Cerner Millennium Explorer 2-3 Source Code Examples

!Review Example

```
! Application file: C:\Program Files\Cerner\CCLUSER\bgr_review_exam.VCL
! Discern Explorer Program file:  ccl_review_exam.PRG
! Generated by Visual Explorer on 2/13/1999

DROP PROGRAM ccl_review_exam GO
CREATE PROGRAM ccl_review_exam

PROMPT       "Output to File/Printer/MINE "  =  MINE

SET MaxSecs = 0
IF (IsOdbc)  SET MaxSecs = 15  ENDIF

SELECT        INTO $1
      P.NAME_FULL_FORMATTED,
      SEX_DISP = UAR_GET_CODE_DISPLAY( P.SEX_CD ),
      P.BIRTH_DT_TM,
      ENCNTR_TYPE_DISP = UAR_GET_CODE_DISPLAY( E.ENCNTR_TYPE_CD ),
      ENCNTR_TYPE_CLASS_DISP = UAR_GET_CODE_DISPLAY(
E.ENCNTR_TYPE_CLASS_CD ),
      age =  cnvtage( P.BIRTH_DT_TM )

FROM  PERSON  P,
      ENCOUNTER  E

PLAN  P WHERE P.PERSON_ID > 0
JOIN  E WHERE E.PERSON_ID =  P.PERSON_ID
      AND E.ENCNTR_ID > 0

ORDER P.PERSON_ID


Head Report
      ROW 2 COL 47 "Example Report"
      ROW 4 COL 7 "Date:"
      curdate "mmm-dd-yyyy;;d"
      ROW 5 COL 7 "Time:"
      curtime "hh:mm;;m"
      ROW + 2

Head Page
      ROW + 1
      COL 7  "Page:"
      curpage "###"
      ROW + 2
      COL 7  "Name:"
      COL 29  "Age:"
      COL 41  "Sex:"
      COL 54  "Encounter Type:"
      ROW + 2
```

```
Head  P.PERSON_ID
      ROW + 1
      age =  cnvtage( P.BIRTH_DT_TM )
      SEX_DISP1 = SUBSTRING( 1, 12, SEX_DISP ),
      NAME_FULL_FORMATTED1 = SUBSTRING( 1, 20, P.NAME_FULL_FORMATTED ),
      COL 7  NAME_FULL_FORMATTED1
      COL 29   age
      COL 41  SEX_DISP1
      ROW + 2

Detail
      if ((ROW + 4) >= maxrow)  break endif
      COL 54  ENCNTR_TYPE_CLASS_DISP
      ROW + 1

WITH  MAXREC = 200, MAXCOL = 500, TIME = VALUE( MaxSecs ),
      NOHEADING, FORMAT = VARIABLE


END
GO
```

;RDBMS OuterJoin Example

```
;cclseclogin go

select  p.person_id,
        name = substring(1,20,p.name_last_key),
        e.encntr_id,
        etype = uar_get_code_meaning(e.encntr_type_cd)

from    person p,
        encounter e

plan    p where p.person_id >0
join    e where outerjoin(p.person_id) = e.person_id

with maxrec = 1000
go
```

;RDBMS OuterJoin Example2

```
SELECT
      P.PERSON_ID,
      E.ENCNTR_ID,
      E_ENCNTR_TYPE_CLASS_DISP = UAR_GET_CODE_DISPLAY(
E.ENCNTR_TYPE_CLASS_CD ),
      O.ORDER_ID,
      O.ORDER_MNEMONIC

FROM  PERSON  P,
      ENCOUNTER  E,
      ORDERS  O

PLAN p WHERE  P.PERSON_ID >  0
JOIN e WHERE  outerjoin(P.PERSON_ID) =   e.PERSON_ID
JOIN o WHERE  outerjoin(E.ENCNTR_ID) = o.ENCNTR_ID

WITH  FORMAT, MAXREC = 100
```

;Discern Explorer Outer Join Example

```
select  p.person_id,
        name = substring(1,20,p.name_last_key),
        e.encntr_id,
        etype = uar_get_code_meaning(e.encntr_type_cd)
from    person p,
        encounter e,
        dummyt d
plan    p where p.person_id >0
join    d
join    e where p.person_id = e.person_id
                and e.encntr_type_cd >0

with    outerjoin = d,
        maxrec = 1000
go
```

;Discern Explorer Outerjoin DontExist Example

```
select  p.person_id,
        name = substring(1,20,p.name_last_key),
        e.encntr_id,
        etype = uar_get_code_meaning(e.encntr_type_cd)
from    person p,
        encounter e,
        dummyt d
plan    p where p.person_id >0
join    d
join    e where p.person_id = e.person_id
                and e.encntr_type_cd >0
with    outerjoin = d,  dontexist,
        maxrec = 1000
go
```

;Discern Explorer Dontcare Join Example

```
select
        p.person_id,
        pname = substring(1,20,p.name_last),
        addid = decode(a.seq,a.parent_entity_id),
        address_D = decode(a.seq,substring(1,20,a.street_addr)),
        address = substring(1,20,a.street_addr),;no decode-invalid data
        o.order_id,
        mne = substring(1,20,o.order_mnemonic)

from
        person p,
        orders o,
        dummyt d1,
        dummyt d2,
        address a

plan    p  where p.person_id > 0
join    d1
join    a  where p.person_id = a.parent_entity_id
                 and a.parent_entity_name = "PERSON"
join    d2
join    o  where p.person_id = o.person_id
                 and o.order_id > 0

with    dontcare = a,
        maxrec = 3000
go
```

;Nested Select Example

```
select  p.name_full_formatted
from    person p
where   p.person_id =
        (select o.person_id
        from    orders o
        where   o.order_mnemonic = "BUN")
go
```

