CA Gen

Host Encyclopedia Administration Guide

Release 8.5

CA Technologies Product References

This document references the following CA Technologies products:

- CA Gen
- AllFusion® Gen

Contact CA Technologies

Contact CA Support

For your convenience, CA Technologies provides one site where you can access the information that you need for your Home Office, Small Business, and Enterprise CA Technologies products. At http://ca.com/support, you can access the following resources:

- Online and telephone contact information for technical assistance and customer services
- Information about user communities and forums
- Product and documentation downloads
- CA Support policies and guidelines
- Other helpful resources appropriate for your product

Providing Feedback About Product Documentation

If you have comments or questions about CA Technologies product documentation, you can send a message to techpubs@ca.com.

To provide feedback about CA Technologies product documentation, complete our short customer survey which is available on the CA Support website at http://ca.com/docs.

Contents

Chapter 1: Methods for Collecting Statistics

This guide recommends ways to improve the performance and process of the CA Host Encyclopedia. Performance improvements involve data that you can modify either within the DBMS or within the encyclopedia. The intent of the modifications is to reduce processing time or increase throughput. Process improvements involve the techniques of encyclopedia use that you can change to reduce contention or decrease processing time.

No single set of performance recommendations can be the right set for all environments. Improving performance is a matter of balancing the workload of the computer to best use available resources. Many variables are involved in performance, such as the size and speed of the central processing unit, the priority assigned to the execution region, the parameters used for the DBMS subsystem, and load levels. Improving performance is an ongoing task because these variables can change over time.

This section contains the following topics:

Audience

The audiences for this guide include the following:

- Encyclopedia administrators
- Model administrators
- Project coordinators
- Database administrators
- System programmers
- Application programmers

Creating an Initial Baseline

The improvements obtained by implementing the recommendations in this guide depend on several factors:

- The current DBMS characteristics of the encyclopedia
- The current procedures in place with respect to encyclopedia usage
- The specific characteristics of the models in the encyclopedia

Because of these factors, you should evaluate the performance of your encyclopedia before implementing the recommendations in this guide. To start with, create an initial baseline using statistics from a set of models and encyclopedia activities. Then, as you collect statistics after implementing an improvement technique, you can compare them with the baseline to determine the actual improvement.

Evaluating Performance

After establishing your initial baseline, you should implement performance improvements to obtain optimal use of the DBMS. Once the performance improvements are implemented, create a new baseline for comparison to future modifications.

Evaluating Processes

After you implement performance improvements, you can implement selected process improvements. Collect statistics after the process improvements and compare them against the recently updated baseline.

Note: You should implement performance improvements first to obtain optimal use of the DBMS, and then implement process improvements. This combination can be an iterative process that continually improves the performance of the encyclopedia.

Tasks You Can Perform

Tasks	Reference
Collect statistics for either specific encyclopedia functions or for an entire model	Chapter 1
Review performance recommendations specific to DB2	Chapter 2

Tasks	Reference
Review process recommendations	Chapter 3
Review Host Encyclopedia utilities	Chapter 4
Review a matrix of which encyclopedia functions can and cannot run together	Encyclopedia Concurrency Matrix section in Chapter 3
Review the names of the Host	Appendix A

Note: For more information on concurrency rules, see *CA Gen Host Encyclopedia Subsetting User Guide.*

Determining What Statistics to Collect

Statistics help you to monitor and optimize encyclopedia performance. You can collect various statistics from tables on the Host Encyclopedia, from DB2 trace data, and logs. This chapter discusses queries and reports that you can use to collect statistics. Most of the queries and reports listed use Structured Query Language (SQL). One query and report is discussed that uses FOCUS, which is a reporting tool from Information Builders. Similar reports can be created with other reporting tools.

The illustrations in this chapter list the queries/reports to use based on the statistics that you want to determine.

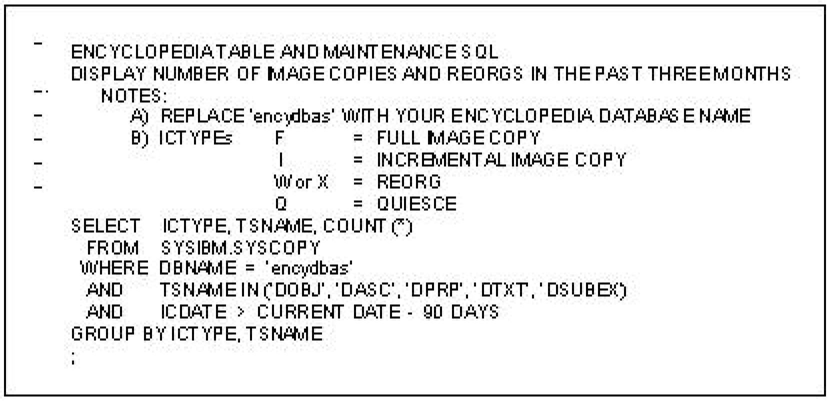
More information:

Process Recommendations (see page 29)

Statistics About Encyclopedia Tables

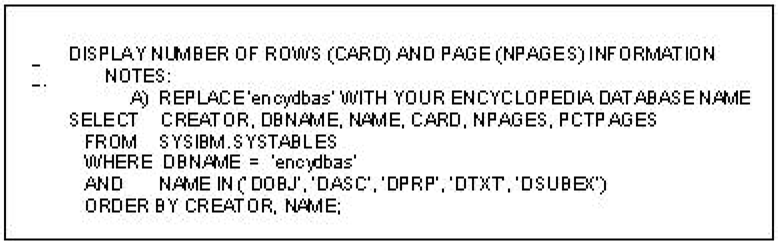
The SQL queries listed in the following illustrations gather statistics from the encyclopedia tables. Note that you replace encydbas with the name of your encyclopedia database.

Number of Image Copies and Reorganizations:



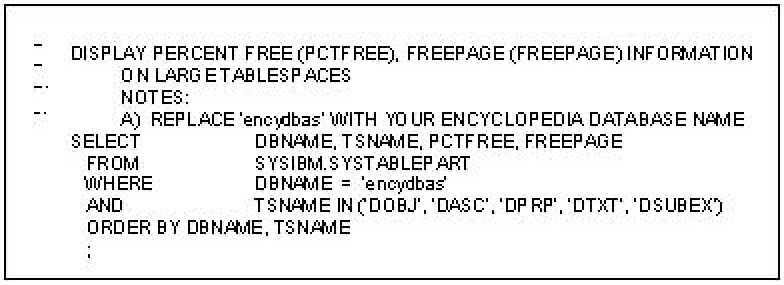
```
-    ENCYCLOPEDIA TABLE AND MAINTENANCE SQL
     DISPLAY NUMBER OF IMAGE COPIES AND REORGS IN THE PAST THREE MONTHS
-        NOTES:
-            A) REPLACE 'encydbas' WITH YOUR ENCYCLOPEDIA DATABASE NAME
-            B) ICTYPEs      F        =  FULL IMAGE COPY
-                            I        =  INCREMENTAL IMAGE COPY
-                            W or X   =  REORG
                             Q        =  QUIESCE
     SELECT   ICTYPE, TSNAME, COUNT (*)
       FROM   SYSIBM.SYSCOPY
      WHERE  DBNAME = 'encydbas'
       AND    TSNAME IN ('DOBJ', 'DASC', 'DPRP', 'DTXT, 'DSUBEX)
       AND    ICDATE > CURRENT DATE - 90 DAYS
     GROUP BY ICTYPE, TSNAME
     ;
```

Number of Rows on Encyclopedia tables:



```
-    DISPLAY NUMBER OF ROWS (CARD) AND PAGE (NPAGES) INFORMATION
-         NOTES:
             A) REPLACE 'encydbas' WITH YOUR ENCYCLOPEDIA DATABASE NAME
     SELECT   CREATOR, DBNAME, NAME, CARD, NPAGES, PCTPAGES
       FROM   SYSIBM.SYSTABLES
       WHERE  DBNAME =  'encydbas'
       AND    NAME IN ( DOBJ', 'DASC', 'DPRP', 'DTXT', 'DSUBEX')
       ORDER BY CREATOR, NAME;
```

Amount of Free Space on Large Tablespaces:



```
-    DISPLAY PERCENT FREE (PCTFREE), FREEPAGE (FREEPAGE) INFORMATION
-           ON LARGE TABLESPACES
-           NOTES:
-           A) REPLACE 'encydbas' WITH YOUR ENCYCLOPEDIA DATABASE NAME
     SELECT                DBNAME, TSNAME, PCTFREE, FREEPAGE
       FROM                SYSIBM.SYSTABLEPART
      WHERE                DBNAME = 'encydbas'
       AND                 TSNAME IN ('DOBJ', 'DASC', 'DPRP', 'DTXT', 'DSUBEX')
       ORDER BY DBNAME, TSNAME
       ;
```

Number of Tablespace Partitions or Segmented Tablespaces:

```
-
    DISPLAY PARTITION INFORMATION ON LARGE ENCYCLOPEDIA TABLES ***
            NOTES:
           A) REPLACE 'encydbas' WITH YOUR ENCYCLOPEDIA DATABASE NAME
-   SELECT              PARTITIONS, NAME, CREATOR, SEGSIZE
      FROM              SYSIBM.SYSTABLESPACE
      WHERE             DBNAME = 'encydbas'
      AND               NAME IN ('DOBJ', DASC', 'DASC', 'DPRP', 'DTXT', 'DSUBEX')
      ORDER BY NAME
      ;
```

More information:

Performance Recommendations (see page 73)

Statistics About the Encyclopedia Profile

Number of Models in Encyclopedia:

```
-
-   DISPLAY NUMBER OF MODELS ON THE ENCYCLOPEDIA
-           NOTES:
           A) REPLACE 'encycreator' WITH YOUR ENCYCLOPEDIA DATABASE CREATOR-ID
    SELECT              COUNT (*)
      FROM              encycreator.DMDL
      ;
```

Number of Models Updated During Last Month:

```
-
    DISPLAY MODELS CHANGED IN THE LAST MONTH
-           NOTES:
           A) REPLACE 'encycreator' WITH YOUR ENCYCLOPEDIA DATABASE CREATOR-ID
-   SELECT              COUNT (*)
      FROM              encycreator.DMDL
      WHERE             DATE (MODEL_DATE) > CURRENT DATE - 1 MONTH;
```

Size of Models and Number of Objects:

```
-
-   DISPLAY NUMBER OF OBJECTS PER MODEL, IN DESCENDING ORDER
-           NOTES:
           A) REPLACE 'encycreator' WITH YOUR ENCYCLOPEDIA DATABASE CREATOR-ID
    SELECT              MODEL_NAME, COUNT (*)
      FROM              encycreator.DMDL,
                        encycreator.DOBJ
      WHERE             OBJ_MODEL_ID = MODEL_ID
      GROUP BY MODEL_NAME
      ORDER BY 2 DESC
      ;
```

Number of Objects in Model and Subset Checkouts:

```
   DISPLAY NUMBER OF OBJECTS IN MODEL AND SUBSET CHECKOUTS
SELECT MODEL_NAME, S_SUBSET_NAME, S_CKO_USER, COUNT (*)
 FROM DSUBID, DMDL, DSUBEX
 WHERE S_MODEL_ID = MODEL_ID
    AND S_CKO_USER ^= '              '
 AND S_SUBSET_ID = SE_SUBSET_ID
 AND S_MODEL_NAME = 'modelname'
 GROUP BY MODEL_NAME, S_SUBSET_NAME, S_CKO_USER
    ;
```

Note: REPLACE 'modelname' with your model name.

User IDs Authorized to Encyclopedia:

```
-      DISPLAY NUMBER OF ENCYCLOPEDIA USERS BY USER TYPE
-           NOTES:
-.         A) REPLACE 'encycreator' WITH YOUR ENCYCLOPEDIA DATABASE CREATOR-ID
    SELECT              U USER_TYPE, COUNT (*)
     FROM               encycreator.DUSR
    GROUP BY   U_USER_TYPE
    ;
```

Number of Active Encyclopedia Users:

```
- .      DISPLAY NUMBER OF ACTIVE ENCYCLOPEDIA USERS
- .             NOTES:
- .           A) REPLACE 'encycreator' WITH YOUR ENCYCLOPEDIA DATABASE CREATOR-ID
         SELECT COUNT (DISTINCT HLG_USERID), HLG_MODEL_NAME

         FROM encycreator.DHLOG
          WHERE ((DATE('19'!!SUBSTR (DIGITS (HLG_DATE), 5,2)!!'-'
         !!SUBSTR (DIGITS (HLG_DATE), 7,2) !!'-'!!SUBSTR (DIGITS (HLG_DATE), 9,2))
         >=CURRENT DATE -1 MONTH)) AND
         HLG_MODEL_NAME> '   '
         GROUP BY HLG_MODEL_NAME;
```

Statistics About Encyclopedia Activity

The encyclopedia DHLOG table is a primary reference for details about encyclopedia activity. The information in DHLOG can provide statistics about the types of functions and activities being performed and the number of times they have occurred over a specified time.

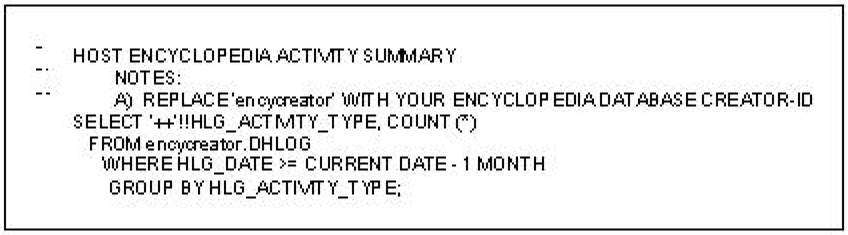
DHLOG provides information about the following encyclopedia activities:

- Upload and check in of models or subsets
- Download and checkout of models or subsets

- Override checkout status of models or subsets
- Code generations
- Model copies
- Migrations and trial migrations
- Adoptions
- Model deletes
- SQL errors encountered
- Failing enqueue waits
- Failing encyclopedia functions

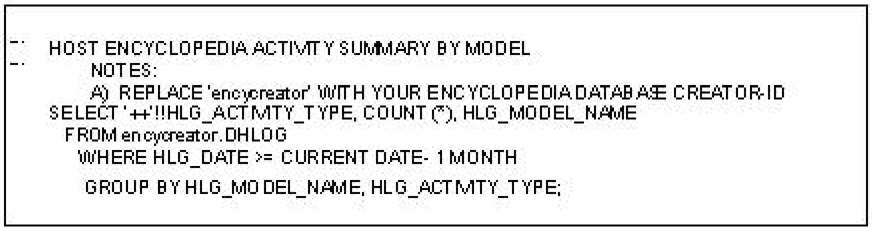
The following illustrations list SQL that gathers statistics from DHLOG. The illustration titles explain the purpose of the SQL or report.

Summary of Encyclopedia activities:



```
-    HOST ENCYCLOPEDIA ACTIVITY SUMMARY
--          NOTES:
--          A) REPLACE 'encycreator' WITH YOUR ENCYCLOPEDIA DATABASE CREATOR-ID
     SELECT '++'!!HLG_ACTIVITY_TYPE, COUNT (*)
       FROM encycreator.DHLOG
         WHERE HLG_DATE >= CURRENT DATE - 1 MONTH
          GROUP BY HLG_ACTIVITY_TYPE;
```

Summary of Encyclopedia activities by model:



```
--   HOST ENCYCLOPEDIA ACTIVITY SUMMARY BY MODEL
--          NOTES:
            A) REPLACE 'encycreator' WITH YOUR ENCYCLOPEDIA DATABASE CREATOR-ID
     SELECT '++'!!HLG_ACTIVITY_TYPE, COUNT (*), HLG_MODEL_NAME
       FROM encycreator.DHLOG
         WHERE HLG_DATE >= CURRENT DATE- 1 MONTH
          GROUP BY HLG_MODEL_NAME, HLG_ACTIVITY_TYPE;
```

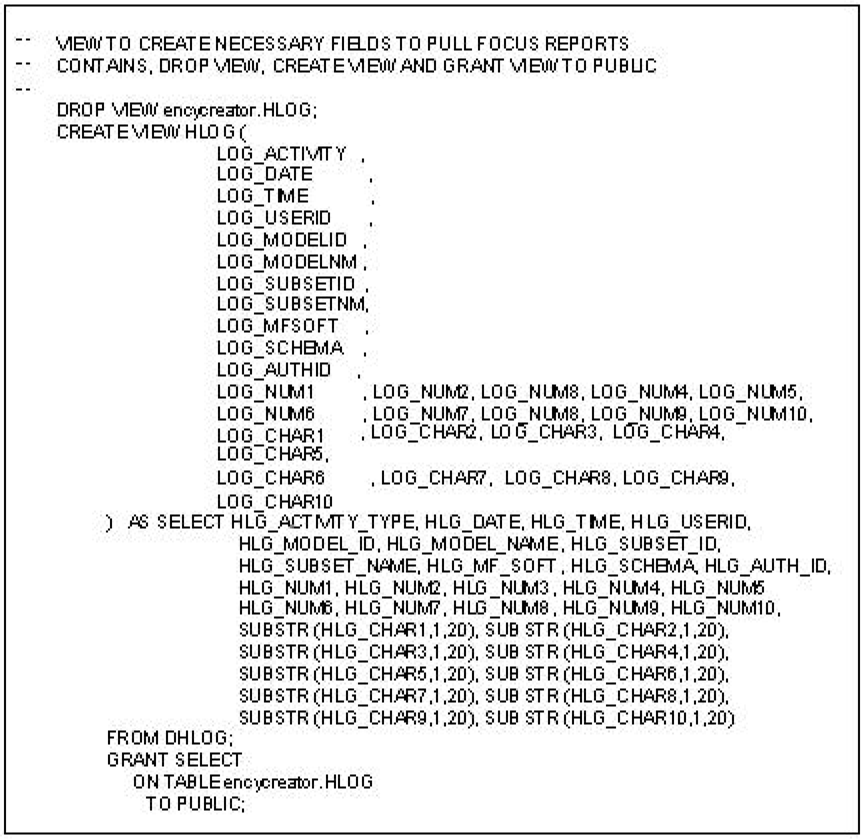
More information:

Process Recommendations (see page 29)

Trend Statistics About Encyclopedia Activity

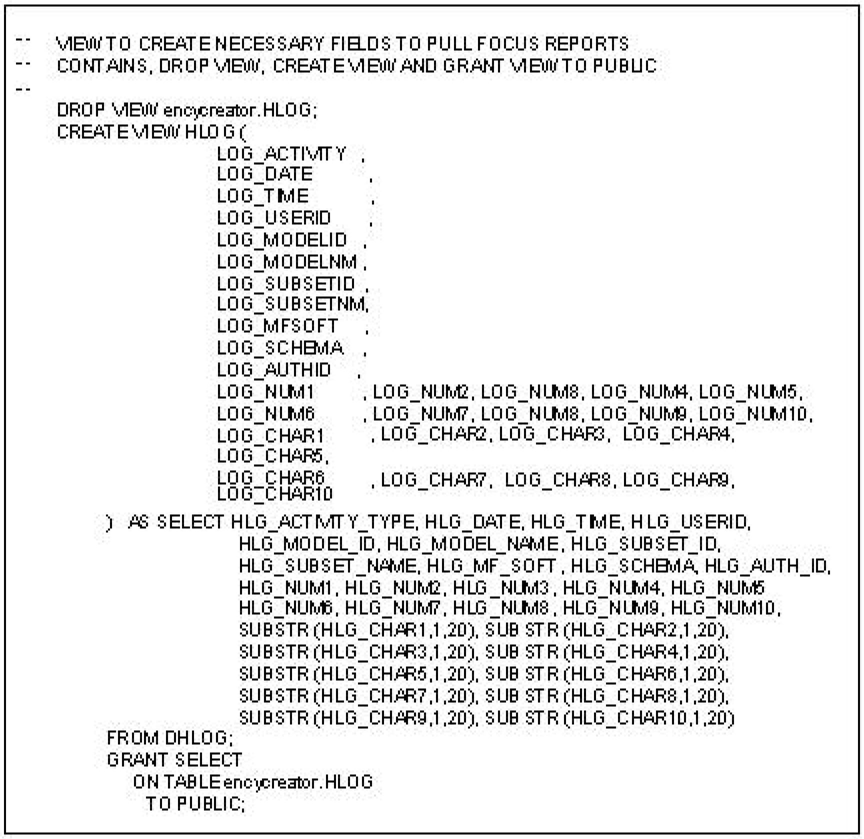
Analyzing trends in encyclopedia activity can be useful to monitor encyclopedia performance. The FOCUS report illustrations list the information required to produce a FOCUS report. The report lists trends in activity over an 8-week period. The illustration titles explain the purpose of the information. As you create the information, replace encycreator with the ID of your encyclopedia database creator.

