



VSAM (Virtual Storage Access Method)

Lesson 00: Overview



Course Goals and Non Goals

Course Goals

- Write JCL to use IDCAMS utilities to handle VSAM objects

Course Non Goals

- Accessing VSAM datasets through CICS

Pre-requisites



COBOL

MVS

JCL

Intended Audience

Software Engineers
Programmers





Day Wise Schedule

Day 1

- Lesson 1: Introduction to VSAM
- Lesson 2: Inside VSAM Datasets
- Lesson 3: IDCAMS Commands

Day 2

- Lesson 4: Using LISTCAT
- Lesson 5: Creating Alternate Indexes
- Lesson 6: Cobol VSAM Considerations

Day 3

- Lesson 7: Reorganizing VSAM datasets
- Lesson 8: VERIFY, PRINT, DELETE, ALTER Command
- Lesson 9: Generation Data Groups



Table of Contents

Lesson 1: Introduction to VSAM

- 1.1. Introduction to VSAM
- 1.2. VSAM History
- 1.3. Features of VSAM
- 1.4. Types of VSAM Datasets
- 1.5. Benefits of VSAM

Lesson 2: Inside VSAM Datasets

- 2.1. Control Interval
- 2.2. Spanned Records
- 2.3. KSDS Structure



Table of Contents

Lesson 3: IDCAMS Commands

- 3.1. Accessing VSAM Data
- 3.2. Types of IDCAMS Commands
- 3.3. Format of IDCAMS Command
- 3.4. IDCAMS Return Codes
- 3.5. The DEFINE CLUSTER Command

Lesson 4: Using LISTCAT

- 4.1. Introduction to LISTCAT
- 4.2. Various LISTCAT Options
- 4.3. To Execute LISTCAT from TSO prompt



Table of Contents

Lesson 5: Creating Alternate Indexes

- 5.1. Defining and Building Alternate Indexes
- 5.2. Defining Paths
- 5.3. Building the Alternate Index

Lesson 6: COBOL VSAM Considerations

- 6.1. Alternate Index Processing
- 6.2. Alternate Index Processing
- 6.3. VSAM I/O Error Processing
- 6.4. COBOL FILE STATUS Key



Table of Contents

Lesson 7: Reorganizing VSAM datasets

- 7.1. REPRO command
- 7.2. The FROMKEY and TOKEY parameters
- 7.3. The FROMADDRESS and TOADDRESS parameters
- 7.4. The FROMNUMBER and TONUMBER parameters
- 7.5. The SKIP and COUNT parameters
- 7.6. Backing up VSAM Datasets



Table of Contents

Lesson 8: VERIFY, PRINT, DELETE, ALTER Command

- 8.1. Verify Command
- 8.2. Delete Command
- 8.3. Print Command
- 8.4. Alter Command

Lesson 9: Generation Data Groups

- 9.1. Generation Data Groups
- 9.2. Specifications to Create a GDG
- 9.3. Features of GDG



Table of Contents

Lesson 9: Generation Data Groups (Contd.)

- 9.4. Creating a GDG
- 9.5. Modifying GDG
- 9.6. Deleting GDG
- 9.7. Deleting GDG Index and Datasets
- 9.8. Add Dataset to GDG
- 9.9. Considerations to Use GDGs
- 9.10. Characteristics of GDGs
- 9.11. Advantages of GDGs

References



MVS/VSAM for Application Programmer
by Brown and Smith
VSAM by Doug Lowe
VSAM for COBOL Programmer by Doug Lowe

Next Step Courses (if applicable)



CICS



VSAM (Virtual Storage Access Method)

Lesson 01: Introduction to VSAM



Lesson Objectives

In this lesson, you will learn about:

- Overview of VSAM
- History of VSAM
- Features of VSAM
- Types of VSAM Datasets
- Advantages and Disadvantages of VSAM

Company Internal



1.1: Introduction to VSAM

Overview

Virtual Storage Access Method (VSAM) is a high performance access method and dataset organization.

It is one of the several access methods that define the technique by which data is stored and retrieved.

It was introduced by IBM in the 1970s as part of the OS/VS1 and OS/VS2 operating systems.

It organizes and maintains data via a cataloged structure.

It allows you to access files of different organization such as sequential, indexed, relative record, and linear datasets.

Company Internal



1.2: History of VSAM Description

Some important landmarks in history of VSAM are as follows:

- 1973: First VSAM introduced - Standard VSAM
 - This version had only ESDS and KSDS.
- 1975: Enhanced VSAM introduced
 - This added RRDS and alternate index for KSDS.
- 1979: DF/EF(Data Facility Extended function) VSAM introduced
 - It was introduced with ICF (Integrated catalog facility) to replace old VSAM catalog of previous version.

Company Internal



Description

1983: DFP/VSAM version 1(Data facility product) introduced

- It was introduced to replace DF/EF VSAM entirely and run under MVS/XA (Multiple virtual storage extended architecture).

1987: DFP/VSAM version 2, Release 3.0 introduced

- It was introduced to replace Version 1 entirely. LDS were added.

1988: DFP/VSAM version 3 introduced

- It was introduced to replace Version 2 entirely to run under MVS/ESA (Enterprise Systems Architecture).

1989: DFP/VSAM version 3, Release 3 introduced

- It was introduced to replace DFP/VSAM Version entirely

Company Internal



1.3: Features of VSAM

Salient Features

Some of the salient features of VSAM are as follows:

- It serves as one coherent file storage system (Stores and Retrieves Data).
- It is not a database management system.
- It is not a programming language.
- It is not a communication system.
- VSAM has no equivalent for a 'PDS'.
- It was introduced by IBM in 1973.
- DFP/VSAM Ver 1 was introduced in 1989.

Company Internal



VSAM Datasets

VSAM provides the file organizations that correspond to the three types of file organizations provided by the native access methods (ESDS, KSDS, RRDS).

- While allocating a VSAM dataset, the following needs to be specified:
 - the dataset name,
 - type of the dataset,
 - units of space (tracks, cylinders, or records),
 - serial number of volume, length of records

Company Internal



1.4: Types Of VSAM Datasets

Explanation

Based on the way in which we store and access the records, there are following types of VSAM Datasets:

- KSDS : Key Sequenced dataset
- ESDS : Entry Sequenced dataset
- RRDS : Relative record dataset
- LDS : Linear dataset

All the four methods contain a data area in which data records are placed.

For KSDS, there is an index area in addition to data area.



Explanation

Entry Sequenced Datasets:

- These are familiar sequential datasets, and they can be read or written only in a sequential order.

Key Sequenced Datasets:

- These are datasets stored in order of key field in the record.

Relative Record Datasets:

- These are datasets stored in some sequential order.

Linear Datasets:

- These are datasets that consist of a long stream of bytes.

Company Internal



VSAM ESDS DATASET

In an ESDS, records are stored in the order in which they were entered.

New records are always added at the end of the dataset i.e. Records can only be appended.

Records can be of fixed & variable length.

Physically an ESDS is stored as only one component (No Index Component) i.e. Comprises of only data component

Company Internal



ESDS Example

- The example shows an entry-sequenced data set (ESDS) that was created in employee number sequence.
- Here records are stored in consecutive order.

Disk Location	Employee Number	First Name	Middle Name	Last Name	Social Security Number
1	01001	Stanley	A	Dcosta	123-23-3333
2	01002	Thomas	B	Dislva	455-78-9876
3	01003	William	S	Andrue	343-12-2212
4	01004	Alice	D	Siebart	565-77-3737
5	01005	Jones	T	Colloins	787-89-3334
6	01007	Jack	E	Cowley	323-78-9879
7	01008	Jill	R	Ababort	123-45-8766
8	01009	Pitter	T	Brothers	343-76-4445
9	01010	Ritchie	Y	Glenn	389-16-8989
10	01013	Loran	S	Andru	444-98-5555

Company Internal



VSAM KSDS DATASET

Comprises of both data and index component

Sequenced based on the key Topic Details

part of record Key should be Unique

Contiguous Free space is provided during the cluster creation

Helps during insertion/updation of records

Company Internal



VSAM KSDS DATASET

- Records can be deleted
- Primary key cannot be changed
- Allows fixed, variable-length records
- Provides sequential, random and dynamic access
- Allows spanned records
- Alternate index can be created

Company Internal



KSDS Example

- The example shows a key-sequenced data set (KSDS) that uses employee number as the key.

Employ ee Number	Disk Locati on	Disk Locatio n	Employ ee Number	First Name	Middle Name	Last Name	Social Security Number
01001	9	1	01003	William
01002	2	2	01002	Thomas
01003	1	3	01007	Jack
01004	6	4	01008	Jill
01005	5	5	01005	Jones
01006	3	6	01004	Jack
01007	4	7	01009	Pitter
01008	7	8	01013	Ioran
01010	10	9	01001	Stanley
01015	8	10	01010	Ritchie

Company Internal



VSAM RRDS DATASET

An RRDS is a series of continuous number of pre-formatted slots of fixed-length.

Each slot may or may not contain a record.

Each slot is identified by its position relative to the first slot in the dataset. This relative position is called the Relative Record Number (RRN) of the slot.

RRDS records can be inserted, retrieved, updated and deleted.

Records can be accessed sequentially and randomly.

Company Internal



RRDS Example

- The example shows a relative record data set (RRDS).
- It has a simple relationship between employee number and relative record number.