;Exception Query Nested Select Not Exists Example

```
;this example selects people without orders
select  p.person_id,
        p.name_full_formatted
from    person p
where   not exists
        (select o.person_id
        from    orders o
        where   p.person_id = o.person_id)
with    maxrec = 1000
go

;this example shows how to use a nested select with plan join clauses

select  p.name_full_formatted,
        o.order_mnemonic

from    person  p,
        orders  o
```

```
plan    p
join    o where o.order_id = p.person_id
               and not exists
               (select r.order_id
                from    result r
                where o.order_id = r.order_id)
go
```

;Discern Explorer Orjoin Example

```
select  name = substring(1,30,p.name_full_formatted),
        order_mnemonic = substring(1,30,o.order_mnemonic),
        check = decode(oc.seq,"c",r.seq,"r","z"),
        result_id = decode(r.seq,r.result_id),
        order_com_id = decode(oc.seq,oc.long_text_id)

from    person p,
        orders o,
        order_comment oc,
        result r,
        dummyt d1,
        dummyt d2

plan p where p.person_id >0
join o where o.person_id = p.person_id
                and o.order_id >0
join d1
join (oc where oc.order_id = o.order_id)
orjoin d2
join (r where r.order_id = o.order_id)

order   p.person_id,
        o.order_id,
        check

with maxqual(oc, 100)
go
```

;Global Local Example1

```
;cclsource:ccl_global_local_exam1.prg
cclseclogin go

drop program ccl_global_local go
create program ccl_global_local

declare disp = c48
set disp = "Global Variable"

select  disp = cv.display,
        cv.code_value
from    code_value cv
where   cv.code_set = 57
        and cv.cdf_meaning = "MALE"
detail
```

```
        col  0 cv.code_value
        col +2 disp      ;shows the value of disp from select
        disp2 = concat("**", disp, "**")
        col +2 disp2     ;shows the value of the global variable disp
        row +1
with counter
call echo(" ")

call echo("The global varaible was NOT set equal to the local
expression")
call echo(concat("Disp = ",disp))

end go
ccl_global_local go
```

;Global Local Example2

```
;cclsource:ccl_global_local_exam2.prg
cclseclogin go

drop program ccl_global_local2 go
create program ccl_global_local2

declare disp = c48
set disp = "Global Variable"


select  disp1 = cv.display,
        cv.code_value
from    code_value cv
where   cv.code_set = 57
        and cv.cdf_meaning = "MALE"
detail
        col  0 cv.code_value
        col +2 disp1     ;shows the value of disp from select
        disp = concat("**", disp1, "**")
        col +2 disp      ;shows the value of the local expression disp1
        row +1
with counter
call echo(" ")
call echo("The global varaible was set equal to the local expression")
call echo(concat("Disp = ",disp))


end
go
ccl_global_local2 go
```

;While Loop Example

```
DROP PROGRAM ccl_while_admit_cnt GO
CREATE PROGRAM ccl_while_admit_cnt
```

```
SELECT updated =  cnvtdate( E.REG_DT_TM ),
       E.REG_DT_TM

FROM  ENCOUNTER  E

WHERE E.REG_DT_TM  BETWEEN cnvtdatetime( curdate - 30, 0) AND
                              cnvtdatetime( curdate,  curtime3)

ORDER updated
Head Report
      predate = updated
      ROW 3 COL 47 "Example Admit Count By Date Report"
      ROW + 1
Head Page
      ROW + 2
      COL 7  "Date:"
      COL 41  "Count:"
      ROW + 1
Foot updated
      ROW + 1
      WHILE ( predate < updated - 1 )
           plusdate = predate +1
           col 8 plusdate "mm/dd/yy;;d"
           row +1
           predate = predate +1
      ENDWHILE
      COL 8  E.REG_DT_TM
      COL 40  count(e.seq)
      predate = updated
END
GO
```