View of Encyclopedia DHLOG for FOCUS reporting:



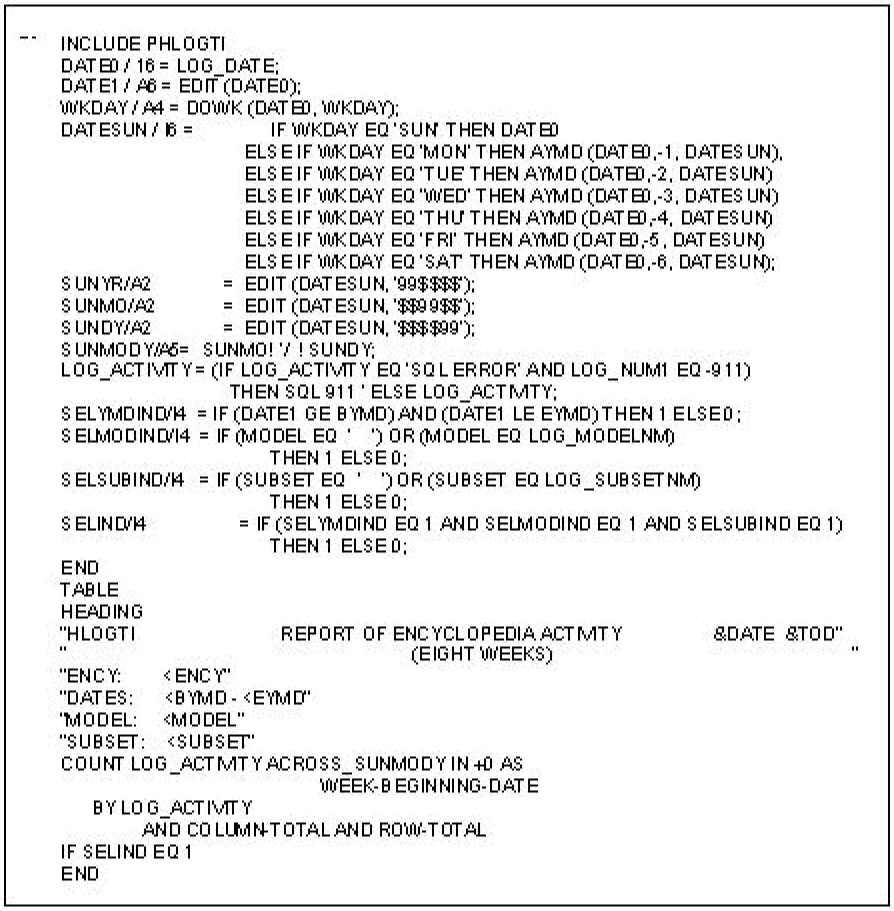
```
--   VIEW TO CREATE NECESSARY FIELDS TO PULL FOCUS REPORTS
--   CONTAINS, DROP VIEW, CREATE VIEW AND GRANT VIEW TO PUBLIC
--
     DROP VIEW encycreator.HLOG;
     CREATE VIEW HLOG (
                         LOG_ACTIVITY  ,
                         LOG_DATE       ,
                         LOG_TIME       ,
                         LOG_USERID     ,
                         LOG_MODELID   ,
                         LOG_MODELNM ,
                         LOG_SUBSETID  ,
                         LOG_SUBSETNM,
                         LOG_MFSOFT     ,
                         LOG_SCHEMA    ,
                         LOG_AUTHID    ,
                         LOG_NUM1       , LOG_NUM2, LOG_NUM8, LOG_NUM4, LOG_NUM5,
                         LOG_NUM6       , LOG_NUM7, LOG_NUM8, LOG_NUM9, LOG_NUM10,
                         LOG_CHAR1     , LOG_CHAR2, LOG_CHAR3, LOG_CHAR4,
                         LOG_CHAR5,
                         LOG_CHAR6       , LOG_CHAR7, LOG_CHAR8, LOG_CHAR9,
                         LOG_CHAR10
          )  AS SELECT HLG_ACTIVITY_TYPE, HLG_DATE, HLG_TIME, HLG_USERID,
                            HLG_MODEL_ID, HLG_MODEL_NAME, HLG_SUBSET_ID,
                            HLG_SUBSET_NAME, HLG_MF_SOFT , HLG_SCHEMA, HLG_AUTH_ID,
                            HLG_NUM1, HLG_NUM2, HLG_NUM3 , HLG_NUM4, HLG_NUM5
                            HLG_NUM6, HLG_NUM7, HLG_NUM8 , HLG_NUM9, HLG_NUM10,
                            SUBSTR (HLG_CHAR1,1,20), SUB STR (HLG_CHAR2,1,20),
                            SUBSTR (HLG_CHAR3,1,20), SUB STR (HLG_CHAR4,1,20),
                            SUBSTR (HLG_CHAR5,1,20), SUB STR (HLG_CHAR6,1,20),
                            SUBSTR (HLG_CHAR7,1,20), SUB STR (HLG_CHAR8,1,20),
                            SUBSTR (HLG_CHAR9,1,20), SUB STR (HLG_CHAR10,1,20)
          FROM DHLOG;
          GRANT SELECT
             ON TABLE encycreator.HLOG
               TO PUBLIC;
```

FOCUS master file description:



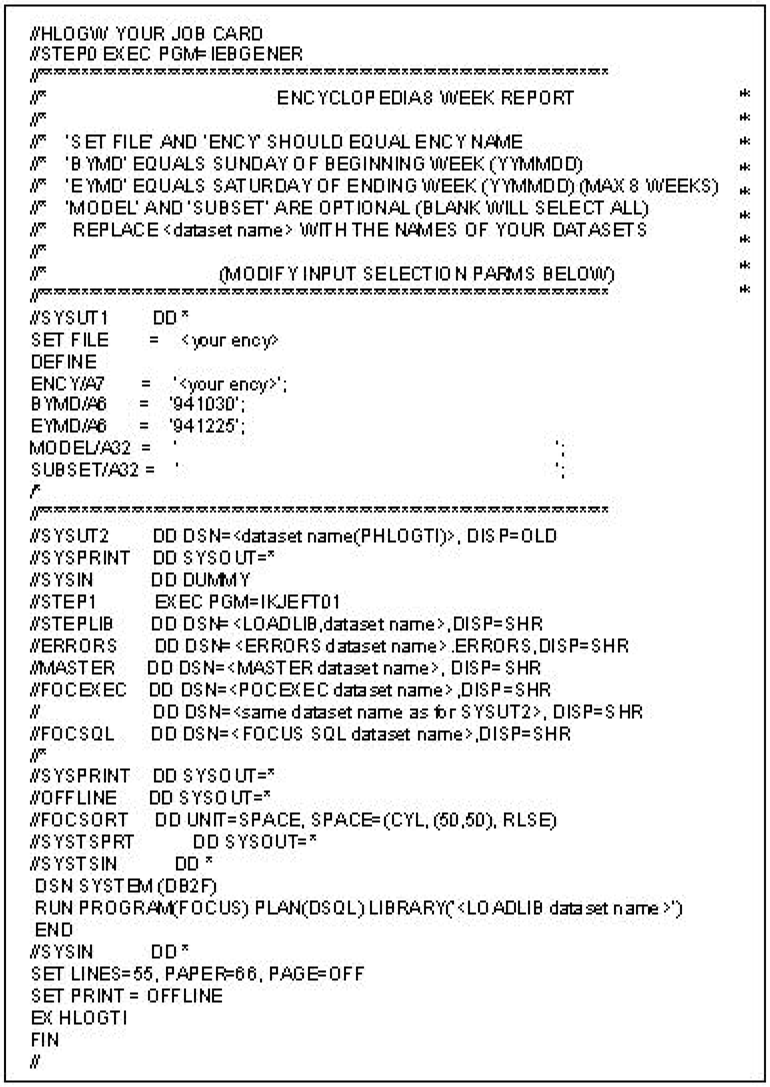
```
--   VIEW TO CREATE NECESSARY FIELDS TO PULL FOCUS REPORTS
--   CONTAINS, DROP VIEW, CREATE VIEW AND GRANT VIEW TO PUBLIC
--
     DROP VIEW encycreator.HLOG;
     CREATE VIEW HLOG (
                       LOG_ACTIVITY  ,
                       LOG_DATE       ,
                       LOG_TIME       ,
                       LOG_USERID     ,
                       LOG_MODELID  ,
                       LOG_MODELNM ,
                       LOG_SUBSETID ,
                       LOG_SUBSETNM,
                       LOG_MFSOFT    ,
                       LOG_SCHEMA   ,
                       LOG_AUTHID    ,
                       LOG_NUM1        , LOG_NUM2, LOG_NUM3, LOG_NUM4, LOG_NUM5,
                       LOG_NUM6        , LOG_NUM7, LOG_NUM8, LOG_NUM9, LOG_NUM10,
                       LOG_CHAR1        , LOG_CHAR2, LOG_CHAR3,  LOG_CHAR4,
                       LOG_CHAR5,
                       LOG_CHAR6        , LOG_CHAR7,  LOG_CHAR8, LOG_CHAR9,
                       LOG_CHAR10
           )  AS SELECT HLG_ACTIVITY_TYPE, HLG_DATE, HLG_TIME, HLG_USERID,
                          HLG_MODEL_ID, HLG_MODEL_NAME, HLG_SUBSET_ID,
                          HLG_SUBSET_NAME, HLG_MF_SOFT , HLG_SCHEMA, HLG_AUTH_ID,
                          HLG_NUM1, HLG_NUM2, HLG_NUM3 , HLG_NUM4, HLG_NUM5
                          HLG_NUM6, HLG_NUM7, HLG_NUM8 , HLG_NUM9, HLG_NUM10,
                          SUBSTR (HLG_CHAR1,1,20), SUB STR (HLG_CHAR2,1,20),
                          SUBSTR (HLG_CHAR3,1,20), SUB STR (HLG_CHAR4,1,20),
                          SUBSTR (HLG_CHAR5,1,20), SUB STR (HLG_CHAR6,1,20),
                          SUBSTR (HLG_CHAR7,1,20), SUB STR (HLG_CHAR8,1,20),
                          SUBSTR (HLG_CHAR9,1,20), SUB STR (HLG_CHAR10,1,20)
           FROM DHLOG;
           GRANT SELECT
               ON TABLE encycreator.HLOG
                 TO PUBLIC;
```

FOCEXEC to create eight-week trend report:



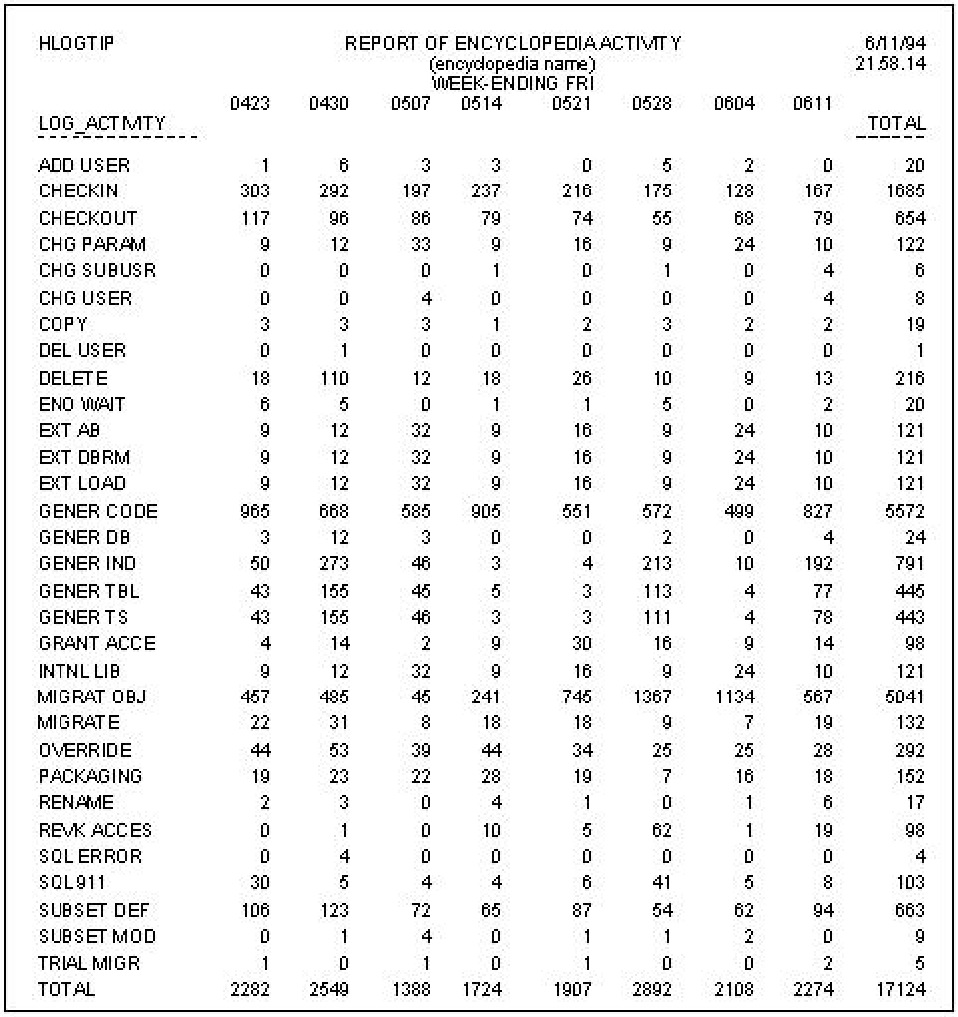
```
--  INCLUDE PHLOGTI
    DATE0 / I6 = LOG_DATE;
    DATE1 / A6 = EDIT (DATE0);
    WKDAY / A4 = DOWK (DATE0, WKDAY);
    DATESUN / I6 =          IF WKDAY EQ 'SUN' THEN DATE0
                          ELSE IF WKDAY EQ 'MON' THEN AYMD (DATE0,-1, DATESUN),
                          ELSE IF WKDAY EQ 'TUE' THEN AYMD (DATE0,-2, DATESUN)
                          ELSE IF WKDAY EQ 'WED' THEN AYMD (DATE0,-3, DATESUN)
                          ELSE IF WKDAY EQ 'THU' THEN AYMD (DATE0,-4, DATESUN)
                          ELSE IF WKDAY EQ 'FRI' THEN AYMD (DATE0,-5 , DATESUN)
                          ELSE IF WKDAY EQ 'SAT' THEN AYMD (DATE0,-6, DATESUN);
    SUNYR/A2          =  EDIT (DATESUN, '99$$$$');
    SUNMO/A2          =  EDIT (DATESUN, '$$99$$');
    SUNDY/A2          =  EDIT (DATESUN, '$$$$99');
    SUNMODY/A5=  SUNMO! '/' ! SUNDY;
    LOG_ACTIVITY = (IF LOG_ACTIVITY EQ 'SQL ERROR' AND LOG_NUM1 EQ -911)
                        THEN SQL 911 ' ELSE LOG_ACTIVITY;
    SELYMDIND/I4  = IF (DATE1 GE BYMD) AND (DATE1 LE EYMD) THEN 1 ELSE 0 ;
    SELMODIND/I4  = IF (MODEL EQ '    ') OR (MODEL EQ LOG_MODELNM)
                            THEN 1 ELSE 0;
    SELSUBIND/I4  = IF (SUBSET EQ '    ') OR (SUBSET EQ LOG_SUBSETNM)
                            THEN 1 ELSE 0;
    SELIND/I4            = IF (SELYMDIND EQ 1 AND SELMODIND EQ 1 AND SELSUBIND EQ 1)
                            THEN 1 ELSE 0;
    END
    TABLE
    HEADING
    "HLOGTI                    REPORT OF ENCYCLOPEDIA ACTIVITY              &DATE  &TOD"
    "                                   (EIGHT WEEKS)                                   "
    "ENCY:      <ENCY"
    "DATES:     <BYMD - <EYMD"
    "MODEL:     <MODEL"
    "SUBSET:    <SUBSET"
    COUNT LOG_ACTIVITY ACROSS _SUNMODY IN +0 AS
                                 WEEK-BEGINNING-DATE
       BY LOG_ACTIVITY
            AND COLUMN-TOTAL AND ROW-TOTAL
    IF SELIND EQ 1
    END
```