Relative Record number	Employee Number	First Name	Middle Name	Last Name	Social Security Number
1	01001	Stanley	A	Dcosta	123-23-3333
2	01002	Thomas	B	Dislva	455-78-9876
3	01003	William	S	Andrue	343-12-2212
4	01004	Alice	D	Siebart	565-77-3737
5	01005	Jones	T	Colloins	787-89-3334
6					
8	01007	Jack	E	Cowley	323-78-9879
9	01008	Jill	R	Ababort	123-45-8766
10	01009	Pitter	T	Brothers	343-76-4445
11	01010	Ritchie	Y	Glenn	389-16-8989
12					
13	01013	Loran	s	andru	444-98-5555

Company Internal



Linear Data Set (LDS)

A Linear Data Set (LDS) contains data that can be accessed as byte-addressable strings in virtual storage.

It is a VSAM data set with a CI size of 4096 bytes. An LDS has no imbedded control information in its CI, that is no RDFs and CIDFs.

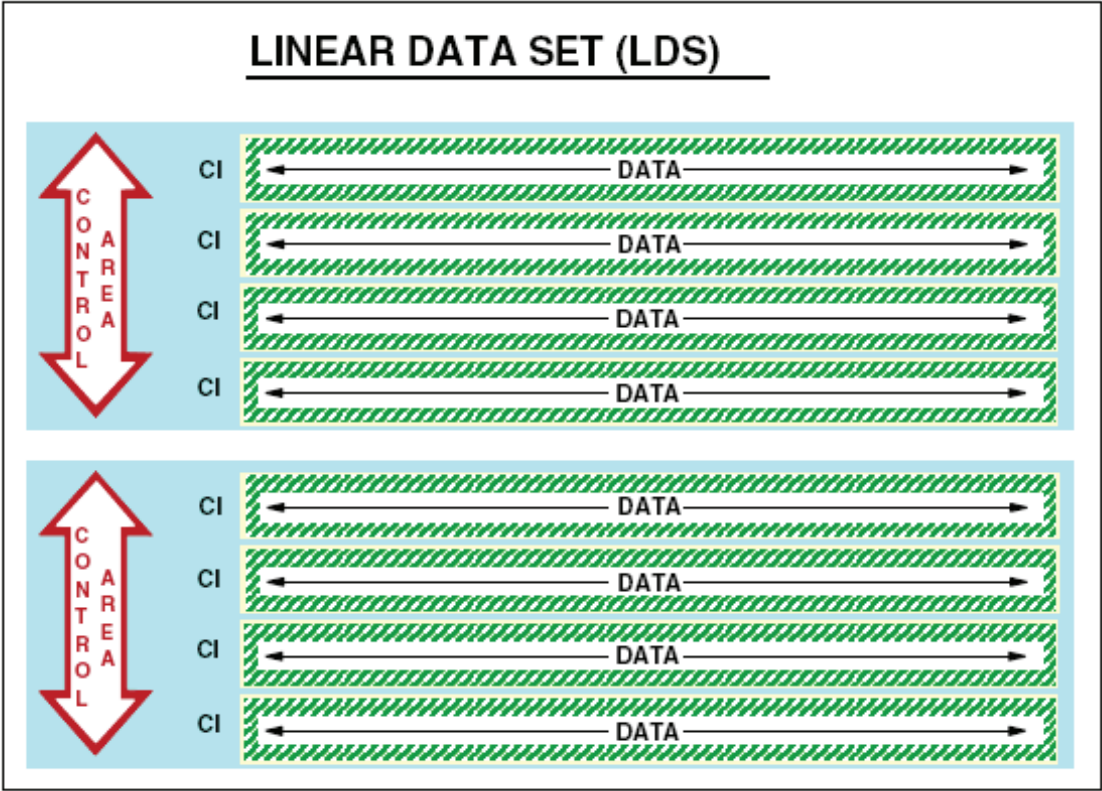
All LDS bytes are data bytes.

Logical records must be “blocked” and “deblocked” by the application program.

However, records do not exist from the point of view of VSAM.

Company Internal

Linear Data Set (LDS)



Company Internal



COMPARISON OF ESDS, KSDS AND RRDS

	ESDS	KSDS	RRDS
Sequential access is by	entry order	primary key order	RRN
Direct access is by	RBA	key RBA	RRN
Record format can be	fixed variable spanned	fixed variable spanned	fixed
Changeable record length	no	yes	no
New records added to	end of file	anywhere	RRN slot (if empty)
Embedded free space defined	no	yes	no
Delete records and reuse space	no	yes	yes
Access through alternate index	yes	yes	no

Company Internal



VSAM Catalogs

VSAM provides comprehensive catalog facility that stores information about VSAM datasets and other files.

The catalog keeps track of the unit and volume on which the dataset resides and can be used for later retrieval of the dataset.

Company Internal



VSAM Catalogs

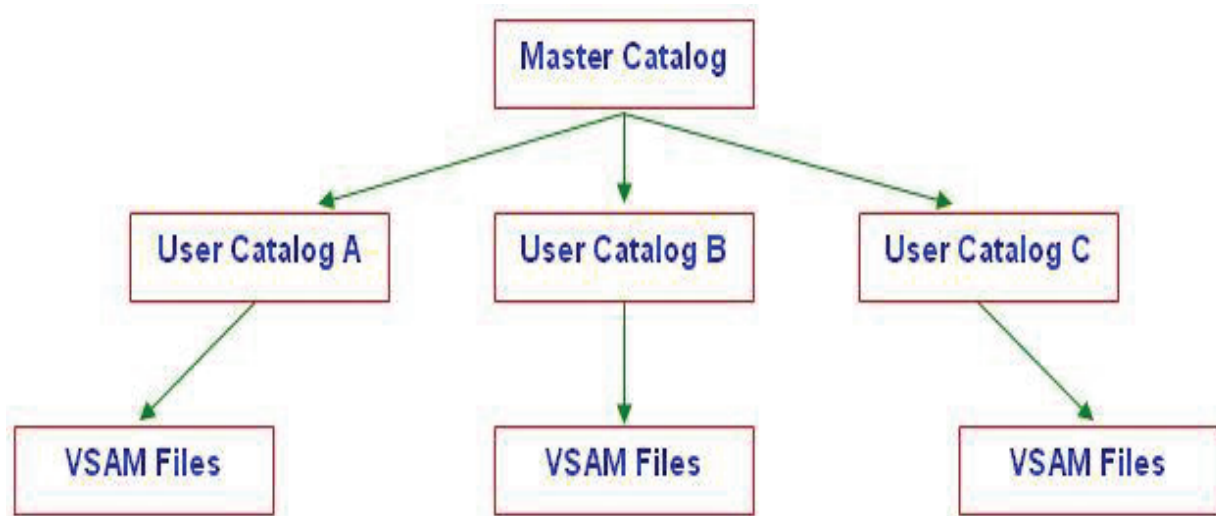
Two types of VSAM catalogs are available, namely:

- Master catalogs
- User catalogs

Company Internal



Master Catalog Hierarchy



Company Internal

Clusters

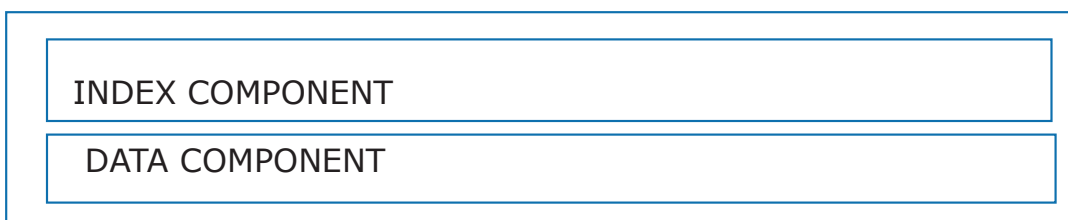


VSAM files are often called clusters.

A cluster is a set of catalog entries that represent a file.

- A cluster consists of one or two components, a data component, which contains the actual records of a file, and for a KSDS only, an index component, which contains the indexes used to access records in the data component.

KSDS Cluster:



Company Internal



VSAM Datasets Advantages

Protection of Data

Cross-system (MVS & VSE) Compatibility

Device Independence (Access Via Catalog)

Provision to include IDCAMS commands in JCL to handle VSAM

Provision for VSCOBOL II/PL-I/ASSEMBLY Language to access VSAM dataset

Company Internal



Disadvantages Of VSAM

VSAM has the following disadvantages:

VSAM requires more storage space than other types of datasets, this is because VSAM datasets carry control information in them, in addition to the actual data that they are comprised of.

VSAM datasets require additional storage free space that must be embedded in them. This free space results in more efficient management of the data contained in them, when records are added, deleted or changed.

Integrity of VSAM datasets in cross-region and cross-system sharing must be controlled by the user.

Company Internal



Summary

In this lesson, you have learnt:

- Overview of VSAM
- History of VSAM
- Features of VSAM
- Types of VSAM datasets
- Advantages of VSAM

Company Internal



Review Questions

Question 1: Records in a ESDS may be accessed sequentially, in order by key value, or directly, by supplying the key value of the desired record.

- True / False

Question 2: A ____ cluster consists of two physical parts, an index component, and a data component.

Question 3: VSAM is single coherent file storage system used to store and retrieve data.

- True/False

Company Internal



VSAM (Virtual Storage Access Method)

Lesson 02: Inside VSAM Datasets



Lesson Objectives

In this lesson, you will learn:

- Control Interval
- Spanned records
- Control Area
- KSDS structure



2.1: Control Interval

What is a Control Interval?

