits compress seqscans
440      0        2331106  0       0        29       5        66

ixda-RA idx-RA   da-RA    RA-pgsused  lchwaits
836      0        16       833         36
```

Figure 3: The `onstat -p` *command provides a server profile.*

Key elements of the `onstat -z` option are listed in the following table:

`onstat -z` Elements	**Description**
Reads %cached	This value is the percentage of reads that use the server's buffers, instead of accessing disk drives (since the records are already in memory). The goal is for 95 percent of reads to be provided through the buffers. If this number is below 95 percent, it may be necessary to increase the `BUFFERS` parameter in the `ONCONFIG` file.
Writes %cached	This value is the percentage of writes that use buffers. The goal is for 85 percent or more of write activity to use the buffers. The one exception is large data loads. The `BUFFERS` parameter in the `ONCONFIG` file will affect this value. Take care when increasing the `BUFFERS` parameter: making the `BUFFERS` too large will take memory away from other processes and may slow down the entire system. When increasing `BUFFERS`, monitor the swapping and paging of your operating system.
ovlock	This value should be zero. Any other number indicates that you have run out of locks since the system was last reset. Increase the `LOCKS` parameter in the `ONCONFIG` file.
ovuserthread	This value should be zero, and is increased each time that a user tries to connect and the number of current users exceeds the maximum number of user threads set in the `ONCONFIG` file. The maximum number of user threads is the third value of the `NETTYPE` parameter in the `ONCONFIG` file.
ovbuff	This value should be zero, and is increased each time that the server tries to acquire more buffers than are set by the `BUFFER` parameter in the `ONCONFIG` file.

bufwaits	This value should be zero, and indicates the number of times that a user thread has waited for a BUFFER.
lokwaits	This value should be zero, and indicates the number of times that a user thread has waited for a LOCK.
deadlks	This value should be zero, and indicates the number of times that a deadlock was detected and prevented.
dltouts	This value should be zero, and indicates the number of times that a distributed deadlock was detected.

Table 3: Key elements of the **onstat -z** *option.*

Display Message Log File: onstat -m

The onstat -m option displays the last lines of the server's message log, refer to Figure 4. This is the message file that contains all messages about the server, and is a key system component to monitor. Figure 4 provides an example. Use this option to view the last 20 messages in the log file.

```
lester@merlin >onstat -m

INFORMIX-OnLine Version 7.23.UC1 -- On-Line -- Up 7 days 12:41:12 --
10656 Kbytes

Message Log File: /u3 / informix7 / online1.log
21:05:15  Checkpoint  Completed:  duration was 8 seconds.
21:05:43  Checkpoint  Completed:  duration was 6 seconds.
21:10:58  Checkpoint  Completed:  duration was 7 seconds.
21:16:06  Checkpoint  Completed:  duration was 7 seconds.
21:21:13  Checkpoint  Completed:  duration was 7 seconds.
21:26:20  Checkpoint  Completed:  duration was 7 seconds.
21:31:28  Checkpoint  Completed:  duration was 7 seconds.
21:36:36  Checkpoint  Completed:  duration was 8 seconds.
21:41:43  Checkpoint  Completed:  duration was 7 seconds.
21:46:51  Checkpoint  Completed:  duration was 8 seconds.
21:52:00  Checkpoint  Completed:  duration was 9 seconds.
21:57:09  Checkpoint  Completed:  duration was 8 seconds.
22:00:42  Logical Log 20 Complete.
22:00:43  Process exited with return code 1: /bin/sh /bin/sh -c
/u3/informix7/lo
g_full.sh 2 23 "Logical Log 20 Complete." "Logical Log 20 Complete."
22:02:17  Checkpoint  Completed:  duration was 8 seconds.
```

Figure 4: Using the **onstat -m** *to display the message log file.*

Note

It may be helpful to keep the DBA's log file always on display in a screen window. To do this, use the UNIX `tail` command with the `-f` option. This command continually reads the last lines of a file as the lines are appended. On my system, I run the following command to continually monitor this log:

```
tail -f $INFORMIXDIR/online.log
```

User Status: `onstat -u`

The `ONSTAT` option used to monitor user activity is `-u`. Figure 5 provides a sample of output from this command. The key field is `sessid` and identifies the user's session ID, which the server uses to track the user internally. This is the number used to kill a specific user's session. For more information, refer to the `onmode -z` command.