JCL to execute the FOCEXEC:



```
//HLOGW YOUR JOB CARD
//STEP0 EXEC PGM=IEBGENER
//**************************************************************
//*                    ENCYCLOPEDIA8 WEEK REPORT                          *
//*                                                                       *
//*   'SET FILE' AND 'ENCY' SHOULD EQUAL ENCY NAME                        *
//*   'BYMD' EQUALS SUNDAY OF BEGINNING WEEK (YYMMDD)                     *
//*   'EYMD' EQUALS SATURDAY OF ENDING WEEK (YYMMDD) (MAX 8 WEEKS)        *
//*   'MODEL' AND 'SUBSET' ARE OPTIONAL (BLANK WILL SELECT ALL)           *
//*    REPLACE <dataset name> WITH THE NAMES OF YOUR DATASETS             *
//*                                                                       *
//*                  (MODIFY INPUT SELECTION PARMS BELOW)                 *
//**************************************************************          *
//SYSUT1       DD *
SET FILE      =    <your ency>
DEFINE
ENCY/A7       =    '<your ency>';
BYMD/A6       =   '941030';
EYMD/A6       =   '941225';
MODEL/A32  =    '                                   ';
SUBSET/A32 =    '                                   ';
/*
//**************************************************************
//SYSUT2       DD DSN=<dataset name(PHLOGTI)>, DISP=OLD
//SYSPRINT     DD SYSOUT=*
//SYSIN        DD DUMMY
//STEP1         EXEC PGM=IKJEFT01
//STEPLIB      DD DSN= <LOADLIB,dataset name>,DISP=SHR
//ERRORS        DD DSN= <ERRORS dataset name>.ERRORS,DISP=SHR
//MASTER      DD DSN=<MASTER dataset name>, DISP= SHR
//FOCEXEC     DD DSN=<POCEXEC dataset name> ,DISP=SHR
//             DD DSN=<same dataset name as for SYSUT2>, DISP=SHR
//FOCSQL       DD DSN=< FOCUS SQL dataset name>,DISP=SHR
//*
//SYSPRINT     DD SYSOUT=*
//OFFLINE      DD SYSOUT=*
//FOCSORT       DD UNIT=SPACE, SPACE=(CYL, (50,50), RLSE)
//SYSTSPRT           DD SYSOUT=*
//SYSTSIN          DD *
 DSN SYSTEM (DB2F)
 RUN PROGRAM(FOCUS) PLAN(DSQL) LIBRARY('<LOADLIB dataset name>')
 END
//SYSIN        DD *
SET LINES=55, PAPER=66, PAGE=OFF
SET PRINT = OFFLINE
EX HLOGTI
FIN
//
```

Sample output of FOCEXEC:



```
HLOGTIP                        REPORT OF ENCYCLOPEDIA ACTIVITY                      6/11/94
                                     (encyclopedia name)                            21.58.14
                                       WEEK-ENDING FRI
                  0423    0430    0507    0514    0521    0528    0604    0611
LOG_ACTIVITY                                                                          TOTAL
--------------                                                                        -------

ADD USER             1       6       3       3       0       5       2       0           20
CHECKIN            303     292     197     237     216     175     128     167         1685
CHECKOUT           117      96      86      79      74      55      68      79          654
CHG PARAM            9      12      33       9      16       9      24      10          122
CHG SUBUSR           0       0       0       1       0       1       0       4            6
CHG USER             0       0       4       0       0       0       0       4            8
COPY                 3       3       3       1       2       3       2       2           19
DEL USER             0       1       0       0       0       0       0       0            1
DELETE              18     110      12      18      26      10       9      13          216
ENQ WAIT             6       5       0       1       1       5       0       2           20
EXT AB               9      12      32       9      16       9      24      10          121
EXT DBRM             9      12      32       9      16       9      24      10          121
EXT LOAD             9      12      32       9      16       9      24      10          121
GENER CODE         965     668     585     905     551     572     499     827         5572
GENER DB             3      12       3       0       0       2       0       4           24
GENER IND           50     273      46       3       4     213      10     192          791
GENER TBL           43     155      45       5       3     113       4      77          445
GENER TS            43     155      46       3       3     111       4      78          443
GRANT ACCE           4      14       2       9      30      16       9      14           98
INTNL LIB            9      12      32       9      16       9      24      10          121
MIGRAT OBJ         457     485      45     241     745    1367    1134     567         5041
MIGRATE             22      31       8      18      18       9       7      19          132
OVERRIDE            44      53      39      44      34      25      25      28          292
PACKAGING           19      23      22      28      19       7      16      18          152
RENAME               2       3       0       4       1       0       1       6           17
REVK ACCES           0       1       0      10       5      62       1      19           98
SQL ERROR            0       4       0       0       0       0       0       0            4
SQL911              30       5       4       4       6      41       5       8          103
SUBSET DEF         106     123      72      65      87      54      62      94          663
SUBSET MOD           0       1       4       0       1       1       2       0            9
TRIAL MIGR           1       0       1       0       1       0       0       2            5
TOTAL             2282    2549    1388    1724    1907    2892    2108    2274        17124
```

The trend report is useful to see if your users are following your project guidelines (such as the recommendations discussed in the Process Recommendations (see page 29) chapter). For example, if the report shows that the number of uploads and downloads are equal, you know that users are not frequently uploading without check in. You can obtain other valuable information from the report also. For example, you can determine a trend of encyclopedia contention as a percent of total activity. You determine this by adding the values for SQL 911 and ENQ WAIT.

Statistics About Encyclopedia Contention

DB2 protects data through locking mechanisms. The monitoring of contention caused by timeouts and deadlocks is important to overall encyclopedia use. Often, as deadlines approach for a particular project, timeouts and deadlocks increase as the work related to a model increases. When many users attempt to funnel work to one place, some users experience timeouts or deadlocks.

DB2 provides some features to assist with monitoring contention.

Timeouts

Each DB2 timeout is reported in the system console log under message ID DSNT376I. If you have the ability to view your system console log, you can determine the users, plans, and resource involved in the timeout situation. Some interactive performance monitors can also provide the same type of information. You may prefer to use a routine to run against a copy of the console log and parse all of the DSNT376I messages found. The routine could run periodically, for example, every 24 hours, and report on that time period. You may want to sort the resulting information by database name to present a report usable by database administrators.

If your performance monitors provide timeout reports, determine what action is required to have the necessary data available. If the requirement is to run the DB2 Performance Trace 6, investigate how much overhead this will add to your DB2 subsystem. The cost may be worthwhile for severe timeout problems, especially if you run the trace only for a short period.

Deadlocks

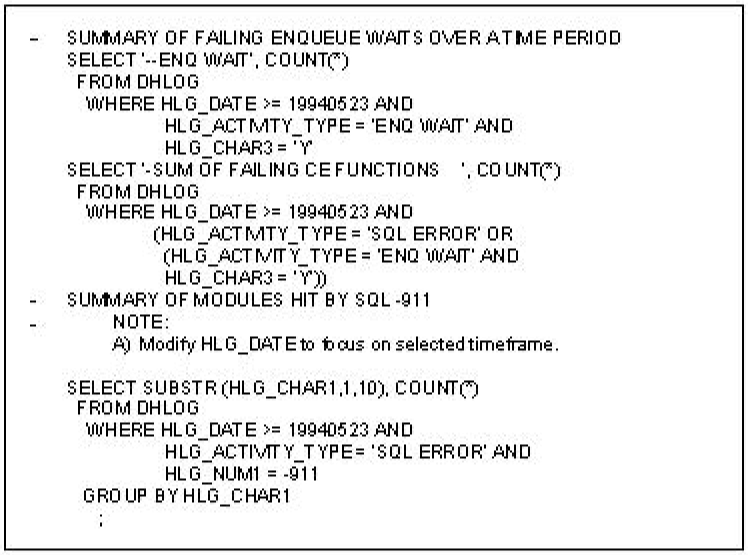
Deadlocks are usually difficult to solve because only one side of the resource conflict is presented by the DB2 deadlock message, DSNT375I. However, a snapshot of the full deadlock resource conflict is available if you are running statistics class 3 traces. These traces reportedly require very little overhead to remain active and are very useful for analyzing deadlock problems.

If the trace is active, a record is placed into the DB2 trace files (usually SMF files) for each deadlock. The record contains the user IDs, plans, and both resources involved in the conflict. An example of the type of output from a performance monitor is provided in the documentation for DB2PMt. Most performance monitors have similar reports that outline both sides of the conflict.

Enqueue Waits and -911 Errors

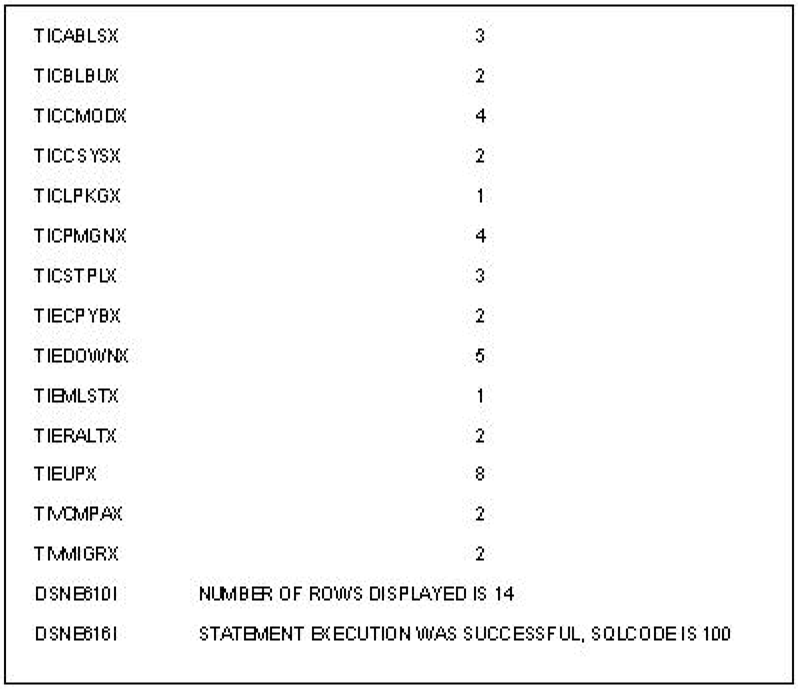
The Waits and Errors illustrations list SQL to determine the amount of enqueue waits and -911 errors. The lead-ins to the illustrations explain the purpose of the SQL or output.

Summary of enqueue waits:



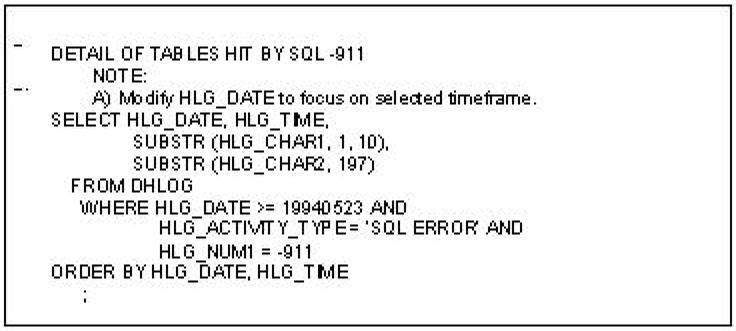
```
-     SUMMARY OF FAILING ENQUEUE WAITS OVER A TIME PERIOD
      SELECT '--ENQ WAIT', COUNT(*)
       FROM DHLOG
        WHERE HLG_DATE >= 19940523 AND
                HLG_ACTIVITY_TYPE = 'ENQ WAIT' AND
                HLG_CHAR3 = 'Y'
      SELECT '-SUM OF FAILING CE FUNCTIONS    ', COUNT(*)
       FROM DHLOG
        WHERE HLG_DATE >= 19940523 AND
               (HLG_ACTIVITY_TYPE = 'SQL ERROR' OR
                (HLG_ACTIVITY_TYPE = 'ENQ WAIT' AND
                HLG_CHAR3 = 'Y'))
-     SUMMARY OF MODULES HIT BY SQL -911
-           NOTE:
            A) Modify HLG_DATE to focus on selected timeframe.

      SELECT SUBSTR(HLG_CHAR1,1,10), COUNT(*)
       FROM DHLOG
        WHERE HLG_DATE >= 19940523 AND
                HLG_ACTIVITY_TYPE= 'SQL ERROR' AND
                HLG_NUM1 = -911
        GROUP BY HLG_CHAR1
         ;
```

Sample output of enqueue waits:



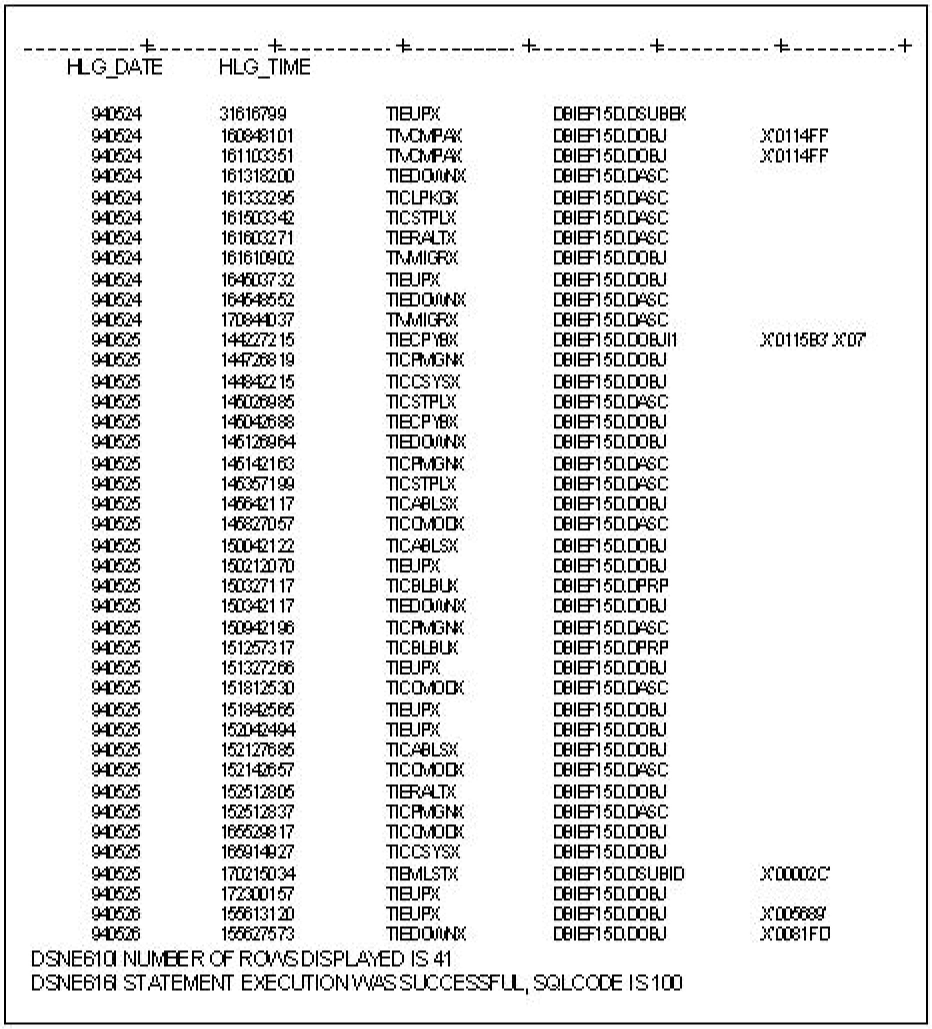
```
TICABLSX                              3
TICBLBUX                              2
TICCMODX                              4
TICCSYSX                              2
TICLPKGX                              1
TICPMGNX                              4
TICSTPLX                              3
TIECPYBX                              2
TIEDOWNX                              5
TIEMLSTX                              1
TIERALTX                              2
TIEUPX                                8
TMCMPAX                               2
TMMIGRX                               2
DSNE610I        NUMBER OF ROWS DISPLAYED IS 14
DSNE616I        STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
```

Tables experiencing SQL -911 errors:



```
-
  DETAIL OF TABLES HIT BY SQL -911
        NOTE:
-.      A) Modify HLG_DATE to focus on selected timeframe.
  SELECT HLG_DATE, HLG_TIME,
              SUBSTR (HLG_CHAR1, 1, 10),
              SUBSTR (HLG_CHAR2, 197)
     FROM DHLOG
      WHERE HLG_DATE >= 19940523 AND
                  HLG_ACTIVITY_TYPE = 'SQL ERROR' AND
                  HLG_NUM1 = -911
  ORDER BY HLG_DATE, HLG_TIME
       ;
```