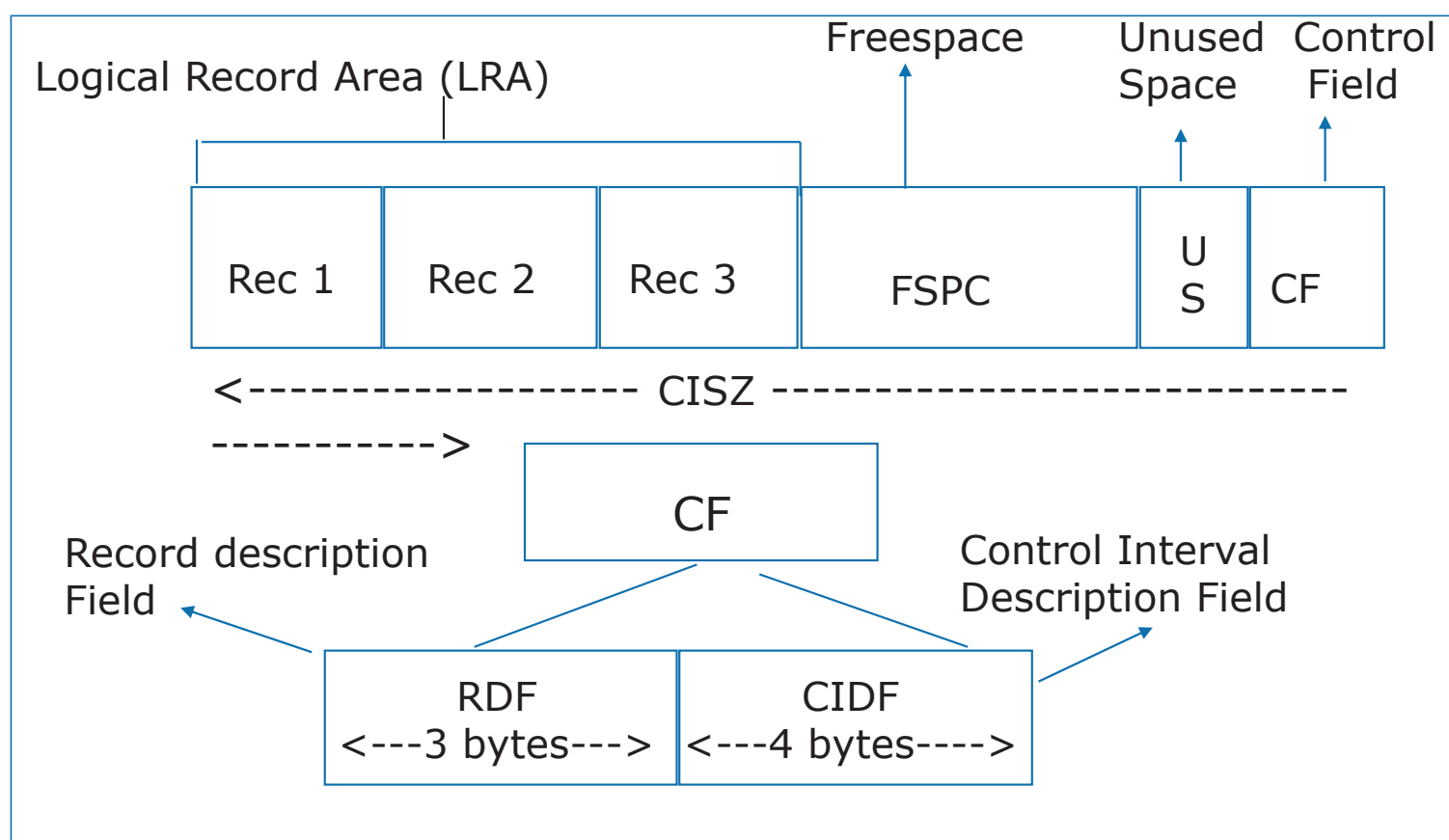
A Control Interval (CI) is the unit of data VSAM transfers between virtual and disk storage.

- It is similar to the concept of blocking in non-VSAM files.
- Generally, it contains more than one record.
- The size of a CI must be between 512 bytes to 32K.
- Up to 8K bytes it must be in a multiple of 512, beyond this it must be in a multiple of 2K.
- It has a fixed length, selected at file creation time.
- For index, the size of CI is 512, 1K, 2K, or 4K bytes.



2.1: Control Interval

CI for the Data Component





2.1: Control Interval

Format of a Control Interval

The logical record area (LRA):

- It contains the actual data records.

The free space (FSPC):

- You can specify the % of a CI to reserve as free space when you create a VSAM dataset.
 - This is done with the FREESPACE parameter of IDCAMS DEFINE CLUSTER.

This area can then be used for adding new records.

Unused Space:

- Due to the record length, the free space may not be a multiple of the record length, and there may be a small amount of the space that cannot be used.
- This can be minimized for fixed-length records by selecting a proper control interval size.



2.1: Control Interval

Format of a Control Interval (contd.)

Control Field:

- It specifies the position and length of the free space within the record.
- CIDF contains information pertaining to control interval itself, such as location and offset in bytes of free space.
- Record Descriptor Field (RDF) contains the length of each record and how many adjacent records are of the same length.
- There is one RDF for each record in variable length records.



2.1: Control Interval

Control Interval Description Field (CIDF)

There is one RDF for each record in variable length records in the simplest case. There will be only two RDF per CI in case of fixed length records.

- One RDF (RDF l) specifies the length of the record, and the second RDF (RDF c) specifies how many adjacent records are there in the CI.
- Each RDF and CIDF is of 3 and 4 bytes respectively for same length records.
- CF is calculated by CIDF size and RDF size.



2.1: Control Interval

How Control Information is stored in CI?

- **Examples:**

Control Interval 1 (1024 bytes)

Logical Record 1	Logical Record 2	Logical Record 3	Logical Record 4	Free space	RDF F 4	RDF3	RDF2	RDF1	CIDF
185	292	128	147	3	3	3	3	3	4

Control Interval 2 (1024 bytes)

Logical Record 1	Logical Record 2	Logical Record 3	Logical Record 4	Logical Record 5	Logical Record 6	RDF C	RDF L	CIDF
169	169	169	169	169	169	3	3	4

Control Interval 3 (1024 bytes)

Logical Record 1	Logical Record 2	Logical Record 3	Logical Record 4	Free space	RDF F 4	RDF C	RDF L	RDF1	CIDF
180	292	292	180	196	3	3	3	3	4



2.1: Control Interval

Control Interval Size

Reserve 10 bytes for Control Information 10

Reserve 20% of the assigned CISZ for free space. 819

Add 1 and 2. 829

Subtract 829 from 4096. 3267

Divide 3267 by 80 to obtain number of data records per data control interval (VSAM round down). 40

The 67 byte remainder represents the portion of control interval that will be unusable.



2.2: Spanned Records

Concept of Spanned Records

VSAM allows maximum record length of 32,767 bytes.

When a record is longer than a CI, VSAM stores as much of the record as will fit in one CI. Subsequently, it spills over the remaining of the record into the next CI, this is concept of spanned records.

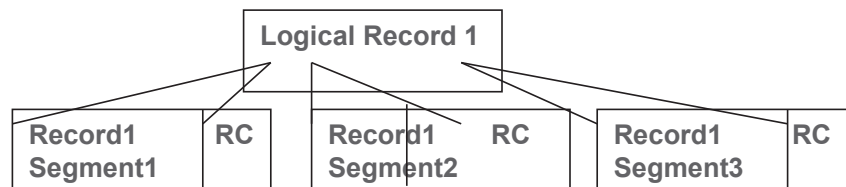
Concept of Spanned Records



The records can span a CI but not a CA.

The start of each spanned record begins in a new CI and spills over to the new CI, as necessary.

The Unused space at the end of last CI is wasted.

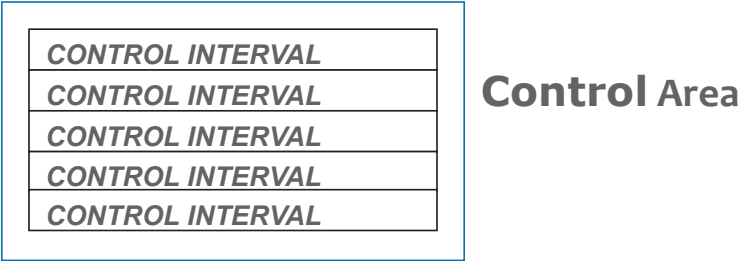




Concept of Control Area

A Control Area is a group of adjacent control intervals.

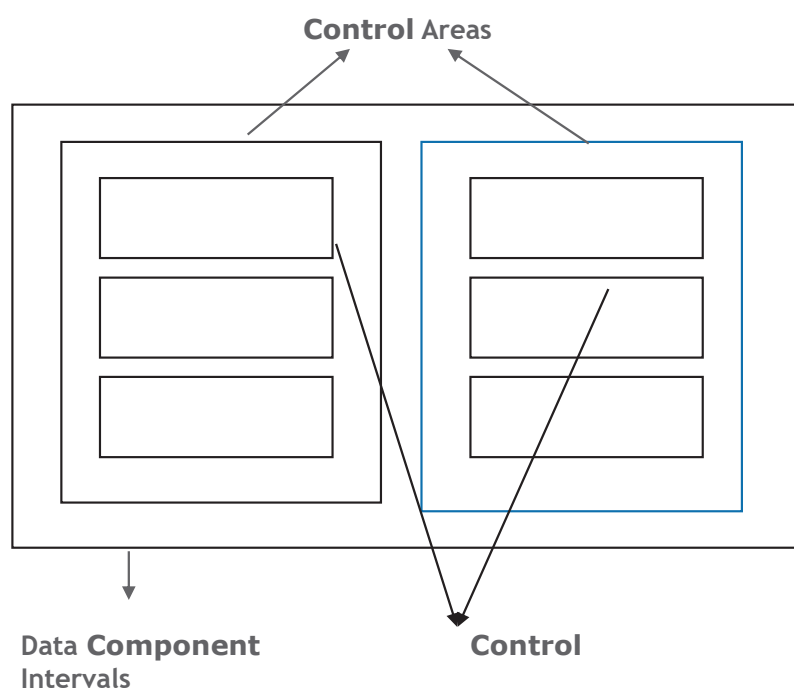
- It can be small as one track of as large as one cylinder.
- You can also specify free space in a control area.
- The total number of CI/CA in a cluster is determined by VSAM.
- VSAM determines the control area size based on space allocation.





