***This module is designed for three contact hours of class time***

**Module Objectives:**

* Understand VSAM concepts with emphasis on Key Sequences Data Sets (KSDS)
* Understand elementary relational database concepts
* Be able to create VSAM clusters using IDCAMS
* Be able to create, populate and query DB2 tables
* Be able to write programs that run against VSAM files and DB2

Background: Chapter 4 of zconcepts.pdf in the Reference Module

**VSAM Organization**

VSAM (Virtual Storage Access Method) allows you to create an indexed cluster consisting of a series of indexes to permit sequential and/or nonsequential access to individual records. The Access Methods Services (AMS) utility IDCAMS is used to create a VSAM clusters with either unique or non-unique indexes for both primary indexes and alternate indexes. The following example will illustrate a creation and testing of VSAM cluster with a unique primary index and a non-unique alternate index.

Assume the following data set:

1 2 3 4 5 6 7

1234567890123456789012345678901234567890123456789012345678901234567890

050555500GRAUER RT11450877200E59001181800118050000118145000180

264805298CLARK JS12450879100E53330880222087928000108126000080

300000000MILGRAM IR06130580200E655510814000681480000580

400000000BENJAMIN BL10531073100E73331073 30000108128000180

638972393GRAUER J 11450877200E59001181800118050000118145000180

800000000SMITH BB09430678100E64440678 42000068036000078

900000000BAKER E 06130580200E655510814000681480000580

955000000GRAUER B 11450877200E59001181800118050000118145000180

The numbers on the top of the file with a gray background are the column numbers. The first nine digits of each record (in yellow) are the social security number of the employee. It is a unique ID and is the primary key. For example, the second record has the unique id of 264805298 and Clark begins in column 10. The first part of the task requires creating a VSAM cluster with a unique index—you have to identify the column locations of the records for the unique key to accomplish this task. This is done via identifying the offset from the beginning of the record. In this example, the social security number has an offset of 0 because it begins in the first column. See if you understand offset—what would be the offset for the last name column? What do you notice about the order of the records? Do you think this is by accident? You are correct, it is no accident – unless otherwise sorted in the program, the input records must be sorted in order of the primary index.

The next 15 characters are the last name of the employee. We want to be able to search on the primary index and also on the last name of the employee. Therefore, the last name of the employee will be used as the alternate index. In this case, the alternate index would not be unique (notice the multiple entries for GRAUER).

Although one long step could be used to accomplish this task, this illustration uses two steps—the first step creates the primary index and the second step creates the alternate index. But first, the IDCAMS utility is discussed.

**IDCAMS Utility**

IDCAMS is a utility program for managing VSAM and non-VSAM files. It provides many access method services; some of them are:

* Create a VSAM cluster with a primary index.
* Create a VSAM alternate index.
* List catalog information for a dataset.
* Print the content of a dataset in hex or character form.
* Do backups of VSAM and non-VSAM datasets.
* Delete a VSAM or non-VSAM dataset.

A listing of some of the IDCAMS Control Statements

* BLDINDEX – Build alternate indexes for existing datasets
* DEFINE ALTERNATE INDEX – Define an alternate index
* DEFINE CLUSTER – Define a cluster for KSDS, ESDS, or RRDS
* DEFINE PATH – Define a path directly to a base cluster or an alternate index and its related base cluster
* DELETE – Delete catalogs, VSAM datasets, and non-VSAM datasets
* PRINT – Print contents of VSAM datasets, non-VSAM datasets, and catalogs
* REPRO – Copy VSAM and non-VSAM datasets
* IF – Conditional execution of commands. Used with LASTCC or MAXCC where CC is Condition Code.

**Defining a VSAM Cluster with a Primary Index**

To create a VSAM cluster with a primary indedx, the IDCAMS utility is used. The REPRO statement is used for all copy operations. The code below creates the cluster UOASXXX.VSAM.MASTER from UOASXXX.WORK.DATA(OLDMAST). Notice the KEYS command. The (9 0) specifies that the key is 9 characters long and begins at offset 0 – the first column in the data set above.