Sample output of tables experiencing SQL -911 errors:



```
----------+---------+---------+---------+---------+---------+---------+
    HLG_DATE      HLG_TIME

      940524      31616799      TIEUPX       DBIEF15D.DSUBEK
      940524      160848101     TMCMPAX      DBIEF15D.DOBJ        X'0114FF'
      940524      161103351     TMCMPAX      DBIEF15D.DOBJ        X'0114FF'
      940524      161318200     TIEDOWNX     DBIEF15D.DASC
      940524      161333295     TICLPKGX     DBIEF15D.DASC
      940524      161503342     TICSTPLX     DBIEF15D.DASC
      940524      161603271     TIERALTX     DBIEF15D.DASC
      940524      161610902     TMMIGRX      DBIEF15D.DOBJ
      940524      164603732     TIEUPX       DBIEF15D.DOBJ
      940524      164648552     TIEDOWNX     DBIEF15D.DASC
      940524      170844037     TMMIGRX      DBIEF15D.DASC
      940525      144227215     TIECPYBX     DBIEF15D.DOBJI1      X'0115B3' X'07'
      940525      144726819     TICPMGNX     DBIEF15D.DOBJ
      940525      144842215     TICCSYSX     DBIEF15D.DOBJ
      940525      146026985     TICSTPLX     DBIEF15D.DASC
      940525      146042688     TIECPYBX     DBIEF15D.DOBJ
      940525      146126964     TIEDOWNX     DBIEF15D.DOBJ
      940525      146142163     TICPMGNX     DBIEF15D.DASC
      940525      146357199     TICSTPLX     DBIEF15D.DASC
      940525      146642117     TICABLSX     DBIEF15D.DOBJ
      940525      146827057     TICOMODX     DBIEF15D.DASC
      940525      150042122     TICABLSX     DBIEF15D.DOBJ
      940525      150212070     TIEUPX       DBIEF15D.DOBJ
      940525      150327117     TICBLBUX     DBIEF15D.DPRP
      940525      150342117     TIEDOWNX     DBIEF15D.DOBJ
      940525      150942196     TICPMGNX     DBIEF15D.DASC
      940525      151257317     TICBLBUX     DBIEF15D.DPRP
      940525      151327266     TIEUPX       DBIEF15D.DOBJ
      940525      151812530     TICOMODX     DBIEF15D.DASC
      940525      151842565     TIEUPX       DBIEF15D.DOBJ
      940525      152042494     TIEUPX       DBIEF15D.DOBJ
      940525      152127685     TICABLSX     DBIEF15D.DOBJ
      940525      152142657     TICOMODX     DBIEF15D.DASC
      940525      152512805     TIERALTX     DBIEF15D.DOBJ
      940525      152512837     TICPMGNX     DBIEF15D.DASC
      940525      165529817     TICOMODX     DBIEF15D.DOBJ
      940525      165914927     TICCSYSX     DBIEF15D.DOBJ
      940525      170215034     TIEMLSTX     DBIEF15D.DSUBID      X'00002C'
      940525      172300157     TIEUPX       DBIEF15D.DOBJ
      940526      155613120     TIEUPX       DBIEF15D.DOBJ        X'005689'
      940526      155627573     TIEDOWNX     DBIEF15D.DOBJ        X'0081FD'
  DSNE610I NUMBER OF ROWS DISPLAYED IS 41
  DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
```

Chapter 2: Maintaining the Host Encyclopedia

CA Gen is a production system. The source code and data it produces must be protected from accidental loss through a regularly scheduled maintenance routine. This routine also optimizes the Encyclopedia's usage of system resources. The aspects of Encyclopedia maintenance discussed in this chapter are:

- The recommended maintenance schedule
- How to delete "orphan" objects
- How to capture an image-copy
- How to expand tablespaces and indexes
- How to improve performance

This section contains the following topics:

Recommended Maintenance Schedule

The recommended daily, weekly, and monthly maintenance schedules for the Host Encyclopedia are described below, along with explanations of the sample jobs you can use to perform the maintenance.

Daily

We recommend an incremental image-copy of all CA Gen Data and Public Interface tablespaces each day. The job CEUINCR (or the alternative job CAUINCR) performs the incremental image-copy.

Weekly

We recommend performing the following weekly maintenance tasks in the order listed:

1. Delete orphan objects from the Host Encyclopedia tables (see the How to Delete Orphan Objects section in this chapter).

2. Capture a full image-copy. A full image-copy before the REORG improves the chances for recovery in the event that the REORG fails and cannot be restarted (see the How to Capture an Image-Copy (see page 27) section in this chapter).

3. Reorganize the DB2 database. The job CEUREOR is provided for you to reorganize your Host Encyclopedia using IBM's DB2 REORG utility. The job CAUREOR is also provided for you to perform a DB2 REORG using CA's DB2 Rapid Reorg utility.

   Depending on the activity level your Encyclopedia experiences, you may want to reorganize some tables, such as DSUBEX, more frequently than weekly.

   To provide a point for consistent recovery (a LOG/RBA), perform a tablespace QUIESCE at the beginning of the REORG step, or start the tablespace in Utility status before starting the REORG.

4. Capture another full image-copy. The image-copy after REORG is required because the tablespaces are in copy-pending state. (CEUREOR and CAUREOR specify LOG NO to eliminate DB2 logging of the REORG.)

How to Delete Orphan Objects

When certain Host Encyclopedia functions (generating, copying, deleting or uploading a model) are unsuccessful, “orphan” objects may be left in the Encyclopedia tables. An orphan object is one that references a model whose ID is not in the DMDL table.

You can use the CEUCLEN job to delete, or “clean” orphan objects from the Encyclopedia. Before executing CEUCLEN, you must set its parameters as shown in the following table:

CEUCLEN Parameter	Required Setting
MODELID	Zero This setting **must** be zero. If you enter a valid MODELID, the model and all its contents are deleted.
SUBSETID	Zero
MODLNAME	Blank
SUBSET	All
TONAME	Blank

CEUCLEN Parameter	Required Setting
SDFCPFLG	Blank
LOCKTBLS	This parameter is not specified, and defaults to N. If you are an Encyclopedia administrator, you can specify the LOCKTBLS parameter and set it to Y to lock users out while the clean-up routine is running. This improves the routine's performance.
MIGRFLAG	Blank
FDELFLAG	N

If you run CEUCLEN in batch mode with the parameters set as defined above, it deletes all rows in the DOBJ table belonging to an orphan object as well as any associated rows in the DASC, DPRP, DSUBID, DSUBDF, DSUBUS, DSUBEX, and DTXT tables.

If extensive Encyclopedia cleanup is required, CEUCLEN may run for a long time. Periodic COMMITS allow the job to be restarted. The job starts from the last COMMIT point.

You may find it useful to run the CEUCLEN job nightly, using a small time parameter (for example, 60 minutes). CEUCLEN performs frequent DB2 commits and checkpoints. If the job times out, the following day's run is restarted from the last checkpoint.

How to Capture an Image-Copy

When capturing an image-copy of databases, it is important to establish a point at which the entire set of tables is consistent. Use one of the two following procedures to establish that point of consistency and to perform the image-copy.

Procedure 1

1. Stop the databases then start them in Utility status. (To ensure a point of consistent recovery, no updates are allowed during image-copy.)
2. Run CEUCOPY (or alternatively CAUCOPY) to execute the DB2 COPY utility to take a full image-copy of all Encyclopedia tablespaces.
3. Start the databases in Read/Write to resume normal processing.

Procedure 2

1. Run CEUCOPY (or alternatively CAUCOPY) in Read/Write status.
2. Use the DB2 QUIESCE utility to establish a point for consistent recovery (a LOG/RBA).

How to Expand Tablespaces and Indexes

You may need to expand a tablespace or index to accommodate a growing model. If you are receiving the following errors, the system is telling you to expand tablespaces and indexes:

```
DB2 error - 904, reason code X'00D70027
```

The job CEUXPND expands tablespaces and indexes. When you run CEUXPND, it does the following:

Follow these steps:

1. Stops the tablespace/indexspace to be expanded.
2. Deletes any existing temporary backup image-copy dataset.
3. Creates a temporary DSN1COPY of table/indexspace.
4. Deletes the VSAM cluster for the table/indexspace.
5. Reallocates the table/indexspace.
6. Restores table/indexspace from temporary DSN1COPy copy.
7. Restarts the table/indexspace.

CEUXPND assumes that Storage Groups are not in use. If Storage groups are in use, use a DB2 ALTER command and a REORG to resize.

Chapter 3: Process Recommendations

This chapter explains the ways to improve the performance of Host Encyclopedia.

Ways to Improve Processes

You can improve the performance of the Host Encyclopedia by implementing processes that help you use the encyclopedia more efficiently, reduce contention among jobs, and decrease the cost associated with failures and poor planning. These process improvements span a variety of areas from scheduling to communications, and aid in the general coordination of development projects.

In many cases, you can use general techniques that do not address specific areas of concern. These techniques are good practices and help increase overall performance by decreasing the time and cost associated with non-successful attempts. The following topics fall under this category of general techniques:

- Understanding what causes contention
- Avoiding known contention
- Making the most of peak and non-peak processing
- Creating subject matter experts
- Using reports to facilitate planning
- Managing and coordinating projects

In other cases, you need to identify and understand the functions that incur the most cost, usage, and contention.

More information:

Specific Encyclopedia Functions (see page 39)

Causes of Contention

The first step in reducing contention is to understand what causes it. Contention can be caused by:

- DB2 timeouts and deadlocks
- Encyclopedia-enforced enqueue waits

DB2 Timeouts and Deadlocks

DB2 timeouts or deadlocks occur when two users attempt to access a DB2 page or table that is locked. Data for different models can be stored on the same DB2 page. Because of this, any two users, regardless of the models being accessed or updated, can experience timeouts and deadlocks. Even though it is not easily predictable, certain encyclopedia functions have a tendency to cause timeouts and deadlocks. These functions are:

- Model deletion
- Model backup and restore

These functions, especially when run on large models, perform thousands of DB2 inserts, updates, and deletes to key encyclopedia tables such as DOBJ, DASC and DPRP. This amount of activity increases the chance of another user attempting to access the same page of a table or index.

Another group of encyclopedia functions have a tendency to cause timeouts and deadlocks because they are not capable of taking interim commits. Because these functions must force a complete rollback if they fail, they cannot take an interim commit, which causes the DB2 page locks to escalate to tablespace locks when run on a large amount of data. These functions are:

- Migration
- Adoption

Encyclopedia Enqueue Waits

The second kind of contention is caused by encyclopedia constraints known as enqueue waits. Enqueue waits are constraints built into the encyclopedia to prohibit two functions that are known to be incompatible from running at the same time. Because enqueue waits can be predicted, it is worthwhile to learn the rules that govern them.

Encyclopedia Concurrency Matrix

The encyclopedia concurrency matrix is the table of rules that determine which functions are compatible to run at the same time. An enqueue wait occurs when a second process is started that violates the rules of the concurrency matrix. The second process is put into wait mode and retried every minute to see if the enqueue still exist. After a specified number of retries per function, the second process terminates (it never really got started) with a resource conflict message. An enqueue wait occurs only at the model and subset level (there is no encyclopedia-wide enqueue contention).

Some of the more common restrictions are:

- A download and an upload cannot run in the same model at the same time.
- No uploads or downloads are allowed in a model while a migration into that model is running.

Encyclopedia users should be aware of the concurrency matrix and know how to use it.

Note: For more information, see *CA Gen Host Encyclopedia Subsetting User Guide.*

Note that the matrix shows only the concurrency enforced by the Host Encyclopedia. Other types of contention, such as that caused by DB2, cannot be determined from the matrix.

Avoiding Contention

You can start to avoid contention in the following ways:

- Know what is currently running.
- Know what is scheduled to run.
- Communicate what is going to be run.

What Is Currently Running

Before you perform an encyclopedia function that may cause contention, see which functions are currently running. You can use several methods to determine what is running. (The methods vary greatly from site to site.) One method is to check DB2 threads; another is to use a performance monitor.

To match the plans with the encyclopedia functions, you need to know the plan suffixes and the plan prefix used for that encyclopedia. See the Encyclopedia Function Plans (see page 77) appendix for a list of all the function plans and a description of each plan. You will probably encounter the following plan suffixes as you work on improving encyclopedia performance:

- **ADPT**-Adopt
- **CKST**-Override checkout status
- **CPYB**-Model copy
- **DSUB**-Model delete
- **DOWN**-Download
- **MIGR**-Migration
- **U07**-Expansion conflict report
- **UP**-Upload

After you identify what is currently running, use the concurrency matrix to determine if the function you want to run is allowed at the same time.

Note: For more information, see Encyclopedia Concurrency Matrix in *CA Gen Host Encyclopedia Subsetting User Guide*.

Since the concurrency matrix deals with activity in the same model, you need to know what model the active function is running against. This is not always easy to find out. You have at least two ways to deal with this:

- Run the jobs in batch whenever possible by using the job name to designate the model affected. Using the same job name within a model sequentially runs the jobs, which avoids contention between the jobs.
- Build a cross-reference list of TSO IDs to models. If a user works on more than one model, assign multiple IDs, each with access to only one model.

What Is Scheduled to Run

By scheduling activities, organizations can give critical tasks higher priority and reduce resource contention. Jobs that affect the entire encyclopedia, such as image copy and reorganization, should be scheduled to run at a consistent time that is communicated to all encyclopedia users. Users should understand that the encyclopedia is unavailable during this time and that all encyclopedia jobs must finish before the scheduled start time.

Schedule encyclopedia jobs that you believe may cause contention with other encyclopedia activity, such as large model copies, deletes, and migrates, during a period when other ongoing activity is curtailed. These time periods are best scheduled during non-peak hours and must be communicated and agreed upon by all encyclopedia users. All users should have access to the schedule.

For example, suppose that a user needs to copy a large model. The user knows from experience that this copy may cause DB2 contention because of the amount of inserts performed. The encyclopedia administrators could set up time periods, such as on Tuesday and Thursday evenings starting at 11:00 pm, for this type of activity. The jobs could be run in batch mode and submitted using the same jobname (single streaming the processing). The user schedules the copy to run during this time period. This increases the likelihood of having the job complete and eliminates the chance of other jobs failing.

Communicate What Is Going to Run

Activity that has a likelihood of causing contention within a model should be communicated to everyone working in that model. Advanced warning allows others to prepare and schedule their activities to avoid contention. It also allows users to respond if the activity being scheduled causes hardship on other priority activities.

For example, if 10 users are actively working in a model and one user needs to run a download, no other user can perform an upload during the download. The user should send a group message to the other users in that model notifying them of the time of the download. The other users can then plan their activities around the download and avoid the cost and frustration of a failed upload.

A variety of successful methods exist for communicating information:

- Electronic mail
- Electronic forms
- Broadcast bulletins
- Voice mail

For each of these means, if possible, set up group IDs for projects, models and encyclopedias.

Peak and Non-Peak Processing Times

Some encyclopedia functions can run during peak processing hours and other functions should run during non-peak hours. You get better at deciding the appropriate times by understanding the concurrency matrix, reviewing performance reports, and continuing to work with the encyclopedia (which increases your experience).

Note: For more information, see *CA Gen Host Encyclopedia Subsetting User* Guide.

Peak Processing

Encyclopedia functions that can run together within the same model are easily run during peak processing hours. A few examples of these types of processes are code generation, testing, reporting, and defining subsets.

Processes that cannot run concurrently, but that are needed during peak hours, can be single streamed by running them in batch using the same job names. The best example of this is upload and download.

Other encyclopedia functions that have a tendency to cause contention should be broken into small jobs (where applicable). For Example, migration and adoption. Both can cause contention when they do not complete in a short time and both can be broken into small pieces. Running 10 migrates that each complete in 10 minutes causes much less contention than one migrate that runs for 100 minutes.

Non-Peak Processing

Some encyclopedia functions that are long running and have a tendency to cause DB2 contention on the encyclopedia tables are best scheduled to run in a single stream mode during non-peak processing hours. Setting up a time period for single stream batch jobs several nights a week can reduce the coordination of getting these jobs done. You can single stream jobs by using a common job name or by setting up a special job class with a single initiator. If you change the locksize on the encyclopedia tables during the period of single streaming from ANY to TABLESPACE, you can additionally increase the throughput of processes such as model copies and model deletes. A job to alter the tablespaces back to a locksize of ANY is needed before normal jobs can begin to process.

Most downloads can be planned. Encyclopedia users can take advantage on non-peak processing charges and decrease the failure rate due to contention by submitting downloads to run overnight. Encyclopedia administrators can set aside time periods during the night just for downloads. Each model can be associated with a jobname and all downloads for that model can be submitted to run under that job name during the time period.

Subject Matter Experts

Training is a very effective way to improve performance. The more information the developers have about their requirements and how all parts of the system interact, the greater their success. Although, it is not always feasible to train all team members in every area, try to have a few individual team members trained in specialized areas. These individuals can share their knowledge with other team members and improve the performance of the entire team.

Two areas in which training can be very effective are subsetting and migration.

Subsetting Expert

An improperly defined subset results in too many or too few objects to complete a task. Users who do not have all objects needed to complete a task have a tendency to over define a subset and get more than is needed. Over defining a subset has a ripple effect of causing other users to get downgraded. The net effect is that subsets must be overridden, redefined and downloaded. This incurs unproductive time and additional cost.

You can usually improve subsetting practices by training one project member for the role of subsetting expert. This person should review each subset for the correct scoping objects, expansion, and protection needed for the task. Each team member should be taught to use the subsetting documentation. These subsets should then be reviewed with the subsetting expert. The subsetting expert should explain any requested changes, so that each developer can gain a better understanding of the subsetting rules.

As team members become better at subsetting, the number of subsets that are incomplete or too large decreases. Well-scoped subsets reduce processing time and improve performance. The number of unnecessary objects included in each subset decreases, as does the number of multiple subsets being downloaded for a single task.

Migration Expert

A migration may fail because of several reasons. Failed migrations take more than twice as long as successful migrations because they result in a DB2 rollback. Trial migrations are equivalent to a failed migration because they also result in a DB2 rollback. You want to reduce the failure rate and number of trial migrates because migrations may cause DB2 contention on the entire encyclopedia. You can do at least three things to eliminate or reduce the number of unsuccessful migrations:

- Understand the rules of migration. For each object, know its enabling and companion object.
- Keep migration aggregate sets small to avoid repeat migration attempts of many objects if one object fails.
- Use reports, such as those discussed in the following section, to understand the model and objects to be migrated.

Each project should have a migration expert who is thoroughly trained in the rules and best practices of migration. This prevents a large number of failed migrations and reduces or eliminates the need for trial migrations. The migration expert should implement a request process to aid in the scheduling of migrations. The process should prevent the need for large migrations by allowing for incremental movement of changed objects.

Reports

Instead of running jobs such as trial migrates and expansion conflict reports, use other reports that can produce the same or similar results. You can run a combination of reports during peak hours that are less costly and resource intensive to help plan for successful jobs such as migrates, downloads, and code generations.

To Determine Objects Needing Migrating

A combination of a When Changed report and a Compare report can provide a complete list of objects needing migration. Run a When Changed report for the development interval on the source model. Use the resulting aggregate set in a Compare report with the destination model. You can then ensure that all changes are tracked and included in the migration plans.

