Use the **onstat -g ses** command to display information about the session.

By default, only the DBSA can view onstat -g ses information. However, when the UNSECURE\_ONSTAT configuration parameter is set to 1, all users can view this information.

You can specify one of the following invocations.

onstat -g ses

Displays a one-line summary for each session

onstat -g ses *session\_id*

Displays information for a specific session

**Example output for all sessions**

Figure 1. onstat -g ses command output

session #RSAM total used dynamic

id user tty pid hostname threads memory memory explain

24 informix - 0 - 0 12288 7936 off

23 informix - 17602 carson 1 57344 48968 off

3 informix - 0 - 0 12288 9168 off

2 informix - 0 - 0 12288 7936 off

Last 20 Sessions Terminated

Ses ID Username Hostname PID Time Reason

46 user\_1 host\_1 21220 01/19/2015.15:20 session limit txn time (60s)

43 user\_1 host\_1 21340 01/19/2015.15:14 session limit memory (5124 KB)

61 user\_1 host\_1 21404 01/19/2015.15:04 session limit logspace (10242 KB)

64 user\_1 host\_1 21458 01/19/2015.15:02 session limit txn time (39548 KB)

**Example output for a specific session**

Figure 2. onstat -g ses *session\_id* command output

session effective #RSAM total used dynamic

id user user tty pid hostname threads memory memory explain

53 informix - 36 18638 apollo11 1 73728 63048 off

Program :

/usr/informix/bin/dbaccess

tid name rstcb flags curstk status

77 sqlexec 4636ba20 Y--P--- 4240 cond wait sm\_read -

Memory pools count 1

name class addr totalsize freesize #allocfrag #freefrag

53 V 4841d040 73728 10680 84 6

name free used name free used

overhead 0 3288 scb 0 144

opentable 0 2904 filetable 0 592

log 0 16536 temprec 0 2208

gentcb 0 1656 ostcb 0 2920

sqscb 0 21296 sql 0 72

hashfiletab 0 552 osenv 0 2848

sqtcb 0 7640 fragman 0 392

sqscb info

scb sqscb optofc pdqpriority optcompind directives

481b70a0 483e2028 0 0 0 1

Sess SQL Current Iso Lock SQL ISAM F.E.

Id Stmt type Database Lvl Mode ERR ERR Vers Explain

53 - sysmaster CR Not Wait 0 0 9.24 Off

Last parsed SQL statement :

Database 'sysmaster@lx1'

Xadatasources participated in this session :

Xadatasource name RMID Active

xabasicdb@atmol10:sitaramv.xads\_t3\_i1 6 YES

xabasicdb@atmol10:sitaramv.xads\_t2\_i1 4 YES

xabasicdb@atmol10:sitaramv.xads\_t1\_i3 3 YES

xabasicdb@atmol10:sitaramv.xads\_t1\_i2 2 YES

xabasicdb@atmol10:sitaramv.xads\_t1\_i1 1 YES

xabasicdb@atmol10:sitaramv.xads\_t2\_i2 5 NO

DRDA client info

Userid:

Wrkstnname: nemea

Applname: db2jcc\_application

Acctng: JCC03510nemea

Programid:

Autocommit:

Packagepath:

Session Limits

Limit Current

Locks 10000 1

Memory(KB) 5120 72

Temp Space(KB) 30720 0

Log Space(KB) 10240 0

Txn Time(s) 120 0

How to kill a session?

First identify a problematic session by using:  
onstat -g ses  
onstat -g sql  
onstat -u  
  
make not of session id and use onmode to kill the session:  
onmode -z <session\_id>  
  
This is similar to Unix kill -9 <pid> interrupt signalling. Onmode takes care about the trasactions in progress condition too, if so it will judge the stage of the transaction, and tries to commit, if possible. But in most of the cases it rolls back the transactions. Such cases it may take a longer period to terminate a session.

How to collect information for all sessions connected to an Informix Dynamic Server

**Answer**

**STEPS**  
  
1. Create an "awk" file

**Example**

vi script.awk

2. Put the following code inside the "awk" file

**Example**

BEGIN {system(":> onstat\_g\_ses\_0")}  
{  
if ($NF > 0) {  
my\_string= "onstat -g ses " $1 "  >> onstat\_g\_ses\_0; echo \"----------------\" >> onstat\_g\_ses\_0"  
system(sprintf(my\_string))  
}  
}

**Note:** This script directs the session information to a file called onstat\_g\_ses\_0. This will be created in the current working directory.  
  
3. Run this script using the command

**Example**

$ onstat -g ses|tail +6|awk -f script.awk

where script.awk is the name of the "awk" file created in the previous step. To improve usability, this command could be written as a shell script (with execute permissions) and located in a directory referenced by your PATH environment variable.  
  