;Report Writer Example

```
DROP PROGRAM INPATS GO
CREATE PROGRAM INPATS

/*
For the UAR functionS to work you must login to the server.
The following command will prompt the user for a username, password,
and domain.  The user only needs to login to the server once for each
Discern Explorer session.
*/
EXECUTE CCLSECLOGIN

;GET THE CODE VALUE FOR INPATIENTS
SET INPATIENTS = 0.0
SET STAT = UAR_GET_MEANING_BY_CODESET(69,"INPATIENT",1,INPATIENTS)

/*
;if the encounter.encntr_type_class_cd is not filled out,
;try using the following select statement to set the inpatients
variable
```

```
;then in the second select statement
;change:
;       E.ENCNTR_TYPE_CLASS_CD = INPATIENTS AND
;to:
;       E.ENCNTR_TYPE_CD = INPATIENTS AND

select into "NL:"
        cv.code_value
from    code_value cv
where   cv.code_set = 71 and cv.display_key = "INPATIENT"
detail
        INPATIENTS = CV.CODE_VALUE
with    nocounter
*/


SELECT
        NAME                 = SUBSTRING(1,45,P.NAME_FULL_FORMATTED),
        SEX_DISP             = UAR_GET_CODE_DISPLAY( P.SEX_CD ),
        P.BIRTH_DT_TM,
        P.NAME_LAST_KEY,
        ENCNTR_TYPE          = UAR_GET_CODE_DISPLAY( E.ENCNTR_TYPE_CD ),
        NURSE_UNIT           = UAR_GET_CODE_DISPLAY(E.LOC_NURSE_UNIT_CD ),
        ROOM                 = UAR_GET_CODE_DISPLAY( E.LOC_ROOM_CD),
        BED                  = UAR_GET_CODE_DISPLAY( E.LOC_BED_CD ),
        LOS                  = DATETIMECMP(E.DISCH_DT_TM,E.REG_DT_TM),
        E.DISCH_DT_TM,
        E.REG_DT_TM

FROM    ENCOUNTER  E,
        PERSON  P
PLAN    P

JOIN    E WHERE  P.PERSON_ID = E.PERSON_ID AND
                 E.ENCNTR_TYPE_CLASS_CD = INPATIENTS AND
                 E.REG_DT_TM > CNVTDATETIME("01-JAN-1900") AND
                 E.DISCH_DT_TM  > CNVTDATETIME("01-JAN-1900")

ORDER   NURSE_UNIT,
        P.NAME_LAST_KEY

;/*** Begin Report Writer Section ***
HEAD REPORT
        ROOM_BED          = FILLSTRING(20," ")    ;store the room and bed
        LINE_D            = FILLSTRING(120,"=")   ;print double line
        LINE_S            = FILLSTRING(120,"-")   ;print single line
        BLANK_LINE        = FILLSTRING(120," ")   ;print a blank line

        MACRO (COL_HEADS)
                COL 0  "Name:"
                COL 50 "Sex:"
                COL 60 "Birth Date:"
                COL 75 "Room-Bed:"
                ROW -1
                COL 95 "Length of Stay"
                ROW +1
                COL 95 "In Days:"
        ENDMACRO

        ;Create Title Page
        ROW 0
        CALL CENTER("*** CERNER'S INPATIENT REPORT ***",0,120)
        COL 0   "Report Date: " , CURDATE "MM/DD/YY;;D"
        COL 100 "Report Time: " , CURTIME "HH:MM;;M"
        ROW +1  LINE_D
        ROW +2


HEAD PAGE
        COL 0  "PAGE: "
        COL 7  CURPAGE "###;L"
        ROW +1 COL_HEADS        ;Calls the col_heads macro.

ROW +1 LINE_S
        ROW +1

HEAD NURSE_UNIT
        ROW +1
        COL  0 "Nursing Unit:"
        COL +2 NURSE_UNIT
        ROW +2
        ;COL_HEADS      ;Uncomment if you want the column headings
                        ;at the top of each nursing unit.

DETAIL
        IF(ROW + 1 >= 57)       ;Verify there are enough blank
                BREAK           ;rows left on the page for
        ENDIF                   ;processing foot clauses
        COL 0  NAME
        COL 50 ;SEX_DISP
        CASE(SEX_DISP)
                OF "Male"   :"M"
                OF "Female" : "F"
                ELSE  "U"
        ENDCASE

        COL 60 P.BIRTH_DT_TM "MM/DD/YYYY;;D"
        IF (ROOM = " " AND BED = " ")
                        ROOM_BED = "No Room Or Bed"
                ELSEIF(ROOM != " " AND BED != " ")
                        ROOM_BED = BUILD(ROOM, "-", BED)
                ELSEIF(ROOM != " ")
                        ROOM_BED = BUILD(ROOM,"-No Bed")
                ELSE
                        ROOM_BED = BUILD("No Room-", BED)
        ENDIF
        COL 75  ROOM_BED
        COL 95  LOS
        col +1  "discharged: ", E.DISCH_DT_TM
        col +1  "registered: ", E.REG_DT_TM
        ROW +1

FOOT NURSE_UNIT
        IF(ROW + 5 >=57 )       ;Verify there are enough blank rows
                BREAK           ;left on the page for processing
        ENDIF                   ;foot clauses
        ROW +1
        COL 45 "    Total number of days for this nursing unit: "
        COL 95 SUM(LOS)
        ROW +1
        COL 45 "Total number of patients for this nursing unit: "
        COL 95 COUNT(NAME)
        ROW +1
        COL 45 "   Patients with LOS > 5 for this nursing unit: "
        COL 95 COUNT(NAME WHERE LOS >5)
        ROW +1
        COL 45 "  Average length of stay for this nursing unit: "
        COL 95 AVG(LOS)
        ROW +1

FOOT PAGE
        ROW 57
        COL 0 LINE_S
        ROW 58
        COL 0 "Report created by the Discern Explorer Program: INPATS"
  ROW 59
        COL 0 LINE_S

FOOT REPORT
        ROW +1          ;Need row+1 to advance past the
        ROW -3          ;page break.
        COL  0 BLANK_LINE
        ROW +1
        COL  0 BLANK_LINE       ;Print blank lines over the normal
        ROW +1          ;page header and column headings
        COL  0 BLANK_LINE
        ROW +5
        CALL CENTER("*** Grand Totals For Report ***",0,120)
        ROW +1
        COL 62 "        Total number of days: "
        COL 95 SUM(LOS)
        ROW +1
        COL 62 "    Total number of patients: "
        COL 95 COUNT(NAME)
  ROW +1
        COL 62 "Total  Patients with LOS > 5: "
        COL 95 COUNT(NAME WHERE LOS >5)
        ROW +1
        COL 62 "      Average length of stay: "
        COL 95 AVG(LOS)
        ROW +5
        CALL CENTER("*** END OF REPORT ***",0,120)
```

```
;*** End Report Writer Section ****/

WITH   ; MAXREC = 500, ;For testing only read 500 rows
        MAXCOL = 250
END
GO
INPATS go
```