To Determine Parent Functions and Processes

A common cause of failed migrations is attempting to migrate a function or process to a model that does not contain the parent objects in the function hierarchy. You can use the Function Hierarchy report to identify all parent functions and processes. Those, which are not contained in the destination model, may be included in the first migration attempt to prevent the need for a second attempt or for a trial migrate.

To Determine Load Module Expansion

The Model Action Block Use Report is a useful tool for all analysts and designers. It lists the calling hierarchy of all action blocks implemented in each load module for a model. It also displays the Dynamic Link attribute associated with each module.

The Dynamic Link attributes associated with each module are:

- Default
- Yes
- No
- Compatibility

The report has three basic modes of operation based on the object range. They are:

- All business systems and load modules
- Selected objects
- Filter objects based on Dynamic Link option

Developers can use this report to find information concerning load module expansion. You can place this report in a common location so that it is available electronically.

To Determine Expansion Conflicts

Developers performing subsetting frequently receive Modify or greater protection on objects included in their subsets as neighborhood objects. This can cause other team members to be downgraded when requesting one of the neighborhood objects for their development work. A common work-around to this problem is to override the downgraded subset, have the first developer upload, allow the second developer to download his/her subset, then have the first developer download his/her subset again. This involves many unnecessary uploads and downloads. This situation can be avoided by running a References report for scoped action diagrams before downloads.

This report lists the "called" action diagrams (which may be included as neighborhood objects depending on the expansion used). The developer can then explicitly scope on these objects for Access or Read Only to prevent future conflicts with other development team members. This also reduces the need to run costly expansion conflict reports.

To View Action Diagram Listings

Developers often use action diagram listings to understand and diagnose existing code. The action diagram listing is at the bottom of the generated source code. Instead of running a report that reads the encyclopedia tables, users can browse the bottom of the source listing and eliminate the need for any encyclopedia access.

Project Planning and Coordination

Good project planning is perhaps the most essential element to improving encyclopedia processes. Understanding and implementing good project planning practices is crucial to reduce overlaps when subsetting, migrating, creating project models, and so forth.

Naming Standards

Naming standards can help identify objects that are obsolete or temporary and reduce object redundancy. Without good naming standards, encyclopedia users create redundant objects or use obsolete ones. When this happens, the users often must re-subset to delete the object usage and include the correct object.

Standard naming conventions can also be used to communicate the purpose or status of a shared object. By having a naming convention, an object list search is quicker and more likely to succeed. A common reason for object searches is to locate an action diagram or exit states to include in new code. In addition to a naming convention for shared objects, a good description further indicates whether the object is suitable for the developer. Good in-line comments communicate how the action block should be implemented.

Another use for naming conventions is to tag obsolete objects. If an obsolete object cannot be deleted (because it is referenced in some way), the object can be renamed so that it's obsolete status is communicated. One possibility is to rename it so that it begins with "ZZ_OBS_" and follow that with as much of its original name as possible.

This naming scheme groups all obsolete objects at the end of an object list. You can easily find them and the name clearly indicates to the users to remove them from their code and subsets. This in turn makes it easier to eventually delete the objects and reduce the size of the model.

A naming convention for temporary objects is also helpful in locating and removing unnecessary objects. One possibility is to start the name with "ZZ_TEMP_" followed by the developer's initials and then whatever descriptive information the developer chooses. This naming scheme also groups temporary objects at the bottom of the list. It makes them easy to find and identifies the developer. The objects can be deleted if the developer forgets.

The last example of naming conventions is to have the developers name their subsets starting with their initials. It reminds the developers of all subsets they have so that they can reuse any that are obsolete. It also identifies to all other developers the owner of the subset.

Project Models

Dividing the development work among project models can greatly aid in the reduction of processing time and conflicts. Developers working in a smaller project model have reduced processing time during downloads since the subsets pull in fewer neighborhood objects. Potential subsetting downgrades and resource conflicts are also reduced.

When feasible, contention can be reduced and developer productivity increased by separating work into one or more project models. Separate models increases the development coordination necessary for a project, but reduces the contention incurred by spreading the work across models.

Coordinating Shared Objects

Careful management and coordination of shared objects is also essential to the success of a development team. By carefully monitoring the modification of shared objects between models, you can reduce the number of migrations necessary to get the most current copy containing all changes into each model (for development or testing). This coordination also prevents the need for multiple coding efforts that become necessary when a shared object is changed in different models and a migration cannot preserve both sets of changes.

Avoid Mass Upload

Calling for all subsets to be checked in to facilitate migration can be very costly and nonproductive. Migration can take place while subsets are checked out.

Consider checking out a subset with the root subject area for Modify and the databases for Delete before any other subset is checked out. You can then migrate the data-related objects (entity types, attributes, relationships, and so forth) by overriding the subset, and then re-checking out the subset.

You can use this same technique for other shared objects. If shared objects are being changed in one model, you should check out them for Modify or Delete in all other models but not download them. This protects them from being changed and/or ending up in another subset for modify or delete. When it is time for them to be migrated, you can override the protective subsets and the migration can proceed without conflict.

When a mass check-in is required, spread out the uploads over time so everyone does not try to upload at the same time (such as at the end of the business day). Another method to avoid contention on a mass upload, is to have everyone using the model submit their upload in batch with the same job name. This runs the uploads sequentially and avoids conflicts.

Following a mass upload, a mass download is usually needed for the next day. To avoid everyone attempting a download at the same time, define subsets in advance and submit the downloads with the same jobname to process overnight. This sequentially downloads the subsets.

Regeneration and Installation

Another way to improve the performance of the encyclopedia is by developing procedures for the regeneration and installation of code. The intelligent regeneration function provided by the encyclopedia is more efficient to regenerate and install code one time rather than relying on individuals to figure out what needs regenerating and installing.

By using intelligent regeneration and good communications, you can reduce duplication of effort and CPU resources. Communication is necessary to determine when an intelligent regeneration should be performed. Also important, after the regeneration, is to notify all individuals that a regeneration has been performed.

An organization should develop procedures for informing the group whenever they modify an object that has a large impact on regeneration. This depends on how objects are shared in an organization. In any case, set up procedures that limit when these changes are made and that indicate to the entire organization when the changes are done.

Testing with Trace

Using trace in the application test facility is useful for debugging an application, but can also cause DB2 contention with others testing against the same application database. Sometimes contention on the application database is confused with contention on the encyclopedia. Testing code with trace on can lock up portions of the application database, which causes other jobs to fail that need access to that portion of the database. Users of trace should access and release trace as quickly as possible. If a user has a need to stay in trace for long periods of time, they should warn others that ma y need to access the database involved.

Specific Encyclopedia Functions

The recommendations previously discussed deal with general areas that you can improve. You can also improve your processes that involve specific encyclopedia functions. The reports discussed in the “Methods of Collecting Statistics” chapter can add insight into encyclopedia use and pinpoint areas to concentrate on. Understanding how these encyclopedia functions work is key to making sound decisions on coordinating their usage. For this reason, the following encyclopedia functions are discussed individually along with tips for improving their performance:

- Download
- Expansion conflict report
- Upload
- Override checkout status
- Migrate

- Adopt
- Add a new model
- Model delete
- Intelligent Regeneration
- Backup and restore
- Mainframe object delete

Download

A download consists of three basic phases: expansion, extraction, and output. During the expansion phase, the scoping objects that are selected during subset definition are expanded to include related (neighborhood) objects. The extraction phase ensures that the subset is complete (that all necessary objects are included in the subset) and downgrades the user's protection to objects if necessary. Downgrades occur because an object is already checked out to another user or because the object was not included in the subset definition. In the output phase, the model is written to the transaction file.

Since download is primarily a read-only process, the encyclopedia allows other read functions to occur while a download is in progress. So, for example, code generation can proceed while a download is in progress, but update/delete operations like an upload to the same model are not allowed.

As objects are extracted from the encyclopedia, the encyclopedia updates the DSUBEX table to indicate that the objects are checked out with a certain level of protection. Updates to the model for this purpose are the source of DB2 locks. Periodic commits are executed to alleviate the scope and duration of the locks. If a download does not complete successfully, the subset is *not* marked as being checked out. However, because commits were taken along the way, the next download must perform a cleanup routine prior to executing the download to remove all checkout updates for the failed download.

Techniques

The following techniques can improve the process involving downloads:

- Plan downloads in advance and submit them to execute in batch during non-peak hours. Set up a time period for downloads during non-peak hours. Allow only downloads during this time period. One job name should be associated with each model. Single stream the download by model during the time period.
- To avoid downloading during prime time, have more than one subset checked out at all times. If work on one subset is finished or needs to be redefined, there is always a second subset to work on.
- If downloads are allowed during the day, require that all users of that model be notified.

- Create a subsetting expert for each project or model. This expert can help users define subsets appropriately, which reduces the number of incomplete subsets and subsets that are too large.
- Users that have read-only access to a model or a subset can do a read-only download that does not mark the original model or subset as *checked* out. This is different from having update access and scoping objects for read protection.

Expansion Conflict Report

If the subset is not already checked out, the logic required to create the expansion conflict report is the same as the logic described in the section on download. The differences are that the conflict report is created rather than the transaction file and a rollback is executed so that checkout information for objects is not saved in the encyclopedia. Because a rollback is performed, the potential for contention is greater when creating this report than when downloading.

If the subset *is* checked out or if the report is run on the entire model, the program uses the existing expansion and no additional expansion is performed. Contention is minimized because no additional expansion is performed.

Techniques

The following techniques can improve the process of using the expansion conflict report:

- Run this report only when necessary. Inform all users in advance when you run the report.
- When the report is necessary, create a new subset definition containing only the object(s) that were downgraded. Do not run on a subset scoped for 10 objects if only one object is being downgraded.
- Use other techniques to find out what subsets are or would cause a downgrade. One of the simplest techniques is to ask other users of the model to check if they have the object(s) required.
- Know the rules of subsetting. Run this report only when a downgrade has occurred because the object is in another subset. Do not run the report when the object is downgraded as incomplete. When an object is downgraded as incomplete, use other encyclopedia reports such as the Delete Prevention report to identify what other objects need to be scoped.

Upload

An upload updates the model in the Host Encyclopedia. If the upload is without check-in, the upload usually runs faster because the object status is not being changed. For an upload with check-in, the following occurs:

- All the transactions in the transaction file are processed

- Intermediate commits occur
- The checkout date and time are set to zeroes
- All objects in the subset are checked in or deleted from DSUBEX with periodic commits taken

If the upload abends during the process of checking in the objects, the next download of the model or any subset of that model cleans up the partial checkout.

Since upload is an update process, the encyclopedia only allows limited access to the model while the model is being updated. An upload cannot run while a download or code generation is occurring in that model. The only activities that can run simultaneously with an upload in the same model are other uploads or overriding, renaming, deleting or copying another subset. DB2 locks occur as objects are inserted and deleted, properties are modified, and associations are added and deleted.

If the upload fails, you cannot continue to work on the subset until the upload is completed. This may be important when you plan for an upload.

Techniques

The following techniques can improve the process involving uploads:

- Upload without check-in frequently. This serves as a primary method of backup in case you need to recover work on the workstation.
- Understand how to perform an upload manually. This can be valuable if you have to restart a failed upload.
- Keep uploads small during peak processing hours. Frequent uploads without check-in accomplishes this. Most uploads that take less than 10 minutes will not fail because of an enqueue wait.
- Perform large uploads (more than one new procedure) during non-peak processing hours.
- Perform uploads with check-in during non-peak processing hours, when possible. Uploads with check-in take substantially longer than uploads without check-in.
- Teach users to select Cancel or Ignore when not making a change with the toolset. Selecting OK, even if there were no changes, can trigger an update.

Override Checkout Status

A table of checked out objects is kept in the Host Encyclopedia. Even if no subsets are defined, objects are added to this table for subset ALL. When the override function is requested, the encyclopedia updates, in the case of subset ALL, or deletes, in the case of a subset, the rows in this table for the model or subset.

Most functions are allowed to run at the same time as an override of a checkout status, but because of the amount of updating that is done, this can cause DB2 contention.

Periodic commits are taken during the delete or update. If a failure occurs during override, the next download of the model or a subset of it cleans up the partial checkout.

Techniques

Use the following technique to improve the process involving override checkout status-for large subsets, consider uploading with check-in in batch during non-peak processing hours even if there have been no updates.

Migration

Migration copies objects from a source model to a destination model. For the source model, migration is a read process. For the destination model, migration is an update process. Periodic commits are not performed because of the difficulty of detecting and correcting the results of a partial migration. Because there are no periodic commits, DB2 can escalate a PAGE lock to a TABLESPACE lock and quickly cause contention across the entire encyclopedia. Trial migration executes the same logic as migration except a rollback is done at the end of processing, which approximately doubles the processing time.

Techniques

The following techniques can improve the process involving migration:

- Migrate at the lowest aggregate level. For example, if you change an attribute, migrate the attribute, not the entity type.
- Keep migrates small (less than 10 minutes) if you must do them during peak processing hours.
- Know what is running prior to a migration. Do not migrate if other large encyclopedia jobs are running such as an upload, download, model copy or another migrate.
- Whenever possible, schedule migrates to run during a time period for single stream batch processing.
- If you must perform migrations during peak processing hours, notify all encyclopedia users in advance.
- Analyze the objects to be migrated. Know the rules of migration for each type of object. Identify the enabling and companion objects for each object to be migrated. You can use the When Changed report to identify changed objects and automatically create aggregate sets. You can use the Compare Report to compare the results between the source and destination models. The Contains/Contained, References/Referenced, and Implements/Implemented reports help you to identify enabling objects that may need to be included in the aggregate set of the migration.
- Avoid large migrations. If one object fails, the entire migration fails.

Adoption/Unadoption

Adoption is used to assign common ancestry to an object for the purpose of migration. The original object ID of the adoptee is updated to the original object ID of the related object. Adoption serves two purposes:

- Moves a new model into an existing family
- Establishes a correspondence between objects that are alike so they can be migrated or compared

The adopt function automatically unadopts component objects. Component objects that do not get adopted are given a new original object ID to sever any common ancestry they may have. Object IDs are assigned as they are to a subset. Incremental commits are not taken during the adopt function. Adopt of large models can cause DB2 to escalate its locks to the tablespace level, thereby causing contention.

With large models, unadoption does no incremental commits and therefore will lock many database pages before the process is completed. To accomplish the effect of incremental commits, you can selectively adopt large models rather than using the All option.

The following table lists the suggested order of adoption for specific object types and the prerequisites that are required for the adoption of the objects:

Object Type	Prerequisite
Subject areas	
Entity types	
Relationships	Entity types
Data records	Entity types, Relationships
Database	
Tablespace	Databases
Work attribute sets	
Business system	
Commands	Business system
Templates	Business system
Exit States	
Functions	Entity types and work attribute sets defined in views
Processes	Entity types and work attribute sets defined in views

Object Type	Prerequisite
BAA common action blocks	Entity types and work attribute sets defined in views
BSD common action blocks	Business system, entity types and work attribute sets defined in views
Procedure	Business system, entity types and work attribute sets defined in views
Dialog flow	Procedures, Exit states
Online load module	Business system
Window load module	Business system
User defined object class	
User defined matrix	User defined object class

All other selectable objects do not have prerequisites and can be adopted in any order.

The unadopt function performs the opposite of an adoption. Uses for the unadopt feature include:

- Removing common ancestry when using template models to seed a model
- Creating a new model family
- Correcting the common ancestry when the adoptee and related models were reversed during an adoption

An unadopted model no longer has common ancestry with models in its prior family. This process is accomplished by replacing all original object IDs with unique IDs, but does not change the appearance of the model. Incremental commits are not taken during the unadopt function. Unadopt of large models can cause DB2 to escalate its locks to the tablespace level, thereby causing contention.

Techniques

The following techniques can improve the process involving adoption:

- Identify the purpose and scope of the adoption. Avoid an adoption on all objects if you require only a limited adoption. Select only system-defined objects if the purpose of the adoption is simply to change the family for the adoptee.
- Keep models within the same family as much as possible to avoid adoptions.
- Refrain from deleting and recreating objects (thereby losing original object IDs) which forces an adopt prior to a migrate.
- Put procedures in place for tracking name changes that are not picked up by an adopt.

- Avoid large adopts/unadopts during peak processing hours. Keep adoptions small (less than 10 minutes) if you must perform them during peak time.
- Know what is running prior to an adopt/unadopt. Do not adopt if other large encyclopedia jobs are running.
- Notify all encyclopedia users in advance if you must perform an adopt/unadopt during peak processing hours.
- Whenever possible, schedule large adopts/unadopts to run during a time period for single stream batch processing.
- Use trial adoption only when necessary to identify and exclude objects that should not be adopted, such as those with the same name, or to ensure that no unintended name changes occur.

Adding a New Model

The following activities can be considered as adding a new model to the encyclopedia:

- Model copy
- Generate new model
- Initial model upload
- Cross copy from another encyclopedia
- Create model from subset

Each of the activities creates objects, associations, properties, and text. The activities place a read lock on the source model to prevent changes to the model while the activity occurs. DB2 locks are acquired during this processing. Periodic commits are executed to alleviate the scope and duration of the DB2 locks.

When you add a new model, all objects, properties, associations and text are added before the DMDL table is updated. Unless the add completes normally and the final step of updating DMDL occurs, orphan objects are left in the database. The orphan objects do not affect other operations, but the incomplete model is unusable. You can clean up the orphan objects with the utility TIECLEAN.

For large models, the overall effect of a failed activity, such as a failed copy, can be substantial. Because a copy does periodic commits, orphan objects are left in the encyclopedia following a failure. For a large model in an encyclopedia where DASD is in short supply, you may need to clean up the orphan objects and REORG the database prior to being able to perform the copy again.

Techniques

The following techniques can improve the process involving the addition of a new model:

- Check current DASD usage and estimate required DASD for a model prior to adding the new model. Increase DASD if necessary prior to performing the activity.
- Schedule model copies and generate new models for overnight processing.
- Create a model from a subset rather than migrating to a new model. This costs less and there is less chance of failure. Migrate to new model requires that all enabling objects be included in the migration selection.
- To reduce CPU and elapse time for a model copy, alter the tablespace locksize from ANY to TABLESPACE. The copy function provides an option to lock tables at the table level. You must be an encyclopedia administrator to use this option.

Note: Altering the locksize to TABLESPACE locks all other users out of the encyclopedia tables. See the discussion below of faster model copy and delete.