Control Area

Contents of Control Area (CA)





2.3; KDS Structure

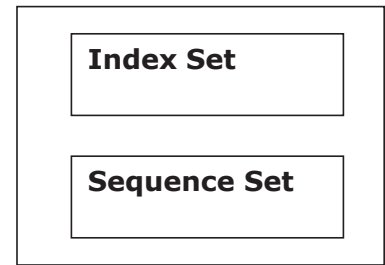
Composition of KSDS Structure

A KSDS consists of two components:

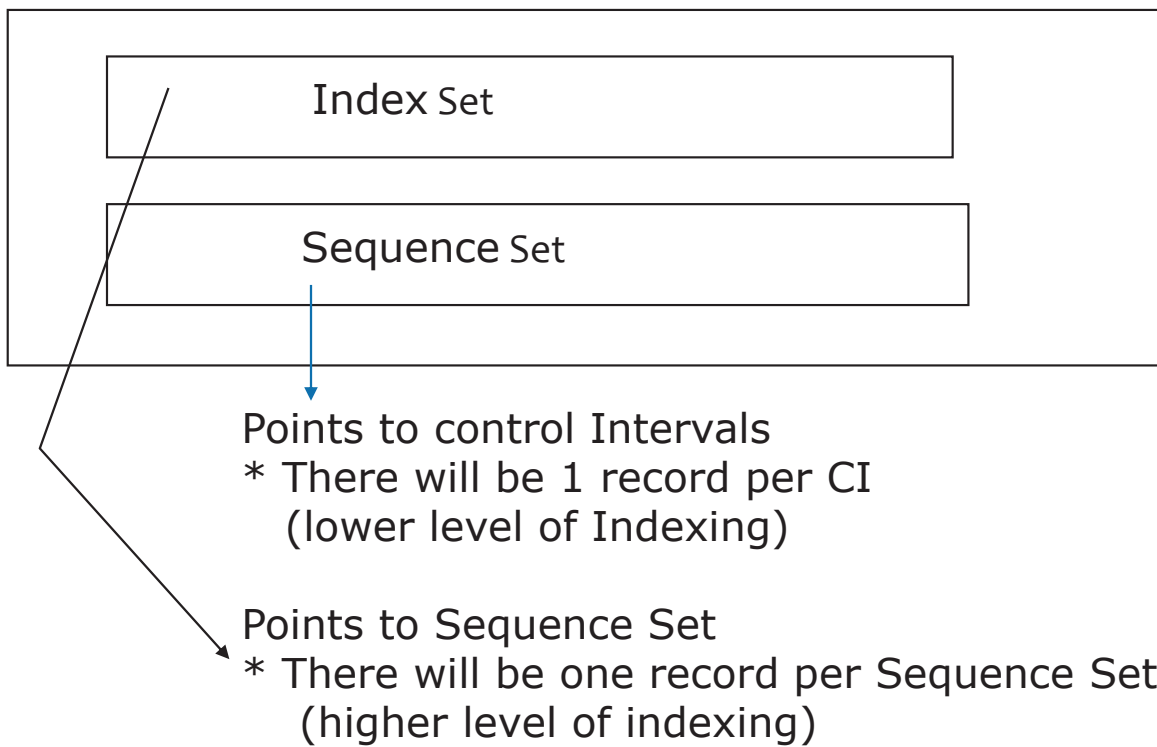
- Data Component
- Index Component

Index component has two parts:

- Sequence set
- Index set



Index Component



Insertion of Records



Before Insertion:

New
Record

Record 1	Record 2	Record 3	FSPC	U S	CF
----------	----------	----------	------	--------	----

After Insertion:

Record 1	Record 2	New Record	Record 3	FSPC	U S	CF
----------	----------	------------	----------	------	--------	----

Deletion of Records



To Delete

Before Deletion:

Record 1	Record 2	New Record	Record 3	FSPC	U S	CF
----------	----------	------------	----------	------	--------	----

After Deletion:

Record 1	Record 2	Record 3	FSPC	U S	CF
----------	----------	----------	------	--------	----



Control Interval Split

Control Interval Splits

Before inserting the record first system locates where the record is to be inserted in the file & then tries to find enough free space in appropriate CI for that record.

If new record requires more space than is available in free space area, then the system tries to locate a free CI within the same CA, to store the record.



Control Interval Split (contd..)

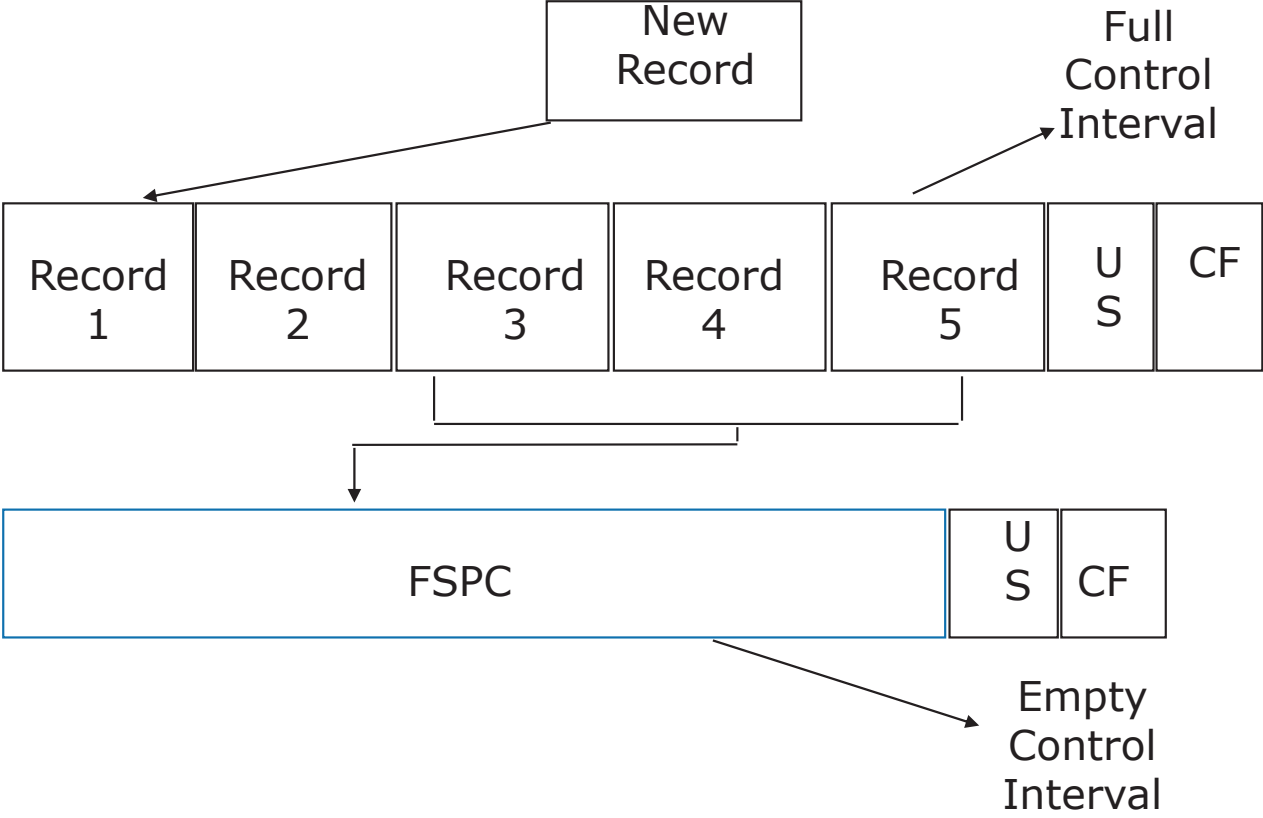
If one is found then approximately half of the records from the current CI are moved to the available CI.

The record is added at this point of split

There are always some percentage of free CI's kept in a CA and CI split may trigger CA split.

Control Interval Splits are possible only in case of KSDS.