;Max Encounter using Report Writer

```
DROP PROGRAM bgr_maxencounter_rpt GO
CREATE PROGRAM bgr_maxencounter_rpt

PROMPT       "Output to File/Printer/MINE "  =  MINE

;Request HNAM sign-on when executed from CCL on host
IF (VALIDATE(IsOdbc, 0) = 0)  EXECUTE CCLSECLOGIN  ENDIF

SET MaxSecs = 0
IF (VALIDATE(IsOdbc, 0))  SET MaxSecs = 15  ENDIF

SELECT        INTO $1
      P.PERSON_ID,
      P.NAME_FULL_FORMATTED,
      E.ENCNTR_ID ";L",
      ENCNTR_TYPE_DISP = UAR_GET_CODE_DISPLAY( E.ENCNTR_TYPE_CD )

FROM  PERSON  P,
      ENCOUNTER  E

PLAN P  WHERE  P.PERSON_ID >  0
JOIN E  WHERE  P.PERSON_ID =   E.PERSON_ID

ORDER BY    P.PERSON_ID,
      E.ENCNTR_ID DESC,
      0 DESC


Head  P.PERSON_ID
      NAME_FULL_FORMATTED1 = SUBSTRING( 1, 30, P.NAME_FULL_FORMATTED ),
      COL 7  P.PERSON_ID
      COL 22  NAME_FULL_FORMATTED1
      COL 54  E.ENCNTR_ID
      COL 69  ENCNTR_TYPE_DISP
      ROW + 2

WITH  MAXREC = 100, TIME = VALUE( MaxSecs ),  NOHEADING, FORMAT=
VARIABLE


END
GO
```

;Max Enounter using Nested Select and Max Function

```
DROP PROGRAM bgr_maxencounter GO
CREATE PROGRAM bgr_maxencounter

PROMPT        "Output to File/Printer/MINE "  =  MINE

;Request HNAM sign-on when executed from CCL on host
IF (VALIDATE(IsOdbc, 0) = 0)  EXECUTE CCLSECLOGIN  ENDIF

SET MaxSecs = 0
IF (VALIDATE(IsOdbc, 0))  SET MaxSecs = 15  ENDIF

SELECT        INTO $1
      P.PERSON_ID,
      P.NAME_FULL_FORMATTED,
      E.ENCNTR_ID,
      ENCNTR_TYPE_DISP = UAR_GET_CODE_DISPLAY( E.ENCNTR_TYPE_CD )

FROM  PERSON  P,
      ENCOUNTER  E

PLAN P  WHERE  P.PERSON_ID >  0
JOIN E  WHERE  P.PERSON_ID =   E.PERSON_ID
             and  E.ENCNTR_ID = (Select max(e2.encntr_id)
                                                       from encounter e2
                                                       where e2.person_id =
p.person_id)


WITH  MAXREC = 100, TIME = VALUE( MaxSecs ),  NOHEADING, FORMAT=
VARIABLE
```

;Bun Lytes Example Using Two Joins to the Orders Table

```
cclseclogin go

drop program bun_lytes_2_join go
create program bun_lytes_2_join

declare bun = f8
declare lytes = f8

set bun = uar_get_code_by("displaykey", 200, "BUN")
set lytes = uar_get_code_by("displaykey", 200, "LYTES")

select  distinct
        p.person_id,
        pname = substring(1,20,p.name_last),
        o.order_id,
        disp = uar_get_code_display(o.catalog_cd),
        o.catalog_cd,
        o2.order_id,
        disp = uar_get_code_display(o2.catalog_cd),
        o2.catalog_cd

from
        person p,
        orders o,
        orders o2

plan    p  where p.person_id > 0
join    o  where p.person_id = o.person_id
                 and o.order_id > 0
                 and o.catalog_cd =bun
join    o2  where o.person_id = o2.person_id
                 and o2.order_id > 0
                 and o2.catalog_cd = lytes

order   p.person_id, o.catalog_cd,0

with    maxrec = 3000
end
go
```

;Bun Lytes Example Using Report Writer Processing

```
drop program bgr_bun_lytes_rpt go
create program bgr_bun_lytes_rpt

declare bun = f8
declare lytes = f8

set bun = uar_get_code_by("displaykey", 200, "BUN")
set lytes = uar_get_code_by("displaykey", 200, "LYTES")


select  distinct
        p.person_id,
        pname = substring(1,20,p.name_last),
        o.order_id,
        disp = uar_get_code_display(o.catalog_cd),
        o.catalog_cd

from
        person p,
        orders o

plan    p  where p.person_id > 0
join    o  where p.person_id = o.person_id
                 and o.order_id > 0
                 and o.catalog_cd in (bun, lytes)

order   p.person_id, o.catalog_cd,0

head p.person_id
        got_bun = "N"
        got_lytes = "N"
detail
        if(o.catalog_cd = bun)
                got_bun = "Y"
        else
                got_lytes = "Y"
        endif
foot p.person_id
        if(got_bun = "Y" and got_lytes = "Y")
        col 10 pname
        row +1
        endif

with    maxrec = 3000
end
go
```

; Execute Example To Get Username

```
;creates the program that populates the Username variable.  Then, this
;program will be called by another program.

DROP PROGRAM   CCL_GET_USERNAME  GO
CREATE PROGRAM  CCL_GET_USERNAME

IF ( ( REQINFO -> UPDT_ID = 0 ) )
SET  USERNAME  =  CURUSER
ELSE
SELECT  INTO  "NL:"
P.NAME_FULL_FORMATTED
FROM ( PRSNL  P )

WHERE (P.PERSON_ID= REQINFO -> UPDT_ID )

DETAIL
 USERNAME = SUBSTRING ( 1 ,  20 , P.NAME_FULL_FORMATTED),
 COL  10 ,
 USERNAME
 WITH  NOCOUNTER
ENDIF

 END GO

; This program executes the CCL_GET_USERNAME program
DROP PROGRAM CCL_CALL_EXECUTE GO
CREATE PROGRAM CCL_CALL_EXECUTE

PROMPT       "Output to File/Printer/MINE "  =  MINE

DECLARE USERNAME = VC
EXECUTE CCL_GET_USERNAME
```

```
SELECT        INTO $1
      P.NAME_LAST_KEY

FROM  PERSON  P


Head Report
      ROW 1 COL 17 "Username that was retrieved from the
CCL_CALL_EXECUTE pgm :"
      ROW 1 COL 69 username
      ROW + 2

Detail
      COL 17  P.NAME_LAST_KEY
      ROW + 1

WITH  NOHEADING, FORMAT= VARIABLE
END GO
```