Model Delete

Model delete deletes objects, associations, properties, text, and subset definitions. It does periodic commits to alleviate the duration of the DB2 locks. Because of the heavy amount of database activity, it has a tendency to escalate to tablespace locks that causes contention across the entire encyclopedia. The first task done by a model delete is to remove the model from the DMDL table. Periodic commits occur during processing. If a model delete does not complete successfully, you can clean up orphan objects with the utility TIECLEAN.

Techniques

The following techniques can improve the process involving model delete:

- Schedule model deletes to run in batch during non-peak processing hours.
- Schedule model deletes prior to a REORG.
- Rename models intended for delete and schedule the delete during a time period for batch single stream processing.
- To reduce CPU and elapse time for a model delete, alter the tablespace locksize from ANY to TABLESPACE. The delete function provides an option to lock tables at the table level. You must be an encyclopedia administrator to use this option.

Note: Altering the locksize to TABLESPACE locks all other users out of the encyclopedia tables. See the discussion of faster model copy and delete.

Faster Model Copy and Delete

You can noticeably improve the performance of the encyclopedia functions Model Copy and Model Delete by altering the DB2 locksize before and after the copy or delete. The purpose of the ALTER command is to change DB2 locksize from PAGE (ANY) to TABLESPACE. Locking at the tablespace level greatly reduces the number of DB2 locks and unlocks that must be performed by IRLM during execution of the function.

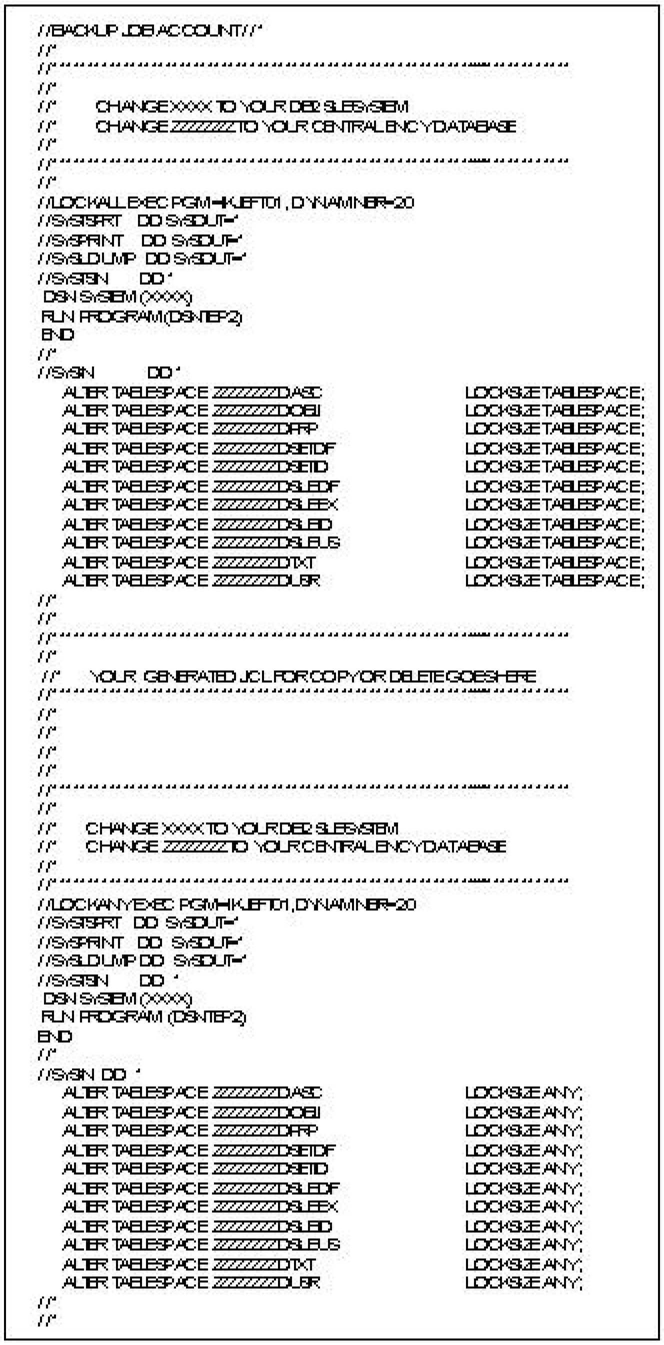
Changing the locksize to TABLESPACE means that the entire encyclopedia is locked during execution. Because of this, the change should probably not be considered for peak processing hours. Note also that you need DB2 authority to perform ALTER.

Techniques

The following technique can improve the process involving faster model copy and delete:

- Set up a batch job to alter the locksize from PAGE to TABLESPACE.
- Begin by generating the JCL for the encyclopedia function in the usual manner.
- The system-generated JCL is placed in the middle of the job.
- Save the combined JCL and execute in the usual manner.
- Sample JCL to alter locksize.

The following illustration provides the additional JCL that is needed:



```
//BACKUP JOB ACCOUNT//*
//*
//*****************************************************************
//*
//*      CHANGE XXXX TO YOUR DB2 SUBSYSTEM
//*      CHANGE ZZZZZZZZ TO YOUR CENTRAL ENCY DATABASE
//*
//*****************************************************************
//*
//LOCKALL EXEC PGM=IKJEFT01 , DYNAMNBR=20
//SYSTSPRT   DD SYSOUT=*
//SYSPRINT   DD SYSOUT=*
//SYSUDUMP  DD SYSOUT=*
//SYSTSIN      DD *
 DSN SYSTEM (XXXX)
 RUN PROGRAM(DSNTEP2)
 END
//*
//SYSIN          DD *
    ALTER TABLESPACE ZZZZZZZZ.DASC                        LOCKSIZE TABLESPACE;
    ALTER TABLESPACE ZZZZZZZZ.DOBJ                        LOCKSIZE TABLESPACE;
    ALTER TABLESPACE ZZZZZZZZ.DPRP                        LOCKSIZE TABLESPACE;
    ALTER TABLESPACE ZZZZZZZZ.DSETDF                      LOCKSIZE TABLESPACE;
    ALTER TABLESPACE ZZZZZZZZ.DSETID                      LOCKSIZE TABLESPACE;
    ALTER TABLESPACE ZZZZZZZZ.DSLEDF                      LOCKSIZE TABLESPACE;
    ALTER TABLESPACE ZZZZZZZZ.DSLEEX                      LOCKSIZE TABLESPACE;
    ALTER TABLESPACE ZZZZZZZZ.DSLEID                      LOCKSIZE TABLESPACE;
    ALTER TABLESPACE ZZZZZZZZ.DSLEUS                      LOCKSIZE TABLESPACE;
    ALTER TABLESPACE ZZZZZZZZ.DTXT                        LOCKSIZE TABLESPACE;
    ALTER TABLESPACE ZZZZZZZZ.DUSR                        LOCKSIZE TABLESPACE;
//*
//*
//*****************************************************************
//*
 //*     YOUR GENERATED JCL FOR COPY OR DELETE GOES HERE
//*****************************************************************
//*
//*
//*
//*
//*****************************************************************
//*
//*     CHANGE XXXX TO YOUR DB2 SUBSYSTEM
//*     CHANGE ZZZZZZZZ TO YOUR CENTRAL ENCY DATABASE
//*
//*****************************************************************
//LOCKANY EXEC PGM=IKJEFT01,DYNAMNBR=20
//SYSTSPRT   DD SYSOUT=*
//SYSPRINT   DD  SYSOUT=*
//SYSUDUMP DD  SYSOUT=*
//SYSTSIN      DD  *
 DSN SYSTEM (XXXX)
 RUN PROGRAM (DSNTEP2)
END
//*
//SYSIN DD  *
    ALTER TABLESPACE ZZZZZZZZ.DASC                        LOCKSIZE ANY;
    ALTER TABLESPACE ZZZZZZZZ.DOBJ                        LOCKSIZE ANY;
    ALTER TABLESPACE ZZZZZZZZ.DPRP                        LOCKSIZE ANY;
    ALTER TABLESPACE ZZZZZZZZ.DSETDF                      LOCKSIZE ANY;
    ALTER TABLESPACE ZZZZZZZZ.DSETID                      LOCKSIZE ANY;
    ALTER TABLESPACE ZZZZZZZZ.DSLEDF                      LOCKSIZE ANY;
    ALTER TABLESPACE ZZZZZZZZ.DSLEEX                      LOCKSIZE ANY;
    ALTER TABLESPACE ZZZZZZZZ.DSLEID                      LOCKSIZE ANY;
    ALTER TABLESPACE ZZZZZZZZ.DSLEUS                      LOCKSIZE ANY;
    ALTER TABLESPACE ZZZZZZZZ.DTXT                        LOCKSIZE ANY;
    ALTER TABLESPACE ZZZZZZZZ.DUSR                        LOCKSIZE ANY;
//*
//*
```

Intelligent Regeneration

Whenever you change a model, the encyclopedia records information about the change. This information is used by intelligent regeneration to determine what has been changed since the last successful generation and installation. This information, combined with the rules of regeneration, enables intelligent regeneration to identify which screens, action blocks, procedure steps, and Dialog/Batch Managers need to be generated and installed.

Techniques

The following techniques can improve the process involving intelligent regeneration:

- Run intelligent regeneration on a regularly scheduled basis. This reduces the amount of work and time to flag objects that require regeneration by keeping the generation current. Run during non-peak hours or during a time period when no other code generation, uploads, or downloads for the model take place.
- Avoid changing environment, external libraries and technical system parameters that cause a major regeneration, or, if you do change them, exclude them from the items to be regenerated.
- When performing regenerations regularly, narrow in by using dates of the previous intelligent regeneration.

Model Backup and Restore

The Backup function copies the model to a sequential file format. The Restore function creates the model from the sequential file by using the DB2 load utility. The backup and restore functions preserve subset definitions, aggregate sets, history data, and user access data. The restore can run for a long time if you have not run the TIECLEAN utility recently. TIECLEAN is part of the restore process.

The Restore function puts the encyclopedia tables in utility status, which prevents other users from using the encyclopedia.

Techniques

The following techniques can improve the process involving model backup and restore:

- Develop schedules for Backup/Restore and consolidate requirements for multiple teams.
- If family ancestry is not important, consider other options like download with the upload option.
- Perform an image copy the encyclopedia prior to running the Restore function.
- Since the encyclopedia database is in utility status during the Restore, consider closing down the encyclopedia or running the Restore during a non-peak time.
- Use Backup for inactive models to reduce DASD requirements.

Object Delete

This function verifies that the object is not checked out and deletes it. The encyclopedia does not allow read or update access to the model (except for limited subset functions) while the delete is taking place.

Techniques

The following techniques can improve the process involving object deletions:

- Use the mainframe delete object whenever possible instead of subsetting to delete an object.
- Flag objects for delete during peak processing by renaming them with a starting prefix (XXDEL_). Delete objects during non-peak processing.

Most deletes run quickly without causing contention. Some objects, such as procedure steps, cause other objects like dialog flows and screens, to be deleted. This causes the process to run longer which may cause some contention.

Chapter 4: Host Encyclopedia Utilities

This chapter describes the Host Encyclopedia utilities.

This section contains the following topics:

DIAG

DIAG clist creates a report of the transaction file. This report can be used for debugging problems associated to a transaction file.

How to Invoke DIAG

DIAG can be invoked from the command line as follows:

```
TSO %DIAG
```

IEFDOWN

IEFDOWN clist creates the transaction file when downloading/extracting a model/subset from the Host Encyclopedia.

How to Invoke IEFDOWN

This function is available from the Main menu, option 1.3.15, or %IEFDOWN can be invoked from the TSO READY prompt.

IEFUP

IEFUP clist uploads or applies a model or a subset to the Host Encyclopedia.

How to Invoke IEFUP

This function is available from the Main menu; option 1.3.16 or %IEFUP can also be invoked from the TSO READY prompt.

IEFVARS

IEFVARS clist displays the significant global variables defined in the TIUGLOB clist. TIUGLOB clist is used by the encyclopedia functions to define global system variables established during the encyclopedia installation.

IEFVARS displays useful information such as the owner ID, DB2 subsystem, DB2 plan prefix, and library names.

How to Invoke IEFVARS

IEFVARS can be invoked from the command line as follows:

```
TSO %IEFVARS
```

Rebind

Rebind Host Encyclopedia packages after you use the Runstat utility. If you plan to investigate or save the access paths, then set EXPLAIN parameter to YES when running the Rebind. When EXPLAIN(YES) option is used, ensure that the corresponding PLAN_TABLE exists.

For instructions about running Rebind, Rebind with EXPLAIN(YES), and about creating and using the PLAN_TABLE, refer to DB2 documentation.

Reorg

Perform reorganization once a week on all data and PI tablespaces. Perform reorganization as needed, once a week is suggested, for the indexes for these tables:

- DASC
- DOBJ
- DPRP
- DSUBEX

Perform a quiesce for each tablespace being reorganized before you start the reorganization. The quiesce provides a point to which you can recover, if necessary. You may also need to reorganize the indexes in between the scheduled table reorganizations if numerous rows are being inserted into the encyclopedia.

The job CEUREOR (or the alternative job CAUREOR) runs the Reorg utility for Host Encyclopedia tablespaces. For more information about running the Reorg utility, refer to DB2 documentation.

Runstat

Use this utility frequently when the Host Encyclopedia is new or growing, once a week or at least once a month is recommended. When the encyclopedia reaches a size where the Runstat information does not change the path used by the optimizer, you do not have to use this utility. Do not run this utility on an empty table.

The job CEJOB06 (or the alternative job CAJOB06) runs the Runstats utility for Host Encyclopedia table spaces and indexes. For more information about running the Runstats utility, refer to DB2 documentation.

Quiesce

Use DB2 Quiesce utility to establish a quiesce point for Host Encyclopedia table spaces. This will provide a point at which you can recover data. For more information about running the Quiesce utility, refer to DB2 documentation.

TIECLEAN

Use this utility once a week. This utility deletes all components of any model that does not have a row in the DMDL table

An orphan model can occur if any of the following actions fail:

- Copying a model
- Creating a model from a subset
- Checking in a new model
- Deleting a model

In order to perform incremental COMMITs during these actions, the system inserts the row into DMDL differently depending on the action:

- The row is inserted last when a model is being created (the first three bullets).
- The row is deleted first when a model is being deleted (the fourth bullet).

This utility removes all of the stranded components of the model and allows for the reuse of space. If you are using non-segmented tablespaces, the space is not available until you REORG the database.

This utility performs COMMITs as it runs. You can run this utility daily with a time parameter. If the utility does not complete before exceeding the time parameter, all components prior to the last COMMIT are deleted. Components that did not get deleted are deleted the next time the utility runs. The next run continues the clean up.

How to Invoke TIECLEAN

This function is available from the Main menu, option 1.6.1 or TIECLEAN can be invoked from the command line as follows:

```
TSO %TIECLEAN
```

TIECOMP

The Model Action Block Use Report lists the calling hierarchy of components in a model. It displays the Dynamic Link attribute associated with each module. The Dynamic Link attributes associated with each module are:

- Yes
- No

- Compatibility
- Default

For each Business System included as part of the report, the default values assigned to the Dynamic Link defaults for Procedure Steps, Screens Managers, and Action Blocks are displayed. The report has three basic modes of operation based on the object range. They are:

- All business systems and load modules
- Selected objects
- Filter objects based on Dynamic Link option

Note: For more information, see *CA Gen Host* Encyclopedia *User Guide*.

How to Invoke TIECOMP

This function is available from the Main menu, option 1.3.13.13, or by invoking %TIECOMP from the TSO READY prompt.

TIEEVAL

TIEEVAL is a problem determination tool. It validates the model according to the schema definition or meta model rules.

The consistency check function checks a model's logical structure from the point of view of the application in question. For example: Does an Entity Type have both attributes and an Identifier? The Encyclopedia Validation program TIEEVAL checks the model at a much more fundamental level against the Schema for that model. For example: Are all Objects, Properties and Associations in place that need to be in place? If a model passes this validation with no errors it is unlikely that the model will hit errors during encyclopedia processing.

When to Run TIEEVAL

The typical use for TIEEVAL is to find association cardinality, ordered association and reference to non-existent object problems.

Example: Cardinality problem:

```
Migration fails with SQLCODE=-811,ONLOC=GET_NAME,
        ONCODE=9 PROCEDURE=TIECCRM
```

Invalid Ordered Association:

```
Upload fails: SQLCODE=-811 in Procedure TIESEQ or
        Invalid Ordered Association
```

Reference to Non-existent Object:

```
Copy fails with “Object id 135692234 not found in XREF table”
```

These errors should be encountered infrequently. They may not necessarily manifest themselves as problems to a user. Some errors may in such cases be regarded more as warnings.

It is recommended to run this report only if you are experiencing several errors or unusual results in the processing of a particular model. It is useful to run it with guidance from Technical Support.

How to Invoke TIEEVAL

TIEEVAL can be run in either foreground or background mode. It is recommended that it be run in background. Because the utility has to check and cross check every Object, Property, and Association against the Schema, there is extensive processing-which can take considerable time, depending on the size of the model. This is why it is recommended to run it in batch.

TIEEVAL clist has many keywords. To get more information, type:

```
TSO %TIEEVAL
```

This will display the required keywords and additional information. Once you know what you want to run, type:

TSO %TIEEVAL MODEL('*model name*') *keywords*

Note: The keyword BATCH generates the JCL to execute as a background job.

TIEMAX

TIEMAX clist resets the maximum ID values in the DMAX table as follows:

MAX_MNEMONIC	MAX_VALUE
MAXDOBJ	Current maximum object ID for subset pool (DSUBID S_MODEL_ID = 1)

MAX_MNEMONIC	MAX_VALUE
MAXDSSID	Last subset ID used (not reset since subset IDs are reused)
MAXSESN	Current maximum session ID for IDNUMBER property in DPRP and COPY_DPRP_INT_MAX property in DCOPY
MAXSETID	Current maximum aggregate set ID in DSETID
MAXSASC	Maximum association type code value for schema release
MAXSDIV	Maximum schema division code value for schema release
MAXSOBJ	Maximum object type code value for schema release
MAXSPRP	Maximum property type code value for schema release

How to Invoke TIEMAX

TIEMAX can be invoked by issuing TSO %TIEMAX from the command line. On the next panel leave all entries blank for schema level, encyclopedia name, and encyclopedia id. These entries are only used during the Host Encyclopedia installation

Note: For more information, see *CA Gen Host Encyclopedia and Host Construction Installation Guide.*

When to Run TIEMAX

Run TIEMAX only if there is a problem with the DMAX table. Typically, this is run only during the encyclopedia installation to set up the DMAX table the first time.

