```
create or replace function system.jps$_to_dec
( p_str in varchar2,
  p_from_base in number default 16 ) return number
is
        l_num   number default 0;
        l_hex   varchar2(16) default '0123456789ABCDEF';
begin
        if ( p_str is null or p_from_base is null )
        then
                return null;
        end if;
        for i in 1 .. length(p_str) loop
                l_num := l_num * p_from_base + instr(l_hex,upper(substr(p_str,i,1)))-1;
        end loop;
        return l_num;
end jps$_to_dec;
/


drop table system.jps$_systab;
create table system.jps$_systab (
  tname varchar2(30),
  indx number primary key,
  baseaddr varchar2(8),
  baseaddr_dec number,
  colcount number,
  rowsize number,
  rowcount number
);

drop table system.jps$_messages;
create table system.jps$_messages (
  tname varchar2(30),
  message varchar2(100)
);

insert into system.jps$_systab(tname, indx, rowsize, colcount)
  select kqftanam, indx, kqftarsz, kqftacoc from x$kqfta;
commit;

declare
  cursor stab is select indx,tname,rowsize from system.jps$_systab;
  baddr varchar2(8);
  baddr_dec number;
  baddr2_dec number;
  baddr_min number;
  baddr_max number;
  rcount number;
  errmsg varchar2(100);
begin
  select count(*) into rcount from x$ksmmem;
  select rowsize into baddr_dec from system.jps$_systab where tname='X$KSMMEM';
  select system.jps$_to_dec(addr) into baddr_min from x$ksmmem where rownum<2;
  baddr_max:=baddr_min+(rcount*baddr_dec);
  for srec in stab loop
    begin
      execute immediate 'select addr from '||srec.tname||' where rownum<=1' into baddr;
      baddr_dec:=system.jps$_to_dec(baddr);
      if baddr_dec between baddr_min and baddr_max then
        execute immediate 'select count(*) from '||srec.tname into rcount;
        update system.jps$_systab set baseaddr=baddr,baseaddr_dec=baddr_dec,
          rowcount=rcount where indx=srec.indx;
        if rcount>1 then
          execute immediate 'select sum(system.jps$_to_dec(addr)) from '||srec.tname||
            ' where rownum<=2' into baddr2_dec;
          baddr2_dec:=baddr2_dec-(baddr_dec*2);
          if srec.rowsize<>baddr2_dec then
            if baddr2_dec>0 then
              update system.jps$_systab set rowsize=baddr2_dec where indx=srec.indx;
              insert into system.jps$_messages(tname,message)
                values(srec.tname,'WARNING: kqftarsz='||srec.rowsize||
                ' but detected '||baddr2_dec||'.  UPDATED.');
            else
              delete from system.jps$_systab where indx=srec.indx;
              insert into system.jps$_messages(tname,message)
                values(srec.tname,'WARNING: kqftarsz='||srec.rowsize||
                ' but detected '||baddr2_dec||'.  DELETED.');
            end if;
          end if;
        end if;
      else
        insert into system.jps$_messages(tname,message)
          values(srec.tname,'baseaddr ('||baddr||') out of sga range.');
        delete from system.jps$_systab where indx=srec.indx;
      end if;
    exception
      when others then
        errmsg:=substr(sqlerrm,1,100);
        insert into system.jps$_messages(tname,message)
          values(srec.tname,errmsg);
        delete from system.jps$_systab where indx=srec.indx;
    end;
  end loop;
  commit;
end;
/

drop table system.jps$_output;
create table system.jps$_output as
select rownum i,tabdef,coldef||
  DECODE(lead(cindx,1,-1) over (partition by tindx order by kqfcooff,cindx),-1,'',',') coldef
from (
select DECODE(rank() over (order by t.indx),1,'   ','}},')||
    '{"'||tname||'",(void*)0x'||baseaddr||','||
    rowsize||','||rowcount||','||colcount||',{' tabdef,
  '{"'||kqfconam||'",'||kqfcodty||','||kqfcosiz||','||kqfcooff||',"'||
    DECODE(kqfcodty,
      1,'%-'||DECODE(sign(length(kqfconam)-kqfcosiz),1,length(kqfconam),kqfcosiz)||'s ',
      2,DECODE(kqfcosiz,0,'',
          '%'||DECODE(sign(length(kqfconam)-10),1,length(kqfconam),10)||
          DECODE(kqfcosiz,8,'llu ',4,'lu ',2,'hu ','hhu ')
        ),
      23,DECODE(kqfcosiz,4,'%-8lX ',''),
      ''
    )||'","'||
    DECODE(kqfcodty,
      1,'%-'||DECODE(sign(length(kqfconam)-kqfcosiz),1,length(kqfconam),kqfcosiz)||'s ',
      2,DECODE(kqfcosiz,0,'',
          '%'||DECODE(sign(length(kqfconam)-10),1,length(kqfconam),10)||'s '
        ),
      23,DECODE(kqfcosiz,4,'%-8s ',''),
      ''
    )||'"}' coldef,
  kqfcooff coloff,