You will create acctid.WORK.DATA if not already created and download the input file OLDMAST that is on Blackboard into your PDS.

Download the file named repro on Blackboard into a PDS of type such as COMPILES that accepts 80 character records. Use the appropriate command to replace all instances of UOASXXX with your acctid. Remember to use capital letters!

Type SUBMIT and press enter to run the program.

//REPRO JOB (AMSEXI),'UOASXXX',NOTIFY=UOASXXX,

// CLASS=A,MSGLEVEL=(1,1)

/\*

//STEP0010 EXEC PGM=IDCAMS

//INDATA DD DSN=UOASXXX.WORK.DATA(OLDMAST),DISP=SHR

//SYSPRINT DD SYSOUT=\*

//SYSOUT DD SYSOUT=\*

//SYSIN DD \*

DELETE (UOASXXX.VSAM.MASTER) CLUSTER

DEFINE CLUSTER ( -

NAME(UOASXXX.VSAM.MASTER) -

INDEXED -

RECORDSIZE(80 80) -

**KEYS(9 0) -**

FREESPACE(19 5) -

VOLUMES (DB1469)-

RECORDS (50 5)-

SHAREOPTIONS (2 3) -

UNIQUE) -

DATA ( -

NAME(UOASXXX.VSAM.MASTER.DATA) -

CISZ(1024)) -

INDEX ( -

NAME(UOASXXX.VSAM.MASTER.INDEX))

REPRO INFILE(INDATA) -

OUTDATASET(UOASXXX.VSAM.MASTER)

IF LASTCC = 0 -

THEN

PRINT INDATASET(UOASXXX.VSAM.MASTER) CHARACTER

Did you receive a condition code of 8?

**Do you remember how to examine the results of a run?**

* TSO SDSF ST is the command to issue.
* Owner acctid
* Type a ? in front of the last job.
* Enter an S in front of the row with a DDNAME of JESYSMSG and press enter.
* What are the condition codes? (Use FIND COND)

The first time this program is submitted, a condition code of 8 is achieved on the DELETE step because the delete statement is trying to delete a cluster that does not yet exist:

DELETE (acctid.VSAM.MASTER) CLUSTER

IDC3012I ENTRY UOASXXX.VSAM.MASTER NOT FOUND

IDC3009I \*\* VSAM CATALOG RETURN CODE IS 8 - REASON CODE IS IGG0CLA3-42

IDC0551I \*\* ENTRY UOASXXX.VSAM.MASTER NOT DELETED

IDC0001I FUNCTION COMPLETED, HIGHEST CONDITION CODE WAS 8

Press F3 to return to the SDSF JOB DATA SET DISPLAY screen.

Type an S in front of the row with a DDNAME of SYSPRINT and press enter.

How many records were processed?

What do you notice about the order of the keys?

Even though you had a condition code of 8 (because the cluster to delete did not exist), the VSAM Cluster is created and a listing should appear near the end of the output display similar to this:

LISTING OF DATA SET -UOASXXX.VSAM.MASTER

KEY OF RECORD - 050555500

050555500GRAUER RT11450877200E59001181800118050000118145000180

KEY OF RECORD - 264805298

264805298CLARK JS12450879100E53330880222087928000108126000080

KEY OF RECORD - 300000000

300000000MILGRAM IR06130580200E655510814000681480000580

KEY OF RECORD - 400000000

400000000BENJAMIN BL10531073100E73331073 30000108128000180

KEY OF RECORD - 638972393

638972393GRAUER J 11450877200E59001181800118050000118145000180

KEY OF RECORD - 800000000

800000000SMITH BB09430678100E64440678 42000068036000078

KEY OF RECORD - 900000000

900000000BAKER E 06130580200E655510814000681480000580

KEY OF RECORD - 955000000

955000000GRAUER B 11450877200E59001181800118050000118145000180

IDC0005I NUMBER OF RECORDS PROCESSED WAS 8

IDC0001I FUNCTION COMPLETED, HIGHEST CONDITION CODE WAS 0

**Note**:After the first run, a condition code of zero is achieved on all steps including DELETE:

DELETE (UOASXXX.VSAM.MASTER) CLUSTER

IDC0550I ENTRY (D) UOASXXX.VSAM.MASTER.DATA DELETED

IDC0550I ENTRY (I) UOASXXX.VSAM.MASTER.INDEX DELETED

IDC0550I ENTRY (C) UOASXXX.VSAM.MASTER DELETED

IDC0001I FUNCTION COMPLETED, HIGHEST CONDITION CODE WAS 0

**Defining an Alternate Key**

After the VSAM cluster has been created (or recreated) using the IDCAMS utility in the JCL code above, the alternative key can be defined. To define the alternate key of last name, the following code is used. Notice that with the RELATE statement, the alternate index is related to the VSAM cluster created above (UOASXXX.VSAM.MASTER). In a simple Cobol program to test the VSAM Cluster alternate index, the name of the alternate index is UOASXXX.VSAM.MASTER.PATH. Notice the KEYS statement with (15 9). The alternate index has a length of 15 and begins at offset 9 in the VSAM cluster. Remember that the first column is really column 0. Columns 0 through 9 contain the social security number. The last name begins in the next column.

To run this program, copy the file named reproai from Blackboard, change all instances of UOASXXX to your acctid, and type SUBMIT.

//REPROAI JOB (AMSEXAI),'UOASXXX',NOTIFY=UOASXXX,

// CLASS=A,MSGLEVEL=(1,1)

/\*

//STEP0010 EXEC PGM=IDCAMS

//SYSPRINT DD SYSOUT=\*

//SYSOUT DD SYSOUT=\*

//ASMDUMP DD SYSOUT=\*

//SYSIN DD \*

DEFINE ALTERNATEINDEX ( -

NAME(UOASXXX.VSAM.MASTER.ALTNDX) -

**RELATE(UOASXXX.VSAM.MASTER) -**

VOLUMES (DB1469)-

RECORDS (50 5)-

**KEYS(15 9) -**

RECORDSIZE(80 80) -

NONUNIQUEKEY -

UPGRADE)

DEFINE PATH -

(NAME(**UOASXXX.VSAM.MASTER.PATH**) -

PATHENTRY(UOASXXX.VSAM.MASTER.ALTNDX))

BLDINDEX INDATASET(UOASXXX.VSAM.MASTER) -

OUTDATASET(UOASXXX.VSAM.MASTER.ALTNDX)

IF LASTCC = 0 -

THEN -

PRINT INDATASET(UOASXXX.VSAM.MASTER.PATH)-

CHARACTER

/\*

//

Follow the same steps as before to examine the output.

The output display should be similar to the following:

LISTING OF DATA SET -UOASXXX.VSAM.MASTER.PATH

KEY OF RECORD - BAKER

900000000BAKER E 06130580200E655510814000681480000580

KEY OF RECORD - BENJAMIN

400000000BENJAMIN BL10531073100E73331073 30000108128000180

KEY OF RECORD - CLARK

264805298CLARK JS12450879100E53330880222087928000108126000080

KEY OF RECORD - GRAUER

050555500GRAUER RT11450877200E59001181800118050000118145000180

KEY OF RECORD - GRAUER

638972393GRAUER J 11450877200E59001181800118050000118145000180

KEY OF RECORD - GRAUER

955000000GRAUER B 11450877200E59001181800118050000118145000180

KEY OF RECORD - MILGRAM

300000000MILGRAM IR06130580200E655510814000681480000580

KEY OF RECORD - SMITH

800000000SMITH BB09430678100E64440678 42000068036000078

IDC0005I NUMBER OF RECORDS PROCESSED WAS 8

IDC0001I FUNCTION COMPLETED, HIGHEST CONDITION CODE WAS 0

**COBOL Program that uses VSAM cluster and Alternate Index.**

The program first searches on primary index (social security numbers of 955000000 and then 245118095) and then by alternate key (last name of GRAUER and then HUMMER). What do you expect the output to be?