Display the contents of the output file to view the details of all active database sessions.  
  
  
**Example**  
Sample output from awk script, for each session you will see something like this:

IBM Informix Dynamic Server Version 7.31.UD9     -- On-Line -- Up 12 days 15:13:05 -- 53048 Kbytes  
  
session                #RSAM    total    used        
id user     tty   pid  hostname threads  memory     memory      
32informix 14  19256  xxxx     1        49152      43872       
  
tid name     rstcb    flags    curstk   status  
60  sqlexec  46395c48 Y--P---  784      cond wait(netnorm)  
  
Memory pools    count 1  
name    class addr     totalsize freesize #allocfrag #freefrag   
32      V     46b30018 49152     5280     278        4           
  
name          free   used   name     free       used        
overhead       0     120    scb        0        96          
opentable      0     1960   filetable  0       544         
blobio         0     5080   log        0      2152        
temprec        0     1608   blob       0       272         
keys           0     96     ralloc     0      7168        
gentcb         0     8544   ostcb      0      2608        
sqscb          0     8056   rdahead    0       160         
hashfiletab    0     280    osenv      0      1248        
buft\_buffer    0     2136   sqtcb      0      1408        
fragman        0     336              
  
Sess  SQL        Current            Iso Lock       SQL  ISAM F.E.  
Id    Stmt type  Database           Lvl Mode       ERR  ERR  Vers  
32    SELECT     db\_with\_log        CR  Not Wait   0    0    7.31  
  
Current statement name : slctcur  
  
Current SQL statement :  
  select \* from catalog  
  
Last parsed SQL statement :  
  select \* from catalog

SQL to identify the users involved in sessions with lock contention

**Troubleshooting**

**Problem**

This example provides sample SQL that can be used to diagnose users involved in lock contention issues.

**Resolving The Problem**

**Q.** How can I identify the users causing lock contention problems?  
  
**A.** In a multi-user Informix® Dynamic Server (IDS) environment where users have their isolation set higher than dirty read, and/or multiple users are performing update activity (i.e. insert, update or delete actions, rather than read-only), multiple users can all be attempting to place mutually exclusive locks on the same record.

You may want to identify the tables/records under contention and reconfigure or change code (possibly using the SET LOCK MODE TO WAIT statement).   
  
Tracing who has which locks and by using onstat involves joining entries from onstat -k, -u and -g sql. As locks are often held for very short periods of time, the evidence can disappear before all the necessary command can be run. The following SQL statements, run against the sysmaster datebase "tables" do all the joining and filtering for you.

This SQL returns information on locks and the users involved:  
  
select t.username waituser, t.sid waitsess, s.username hasuser,  
s.sid hassess, l.type locktype, l.dbsname database,  
l.tabname table, hex(l.rowidlk) rowid  
from sysmaster:syslocks l, sysmaster:syssessions s, sysmaster:syssessions t  
where s.sid = l.owner  
and l.waiter = t.sid ;  
  
  
**Note:** The commented out clause "dbsname <> 'sysmaster'", if un-commented, will avoid returning the shared lock every user places when they connect to a database, and the locks that this monitoring SQL places when running.  
  
The output looks like this:

user   informix  
sessn  168  
type   S  
dbase  sysmaster  
table  sysdatabases  
rowid  0x00000205  
  
user   informix  
sessn  167  
type   S  
dbase  sysmaster  
table  sysdatabases  
rowid  0x00000205  
  
user   informix  
sessn  173  
type   S  
dbase  sysmaster  
table  sysdatabases  
rowid  0x00000201  
  
user   informix  
sessn  167  
type   X  
dbase  stores9  
table  state  
rowid  0x00000000  
  
user   informix  
sessn  173  
type   S  
dbase  sysmaster  
table  sysdatabases  
rowid  0x00000205

A variation on the SQL is this:  
  
select trim(s.username)||":"||s.sid||" has "||trim(l.type)||  
" lock on "||trim(l.dbsname)||":"||trim(l.tabname)||"-"||hex(l.rowidlk) L  
from sysmaster:syslocks l, sysmaster:syssessions s  
where s.sid = l.owner  
-- and dbsname <> 'sysmaster'  
order by 1 ;  
  
**Note**: The select portion of the query must be entered all on one line, not split over several as it appears here. This query returns the same data as above, but with the columns wrapped with text onto one line per session and lock, like this:  
  
l  informix:167 has S lock on sysmaster:sysdatabases-0x00000205  
l  informix:167 has X lock on stores9:state-0x00000000  
l  informix:168 has S lock on sysmaster:sysdatabases-0x00000205  
l  informix:173 has S lock on sysmaster:sysdatabases-0x00000201  
l  informix:173 has S lock on sysmaster:sysdatabases-0x00000205