  lag(kqfcooff,1,-1) over (partition by t.indx order by kqfcooff,c.indx) prevoff,
  t.indx tindx, kqfcooff, c.indx cindx
from system.jps$_systab t, x$kqfco c
where c.kqfcotab=t.indx and kqfconam not in ('ADDR','INDX','INST_ID')
  and kqfcodty in (1,2,23) -- right now only coding for number,varchar,raw
) where coloff>prevoff
order by tindx,kqfcooff,cindx;


set heading off
set pagesize 0
set linesize 9999
set feedback off
set termout off
set trimspool on
spool orasga.c

select chr(10)||
'#include <stdio.h>    '||chr(10)||
'#include <sys/shm.h>  '||chr(10)||
'' from dual;

select '#define SGA_BASE 0x'||addr from x$ksmmem where rownum<=1;
select '#define TCNT '||count(distinct(tabdef)) from system.jps$_output;
select chr(10)||
'typedef struct {      '||chr(10)||
'  char* fname;        '||chr(10)||
'  unsigned int otype; '||chr(10)||
'  unsigned int size;  '||chr(10)||
'  unsigned int ofs;   '||chr(10)||
'  char* type;         '||chr(10)||
'  char* htype;        '||chr(10)||
'} FIELD;              '||chr(10)||
'                      '||chr(10)||
'typedef struct {      '||chr(10)||
'  char* tname;        '||chr(10)||
'  void* addr;         '||chr(10)||
'  unsigned int rsz;   '||chr(10)||
'  unsigned int rcnt;  '||chr(10)||
'  unsigned int fcnt;  '||chr(10)||
'' from dual;

select '  FIELD fields['||max(colcount)||'];' from system.jps$_systab;

select chr(10)||
'} TABLE;              '||chr(10)||
'                      '||chr(10)||
'TABLE master[] = {    '||chr(10)||
'' from dual;

select DECODE(lag(tabdef,1,'') over (order by i),tabdef,'          ',tabdef)||coldef data
from system.jps$_output order by i;


select chr(10)||
'}}};                                       '||chr(10)||
'                                           '||chr(10)||
'int main( int argc, const char* argv[] )   '||chr(10)||
'{                                          '||chr(10)||
'  int shmid;                               '||chr(10)||
'  void *shmaddr=(void*)SGA_BASE;           '||chr(10)||
'  short int headers=0;                     '||chr(10)||
'  char tname[33]="";                       '||chr(10)||
'  char *p=NULL;                            '||chr(10)||
'  int tindx;                               '||chr(10)||
'  int findx;                               '||chr(10)||
'  int rindx;                               '||chr(10)||
'  void *row;                               '||chr(10)||
'  char fmt[20];                            '||chr(10)||
'                                           '||chr(10)||
'  if(argc<3||argc>4) {                     '||chr(10)||
'    printf("Usage: %s shmid-from-ipcs [x$]tname [-headers]\n", *argv);  '||chr(10)||
'    exit(1);                               '||chr(10)||
'  }                                        '||chr(10)||
'                                           '||chr(10)||
'  if(argc==4&&'||'strlen(argv[3])>1)       '||chr(10)||
'    if(argv[3][0]==''-''&&'||'tolower(argv[3][1])==''h'') headers=1;  '||chr(10)||
'                                           '||chr(10)||
'  if(strncasecmp(argv[2],"X$",2)) strcat(tname,"X$");  '||chr(10)||
'  strncat(tname,argv[2],30);               '||chr(10)||
'  for(p=tname;*p;p++) *p=toupper(*p);     // convert tname to uppercase  '||chr(10)||
'                                           '||chr(10)||
'  shmid=atoi(argv[1]);                     '||chr(10)||
'  if( shmat(shmid,shmaddr,SHM_RDONLY) == (void*)-1 ) {  '||chr(10)||
'    printf("shmat: error attaching to SGA\n");  '||chr(10)||
'    exit(2);                               '||chr(10)||
'  }                                        '||chr(10)||
'                                           '||chr(10)||
'  for(tindx=0;tindx<TCNT;tindx++) {        '||chr(10)||
'    if(strcmp(tname,master[tindx].tname)==0) {  '||chr(10)||
'      if(headers) {                        '||chr(10)||
'        printf("%s\n",tname);              '||chr(10)||
'        printf("%-8s %4s ","ADDR","INDX"); '||chr(10)||
'        for(findx=0;findx<master[tindx].fcnt;findx++)  '||chr(10)||
'          printf(master[tindx].fields[findx].htype,  '||chr(10)||
'            master[tindx].fields[findx].fname);  '||chr(10)||
'        printf("\n");                      '||chr(10)||
'      };                                    '||chr(10)||
'      row=master[tindx].addr;               '||chr(10)||
'      for(rindx=0;rindx<master[tindx].rcnt;rindx++) {  '||chr(10)||