Before Control Interval Split



After Control Interval Split



Record 1	Record 2	New Record	FSPC	U S	CF
Record 3	Record 4	Record 5	FSPC	U S	CF

Before Control Interval Split



Sequence set

I	0	K	100	
---	---	---	-----	--

Record C

Data Component

0

Record A	Record B	Record F	Record G	US	CF
----------	----------	----------	----------	----	----

100

Record J	Record K	FSPC	US	CF
----------	----------	------	----	----

200

FSPC	US	CF
------	----	----

After Control Interval Split



Sequence set	E	0	I	200	K	100
--------------	---	---	---	-----	---	-----

Data Component	0	Record A	Record B	Record C	FSPC	U S	CF
	100	Record J	Record K	FSPC		U S	CF
	200	Record F	Record G	FSPC		U S	CF

Control Interval Split



Seq Set

060	180	190		
00	012	018	025	050
061	100	145	170	
181	190			

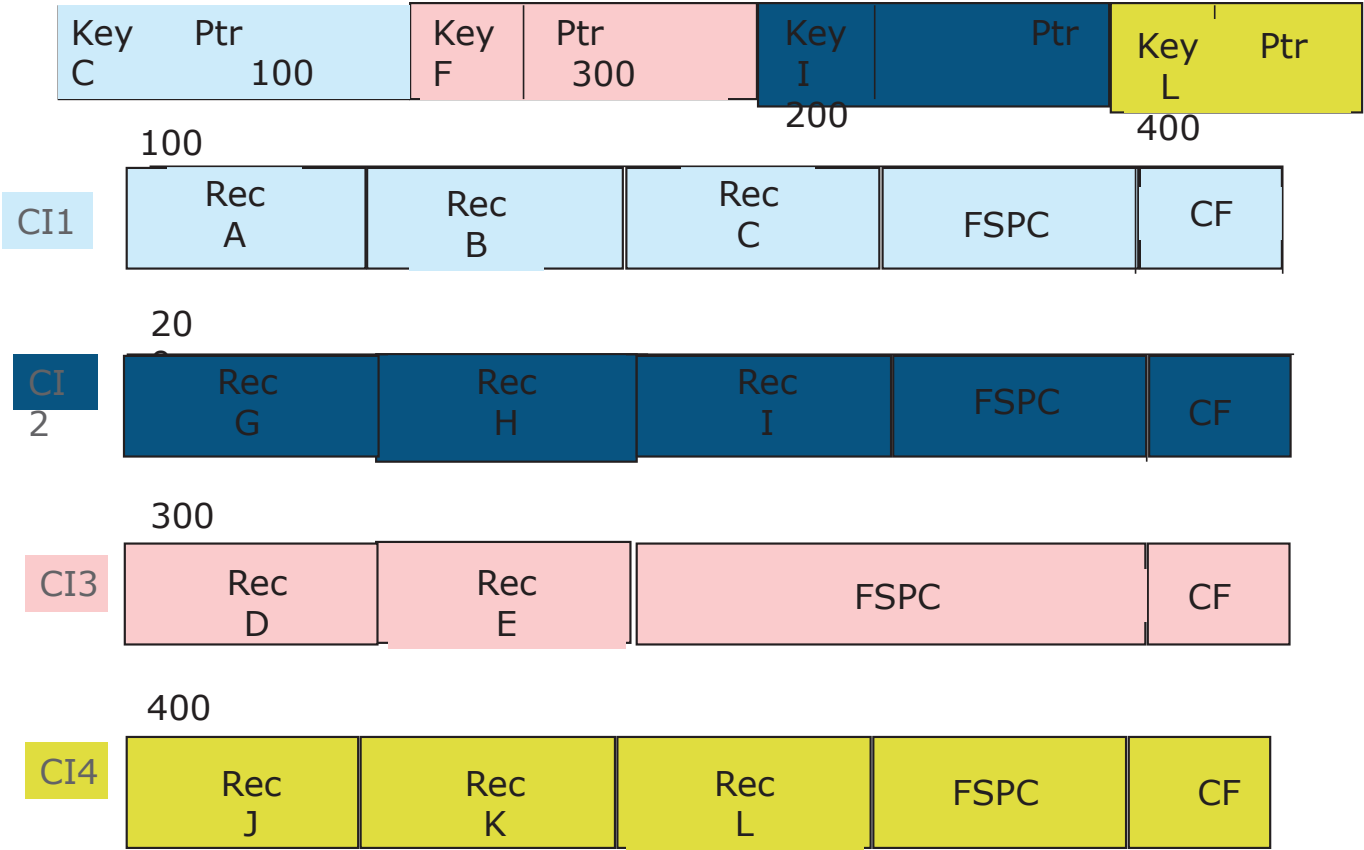
DATA AREA

After adding 030 and 110

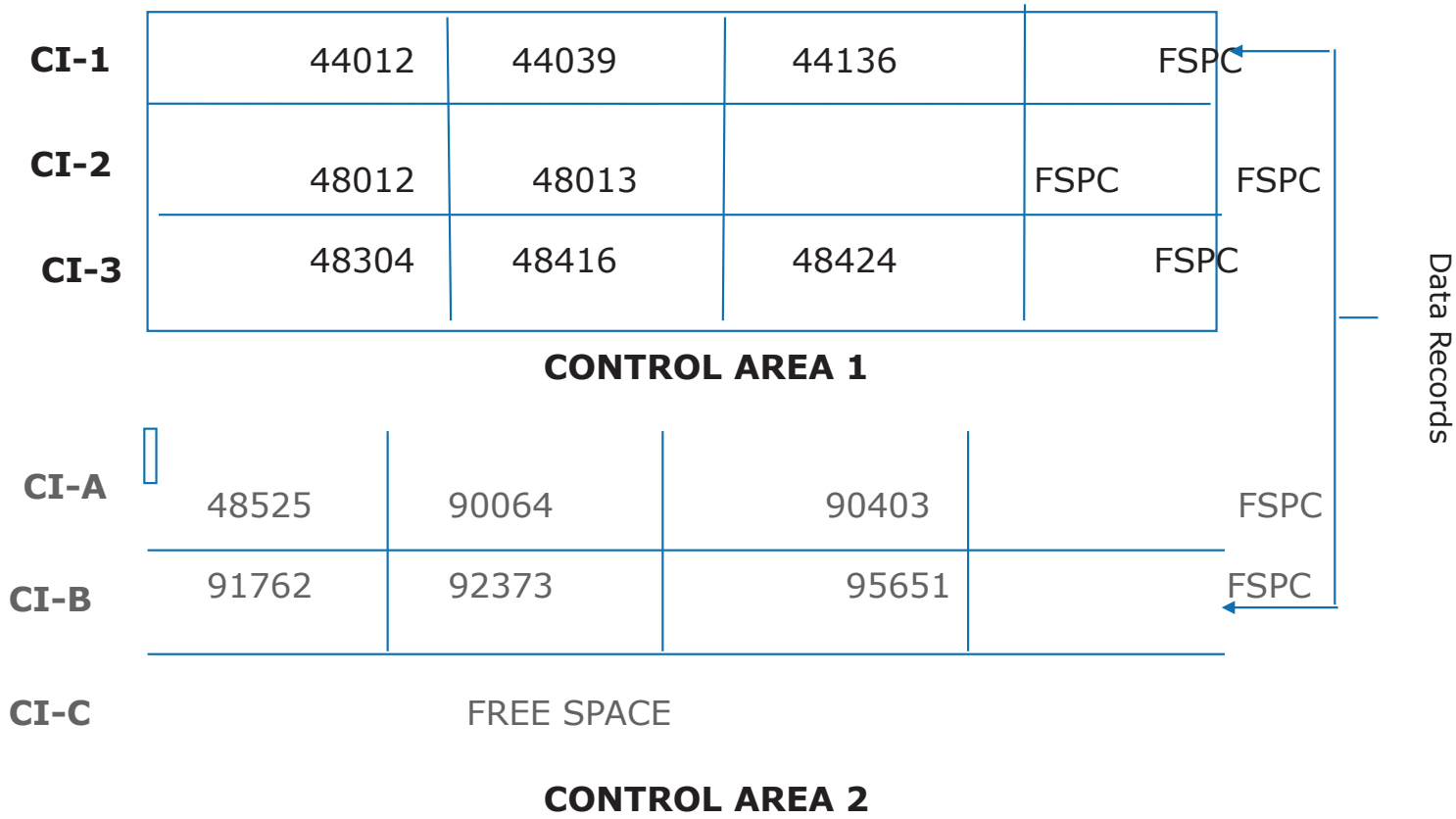
Seq Set

020	060	180	190	
001	012	018		
061	100	110	145	170
181	190			
025	030	050		

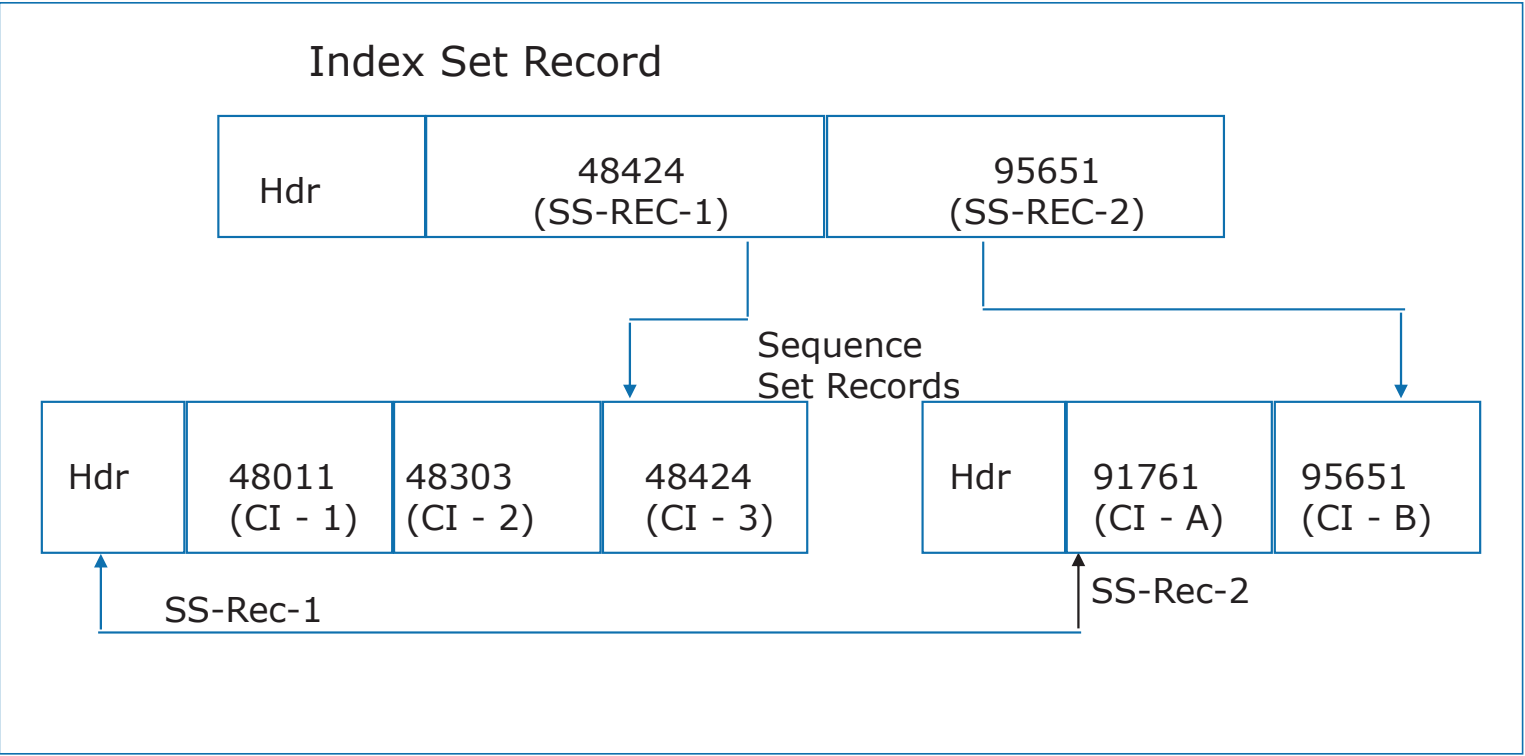
Sequence of KSDS



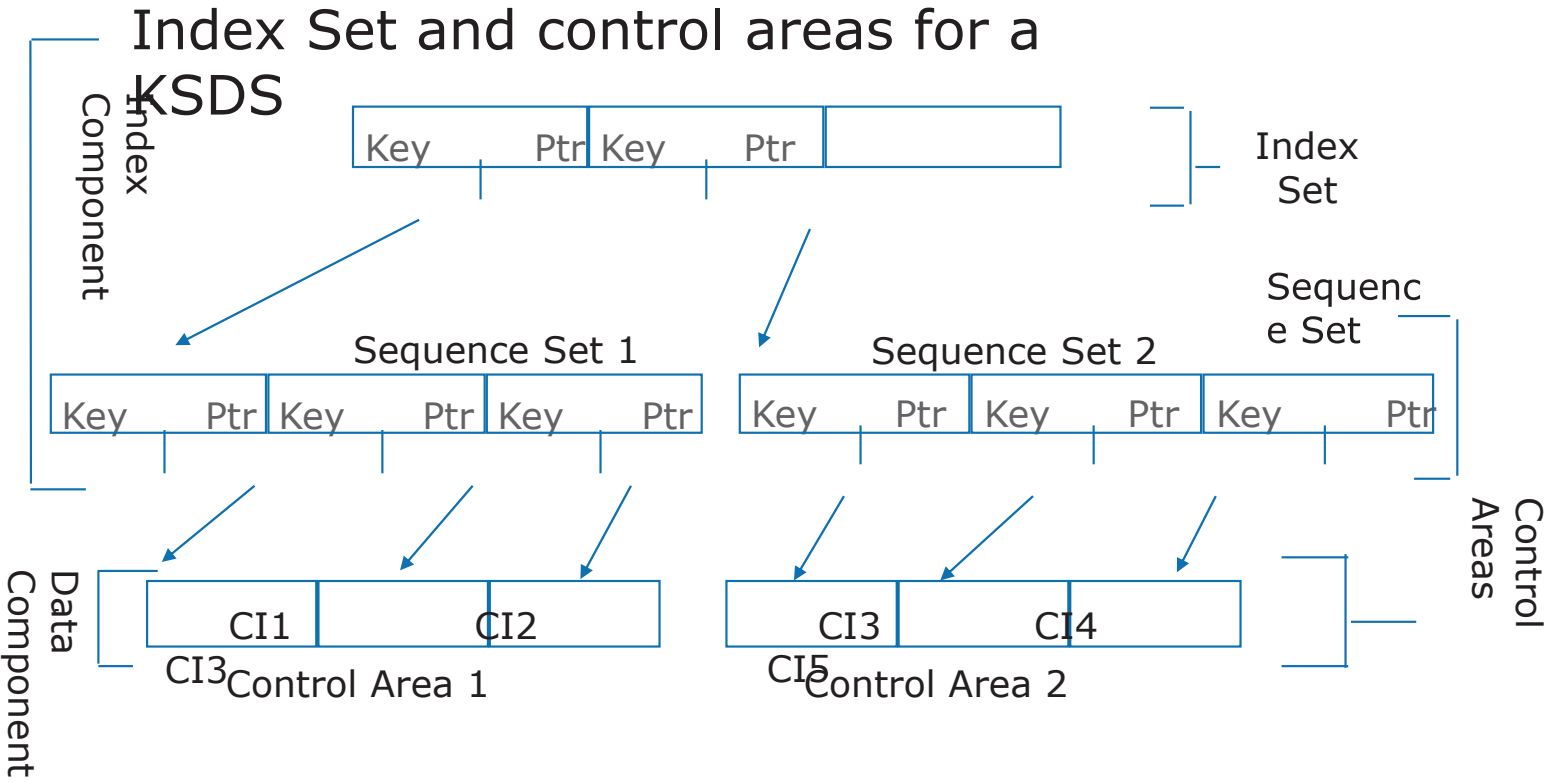
KSDS Structure



KSDS Structure (contd.)



KSDS Structure (contd.)



VSAM Terminology



VSAM Terminology

VSAM Term	Non VSAM Term
Cluster	File or Dataset
Control Interval	Block
Control Interval Size	Block Size
Record Size	Logical Record Length
Control Area	No Parallel Term

Summary



In this lesson, you have learnt the following concepts:

- Control Interval
- Spanned records
- KSDS structure



Review Question

Question 1: VSAM stores records in the data component in units called ____.

- Option 1: Control areas
- Option 2: Records
- Option 3: Control intervals

Question 2: A Control Interval consists of records, free space, and control field information.

- True/ False

Question 3: Spanned records are records ____ than the specified control interval size.

- Option 1: larger
- Option 2: smaller



VSAM (Virtual Storage Access Method)

Lesson 03: IDCAMS Commands



Lesson Objectives

In this lesson, you will learn:

- Functional-IDCAMS Commands
- BUILDINDEX
- REPRO
- PRINT
- DELETE
- VERIFY
- ALTER

LISTCAT

- Modal
- SET
- IF-THEN-ELSE



Lesson Objectives

In this lesson, you will learn:

- Format of IDCAMS command
- IDCAMS return codes



3.1: Accessing VSAM Data

Concept of Accessing VSAM Data

VSAM data sets can be accessed using several methods.

- For example: IDCAMS, batch and CICS application programs, and DB2

Access Method Services (AMS) is a service program used with VSAM to establish and maintain catalogs and data sets invoked by JCL or TSO.

The IDCAMS program can be run with the AMS command and its parameters as input to the IDCAMS program.



Access Method Service (AMS)

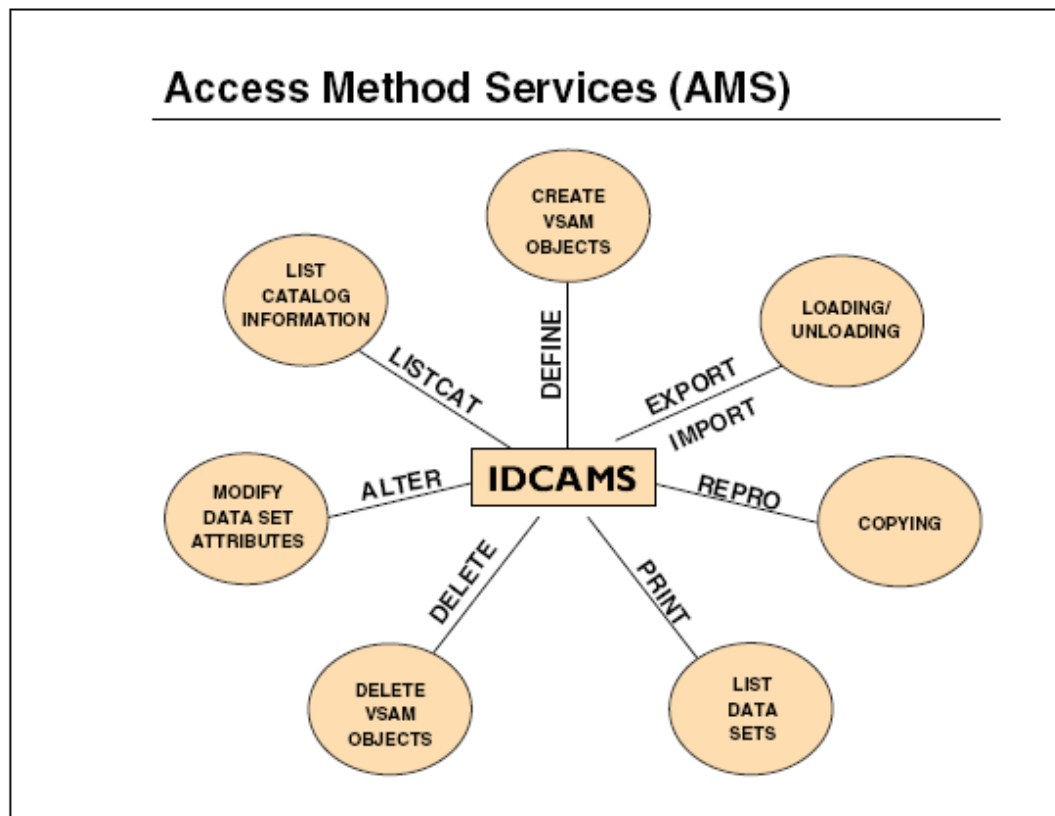
A powerful service program which perform various functions on VSAM & NON-VSAM files.