```
lester@merlin >onstat -u

INFORMIX-OnLine Version 7.23.UC1 -- On-Line -- Up 7 days 12:52:40 --
10656 Kbytes

Userthreads
address  flags    sessid user      tty  wait    tout locks nreads nwrites
a2d0018  ---P--D  1      informix  -    0       0    0     84     250
a2d0458  ---P--F  0      informix  -    0       0    0     0      177701
a2d0898  ---P--B  8      informix  -    0       0    0     0      0
a2d1558  ---P--D  12     informix  -    0       0    0     0      0
a2d2218  Y--P---  264    lester    4    a35f190 0    1     35063  14836
   5 active, 128 total, 17 maximum concurrent
```

Figure 5: The `onstat -u` *option provides user status.*

The "flags" column provides an indication of user activity. Following are the critical flags, based on their position within the flag field:

Flags in position 1	Description
B	Waiting on a buffer
C	Waiting on a checkpoint
G	Waiting on a logical log buffer write
L	Waiting on a lock
S	Waiting on a mutex
T	Waiting on a transaction
Y	Waiting on a condition
X	Waiting on a transaction rollback

Flags in position 2	Description
*	Transaction active during I/O error

Flags in position 3	Description
A	Dbspace backup thread
B	Begin work
P	Prepared for commit work
X	TP/XA prepared for commit work
C	Committing work
R	Rolling back work
H	Heuristically rolling back work

Flags in position 4	Description
P	Primary thread for a session

Flags in position 5	Description
R	Reading call
X	Transaction is committing

Flags in position 6	Description
None	

Flags in position 7	Description
B	B+Tree cleaner thread
C	Cleanup of terminated user
D	Daemon thread
F	Page flusher thread
M	ON-Monitor user thread

Table 4: Flags and their position within the flag field.

Logical Logs Status: `onstat -l`

The `-l` option of `ONSTAT` displays the current status of the logical logs. Figure 6 provides a sample display. One problem with this display is that it does not indicate which logs are ready for reuse. In versions 5.x of INFORMIX-OnLine, as soon as a log was backed up and had no open transactions, it was marked as free with an "F" in the flags column. In versions 7.x of Informix Dynamic Server, logs are not marked as free until just before their reuse. One way to use `ONSTAT` to indicate which logs can be reused is to use `onstat -l` with `onstat -x` to display all active sessions.

```
Physical Logging
Buffer  bufused   bufsize  numpages  numwrits  pages/io
P-1     0         16       236       60        3.93
        phybegin  physize  phypos    phyused   %used
        10003f    1000     967       0         0.00

Logical Logging
Buffer  bufused  bufsize  numrecs  numpages  numwrits recs/pages pages/io
L-3     0        16       90303    1522      275      59.3       5.5

address  number  flags     uniqid  begin     size   used   %used
a1ee3e4  1       U-B----   13      100427    500    500    100.00
a1ee400  2       U-B----   14      10061b    500    500    100.00
a1ee41c  3       U-B----   15      10080f    500    500    100.00
a1ee438  4       U-B----   16      100a03    500    500    100.00
a1ee454  5       U-B----   17      100bf7    500    432     84.40
a1ee470  6       U-B----   18      100deb    500    500    100.00
a1ee48c  7       U-B----   19      100fdf    500    500    100.00
a1ee4a8  8       U-B----   20      1011d3    500    500    100.00
a1ee4c4  9       U---C-L   21      1013c7    500     23      4.60
a1ee4e0  10      U-B----   10      1015bb    500    500    100.00
a1ee4fc  11      U-B----   11      1017af    500    500    100.00
a1ee518  12      U-B----   12      1019a3    500    500    100.00
```

Figure 6: The `onstat -l` *command indicates the logical logs' status.*

The flags column provides status information about each log, as follows:

Flag	Description
A	Newly added: must run an archive before use
B	Backed up to tape or "/dev/null"
C	Current logical log file
F	Free and available for use. You will rarely see this flag, as logs are not marked as free until they are needed
L	Last checkpoint is in this logical log
U	Used logical log: it may be free if it is backed up and contains no active transactions

Table 5: Flags for **`onstat -l`.**

Display Transactions: `onstat -x`

The `onstat -x` option displays all current transactions. The most useful column is "log begin." This option can provide information about the logical log in which a transaction is started, and the option can be used with the `onstat -l` command to determine which logs are free and for reuse. To find out which logical log has the earliest active transaction, locate the earliest logical log number in the "log begin" column. Any logical logs that are backed up before the log with the earliest transaction will be automatically reused by the server.