Notice the two JCL statements at the bottom of the program that define the VSAM cluster and alternate inces (repeated here for reference). Can you find acctid.VSAM.MASTER and acctid.VSAM.MASTER.PATH in the two programs above?

**STATEMENT TO REFERENCE THE VSAM CLUSTER:**

//GO.VSAMMAST DD DSN=UOASXXX.VSAM.MASTER,DISP=SHR

**STATEMENT JCL TO REFERENCE THE VSAM ALTERNATE KEY:**

//GO.VSAMMAS1 DD DSN=UOASXXX.VSAM.MASTER.PATH,DISP=SHR

Copy the file vsamex01 on Blackboard to a PDS with a type such as COMPILES that accepts 80 column characters. Change all instances of UOASXXX to your ACCTID. Type Submit and press enter.

**COBOL PROGRAM:**

//AIXEXAMP JOB (AIVSAM),'UOASXXX',NOTIFY=UOASXXX,

// CLASS=A,MSGLEVEL=(1,1),TIME=1,MSGCLASS=A

//STEP1 EXEC PROC=IGYWCLG

//COBOL.SYSIN DD \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* \*

\* This program was modified from the code in text-- \*

\* The IBM Cobol Environment by Robert T. Grauer, page 120 \*

\* \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

IDENTIFICATION DIVISION.

PROGRAM-ID. ALTINDEX.

AUTHOR. Douglas.

ENVIRONMENT DIVISION.

INPUT-OUTPUT SECTION.

FILE-CONTROL.

SELECT INDEXED-FILE

ASSIGN TO DA-VSAMMAST

ORGANIZATION IS INDEXED

ACCESS IS DYNAMIC

RECORD KEY IS INDEX-SOC-SEC-NUMBER

ALTERNATE RECORD KEY IS INDEX-NAME

WITH DUPLICATES.

DATA DIVISION.

FILE SECTION.

FD INDEXED-FILE

LABEL RECORDS ARE STANDARD

RECORD CONTAINS 80 CHARACTERS

DATA RECORD IS INDEXED-RECORD.

01 INDEXED-RECORD.

05 INDEX-SOC-SEC-NUMBER PIC X(9).

05 INDEX-NAME PIC X(15).

05 REST-OF-INDEXED-RECORD PIC X(56).

WORKING-STORAGE SECTION.

01 FILLER PIC X(14)

VALUE 'WS BEGINS HERE'.

01 WS-NDX-MAST-RECORD.

05 NDX-SOC-SEC-NUMBER PIC X(9).

05 NDX-NAME.

10 NDX-LAST-NAME PIC X(15).

10 NDX-INITIALS PIC XX.

05 FILLER PIC X(54).

01 WS-ACTIVE-NAME PIC X(15).

01 WS-BALANCE-LINE-SWITCHES.

05 WS-RECORD-KEY-ALLOCATED-SWITCH PIC X(3).

05 WS-END-INDEX-FILE PIC X(3).

PROCEDURE DIVISION.

0010-PROCESS-NAME-FILE.

OPEN INPUT INDEXED-FILE.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\* RETRIEVE RECORDS BY SOCIAL SECURITY NUMBER \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

MOVE '955000000' TO INDEX-SOC-SEC-NUMBER.

PERFORM 0040-READ-INDEX-FILE-BY-NUMBER.

IF WS-RECORD-KEY-ALLOCATED-SWITCH = 'YES'

DISPLAY WS-NDX-MAST-RECORD

ELSE

DISPLAY ' '

DISPLAY 'NO MATCH FOUND FOR: 955000000'.

MOVE '245118095' TO INDEX-SOC-SEC-NUMBER.

PERFORM 0040-READ-INDEX-FILE-BY-NUMBER.

IF WS-RECORD-KEY-ALLOCATED-SWITCH = 'YES'

DISPLAY WS-NDX-MAST-RECORD

ELSE

DISPLAY ' '

DISPLAY 'NO MATCH FOUND FOR: 245118095'.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\* RETRIEVE RECORDS BY NAME - THE ALTERNATE RECORD KEY \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

MOVE 'GRAUER ' TO INDEX-NAME

WS-ACTIVE-NAME.