;BUN Lytes Example Creating and Using a Record Structure

;Part 1 defining the record structure.

```
drop program bgr_rec_struc_exam go
create program bgr_rec_struc_exam

;declare global variables
declare cnt = i4
declare bun = f8
declare lytes = f8

;get the code value for catalog_cd BUN
set bun = uar_get_code_by("displaykey", 200, "BUN")


free record bgr_bun
;create a record structure to store the person id's of people with buns
record bgr_bun(
        1  list[*]
           2  person_id = f8 )
```

;Part 2 buld and populate the record structure

```
select  into "NL:"
        o.person_id
from    orders o
where   o.catalog_cd = bun

head report
        ;initialize the list
        stat = alterlist(bgr_bun->list, 10)
        cnt = 0
detail
        cnt = cnt+1
        ;check to see if more positions are needed in the list
        if(mod(cnt,10) = 1 and cnt > 1)
                stat = alterlist(bgr_bun->list, cnt + 9)
        endif
        ;populate the record sturcture
        bgr_bun->list[cnt].person_id = o.person_id
foot report
        ;remove un-used positions from the list
        stat = alterlist(bgr_bun->list, cnt)
with nocounter
```

;Part 3 verify that the record structure is populated.

```
for(lcnt = 1 to cnt)
        call echo(bgr_bun->list[lcnt].person_id)
endfor
```

;Part 4 Using the record structure in a join

```
;Use the record structure to join to the orders table to get
;the person ids of people that have lytes
;Also join to the person table to get the names

;get the code value for catalog_cd LYTES
set lytes = uar_get_code_by("displaykey", 200, "LYTES")


select distinct
        p.name_full_formatted
from    (dummyt d with seq = value(size(bgr_bun->list,5))),
        orders o,
        person p
plan    d
join    o where bgr_bun->list[d.seq].person_id = o.person_id and
                o.catalog_cd = lytes
join    p where o.person_id = p.person_id

order o.person_id
end
go
```

Record Structure Example, Print Orders in Columns

```
DROP PROGRAM ccl_order_cols GO
CREATE PROGRAM ccl_order_cols

PROMPT   "Output to File/Printer/MINE "  =  MINE

;Request HNAM sign-on when executed from CCL on host
```

```
IF (VALIDATE(IsOdbc, 0) = 0)  EXECUTE CCLSECLOGIN  ENDIF

declare future = f8
declare completed = f8
declare canceled = f8

set future = uar_get_code_by("meaning", 6004, "FUTURE")
set completed = uar_get_code_by("meaning", 6004, "COMPLETED")
set canceled = uar_get_code_by("meaning", 6004, "CANCELED")

record person(
   1 person[*]
      2 most_cnt = i4
      2 p_id = f8
      2 name = C20
      2 canceled[*]
         3 odisplay = c40
      2 completed[*]
         3 odisplay = c40
      2 future[*]
         3 odisplay = c40 )




SELECT    INTO $1
        name = substring(1, 25, P.NAME_FULL_FORMATTED),
        P.PERSON_ID,
        O.ORDER_STATUS_CD,
        ORDER_STATUS_DISP = UAR_GET_CODE_DISPLAY( O.ORDER_STATUS_CD ),
        O.ORDER_ID,
        O.CATALOG_CD,
        ord = UAR_GET_CODE_DISPLAY( O.CATALOG_CD )

FROM    PERSON  P,
        ORDERS  O

plan p where p.person_id >= 6001
join o where o.person_id = p.person_id
        and  O.ORDER_STATUS_CD+0 in (canceled, completed, future)

ORDER   P.PERSON_ID,
        O.ORDER_STATUS_CD,
        O.CATALOG_CD,
        0

Head Report
        cntp = 0
        cntpx = 0
        stat_cnt = 0
        most_cnt = 0
        avail_cnt = 0
```

```
Head Page
        COL 5  "Name:"
        COL 25  "Canceled:"
        COL 65  "Completed:"
        COL 105  "Future:"
        ROW + 2

Head P.PERSON_ID
        most_cnt = 0
        avail_cnt = 0
        cntp = cntp + 1

        if(mod(cntp, 10) = 1)
          stat = alterlist(person->person, cntp + 9)
        endif

        person->person[cntp].p_id = p.person_id
        person->person[cntp].name = name

        stat = alterlist(person->person[cntp].canceled,  10)
        stat = alterlist(person->person[cntp].completed, 10)
        stat = alterlist(person->person[cntp].future,    10)

Head O.ORDER_STATUS_CD
        stat_cnt = 0

Detail
        stat_cnt = stat_cnt + 1

        if(stat_cnt > avail_cnt)
           avail_cnt = avail_cnt + 10
           stat = alterlist(person->person[cntp].canceled, stat_cnt+9)
           stat = alterlist(person->person[cntp].completed, stat_cnt+9)
           stat = alterlist(person->person[cntp].future, stat_cnt + 9)
        endif

        If( O.ORDER_STATUS_CD = canceled)
           person->person[cntp]->canceled[stat_cnt].odisplay = ord
        elseif( O.ORDER_STATUS_CD = completed)
          person->person[cntp]->completed[stat_cnt].odisplay = ord
        elseif( O.ORDER_STATUS_CD = future)
           person->person[cntp]->future[stat_cnt].odisplay = ord
        endif

Foot O.ORDER_STATUS_CD
        if(stat_cnt > most_cnt)
           most_cnt = stat_cnt
        endif

Foot P.PERSON_ID
        stat = alterlist(person->person[cntp]->canceled, most_cnt)
        stat = alterlist(person->person[cntp]->completed, most_cnt)
        stat = alterlist(person->person[cntp]->future, most_cnt)
        person->person[cntp].most_cnt = most_cnt

Foot Report
        stat = alterlist(person->person, cntp)

        for(cntpx = 1 to cntp)
          col 5 person->person[cntpx].name
          row + 1

           for(x = 1 to person->person[cntpx].most_cnt)
                    col 25 person->person[cntpx]->canceled[x].odisplay
                    col 65 person->person[cntpx]->completed[x].odisplay
                    col 105  person->person[cntpx]->future[x].odisplay
                    row + 1
              endfor
        endfor

WITH  MAXREC = 200

END
GO
```