```
lester@merlin >onstat -x

INFORMIX-OnLine Version 7.23.UC1 -- On-Line -- Up 01:21:38 -- 10656 Kbytes

Transactions
address  flags  userthread  locks  log begin isolation  retrys coordinator
a2f4018  A----  a2d0018     0      0         COMMIT     0
a2f413c  A----  a2d0458     0      0         COMMIT     0
a2f4260  A----  a2d0898     0      0         COMMIT     0
a2f4384  A----  a2d1118     0      0         NOTRANS    0
a2f44a8  A----  a2d1558     0      0         COMMIT     0
a2f45cc  A-B--  a2d1118     2      21        NOTRANS    0
  6 active, 128 total, 7 maximum concurrent
```

Figure 7: The **`onstat -x`** *option provides a status of transactions.*

Display Locks: `onstat -k`

The `onstat -k` option will display all active locks. However, be aware that this display can be lengthy. If a large number of LOCKS are defined in the ONCONFIG file and there are many users, it is possible that thousands of rows will be displayed by this command. Figure 8 provides an example of the display.

```
lester@merlin >onstat -k

INFORMIX-OnLine Version 7.23.UC1 -- On-Line -- Up 7 days 12:53:31 --
10656 Kbytes

Locks
address  wtlist  owner    1klist  type   tblsnum rowid key#/bsiz
a103e74  0       a2d2218  0       HDR+S  100002  20a      0
  1 active, 20000 total, 16384 hash buckets
```

Figure 8: The **`onstat -k`** *option displays all locks.*

Who Owns a Lock?

The "owner" column lists the address in shared memory of the user who owns a lock. Use this column with the `onstat -u` option to view all users, and compare this output with the "address" column to identify the `username` of owners.

What Table is Locked?

The "tblsnum" column identifies the table that is being locked. Compare this output with that of the following SQL statement to convert a table's `partnum` to hex. The output of this statement will identify which table is locked.

```
select tabname, hex(partnum) tblsnum from systables where tabid > 99;
```

This SQL statement will also provide a list of tables and their associated `tblsnum` to identify which table has a lock placed on it. Figure 9 provides an example for identifying which table is locked.

```
1. Find a list of tblsnum

dbaccess database - <<EOF
     select tabname, hex(partnum) tblsnum
     from systables where tabid > 99;
EOF

database selected

tabname      tblsnum
genjournal   0x0010009E
gjsum        0x0010009F

2. Find what is locked
onstat -k

INFORMIX-OnLine Version 7.23.UC1 -- On-Line -- Up 01:47:38 --
10656 Kbytes

Locks
address  wtlist  owner    lklist  type   tblsnum rowid key#/bsiz
a103e44  0       a2d1118  a103de4 HDR+X  10009f  0        0
  3 active, 20000 total, 16384 hash buckets

3. Compare tblsnum from step 1 and step 2. This identifies the
table gjsum as the one that is locked.
```

Figure 9: Output to determine what table is locked.

The `tblsnum 100002` indicates a database lock. Each user who opens a database will place a shared lock on the database.

Types of Locks

Following is a list of the types of locks available and information regarding the identification of locks:

Lock Type	Description
Database	Lock is placed on `tablespace 1000002`
Table	Lock is placed on actual `tablespace` with `rowid` of 0
Page	Lock is placed on `tablespace` with `rowid` ending in 00
Row	Lock is placed on `tablespace` with actual `rowid` (not 00)
Byte	Lock is placed on `tablespace/page` with size of bytes
Key	Lock is placed on `tablespace` hex `rowid` (starting with f)

Table 6: Locks for **`onstat -k`.**

Lock-Type Flags

Following is a list of the lock flags in the "flags" column of `onstat -k`:

Lock Flag	Description
HDR	Header
B	Bytes lock
S	Shared lock
X	Exclusive
I	Intent
U	Update
IX	Intent-exclusive
IS	Intent-shared
SIX	Shared, intent-exclusive

Table 7: Lock flags for **`onstat -k`.**

Dbspaces and Chunks Status: `onstat -d`

The `onstat -d` command provides important information about the layout of dbspaces and disk chunks, and the status of each chunk and dbspace. Print the output of this command and save it. If a restore must be performed, this output will be necessary. The output identifies each dbspace and chunk needed to rebuild the system. Figure 10 contains an example output.