The utility IDCAMS (Integrated Data Cluster Access Method Service) (IDC is a prefix given by IBM.) used for VSAM operation

IDCAMS performs the following functions

- Allocating, maintaining and deleting Non-VSAM datasets.
- Reorganizing datasets
- Defining the Cluster & Defining Alternate Index
- Build Alternate Index & Define Path for Alternate Index
- Delete Cluster or AIX & Alter characteristics of cluster
- Load, Update or Copy & Print a Cluster

AMS Commands



IDCAMS



Application Programmers use VSAM in the following ways:

- Write IDCAMS utility program commands and execute them to create VSAM dataset.
- Write application programs (in COBOL, PL/I Assembler Language, in CICS)
- Use the statements provided by these languages to write and read VSAM datasets.
- Use IDCAMS utility program commands to list, examine, print, tune, backup, and export/import VSAM datasets.

IDCAMS



The IDCAMS utility can be invoked in the following three ways:

- In batch mode with JCL
- Interactively with TSO commands
- Via calls from an application program



3.2: Types of IDCAMS Commands

IDCAMS Commands

IDCAMS commands are of the following types:

- FUNCTIONAL Commands
 - DEFINE
 - BUILDINDEX
 - PRINT
 - VERIFY

LISTCAT

- MODAL Commands
 - REPRO
 - DELETE
 - ALTER

JCL for executing IDCAMS Commands



Example

```
// jobname    JOB (parameters)
// stepname   EXEC PGM=IDCAMS
// SYSPRINT DD SYSOUT = *
[// ddname    DD DSN=datasetname,
                        DISP= SHR/ OLD
]
//SYSIN       DD *
```

IDCAMS command/s coded freely between 2 to 72 cols.

```
/*
//
```

JCL for executing IDCAMS Commands



Alternate method:

```
// JOBCAT DD DSN = catalogname, DISP= SHR  
// STEPCAT DD DSN = catalogname, DISP = SHR
```

Optionally the above may be used to indicate catalog names for a job/step, in which concerned dataset may be cataloged.



3.3: Format of IDCAMS Command

IDCAMS Command

Format of IDCAMS command:

verb object (parameters)

Example :

```
DEFINE CLUSTER                                -  
NAME(DA0001T.LIB.ESDS.CLUSTER)              -  
      CYLINDERS(5, 1)                        -  
      VOLUMES (BS3013)                       -  
      INDEXED                                -  
)
```

Comment:

```
/* comment */
```



3.4: IDCAMS Return Codes

List of Codes

Condition Code:

- 0: command executed with no errors
- 4 : warning - execution may go successful
- 8 : serious error - execution may fail
- 12: serious error - execution impossible
- 16: fatal error - job step terminates

Condition codes are stored in LASTCC/MAXCC.

Both LASTCC and MAXCC contain zero by default at the start of IDCAMS execution.



Syntax

```
IF {LASTCC/MAXCC} operator number  
THEN[ command'  
DO  
command set  
END]  
[ELSE[ command'  
DO  
command set  
END]]
```



IDCAMS Commands

IF Syntax

For an IF syntax:

- Comparands are EQ/NE/GT/LT/GE/LE.
- Hyphen is the continuation character.
- Comment is assumed as Null statement.
- ELSE is optional.



Example of IF Syntax

```
.....  
      REPRO INFILE (INDD)          -  
      OUTFILE (OUTDD)  
      .....  
      IF LASTCC EQ 0              -  
      THEN                        -  
          PRINT OUTFILE (OUTDD)  
      ELSE  
          PRINT INFILE (INDD)
```



```
IF MAXCC LT 4      -  
THEN              -  
DO  
/* COMMENT */  
    Command  
    Command  
END  
ELSE  
    Command
```


SET COMMAND



LASTCC and MAXCC can be set to a value between 0-16:

- setting MAXCC has no effect on LASTCC
- setting LASTCC changes the value of MAXCC, if LASTCC is set to a value larger than MAXCC
- setting MAXCC = 16 terminates the job

For example:

```
DEFINE CLUSTER  
IF LASTCC > 0 THEN -  
    SET MAXCC = 16  
ELSE REPRO
```



3.5: The DEFINE CLUSTER Command

Concept of DEFINE CLUSTER Command

DEFINE CLUSTER Command:

- The command tells IDCAMS to create and name a VSAM cluster.
- Format: DEFINE CLUSTER (NAME (cluster name) parameters)
- Abbreviation: DEF CL
- Used for: ESDS, KSDS, RRDS
- Default: None

Example:

```
DEFINE CLUSTER (NAME (DA0004T.LIB.KSDS.CLUSTER)) -  
                DATA (NAME (DA0004T.LIB.KSDS.DATA)) -  
                INDEX (NAME (DA0004T.LIB.KSDS.INDEX))
```



The DEFINE CLUSTER Command

Concept of Name Parameter

Name Parameter:

- This is a required positional parameter.
- Format: NAME (cluster name)
- cluster name: The name to be assigned to the cluster.
- Abbreviation: None
- Used for: ESDS, KSDS, RRDS
- Default: None

Example:

```
DEFINE CLUSTER (NAME (DA0004T.LIB.KSDS.CLUSTER)) -  
              DATA (NAME (DA0004T.LIB.KSDS.DATA)) -  
              INDEX (NAME (DA0004T.LIB.KSDS.INDEX))
```



The DEFINE CLUSTER Command

Concept of DATA Parameter

DATA Parameter:

The DATA parameter tells IDCAMS that you going to create a separate data component.

- Format: DATA (NAME (data name) parameters)
 - NAME parameter: It is optional, but you should usually code it. If coded, it must be the first DATA parameter.

data name: The name you choose to name the data component.

Abbreviation: None

Used for: ESDS, KSDS, RRDS

Required: No



The DEFINE CLUSTER Command

Concept of DATA Parameter (contd.)

Default: None

Example:

```
DATA (NAME (DA0004T.LIB.KSDS.DATA))  
INDEX (NAME (DA0004T.LIB.KSDS.INDEX))
```

There are several options for the DATA parameter. However NAME is the most common one. The NAME parameter need not be coded but it should be done for a KSDS so that we can operate on the data component by itself.



The DEFINE CLUSTER Command

Concept of INDEX Parameter

INDEX Parameter:

- The INDEX parameter creates a separate index component.
 - Format: INDEX (NAME (index name) parameters)

index name: The name you choose for the index component.

- Abbreviation: IX
- Used for: KSDS
- Required: No
- Default: None
- Example: INDEX (NAME (DA0004T.LIB.KSDS.INDEX))



The DEFINE CLUSTER Command

Concept of Space Allocation Parameter

Space Allocation parameter:

- The space allocation parameter specifies space allocation values in the units shown below:
 - Format: Code only one of the following:
 - CYLINDERS (primary secondary) or
 - TRACKS (primary secondary) or
 - RECORDS (primary secondary) or
 - KILOBYTES (primary secondary) or
 - MEGABYTES (primary secondary)
 - Abbreviation: CYL, TRK, REC, KB, MB
 - Used for: KSDS, ESDS, RRDS



The DEFINE CLUSTER Command

Concept of Space Allocation Parameter

Required: Yes

Default: None

Example:

```
CYLINDERS (10 4)  
RECORDS (400 500)
```




The DEFINE CLUSTER Command

Concept of Space Allocation Parameter

```
DEFINE CLUSTER (NAME (DA0004T.LIB.KSDS.CLUSTER) -  
                CYLINDERS (5 3)  
                )  
-  
  DATA (NAME (DA0004T.LIB.KSDS.DATA))  
-  
  INDEX (NAME (DA0004T.LIB.KSDS.INDEX))
```

```
DEFINE CLUSTER (NAME (DA0004T.LIB.KSDS.CLUSTER) -  
                DATA (NAME (DA0004T.LIB.KSDS.DATA))  
-  
                CYLINDERS (5 3)  
                )  
-  
  INDEX (NAME (DA0004T.LIB.KSDS.INDEX))
```



The DEFINE CLUSTER Command

Concept of Space Allocation Parameter

Primary: Allocated once when the dataset is created

Secondary: Allocated a maximum of 122 times as needed during the life of the dataset



The DEFINE CLUSTER Command

Concept of Volumes Parameter

Volumes Parameter:

- The VOLUMES parameter assigns one or more storage volumes to your dataset.
- Multiple volumes must be of the same device type (for example: 3390).
 - Format: VOLUMES (vol ser) or VOLUMES (vol ser vol ser)
 - vol ser: The six-digit volume serial number of a volume.
 - Abbreviation: VOL
 - Used for: KSDS, ESDS, RRDS
 - Required: Yes



The DEFINE CLUSTER Command

Concept of Volumes Parameter

Concept of Volumes Parameter

VOLUMES (BS3011)

VOLUMES (BS3011 BS3040 BS3042)



The DEFINE CLUSTER Command

Concept of Volumes Parameter

You can store the data and index on separate volumes and this may provide a performance advantage for large dataset.