;Record Structure Example

```
cclseclogin go

free record person go

;create the record strucutre
record person(
        1 person[*]
          2 id    = f8
          2 name  = c20
          2 addr[*]
            3 id     = f8
            3 street = c40
            3 type   = c40   )
go

select  p.person_id,
        name = substring(1,20,p.name_full_formatted),
        a.address_id,
        street = substring(1,40,a.street_addr),
        type = uar_get_code_meaning(a.address_type_cd)

from    person p,
        address a

where   p.person_id = a.parent_entity_id
        and a.parent_entity_name = "PERSON"
        and p.person_id >0

order   p.person_id
```

```
head report
        ;initialize counter variables
        cntp  = 0
        cntpx = 0
        cnta  = 0
        cntax = 0

;initialize 10 positions in the person segment of the record structure
        stat = alterlist(person->person,10)

head p.person_id
        cntp = cntp +1

        ;add positions to the person segment if needed
        stat = mod(cntp,10)
        if(stat = 1 and cntp != 1)
                stat = alterlist(person->person,cntp + 10)
        endif
        ;store the data in the record structure
        person->person[cntp].id = p.person_id
        person->person[cntp].name = name

        cnta = 0
        stat = alterlist(person->person[cntp].addr, 10)

detail
        cnta = cnta +1
        ;add positions to the address segment if needed
        stat = mod(cnta,10)
        if(stat = 1 and cnta != 1)
                stat = alterlist(person->person[cntp].addr, cnta + 10)
        endif
         ;store the data in the reocrd structure
        person->person[cntp].addr[cnta].id = a.address_id
        person->person[cntp].addr[cnta].street = street
        person->person[cntp].addr[cnta].type = type

foot p.person_id
        ;remove unused address postions
        stat = alterlist(person->person[cntp].addr, cnta)

foot report
        ;remove unused person positions
        stat = alterlist(person->person, cntp)

;display the data stored in the record structure
        for(cntpx = 1 to cntp)
                col  1  person->person[cntpx].id
                col +1  person->person[cntpx].name
                row +1
                cnta = size(person->person[cntpx].addr, 5)
                for(cntax = 1 to cnta)
                       col 15  person->person[cntpx].addr[cntax].id
                       col +1  person->person[cntpx].addr[cntax].street
```

```
                                          col +1  person->person[cntpx].addr[cntax].type
                                          row +1
                                 endfor
                      endfor

           go
```

;Join to Record Structure Example

```
;cclseclogin go

free record person go
;create the record structure
record person(
        1 person[*]
          2 id    = f8
          2 name  = c20 )
go

select  p.person_id,
        name = substring(1,20,p.name_full_formatted)

from    person p

where   p.person_id >0

order   p.person_id

head report
        ;initialize counter variables
        cntp  = 0
        cntpx = 0

 ;initialize 10 positions in the person segment of the record structure
        stat = alterlist(person->person,10)

detail
        cntp = cntp +1

        ;add positions to the person segment if needed
        stat = mod(cntp,10)
        if(stat = 1 and cntp != 1)
                stat = alterlist(person->person,cntp + 10)
        endif
        ;store the data in the record structure
        person->person[cntp].id = p.person_id
        person->person[cntp].name = name


foot report
        ;remove unused person positions
        stat = alterlist(person->person, cntp)

;display the data stored in the record structure
        for(cntpx = 1 to cntp)
                col  1  person->person[cntpx].id
                col +1  person->person[cntpx].name
                row +1
        endfor
go
```

```
;join the record structure to the person_alias table

select into "NL:"  ;do the select in memory
        pa.person_id,
        pa.alias
            ;the value and size functions are used to determine the
            ;number of positions in the person record structure
from    (dummyt d with seq = value(size(person->person,5))),
        person_alias pa

plan    d
join    pa where person->person[d.seq].id = pa.person_id

detail  ;echo information to the screen with out using the displayer
        call echo(build(
                "-- pid =",
                person->person[d.seq].id,
                "-- name =",
                person->person[d.seq].name,
                "-- alias =",
                pa.alias))

go
```