```
lester@merlin >onstat -d

INFORMIX-OnLine Version 7.23.UC1 -- On-Line -- Up 7 days 12:54:44 --
10656 Kbytes

Dbspaces
address  number  flags    fchunk  nchunks   flags owner    name
a2ce100  1       1        1       1         N     informix rootdbs
a2ce508  2       1        2       1         N     informix dbspace1
a2ce578  3       1        3       1         N     informix dbspace2
a2ce5e8  4       1        4       1         N     informix dbspace3
  4 active, 2047 maximum

Chunks
address  chk/dbs  offset  size    free   bpages flags  pathname
a2ce170  1   1    0       250000  62047          PO-    /u3/dev/rootdbs1
a2ce280  2   2    0       10000   9587           PO-    /u3/dev/dbspace1
a2ce358  3   3    0       10000   9947           PO-    /u3/dev/dbspace2
a2ce430  4   4    0       10000   9947           PO-    /u3/dev/dbspace3
  4 active, 2047 maximum

```

Figure 10: The `onstat -d` *output provides dbspaces and chunk status.*

The output of the `onstat -d` command also indicates how much free space each chunk has, and the status of each chunk.

The "flags" for Dbspaces are:

Position 1

- M - Mirrored Dbspace
- N - Not Mirrored Dbspace

Position 2

- X - Newly mirrored
- P - Physical recovery underway
- L - Logical recovery underway
- R - Recovery underway

Position 3

- B - Blobspace

The "flags" for Chunks are:

Position 1

- P - Primary
- M - Mirror

Position 2

- O - On-line
- D - Down
- X - Newly mirrored
- I - Inconsistent

Position 3

- B - Blobspace
- - - Dbspace
- T - Temporary Dbspace

***Table 8: Flags for* `onstat -d`.**

Dbspaces and Chunks I/O: `onstat -D`

The `onstat -D` option displays I/O by chunk. This option is very helpful in performance tuning. The goal is to spread reads and writes evenly across all chunks. Figure 11 provides an example wherein one chunk is utilized for all I/O, and all other chunks are inactive. The I/O is not spread out among chunks, which is not an effective use of disk space.

```
lester@merlin >onstat -D

INFORMIX-OnLine Version 7.23.UC1 -- On-Line -- Up 7 days 12:55:09 --
10656 Kbytes

Dbspaces
address  number  flags    fchunk  nchunks   flags owner    name
a2ce100  1       1        1       1         N     informix rootdbs
a2ce508  2       1        2       1         N     informix dbspace1
a2ce578  3       1        3       1         N     informix dbspace2
a2ce5e8  4       1        4       1         N     informix dbspace3
  4 active, 2047 maximum

Chunks
address  chk/dbs  offset  page Rd   page Wr   pathname
a2ce170  1   1    0       36563     179558    /u3/dev/rootdbs1
a2ce280  2   2    0       3         0         /u3/dev/dbspace1
a2ce358  3   3    0       2         0         /u3/dev/dbspace2
a2ce430  4   4    0       2         0         /u3/dev/dbspace3
  4 active, 2047 maximum

```

Figure 11: The `onstat -D` *option displays the I/O utilization of dbspaces and chunks.*

Page Write Status: `onstat -F`

There are three ways that the server writes pages from shared memory buffers to disk. Foreground writes occur when the server needs a buffer and must interrupt processing to flush buffers to disk in order to free a buffer. These are the least desirable types of writes. Background writes (`LRU Writes`) occur when a set percentage of the buffers are dirty. This is controlled by the `LRU` parameters in the `ONCONFIG` file. These writes do not interrupt user processing and are best for interactive systems. Chunk writes occur at checkpoints, and all dirty buffer pages are written to disk. With more dirty pages, the checkpoint takes longer to complete. Checkpoint writes are sorted and optimized; however, the longer the checkpoint, the longer it will block user activity. Checkpoint writes are best for batch systems. The `ONSTAT` option to monitor this activity is `-F`. The goal should be to see zero foreground writes (`Fg Writes`). Figure 12 provides an example.