PERFORM 0020-READ-INDEX-FILE-BY-NAME.

IF WS-RECORD-KEY-ALLOCATED-SWITCH = 'NO'

DISPLAY ' '

DISPLAY 'NO MATCH FOUND FOR: ' WS-ACTIVE-NAME

ELSE

PERFORM 0030-READ-DUPLICATES

UNTIL WS-ACTIVE-NAME NOT = NDX-LAST-NAME

OR WS-END-INDEX-FILE = 'YES'.

MOVE 'HUMMER ' TO INDEX-NAME

WS-ACTIVE-NAME.

PERFORM 0020-READ-INDEX-FILE-BY-NAME.

IF WS-RECORD-KEY-ALLOCATED-SWITCH = 'YES'

DISPLAY WS-NDX-MAST-RECORD

ELSE

DISPLAY ' '

DISPLAY 'NO MATCH FOUND FOR: ' WS-ACTIVE-NAME

CLOSE INDEXED-FILE.

STOP RUN.

0020-READ-INDEX-FILE-BY-NAME.

MOVE 'YES' TO WS-RECORD-KEY-ALLOCATED-SWITCH.

READ INDEXED-FILE INTO WS-NDX-MAST-RECORD

KEY IS INDEX-NAME

INVALID KEY

MOVE 'NO' TO WS-RECORD-KEY-ALLOCATED-SWITCH.

0030-READ-DUPLICATES.

DISPLAY WS-NDX-MAST-RECORD.

READ INDEXED-FILE NEXT RECORD

INTO WS-NDX-MAST-RECORD

AT END

MOVE 'YES' TO WS-END-INDEX-FILE.

0040-READ-INDEX-FILE-BY-NUMBER.

MOVE 'YES' TO WS-RECORD-KEY-ALLOCATED-SWITCH.

READ INDEXED-FILE INTO WS-NDX-MAST-RECORD

INVALID KEY

MOVE 'NO' TO WS-RECORD-KEY-ALLOCATED-SWITCH.

//\*

//GO.VSAMMAST DD DSN=UOASXXX.VSAM.MASTER,DISP=SHR

//GO.VSAMMAS1 DD DSN=UOASXXX.VSAM.MASTER.PATH,DISP=SHR

/\*

**OUTPUT from the Program**

Follow the same steps as before to examine the output.

Are the condition codes all 0 when you show the row with the DDNAME of JESYSMSG?

To see your output, enter an s in front of the row with the ProcStep equal to GO.

The output from the program you should see is:

\*\*\*\* END OF MESSAGE SUMMARY REPORT \*\*\*\*

955000000GRAUER B 11450877200E59001181800118050000118145000180

NO MATCH FOUND FOR: 245118095

050555500GRAUER RT11450877200E59001181800118050000118145000180

638972393GRAUER J 11450877200E59001181800118050000118145000180

955000000GRAUER B 11450877200E59001181800118050000118145000180

**NO MATCH FOUND FOR: HUMMER**

Did your expected output match the actual output?

**Review Questions:**

* Does the order of the records matter? Explain.
* Must the alternate index have unique values?
* Can you explain the KEYS statement?
* What if the original data file were changed to the following – what would be used for the KEYS statement in each of the two programs to generate the VSAM cluster and the alternate index? For your reference, the first row is included so that you can count the column.

1 2 3 4 5 6 7

1234567890123456789012345678901234567890123456789012345678901234567890

050500GRAUER RT11450877200E59001181800118050000118145000180

264898CLARK JS12450879100E53330880222087928000108126000080

300000MILGRAM IR06130580200E655510814000681480000580

400000BENJAMIN BL10531073100E73331073 30000108128000180

638993GRAUER J 11450877200E59001181800118050000118145000180

800000SMITH BB09430678100E64440678 42000068036000078

900000BAKER E 06130580200E655510814000681480000580

955000GRAUER B 11450877200E59001181800118050000118145000180