TIEU31K

TIEU31K is a report utility to detect and report all procedure steps within a model where the view mapping fails the 31 KB limit calculation.

The consistency check report also displays severe warnings if the 31 KB limit is met:

- ICCBR03S The total import view length will exceed the supported maximum length of 31744 bytes (31 KB) if generating COBOL code for non-cooperative applications.
- ICCBR04S The total export view length will exceed the supported maximum length of 31744 bytes (31 KB) if generating COBOL code for non-cooperative applications.
- ICCBR11S The total import view length will exceed the supported maximum length of 31744 bytes (31 KB) if generating C code for non-cooperative applications.
- ICCBR12S The total export view length will exceed the supported maximum length of 31744 bytes (31 KB) if generating C code for non-cooperative applications.
- ICCBR16S The total import view length will exceed the supported maximum length of 31400 bytes if generating C code for cooperative servers.
- ICCBR17S The total export view length will exceed the supported maximum length of 31400 bytes if generating C code for cooperative servers.
- ICCBR18S The total import view length will exceed the supported maximum length of 31400 bytes if generating COBOL code for cooperative servers.
- ICCBR19S The total export view length will exceed the supported maximum length of 31400 bytes if generating COBOL code for cooperative servers.

How to Invoke TIEU31K

TIEU31K can be run for one model or for all models in the encyclopedia. It can be run in either foreground or background mode.

The syntax to run TIEU31K for one model only is the following:

```
TSO %TIEU31K MODEL('model name') BATCH
```

The BATCH parameter (background) is optional. The default is ONLINE (foreground).

The syntax to run TIEU31K against all models in the encyclopedia is the following:

```
%TIEU31K MODEL('REPORT ALL MODELS IN ENCY') BATCH
```

The MODEL parameter must be 'REPORT ALL MODELS IN ENCY'. The BATCH parameter (background) is optional, but is recommended if you run against the encyclopedia. The default is ONLINE (foreground).

When to Run TIEU31K

TIEU31K can be run to determine if there are any procedure steps within the model where the view mapping will fail the 31 KB limit calculation. It can also be run against all models in the encyclopedia. Note that the consistency check report also displays severe warnings if the 31 KB limit is met.

TIEUGVW

TIEUGVW is a report utility to detect and report all occurrences of the max size of group views that are greater than 9999 in a model.

How to Invoke TIEUGVW

TIEUGVW can be run for one model or for all models in the encyclopedia. It can be run in either foreground or background mode.

The syntax to run TIEUGVW for one model only is the following:

```
TSO %TIEUGVW MODEL('model name') BATCH
```

The MODEL parameter is required. The BATCH parameter (background) is optional. The default is ONLINE (foreground).

The syntax to run TIEUGVW against all models in the encyclopedia is the following:

```
%TIEUGVW MODEL('REPORT ALL MODELS IN ENCY') BATCH
```

The MODEL parameter must be 'REPORT ALL MODELS IN ENCY'. The BATCH parameter (background) is optional, but is recommended if you run against the encyclopedia. The default is ONLINE (foreground).

When to Run TIEUGVW

TIEUGVW can be run to determine if there are any group views with the max size greater than 9999. It can also be run against all models in the encyclopedia.

TIEWENCY

TIEWENCY clist enables on-line navigation of the Host Encyclopedia. It is a problem research tool. It also assists in Version Control. It is sometimes referred to as WALKENCY (“WALK around the host ENCYclopedia).

TIEWENCY can be used for the following activities:

- Querying Encyclopedia using model name and object type or by discrete object Id
- Following associations between one object and another
- Displaying properties for specific objects
- Optionally displaying only To (T) or From (F) associations

When Should You Use TIEWENCY?

- For problem determination within the Host Encyclopedia
- For detailed verification of internal model structures
- Under the direction of Technical Support

From time to time a model corruption may occur causing unusual errors. The encyclopedia software can normally detect a model corruption. It gives some diagnostic information against which to act. This may be a message relating to an invalid or missing mandatory association, or duplicate object for example. Correction may need to be made outside of the scope of the normal Encyclopedia Software. In these instances, the TIEWENCY tool can be used to investigate problematic objects and to identify the required action.

TIEWENCY can take locks that are held while waiting for the user to input a selection on a panel. It is possible that if you are working in TIEWENCY for some time that you can conflict with another encyclopedia user causing their application to deadlock and timeout.

DPRP and Walkency

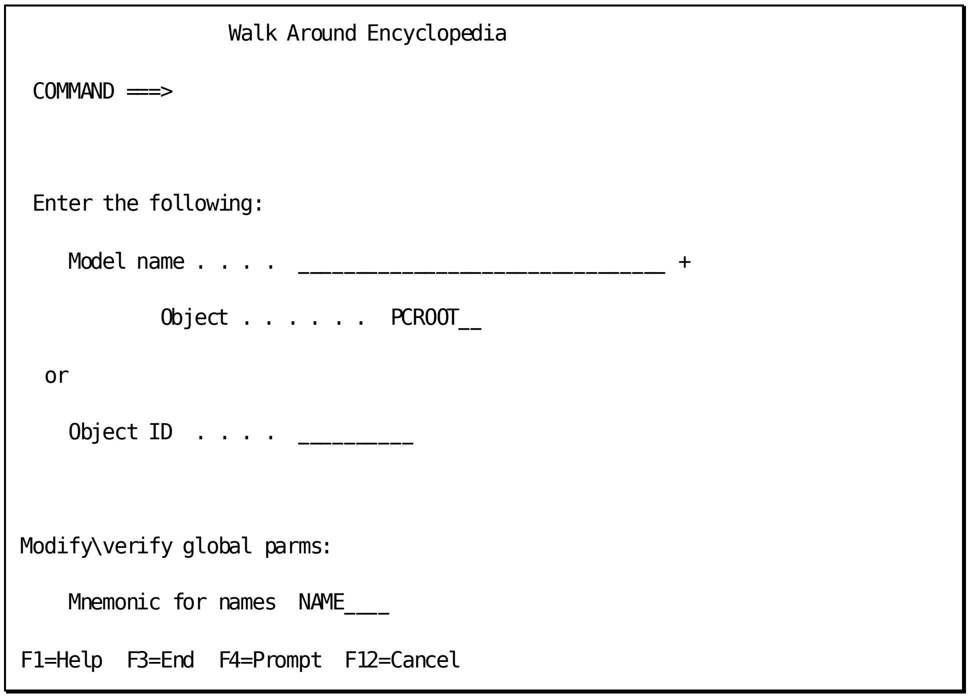
Walkency will display all properties for that object, as held in the DPRP table. It will also display all possible valid properties for that object, as held in the Schema table SPRP.

DTXT and Walkency

Long text properties stored in DTXT table can be displayed by using an option “T” from within the Property List panel for a particular object.

How to Invoke TIEWENCY

TIEWENCY can be invoked by issuing TSO %TIEWENCY from the command line. It displays the following panel:



```
                 Walk Around Encyclopedia

 COMMAND ===>


 Enter the following:

    Model name . . . .  ______________________________ +

            Object . . . . . .  PCROOT__

  or

    Object ID  . . . .  __________


Modify\verify global parms:

    Mnemonic for names  NAME____

F1=Help  F3=End  F4=Prompt  F12=Cancel
```

If invoking TIEWENCY by Object Type, the mnemonic name needs to be used. This can be obtained from SOBJ interrogation if you are not sure. You need to specify the Model name to restrict the number of objects returned if you use this method.

The object id is unique within the encyclopedia. Therefore if you enter Object ID, you do not need to specify the model name. The object id is usually given in an error message issued by the encyclopedia functions when there is a model corruption.

To back out path levels, press the Cancel key. This will also return to the previous panel when no path is displayed. To exit from Walkency at any point, press the End key. Find and locate Commands are also available. See Help within TIEWENCY dialog.

TIUADDF

TIUADDF clist adds IEF_SUPPLIED functions to a model if they do not exist.

How to Invoke TIUADDF

TIUADDF can be invoked by issuing TSO %TIUADDF from the command line. Enter name of model or enter ALL for model name to execute utility on every model in the encyclopedia.

When to Run TIUADDF

Run TIUADDF if any IEF_SUPPLIED functions are missing from a model.

TIUDEBUG

TIUDEBUG clist can be used to set various debugging and tracing options for all CLISTs.

Note: For more information, see *CA Gen Host Encyclopedia Construction User Guide*.

How to Invoke TIUDEBUG

TIUDEBUG can be invoked from the command line as follows:

```
TSO %TIUDEBUG
```

TIUSESS

TIUSESS clist can be used to delete SESSION objects that do not have any objects associated with them. The SESSION objects identify changes made during a workstation session or some Encyclopedia tasks such as Migration. The SESSION objects drive the Intelligent Regeneration functions.

How to Invoke TIUSESS

TIUSESS can be invoked USING the command line as follows:

```
TSO %TIUSESS Model ('model name') other parameters
```

TIUSIDU

Each Host Encyclopedia has a limit of 8192 subset identifiers. If a Host Encyclopedia is running critically short of subset identifiers, this utility will free up those that are identified as ancestry implicit. Ancestry implicit subset IDs are those where any object identifier, within the subset's available range, only exist within the encyclopedia as an ancestry id. Ancestry implicit subset ID no longer has any objects in its range, because the model or the objects have been deleted, but these objects had been used as ancestry, so the object IDs from its range are used in the original object IDs (OBJ_ORG_ID) of other objects. This utility updates the original object IDs (OBJ_ORG_ID) pointing to non-existing objects to reference an existing object. Therefore, it makes it possible to free ancestry implicit subset IDs and still preserve ancestry.

Steps to Determine Whether It Is Necessary to Run TIUSIDU

Follow these steps:

1. Run encyclopedia cleanup (option 1.6.1 from Main menu or TIECLEAN clist).
2. Run TIUSSID utility to determine how many subset identifiers are available.
3. Consult with Technical Support.

It is important to note that TIUSIDU uses extensive resources. Other measures and techniques that can be used to free up subset identifiers are to delete unnecessary models, delete unused subset definitions, delete any unnecessary backups from DCOPY and DCPYUS tables, and execute the encyclopedia cleanup utility. These techniques should be tried first before running this utility.

Recommendations Prior to Running Utility

Follow these steps:

1. Backup encyclopedia database.
2. For performance reasons, it may be desirable to create an additional temporary DB2 index on encyclopedia table DSETDF and bind the TIUSIDU plan or package prior to submitting the generated utility JCL.

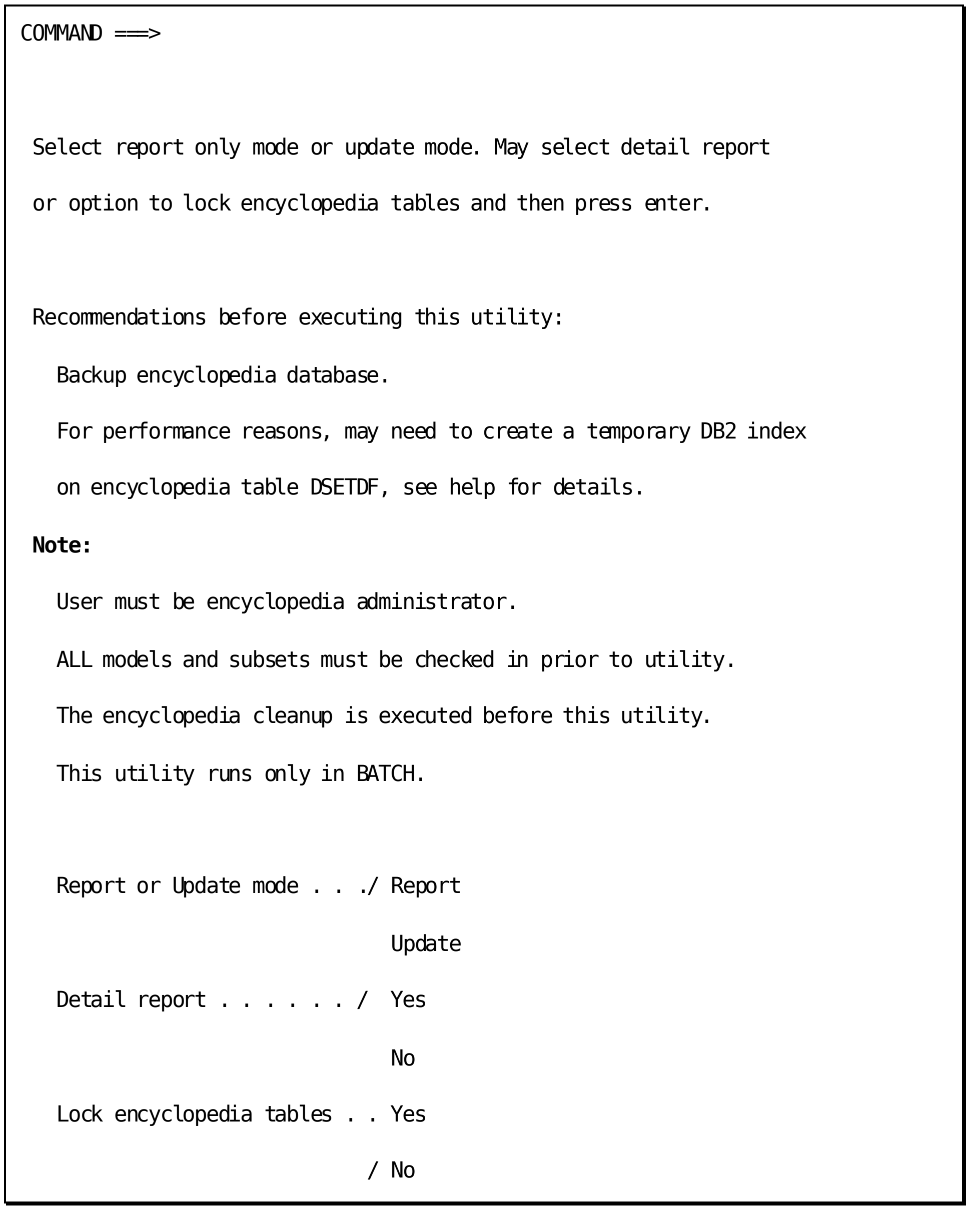
TIUSIDU performance of aggregate set processing can be significantly improved by creating a temporary index on DSETDF as follows:

```
CREATE TYPE 2 UNIQUE INDEX DSETDFI2
ON DSETDF (ASD_AGGREGATE_ID, ASD_SET_ID)
USING STOGROUP xxxx
             PRIQTY xxxx
             SECQTY xxx
             CLOSE NO;
```

If you create this index be sure to bind TIUSIDU again so that the new index will be included in the DB2 plan access path. Also, remember to drop this index when you are finished running TIUSIDU so that this index does not degrade other host encyclopedia functions. Do not bind any other encyclopedia plan/package while this temporary index exists.

How to Invoke TIUSIDU

TIUSIDU is a batch utility. The JCL to run this will be generated in the foreground by entering "TSO %TIUSIDU" from the command line. The following Ancestry Implicit Subset Identifier Cleanup Utility panel will appear, after which JCL will be generated:



```
COMMAND ===>


 Select report only mode or update mode. May select detail report

 or option to lock encyclopedia tables and then press enter.


 Recommendations before executing this utility:

   Backup encyclopedia database.

   For performance reasons, may need to create a temporary DB2 index

   on encyclopedia table DSETDF, see help for details.

 Note:

   User must be encyclopedia administrator.

   ALL models and subsets must be checked in prior to utility.

   The encyclopedia cleanup is executed before this utility.

   This utility runs only in BATCH.


   Report or Update mode . . ./ Report

                                Update

   Detail report . . . . . . /  Yes

                                No

   Lock encyclopedia tables . . Yes

                              / No
```

The user must then submit the JCL for execution. The generated JCL contains two steps. Encyclopedia Cleanup is the first step (to ensure all incomplete models are deleted) followed by the TIUSIDU utility. All models and subsets must be checked in before running this utility in UPDATE mode. The default REPORT mode does not have this requirement. It produces a report with the results without updating the database. Note that if any updates to database occur between the REPORT mode and the UPDATE mode, then results may vary. Be sure to allow plenty of time and output lines on the job card. The detail report can be very lengthy. It produces a detail report that contains a list of every OBJ_ORG_ID that is replaced. You might also like to run with DEBUG=P specified in the JCL PARM statement of the second step to get additional report log file information about each subset ID from 0 to 8191. This information is helpful to understand how each subset ID is used.