```
lester@merlin >onstat -F

INFORMIX-OnLine Version 7.23.UC1 -- On-Line --
Up 7 days 12:55:32 -- 10656 Kbytes

Fg Writes  LRU Writes   Chunk Writes
168        172280       5277

address  flusher  state  data
a2d0458  0        I      0      = 0X0
      states: Exit Idle Chunk Lru
```

Figure 12: The `onstat -D` *option provides a status of page writes.*

New Monitoring and Debugging Commands (Informix Dynamic Server, versions 7.x): `onstat -g`

The `-g` commands are a new subset of commands available beginning with versions 7.x of Informix Dynamic Server. Table 2 displays a list of the `-g` commands. This section will discuss some of these `-g` commands.

List all Threads: `onstat -g ath`

The `onstat -g ath` option lists all active threads. Figure 13 provides an example.

```
INFORMIX-OnLine Version 7.23.UC1 -- On-Line -- Up 7 days 12:56:09 -- 
10656 Kbytes
 
Threads
tid  tcb      rstcb     prty  status              vp-class  name
2    a336b70  0         2     sleeping(Forever)   3lio      lio vp 0
3    a336dd0  0         2     sleeping(Forever)   4pio      pio vp 0
4    a337088  0         2     sleeping(Forever)   5aio      aio vp 0
5    a337340  0         2     sleeping(Forever)   6msc      msc vp 0
6    a337af8  0         2     sleeping(Forever)   7aio      aio vp 1
7    a337e00  a2d0018   4     sleeping(secs: 1)   1cpu      main_loop()
8    a34ab48  0         2     running             1cpu      sm_poll
9    a34b770  0         2     running             8tli      tlitcppoll
10   a34bce0  0         2     sleeping(Forever)   1cpu      sm_listen
11   a3c4a28  0         2     sleeping(secs: 2)   1cpu      sm_discon
12   a3c4e58  0         3     sleeping(Forever)   1cpu      tlitcplst
13   a3d0680  a2d0458   2     sleeping(Forever)   1cpu      flush_sub(0)
14   a3d0e40  a2d0898   2     sleeping(secs: 8)   1cpu      btclean
30   a35ea58  a2d1558   4     sleeping(secs: 1)   1cpu      onmode_mon
283  a39ef38  a2d2218   2     cond wait (sm_read) 1cpu      sqlexec
```

Figure 13: The **`onstat -g ath`** *command lists all active threads.*

List Virtual Processor Status: onstat –g sch

The `onstat –g sch` option provides a means to identify which `oninit` UNIX process corresponds to which server virtual processor (VP). When performing a `ps –ef` command in UNIX, it is possible to see many `oninit` processes running. Each one performs a specific task for the database server. Use the UNIX "pid" column from `ps –ef` to correlate a process to the "pid" column from `onstat –g sch`. Figure 14 provides an example output of this command.

```
lester@merlin >onstat -g sch

INFORMIX-OnLine Version 7.23.UC1 -- On-Line --
Up 7 days 12:56:46 -- 10656 Kbytes

VP Scheduler Statistics:

vp  pid  class    semops  busy waits  spins/wait
1   230  cpu          21           0           0
2   231  adm           0           0           0
3   232  lio         277           0           0
4   233  pio          62           0           0
5   234  aio      144794           0           0
6   235  msc         756           0           0
7   236  aio       64028           0           0
8   237  tli           3           0           0
```

Figure 14: The **`onstat –g sch`** *command provides the status of virtual processors.*

List SQL Statement Types: `onstat –g sql`

This is the most interesting of the new options. This option can be used to drill down and see the actual SQL statement that a user is executing. The example in Figure 15 provides a summary of all the SQL statements which are running. By using the `session id`, it is possible to see the details and actual SQL statements being run. Figure 16 provides an example of this detail.