;Example DRG Report

```
/*The following example report shows how to use a record structure
  to store the total number of encounters with DRG's and the number
  of encounters with each DRG.  It then produces a report that is
  sorted from the most frequent DRG to the least frequent DRG.
  To order the output by the most common DRG and calculate the
  percentage of total DRG's that each DRG represents, you need
  to know the total number of encounters with DRG's and how
  many times each DRG occurs.
*/


DROP PROGRAM ccl_drg_report GO
CREATE PROGRAM ccl_drg_report

;The following record structure
;is used to store total number of encounters with DRG's and
;the number of times each DRG occurs.

record drg(
        1 total_drg = f8                ;total DRG's
        1 qual[*]
          2  nomenclature_id = f8       ;nomenclature id of each DRG
          2  sort_num        = f8 )     ;number of times the DRG occurs

;select the encounters that have DRG's
;load the required information into the record structure

SELECT  into "NL:"
        E.ENCNTR_ID,
        D.ENCNTR_ID,
        D.NOMENCLATURE_ID
FROM    ENCOUNTER  E,
        DRG  D
PLAN  E
JOIN  D WHERE D.ENCNTR_ID =  E.ENCNTR_ID

ORDER   D.NOMENCLATURE_ID


head report
        cnt = 0
        stat = alterlist(drg->qual,10)

head d.nomenclature_id
        cnt = cnt +1

        ;add positions to the qual segment if needed
        stat = mod(cnt,10)
        if(stat = 1 and cnt != 1)
                stat = alterlist(drg->qual,cnt + 10)
        endif
```

```
detail
        row +0

foot d.nomenclature_id
        drg->qual[cnt].nomenclature_id = d.nomenclature_id
        drg->qual[cnt].sort_num = count(d.seq)

        ;uncomment the following three lines for debugging
        ;col 0  drg->qual[cnt].nomenclature_id
        ;col +2 drg->qual[cnt].sort_num
        ;row +1

foot report
        drg->total_drg = count(d.seq)
        stat = alterlist(drg->qual,cnt)

        ;uncomment the following three lines for debugging
        ;row +1
        ;col 10 "totat"
        ;col +2 drg->total_drg



;Join to the record structure to select the encounters with DRG's
;ordered from the most frequent DRG to the least frequent DRG

SELECT  N.SOURCE_STRING,
        sort_num = drg->qual[d1.seq].sort_num,
        encntr_type=UAR_GET_CODE_DISPLAY(E.ENCNTR_TYPE_CLASS_CD),
        E.ENCNTR_ID,
        D.DRG_ID,
        D.NOMENCLATURE_ID

FROM    ENCOUNTER  E,
        DRG  D,
        NOMENCLATURE  N,
        (dummyt d1 with seq = value(size(drg->qual,5)))

PLAN  d1
join  d where drg->qual[d1.seq].nomenclature_id = d.nomenclature_id
JOIN  e WHERE D.ENCNTR_ID =  E.ENCNTR_ID
JOIN  N WHERE N.NOMENCLATURE_ID =  D.NOMENCLATURE_ID

ORDER   sort_num desc,
        N.NOMENCLATURE_ID

head report
        tpercent = 0.0
        rdate = format(curdate,"dd-mmm-yyyy;;d")
        col  0 rdate
        row +1
        rtime = format(curtime,"hh:mm;;m")
        col  0 rtime
        col 30 "Example DRG Report"
        row +3


head n.nomenclature_id
        nomen = substring(1,120,N.SOURCE_STRING)
        col  0 nomen
        row +2

detail
        col 10 e.encntr_id
        col 25 encntr_type
        row +1

foot n.nomenclature_id
        row +1
        col  0 "                    Encounters with this DRG: "
        col +1 sort_num "#########"
        row +1
        rpercent = (sort_num / drg->total_drg) * 100
        col  0 "Percentage of total that this DRG represents: "
        col +2 rpercent "###.####%"
        row +2
        tpercent = tpercent + rpercent

foot report
        col 0 "****** End of Report *****"


WITH    MAXREC = 1000


END GO
```

;Select Into File Example

```
DROP PROGRAM ccl_into_file_exam GO
CREATE PROGRAM ccl_into_file_exam

SELECT  INTO BGR_TEST
        O.ORIG_ORDER_DT_TM,
        O.CATALOG_CD,
        CATALOG_DISP = UAR_GET_CODE_DISPLAY( O.CATALOG_CD )


FROM    ORDERS  O

WHERE   O.ORIG_ORDER_DT_TM BETWEEN CNVTDATETIME(CURDATE - 7,0)
                                AND CNVTDATETIME(CURDATE, 235959)

ORDER   cnvtdatetime( O.ORIG_ORDER_DT_TM ),
        O.CATALOG_CD

WITH    FORMAT =  PCFORMAT,
        NOHEADING

END GO
ccl_into_file_exam GO
```

;Bun Lytes Example Creating and Using a Custom Table

```
cclseclogin go

drop program bgr_test go
create program bgr_test

declare bun = f8
declare lytes = f8

set bun = uar_get_code_by("displaykey", 200, "BUN")
set lytes = uar_get_code_by("displaykey", 200, "LYTES")

select  distinct into table bgr_test
        p.person_id,
        o.catalog_cd,
        o.order_id,
        name = substring(1,20,p.name_last_key),
        disp = uar_get_code_display(o.catalog_cd)

from
        person p,
        orders o

plan    p  where p.person_id > 0
join    o  where p.person_id = o.person_id
                 and o.order_id > 0
                 and o.catalog_cd =bun

order   p.person_id, o.catalog_cd

with    organization = i


select  distinct
        b.person_id,
        b.catalog_cd,
        b.name,
        b.order_id,
        b.disp,
        o.catalog_cd,
        o.order_id,
        disp = uar_get_code_display(o.catalog_cd)


from
        bgr_test b,
        orders o
```

```
plan    b  where b.person_id > 0
join    o  where b.person_id = o.person_id
                 and o.order_id > 0
                 and o.catalog_cd =lytes

order   b.person_id, b.catalog_cd, o.catalog_cd, 0

with    maxrec = 3000

end
go
bgr_test go
```