'        printf("%-8lX %4u ",row,rindx);     '||chr(10)||
'        for(findx=0;findx<master[tindx].fcnt;findx++)  '||chr(10)||
'          switch(master[tindx].fields[findx].otype) {'||chr(10)||
'          case 1:printf(master[tindx].fields[findx].type,  '||chr(10)||
'              row+master[tindx].fields[findx].ofs);  '||chr(10)||
'            break;                          '||chr(10)||
'          case 2:switch(master[tindx].fields[findx].size) {'||chr(10)||
'            case 1:printf(master[tindx].fields[findx].type,  '||chr(10)||
'                *((unsigned char*)(row+master[tindx].fields[findx].ofs)));  '||chr(10)||
'              break;                        '||chr(10)||
'            case 2:printf(master[tindx].fields[findx].type,  '||chr(10)||
'                *((unsigned short int*)(row+master[tindx].fields[findx].ofs)));  '||chr(10)||
'              break;                        '||chr(10)||
'            case 4:printf(master[tindx].fields[findx].type,  '||chr(10)||
'                *((unsigned long int*)(row+master[tindx].fields[findx].ofs)));  '||chr(10)||
'              break;                        '||chr(10)||
'            case 8:printf(master[tindx].fields[findx].type,  '||chr(10)||
'                *((unsigned long long int*)(row+master[tindx].fields[findx].ofs)));  '||chr(10)||
'              break;                        '||chr(10)||
'            }                               '||chr(10)||
'            break;                          '||chr(10)||
'          case 23:printf(master[tindx].fields[findx].type,  '||chr(10)||
'              *((unsigned long int*)row+master[tindx].fields[findx].ofs));  '||chr(10)||
'            break;                          '||chr(10)||
'          }                                 '||chr(10)||
'        printf("\n");                       '||chr(10)||
'        row+=master[tindx].rsz;             '||chr(10)||
'      };                                    '||chr(10)||
'    }                                       '||chr(10)||
'  }                                         '||chr(10)||
'                                            '||chr(10)||
'  shmdt(shmaddr);                           '||chr(10)||
'}                                           '||chr(10)||
'' from dual;

spool off
```

Sample output:

```
[oracle@rh4lab12 ~]$ sqlplus

SQL*Plus: Release 10.2.0.1.0 - Production on Wed Nov 1 09:16:11 2006

Copyright (c) 1982, 2005, Oracle.  All rights reserved.

Enter user-name: / as sysdba

Connected to:
Oracle Database 10g Enterprise Edition Release 10.2.0.1.0 - Production
With the Partitioning, OLAP and Data Mining options

SQL> @orasga

Function created.


Table dropped.


Table created.


Table dropped.


Table created.


597 rows created.


Commit complete.


PL/SQL procedure successfully completed.


Table dropped.


Table created.

SQL> quit
Disconnected from Oracle Database 10g Enterprise Edition Release 10.2.0.1.0 - Production
With the Partitioning, OLAP and Data Mining options
[oracle@rh4lab12 ~]$ gcc orasga.c -o orasga
[oracle@rh4lab12 ~]$ ipcs -m|grep -v 0x00000000

------ Shared Memory Segments --------
key        shmid      owner      perms      bytes      nattch     status
0x22c67678 196614     oracle    640        289406976  20
0x3f8f1600 229383     oracle    640        73662464   13

[oracle@rh4lab12 ~]$ ./orasga 196614 ksqdn -h
X$KSQDN
ADDR     INDX KSQDNGDN  KSQDNGLN                        KSQDNGUN                        KSQDNGUNID
20003C9C    0 ORCL      ORCL                            orcl                            1534721008
[oracle@rh4lab12 ~]$ ./orasga 196614 ksmsd -h
X$KSMSD
ADDR     INDX KSMSDNAM               KSMSDVAL
20004C1C    0 Fixed Size              1218992
20004C38    1 Variable Size          83887696
20004C54    2 Database Buffers      197132288
20004C70    3 Redo Buffers            2973696
[oracle@rh4lab12 ~]$ ./orasga 196614 ksmfs -h
X$KSMFS
ADDR     INDX KSMSSNAM                     KSMSSLEN
20045EE4    0 fixed_sga                     1218992
20045F0C    1 buffer_cache                197132288
20045F34    2 log_buffer                    2973696
```

Suggested example tables:

```
SQL> select tname from system.jps$_systab
where colcount<10 and rowcount<10;

TNAME
------------------------------
X$KSMSD
X$KSMFS
X$KSMJS
X$KSQDN
X$KSRMPCTX
X$KCBSDS
X$LE_STAT
X$KQRSD
X$KGICS
X$KMCQS
```