```
lester@merlin >onstat -g sql

INFORMIX-OnLine Version 7.23.UC1 -- On-Line --
Up 7 days 12:52:02 -- 10656 Kbytes

Sess  SQL        Current    Iso   Lock       SQL   ISAM  F.E.
ID    Stmt type  Database   Lvl   Mode       ERR   ERR   Vers
264   INSERT     ffsdw      NL    Not Wait   -264  0     7.23
```

Figure 15: The **`onstat –g sql`** *command lists all SQL statements.*

List SQL Statement for a Specific User: `onstat –g sql sid`

```
lester@merlin >onstat -g sql 264

INFORMIX-OnLine Version 7.23.UC1 -- On-Line --
Up 7 days 12:51:10 -- 10656 Kbytes

VP Scheduler Statistics:

Sess  SQL        Current    Iso   Lock       SQL   ISAM  F.E.
ID    Stmt type  Database   Lvl   Mode       ERR   ERR   Vers
264   INSERT     ffsdw      NL    Not Wait   -264  0     7.23

Current SQL statement :
  insert into gjsum select exp_org, exp_prog, bud_obj_code, job_num,
    sum (exp_amount) from genjournal group by 1, 2, 3, 4

Last parsed SQL statement :
  insert into gjsum select exp_org, exp_prog, bud_obj_code, job_num,
    sum (exp_amount) from genjournal group by 1, 2, 3, 4
```

Figure 16: The **`onstat –g sid`** *command provides an SQL statement for a specific user.*

This option can be very useful in a couple of cases, such as when access to the SQL code is not available and it is necessary to optimize the database tables and indexes. By running this command repeatedly, it is possible to see the SQL statements that are being processed. Then, by collecting and examining the SQL, you can determine where to add indexes to improve the overall performance of the system.

A second use for this option is in debugging program transactions. I used this option to help a programer debug a program by running `onstat -g sql sid` while the program was operating. Error conditions and SQL errors that the programmer was not catching in the program were available thusly.

List Users Sessions: onstat –g ses

The `onstat -g ses` option provides additional information about users' sessions, including how much memory each session uses. Figure 17 provides an example. This option can also be used to display detailed information about a session.

```
lester@merlin >onstat -g ses

INFORMIX-OnLine Version 7.23.UC1 -- On-Line --
Up 7 days 12:57:48 -- 10656 Kbytes

session                                  #RAM     total    used
id   user       tty  pid   hostname     threads  memory   memory
265  informix   -    0     -            0        8192     4680
264  lester     4    4249  merlin       1        106496   97840
10   informix   -    0     -            0        8192     4680
7    informix   -    0     -            0        16384    13144
6    informix   -    0     -            0        8192     4680
4    informix   -    0     -            0        16384    13144
3    informix   -    0     -            0        8192     4680
2    informix   -    0     -            0        8192     4680
```

Figure 17: The `onstat –g ses` *command provides a list of users' sessions.*

Repeat ONSTAT Commands: `-r`

To continually repeat an `ONSTAT` command, use the `-r # of seconds` option. This option is very useful when you need to monitor a situation. The following example displays the status of the logical logs every ten seconds.

```
onstat -l -r 10
```

Clear ONSTAT Shared Memory Statistics: `onstat -z`

The server statistics are reset each time that the server is restarted. To reset all statistics while the server is running (without shutting it down), use the following command:

```
onstat -z
```

The above command will clear all statistics for the `ONSTAT` and the SMI tables.

Conclusion

The `ONSTAT` utility provides a powerful toolkit for the DBA to care for and monitor a database server. It is hoped that this article provides a starting point for the use of the `ONSTAT` utility.

About the Author

Lester Knutsen is a database consultant and trainer, and is president of Advanced DataTools Corporation. Knutsen is also president of the Washington Area Informix User Group and was one of the founding board members of the International Informix Users Group.

About Advanced DataTools

Advanced DataTools can be contacted via phone at 703 256 0267; via the Web at www.advancedatatools.com; or via email at lester@advancedatatools.com. TN