;Include File Example

;The Include File

;Save the Include file as **ccl_head_rpt.inc**

```
row 5
call center("Example Report Heading", 0,80)
row +1
call center("Created Using an Include File", 0,80)
row +1
datetime=concat(format(curdate,"mmm-dd-yyyy;;d"),
                " ",format(curtime,"hh:mm;;s"))
call center(datetime,0,80)
row +5
```

;Source code that uses the include file

```
cclseclogin go

drop program ccl_include_exam go
create program ccl_include_exam

select  p.name_last_key
from    person p

head report

%i ccl_head_rpt.inc

head page
        col 0 "Page:"
        col +1 curpage "###;l"
        row +2
detail
        if(row > 50)
        break
        endif
        col 1 p.name_last_key
        row +1
with maxrec = 300
end
go
ccl_include_exam go
```

;Control Index Example

```
select  p.person_id,
        p.updt_dt_tm

from    person p

where   p.updt_dt_tm +0 > cnvtdatetime("01-jan-1998") and
        p.person_id >352287
        ;p.updt_dt_tm  > cnvtdatetime("01-jan-1998") and
        ;p.person_id +0 >352287



with    nocounter

go
```

;Efficiency Testing Example

```
/***Select people with last names = A* NOT using an indexed field*****/
drop program ccl_eff_test go
create program ccl_eff_test
select  into "NL:"
        name = substring(1,20,p.name_last),
        name_key = substring(1,20,p.name_last_key)
from    person p
where   p.name_last = "A*"
end go
call echo("Names not using indexed field") go
ccl_eff_test go
/***Select people with last names = A* using an indexed field********/
drop program ccl_eff_test go
create program ccl_eff_test
select  into "NL:"
        name = substring(1,20,p.name_last),
        name_key = substring(1,20,p.name_last_key)
from    person p
where   p.name_last_KEY = "A*"
end go
call echo("Names using indexed field") go
ccl_eff_test go
/***Select people w/o encounters using nested select****************/
drop program ccl_eff_test go
create program ccl_eff_test
```

```
select  into "NL:"
        p.person_id,
        name = substring(1,20,p.name_last_key)
from    person p
where   not exists
        (select e.person_id
        from    encounter e
        where   e.person_id = p.person_id)
end go
call echo("People without encounters Nested Select") go
ccl_eff_test go
/***Select people w/o encounters using outerjoin dontexist************/
drop program ccl_eff_test go
create program ccl_eff_test
select  into "NL:"
        p.person_id,
        name = substring(1,20,p.name_last_key)
from    person p,
        encounter e,
        dummyt d
plan    p where p.person_id >0
join    d
join    e where p.person_id = e.person_id and e.encntr_type_cd >0
with    outerjoin = d,  dontexist
end go
call echo("People without encounters Outerjoin Dontexist") go
ccl_eff_test go
```

;Prompt, Standard Join, Report Writer Example

```
DROP PROGRAM personaddress GO
CREATE PROGRAM personaddress
PROMPT   "Output to File/Printer/MINE "  =  MINE
SET MaxSecs = 0  IF (IsOdbc)  SET MaxSecs = 60  ENDIF
SELECT    INTO $1
        P.NAME_FULL_FORMATTED,
        A.STREET_ADDR,
        A.CITY,
        A.STATE,
        A.ZIPCODE,
        ADDRESS_TYPE_CDF = UAR_GET_CODE_MEANING( A.ADDRESS_TYPE_CD ),
        A.ADDRESS_ID,
        P.PERSON_ID,
        A.PARENT_ENTITY_ID,
        A.PARENT_ENTITY_NAME
FROM    ADDRESS  A,
        PERSON  P
PLAN p
JOIN a  WHERE P.PERSON_ID =  A.PARENT_ENTITY_ID
            AND A.PARENT_ENTITY_ID = "PERSON"
ORDER   P.PERSON_ID,
        P.NAME_FULL_FORMATTED,
        ADDRESS_TYPE_CDF
Head Report
        ROW 1 COL 53 "Address Report"
        ROW + 1
Head  P.NAME_FULL_FORMATTED
        ROW + 2
        NAME_FULL_FORMATTED1 = SUBSTRING( 1, 40, P.NAME_FULL_FORMATTED),
        COL 1  NAME_FULL_FORMATTED1
        ROW + 1
Head  ADDRESS_TYPE_CDF
        COL 6  ADDRESS_TYPE_CDF
        ROW + 1
Detail
        if ((ROW + 3) >= maxrow)  break endif
        ROW + 1
        STREET_ADDR1 = SUBSTRING( 1, 50, A.STREET_ADDR ),
        COL 18  STREET_ADDR1
        ROW + 1
        STATE1 = SUBSTRING( 1, 10, A.STATE ),
        CITY1 = SUBSTRING( 1, 50, A.CITY ),
        COL 18  CITY1
        COL 67  STATE1
        COL 78  A.ZIPCODE
        ROW + 1
WITH  MAXREC = 100, MAXCOL = 250, TIME = VALUE( MaxSecs ), NOHEADING,
FORMAT = VARIABLE
END GO
```

;Array Example

```
select  distinct p.name_last_key

from    person p

where   p.name_last_key > "A"

order   p.name_last_key

head report
        name[100] = fillstring(100, " ") ;define a static array
        x = initarray(name, " ") ;set all array values equal to spaces
        cnt = 0
        cntx = 0
detail

        cnt = cnt+1
        name[cnt] = p.name_last_key

foot report
        for(cntx = 1 to cnt)
                col 0  name[cntx]
                row +1
        endfor

with maxrec = 100

go
```