Sample Debug Information in Log File for DEBUG=P

```
SUBSET ID= 0 IS NOT AVAILABLE (DEFINED IN DSUBID)
SUBSET ID= 1 IS AVAILABLE (NOT USED)
SUBSET ID= 2 IS NOT AVAILABLE (DEFINED IN BACKUP)
SUBSET ID= 3 IS NOT AVAILABLE (OBJECTS DEFINED IN DOBJ)
SUBSET ID= 4 IS NOT AVAILABLE (S_MAX_OBJ_ID IN RANGE)
SUBSET ID= 5 IS ANCESTRY IMPLICIT
```

Sample Detail Report

```
CLEANUP OF ANCESTRY IMPLICIT SUBSET IDENTIFIERS ON ENCYCLOPEDIA ENCYH60
DETAIL REPORT *** NO DATABASE UPDATES ***

DATE: 2001-09-04, TIME: 15:26

 Subset ID     Object ID       Old Org ID      New Org ID      Model ID
18             6553601 4718593        6553601         24
18             6815745 4718593        6553601         25
18             7077889 4718593        6553601         26
18             7340033 4718593        6553601         27
18             8388609 4718593        6553601         31
18             8650753 4718593        6553601         32
18             8912897 4718593        6553601         33
```

Sample Summary Report

The report is displayed as follows:

```
CLEANUP OF ANCESTRY IMPLICIT SUBSET IDENTIFIERS ON ENCYCLOPEDIA ENCYH60
               SUMMARY REPORT *** NO DATABASE UPDATES ***

                        DATE: 2001-09-04, TIME: 15:26

The following ancestry implicit subsets are obsolete.
The models that will be affected by the release of each subset ID are listed.


Subset ID               Model Name                               Model ID

       18               CBD CONSUME BASE 12 BUS SYS              24
                        CBD CONSUME BASE 13 BUS SYS              25
                        CBD CONSUME BASE 14 BUS SYS              26
                        CBD CONSUME BASE 15 BUS SYS              27
                        CONSUME FROM LOCAL                       33
                        CONSUME10                                        50
                        CONSUME2                                         42
                        CONSUME3                                         43
                        CONSUME4                                         44
                        CONSUME5                                         45
                        CONSUME6                                         46
                        CONSUME7                                         47
                        CONSUME8                                         48
                        CONSUME9                                         49
                        CUR SCHEMA CBD99 CONSUME INTERFA 32
                        CUR SCHEMA CBD99 CONSUME OPERATI 31
                        COPY OF CONSUME1                 72
                        COPY RENAME TEST 1 RENAMED               73
                        COPY RENAME TEST 2                       74
                        COPY RENAME TEST 3                       75
                        PRIOR SCHEMA CONSUME1                    40
       42               COPY OF CONSUME1                 72
```

```
The original object identifiers which use obsolete ancestry implicit subset
identifiers will be replaced.  See Detail Report.

The following models will be affected:

Model Name                                                Model ID

CBD CONSUME BASE 12 BUS SYS                                       24
CBD CONSUME BASE 13 BUS SYS                                       25
CBD CONSUME BASE 14 BUS SYS                                       26
CBD CONSUME BASE 15 BUS SYS                                       27
CONSUME FROM LOCAL                                                33
CONSUME10                                                         50
CONSUME2                                                          42
CONSUME3                                                          43
CONSUME4                                                          44
CONSUME5                                                          45
CONSUME6                                                          46
CONSUME7                                                          47
CONSUME8                                                          48
CONSUME9                                                          49
CUR SCHEMA CBD99 CONSUME INTERFA                                  32
CUR SCHEMA CBD99 CONSUME OPERATI                                  31
COPY OF CONSUME1                                                  72
COPY RENAME TEST 1 RENAMED                                        73
COPY RENAME TEST 2                                                74
COPY RENAME TEST 3                                                75
PRIOR SCHEMA CONSUME1                                     40

The following aggregate sets will be affected:

Aggregate Set Name            Aggregate Set Id

Aggregate Set 1               1

Total # of ancestry implicit subset IDs to be released=           2
Total # of original object IDs to be replaced=                    43832
Total # of models to be updated=                                  21
Total # of aggregate IDs to be replaced=                          2
Total # of aggregate sets to be updated=                          1

Total # of subset IDs already used (not available)=                           45
Total # of subset IDs free (available) before cleanup=            8145
Total # of ancestry implicit subset IDs to be released=           2
Total # of subset IDs free (available) after cleanup=             8147
```

TIUSSID

TIUSSID clist produces a report to display the subset usage. Each subset reserves an object range for use by new objects added with that subset. This report displays the subsets that have no objects in the encyclopedia that correspond to the respective object range. If these subsets are not needed, then they should be deleted. This will enable the subset identifiers to be available for reuse. At the end of the report, the totals for subset identifiers are displayed as follows:

Follow these steps:

1. Total # of subsets with no objects associated-These subsets should be deleted if no longer needed.
2. Total # of explicit subsets-This represents the rows in DSUBID table where column S_MODEL_ID > 0.
3. Total # of implicit subsets-This represents the subset ID where no row exist in DSUBID table, but there are objects in a data table (DOBJ, DPRP, DTXT, or DASC) that represent the subset ID.
4. Total # of reserved subsets-This represents the rows in DSUBID table where column S_MODEL_ID < 0.
5. Total # of object subsets defined in backups only-This represents subset IDs referenced in DCPYUS table only.
6. Grand total # of subsets used.
7. Total # of subsets available.

How to Invoke TIUSSID

TIUSSID can be invoked by issuing TSO %TIUSSID from the command line. This is a batch utility only. The JCL to run this utility will be generated. Note that this report can be run for one model only by specifying MODEL='model name' in the JCL PARM statement.

When to Run TIUSSID

TIUSSID can be run to evaluate how many subset identifiers are available for use.

Chapter 5: Performance Recommendations

This chapter describes the ways to improve the performance of the Host Encyclopedia.

Ways to Improve Performance

You can improve performance of the Host Encyclopedia with the following:

- Settings of DB2 parameters
- Definitions of tablespaces and indexspaces

If your encyclopedia resides on a machine that is running at or near 100 percent capacity, these recommendations have little effect on how your encyclopedia performs.

DB2 Parameters

The following table lists DSNZPARM value recommendations:

Value	Recommendation
DSMAX	Set DSMAX to the same value as the z/OS maximum number of datasets open.
EDMPOOL	Set the size to a minimum of 7 megabytes.
BUFFER	Define BP1 and BP2 to DB2 even if you do not allocate space for them. Defining them helps the RID list processing. The RID pool is increased as needed until it reaches the **lower** of two values: 50% of the sum of the buffer pools' sizes (BP0 + BP1 + BP2 + BP32) 200 megabytes
NUMLKTS	Set the size to 2000.
NUMLKUS	Set the size to 10000.
BLKSIZE	Set the size to 28K
LOGLOAD	Set the size large enough to hold enough records for about 15 minutes of processing at peak times. Also consider archiving the log to DASD.

Value	Recommendation
DSN07	Set the DSN07 to be multiple tablespaces. Allocate the tablespaces large enough so that the datasets do not use secondary extents.

Tablespaces and Indexspaces

Segmenting tablespaces allows DB2 to map additional space for inserts. The mapping provides a more efficient method of finding available space for the inserts. The default installation sets the segment size to 64K for the following tablespaces:

- DASC
- DOBJ
- DPRP
- DSUBEX

Partitioning encyclopedia tablespaces enables parallel execution of DB2 utilities, in that any utility can be run on any single partition. This increases the overall throughput, especially in large encyclopedia sites.

However, the partition increments are very data dependent and site specific, and the values of these increments may require periodic adjustment as the distribution of data changes within the encyclopedia tables.

There are five tablespaces in the current host encyclopedia that benefit from partitioning. They are DASC, DOBJ, DPRP, DSUBEX and possibly DTXT depending on the textual information in models. Below is the Host Encyclopedia Partitioning Guidelines.

These tablespaces can be partitioned as follows:

Tablespace	Comments
DASC	Make the current unique clustering index (DASCI1) a unique, non-clustering index. Define a new, non-unique clustering and partitioning index on columns ASSOC_FROM_OBJ_ID, ASSOC_TYPE_CODE.
DOBJ	Partition using the current unique clustering index (DOBJI1).
DPRP	Partition using the current unique clustering index (DPRPI1).

Tablespace	Comments
DSUBEX	Make the current unique clustering index (DSUBEXI1) a unique, non-clustering index. Define a new, unique clustering and partitioning index on columns SE_SUBSET_ID, SE_OBJ_ID.
DTXT	Partition using the current unique clustering index (DTXTI1).

The encyclopedia installation makes use of the following DB2 features to reduce contentions:

- Eliminate index-locking altogether by defining indexes as type 2 indexes.
- Reduce data page locking by binding encyclopedia plans with CURRENTDATA (NO) parameter.

It is possible to reduce the contentions further by using row level locking (RLL) on DB2 objects that are subject to contentions. The DMAX table is a good candidate for row level locking.

The DB2 optimizer may not choose the correct path when the first key cardinality of DOBJI2 is small compared to the first key cardinality of DOBJI1. This situation occurs when an encyclopedia contains a small number of models with a large number of objects. To help influence DB2 to choose the optimal path, the encyclopedia installation process inserts 1300 rows into the DOBJ table. The rows contain negative numbers for the model IDs and object IDs.

Appendix A: Encyclopedia Function Plans

This appendix describes the encyclopedia function plans and their purpose.

Function Plans and Purpose

The following table lists out the encyclopedia plans their purpose:

Plan	Purpose
A15	List models in encyclopedia
ADDF	Add system-defined functions to a model if the functions do not already exist
ADPT	Adopt model into another model family
AGGR	Aggregate set reports: display aggregate sets within a model and display objects within an aggregate set
ALST	Display aggregate set selection list
BCMP	Recompile of selected generated source member
BLBC	Retrieve and update RI trigger options and libraries
BLBE	Add/Update techsys objects (target environment)
BLBL	Add/Update user specified libraries
BLBR	Retrieve the technical system user specified options and libraries
BLBT	Retrieve TECHDESN libraries
BP	Maintain bind package defaults and package list order
BPKG	Batch load module packaging
CCG	Referential integrity trigger installation control generation driver
CCMP	Check an existing model or an existing subset for consistency
CJOB	List batch job load modules for generation
CKST	Override checkout status for a model or for a subset
CLEN	Used by a standalone program that deletes all codegen objects for the encyclopedia or for a model

Plan	Purpose
CLST	Build a list of backed up models
CMG	Server manager driver program
CMOD	List online load module for generation
CMPA	Compare aggregate objects report
CN02	Encyclopedia upgrade utility
COMP	Model action block use report
CONV	Convert DSUBUS table from 4.0 to 5.0 format
CPYA	Copy an existing model or an existing subset definition
CPYB	Copy an existing model or an existing subset definition
CRUD	Provide a load-module-to-entity-action matrix report for a business system
CSYS	Verify model/business system name and build business system table
CVTS	Convert model to the new schema
DELA	Delete model online
DELB	Delete model in batch
DIAG	Used by the %DIAG Clist which provides a formatted listing of the transaction file 'userid ief tran'
DLA	Display a list of DDL information or generate the DDL/VSAM statements
DLDB	Build a list of Data Bases
DNR	Duplicate object name: identify objects that appear in more than one model
DOWN	Check out a model or a subset Cross copy: Checkout model/subset from the encyclopedia to the workstation and cross copy a model from one encyclopedia to another
DSUB	Delete a model or a subset definition
EBG	Referential integrity entity trigger generation (E modules)
EVAL	Model validate
EXST	Where exists report
FBG	Referential integrity relationship trigger generation (F modules)

Plan	Purpose
FNCQ	Add subset: Expansion query report Modify subset: Expansion query report
GDRV	Code generator driver program
GPAD	Action block code generator
GRP	Group authorizations: add/delete/modify group IDs
HLOG	Used by functions to record the model history activity in the DHLOG table
IEFM	Build Public Interface definition records
IMPT	Import model into encyclopedia
INST	Drives z/OS load module installation
IRI	Installation of RI trigger modules
IRMN	Drives intelligent regeneration
IUPD	Update link-edit date and time properties of the EXECUNIT object
KWIC	KWIC index of entity types, subtypes and attributes
MAIN	Model and object selection dialog
MAX	Used to initialize the DMAX Table when installing CA Gen and to update the DMAX Table when the DMAX and DOBJ Tables get out of synchronization
MCR	Detail of encyclopedia data for a model or for a subset
MDL	Model authorizations: grant access, revoke access, change access, change owner
MHA	Encyclopedia model history activity list
MHAR	Model history archive
MHD	Encyclopedia model history detail
MHOF	Display/Update model history flag
MHS	Encyclopedia model history model selection
MIGR	Migrate aggregate object list
MLST	Used by all functions that allow the model selection list to be requested
MPGN	Map generation
MSMD	Moves the system-defined function model from DMDL to SMDL table

Plan	Purpose
NMDL	Create a new model from an existing subset
OCR	Object cross reference report generator
OPKG	Online load module packaging
PFIX	Deletion (clean up) of dangling PSTEPEU objects
PIML	Action diagrams: print action diagrams from a model Function points: calculate function points for a model KWIC index: KWIC index of entity types, subtypes and attributes
PMGN	Build the Dialog Manager for a load module
RACD	Print action diagram
RALT	List all action diagrams
RENM	Rename an existing model or an existing subset definition
RIGN	Selective generation and install of RI trigger modules
RINM	Rename RI trigger modules
RPT	User access reports
RSTH	Update model history following model restore from backup
RSTV	Model restore timestamp validation
SESS	Delete session objects no longer used
SIDU	Ancestry implicit subset identifier cleanup utility
SLST	Subset selection list
SREL	Set release
SSID	Subset identifier usage report
SSTE	Update server load modules target environment
STAT	Display model statistics or subset statistics
STPL	Build table of action blocks to be generated or listed when load module is expanded
SUB	Add subset: scoping object type list Modify subset: scoping object type list Subset summary: scoping object type list
SUBU	Change checkout user ID for a subset
TABL	Export model from encyclopedia

Plan	Purpose
TRMP	Create load module to trancode dataset
U01	Entity definition report
U02	Function definition report
U04	Elementary process information view definition report
U05	Attribute definition report
U06	Scoping object where used report
U07	Expansion conflict report
UADP	Unadoption of a complete model from its model family
UAR	User access: users' names, user ID's and user types (E,P,A)
UFPT	Function point calculation
UNAD	Unadoption model selection list
UNLD	DB2 tablespace unload: unload data to SYSRECNN and write load control records to SYSPUNCH
UP	Check in a model or a subset Cross copy: used to check model/subset into the encyclopedia from the workstation and to copy a model from one encyclopedia to another
UPDT	Used when installing CA Gen to perform an update against the schema tables
USEC	Used by all user report functions to ensures proper access of the user reports
USQL	Used when installing CA Gen
USR	User authorization functions: add/delete/change user
VER	Verification of model name and subset name
WCHG	Intelligent regeneration when changed report
WENY	TIEWENCY utility
XABR	External action block resolution process: allows the source name to be modified and determines whether it contains DBRMs
XLAT	Code page translation
XRCT	Model backup
XSQL	Used by the dynamic SQL program for TSO and PL/I and by the installation process when installing CA Gen

Appendix B: JCL for Batch Check In and Check Out

Check In JCL

Following sections discuss how to check in a model or subset to the Host Encyclopedia in batch.

Create JCL for Subset Check In

Follow these steps:

1. Modify jobcard to be valid for your site.
2. Make sure that the job will run on the DB2 subsystem where the Host Encyclopedia resides.
3. Change 'YOUR.CLIST.LIBRARY' on the //SYSPROC statement to the dataset name of the CLIST library for your Host Encyclopedia.
4. Modify PROF PREFIX(TSOID) line to contain a TSO ID that is authorized to check in a model to the encyclopedia. If you are checking in a subset, the TSOID should be the same TSO ID that has the subset checked out or TSOID of the Host Encyclopedia administrator.

5. Modify the DATASET parameter of the //SYSTSIN DD statement:

   a. % parameter-If you are uploading the model from a Windows workstation, change the %IEFUP to %IEFUW

   b. DATASET-The dataset that contains the UPDATE.TRN file.

   c. The TSO ID specified in 'PROF PRFIX(TSOID)' will automatically prefix the 'IEF.DATASET' name. Do not specify the TSO ID in this parameter. If you omit this parameter, the dataset name defaults to <tsoid>.IEF.TRAN.

```
//************************************************
//IEFUP JOB 'ACCTING INFO','YOUR NAME', *
// TIME=(1400),MSGCLASS=W,NOTIFY=TSOID,REGION=4096K *
//IEF EXEC PGM=IKJEFT01,DYNAMNBR=25
//SYSPRINT DD SYSOUT=*
//PRINTER DD SYSOUT=*
//SYSTSPRT DD SYSOUT=*
//SYSOUT DD SYSOUT=*
//SYSPROC DD DSN=YOUR.CLIST.LIBRARY,DISP=SHR
//SYSTSIN DD *
PROF PREFIX(TSOID)
%IEFUP +
TRFILE('IEF.DATASET')
//*
//
//**************************************************
```

Check Out JCL

If you download a model or subset and choose the batch rather than online option, the Update JCL for Encyclopedia Functions panel will appear when you exit the Download panel. Use the following procedure to update or create the needed JCL for subset checkout processing.

Update JCL for Model or Subset Checkout

Follow these steps:

1. Modify jobcard to be valid for your site.
2. Make sure that the job will run on the DB2 subsystem where the Host Encyclopedia resides.
3. Change 'YOUR.CLIST.LIBRARY' on the //SYSPROC statement to the dataset name of the IEF CLIST library for your Host Encyclopedia.
4. Modify 'PROF PREFIX(TSOID)' line to contain a TSO ID that is authorized to checkout the model or subset.

5. Modify the following parameters of the //SYSTSIN DD statement:

   a. J% parameter-If you are checking out the model for use on a Windows workstation, change the %IEFDOWN to %IEFDOWW

   b. MODEL('model name')-The mainframe name of model to download

   c. SUBSET('subset name')-Name of your subset; 'ALL' if entire model

   d. SOFTVERS(????)- ???? should be 9.2.*xx* if your model is a CA Gen 8 model or CA Gen 8.5 model, 9.1.A5 if your model is an AllFusion Gen 7.6, 7.5, or 7.0 model

      Note: For the exact model schema, see the Release Notes for the value of *xx*.

   e. The TSO ID specified in 'PROF PREFIX(TSOID)' will automatically prefix the 'IEF.DATASET' name. Do not specify the TSO ID in this parameter. If you omit this parameter, the dataset name defaults to <tsoid>.IEF.TRAN

   f. CPID('nnnn') nnnn is the codepage identifier for the model's language on the destination platform.

```
///*************************************************
//IEFDOWN JOB 'ACCTING INFO','YOUR NAME',
// TIME=(1400),MSGCLASS=W,NOTIFY=TSOID,REGION=4096K
//IEF EXEC PGM=IKJEFT01,DYNAMNBR=25
//SYSPRINT DD SYSOUT=*
//PRINTER DD SYSOUT=*
//SYSTSPRT DD SYSOUT=*
//SYSOUT DD SYSOUT=*
//SYSPROC DD DSN=YOUR.CLIST.LIBRARY,DISP=SHR
//SYSTSIN DD *
PROF PREFIX(TSOID)
%IEFDOWW +
MODEL('MODEL NAME') +
SUB('ALL') +
SOFT('9.2.xx') +
TRFILE('IEF.DATASET')
CPID('1252')
//*
//
///*****************************************************
```

Note: For more information, see *CA Gen Client Server Encyclopedia User Guide*

Index