The DEFINE CLUSTER Command

Concept of RecordSize Parameter

RecordSize Parameter:

- This parameter tells VSAM what size records to expect.
- The avg and max are average and maximum values for variable-length records.
- If records are of fixed length, avg and max should be the same.
 - Format: RECORDSIZE (avg max)
 - avg: Average length of records
 - max: Maximum length of records
 - Abbreviation: RECSZ
 - Used for: KSDS, ESDS, RRDS
 - Required: No



The DEFINE CLUSTER Command

Concept of RecordSize Parameter

Default: RECORDSIZE (4086 4086)

Example:

RECORDSIZE (80 80) [Fixed Length records]

RECORDSIZE (80 120) [Variable Length records]

RECORDSIZE can be assigned at the cluster or data level.

For a KSDS, VSAM calculates the RECORDSIZE of the index based on the KEYS parameter.



The DEFINE CLUSTER Command

Concept of Keys Parameter

KEYS Parameter:

- This parameter defines the length and offset of the primary key in a KSDS record.
- The offset is the primary key's displacement (in bytes) from the beginning of the record.
- Format: KEYS (length offset)
- length: length in bytes of the primary key
- offset: Offset in bytes of the primary key with records (0 to n)
- Abbreviation: None
- Used for: KSDS
- Required: No



The DEFINE CLUSTER Command

Concept of Keys Parameter

Default: KEYS (64 1)

Example:

KEYS (8 0)

VSAM records begin in position zero.



The DEFINE CLUSTER Command

Concept of DataSet Type

DataSet Type:

- This parameter specifies whether the dataset is INDEXED (KSDS), NONINDEXED (ESDS), or NUMBERED (RRDS).

Format: INDEXED | NONINDEXED | NUMBERED

- INDEXED: Specifies a KSDS and is the default
- NONINDEXED: Specifies an ESDS.
- No index is created and records are accessed sequentially or by relative byte address.
- NUMBERED: Specifies an RRDS
 - Abbreviation: IXD | NIXD | NUMD
 - Used for: KSDS, ESDS, RRDS



The DEFINE CLUSTER Command

Concept of DataSet Type

Required: No Because of the default

Default: INDEXED

Example:

INDEXED

NONINDEXED

NUMBERED

The DEFINE CLUSTER Command

Sample ESDS



Let us see a sample of ESDS:

```
DEFINE CLUSTER                                -  
    (NAME(DA0001T.LIB.ESDS.CLUSTER)          -  
    NONINDEXED                               -  
    RECORDSIZE(125 125)                       -  
    RECORDS(100 10)                           -  
    NONSPANNED                                -  
    VOLUMES (BS3013)                          -  
    REUSE                                     -  
    )                                          -  
    DATA(NAME(DA0001T.LIB.ESDS.DATA))  
/*  
//
```



The DEFINE CLUSTER Command

Sample ESDS

Let us see a sample of ESDS:

```
//DA0001TA JOB LA2719, PCS,MSGLEVEL=(1,1),
//          MGCLASS=A,NOTIFY=DA0001T
// * Delete/Define Cluster for KSDS VSAM Dataset
//STEP1      EXEC PGM=IDCAMS
// SYSPRINT   DD SYSOUT=*
// SYSIN      DD *
DELETE DA0001T.LIB.KSDS.CLUSTER
DEFINE CLUSTER              -
(NAME(DA0001T.LIB.KSDS.CLUSTER) -
INDEXED                      -
KEYS(4 0)                    -
FSPC(10 20)                  -
RECORDSIZE(125 125)         -
```



The DEFINE CLUSTER Command

Sample ESDS

Contd.

```
RECORDS(100 10)           -  
    NONSPANNED             -  
    VOLUMES (BS3013)       -  
    NOREUSE                 -  
    )                       -  
    DATA(NAME(DA0001T.LIB.KSDS.DATA)) -  
    INDEX(NAME(DA0001T.LIB.KSDS.INDEX))  
/*  
//
```



The DEFINE CLUSTER Command

Concept of CONTROLINTERVALSIZE Parameter

CONTROLINTERVALSIZE Parameter:

- This parameter specifies the Control Interval size. It is usually abbreviated CISZ.
- Format: CONTROLINTERVALSIZE (bytes)
 - bytes: The size of control interval in bytes.
- Abbreviation: CISZ
- Used for: KSDS, ESDS, RRDS
- Required: No, because of the default



The DEFINE CLUSTER Command

Concept of CONTROLINTERVALSIZE Parameter

Default: Calculated by VSAM

Example:

CISZ (4096)

Note: If omitted VSAM calculates CISZ based on record size and other control information



The DEFINE CLUSTER Command

Concept of FREESPACE Parameter

FREESPACE Parameter:

- This FREESPACE parameter, which applies to the KSDS, allocates some percentage of control interval and control area for planned free space. This free space can be used for adding new records or for expanding existing variable records.
- FREESPACE applies only to the data component although it is syntactically correct to code it at the cluster as well.
- Too much free space results in more i/o, especially when doing sequential processing. Too little results in excessive control interval and control area split.



The DEFINE CLUSTER Command

Concept of Freespace Parameter

Format: FREESPACE (ci% ca%)

ci%: Percentage of control interval to leave free for expansion

ca%: Percentage of control area to leave free for expansion

Abbreviation: FSPC

Used for: KSDS

Required: No, because of default.

Default: FREESPACE (0 0)

Example:

FREESPACE (20 10)

FREESPACE (20)

FREESPACE (0 30)

Control interval only

Control area only



The DEFINE CLUSTER Command

Concept of SPANNED/NONSPANNED Parameter

SPANNED / NONSPANNED Parameter:

- This parameter allows large record to span more than one control interval.
- However records cannot span Control Areas.
- The resulting free space in the spanned control interval is unusable by other records, even if they fit logically in the unused bytes.



The DEFINE CLUSTER Command

Concept of SPANNED/NONSPANNED Parameter

[NONSPANNED is the default] and it means that records cannot span control intervals.

Format: SPANNED | NONSPANNED

Abbreviation: SPND | NSPND

Used for: KSDS, ESDS

Required: No, because of default.

Default: NONSPANNED

Example:

NONSPANNED

SPANNED



The DEFINE CLUSTER Command

Unloaded Dataset versus Empty Dataset

An unloaded dataset is a dataset that is empty and has never contained any records.

- An unloaded dataset has the HURBA value of zero.

An empty dataset is one that has previously held records, however is now empty because all the records have been removed with COBOL delete statement.

- An empty dataset has the HURBA value greater than zero.



The DEFINE CLUSTER Command

Empty Dataset

To load an empty dataset with records, the dataset has to be opened in either I-O or EXTEND mode.

- It cannot be opened in OUTPUT mode because the value of HURBA would be greater than zero in that case.

If an application program opens an empty dataset in OUTPUT mode, then COBOL will set the status key to a value of 95 and open will fail.

- This may also happen if the dataset already contains records.



The DEFINE CLUSTER Command Empty Dataset

The one exception is a KSDS that has been defined with the reuse option.

Even if the dataset contains records and the HURBA value is greater than zero, the reuse option forces the logical deletion of all existing records before the loading process and the application program can open the dataset in output mode.



The DEFINE CLUSTER Command Unloaded Dataset

An unloaded dataset cannot be opened in I-O mode initially.



The DEFINE CLUSTER Command

Concept of REUSE/NOREUSE

REUSE/NOREUSE Parameter:

- The REUSE parameter specifies that the cluster can be opened a second time as a reusable cluster.
- NOREUSE is the default, and specifies the cluster as non-reusable.
 - Format: REUSE | NOREUSE
 - Abbreviation: RUS | UNRUS
 - Used for: KSDS, ESDS, RRDS
 - Required: No, because of default.



The DEFINE CLUSTER Command

Concept of REUSE/NOREUSE

Default: NOREUSE

Example:

```
REUSE  
NOREUSE
```



The DEFINE CLUSTER Command

Concept of REUSE

REUSE:

- It signifies some application call for temporary dataset or workfile that must be created, used, and deleted each time the application is run.
- To simplify these applications, VSAM lets you create reusable files.
- REUSE sets the HURBA to zero when the dataset is opened for output by an application program or if REPRO is executed.



The DEFINE CLUSTER Command

Concept of BUFFERSPACE Parameter

BUFFERSPACE Parameter:

- Bufferspace represents the amount of storage (in bytes) required to process the contents of a minimum of one control interval's worth of data.
 - Format: BUFFERSPACE (bytes)
 - Bytes: The number of bytes to allocate to each buffer.
 - Abbreviation: BUFSP
 - Used for: KSDS, ESDS, RRDS



The DEFINE CLUSTER Command

Concept of BUFFERSPACE Parameter

Required: No, because of default.

Default: Two data buffers for all types of datasets, plus one additional index buffer for KSDS.

Example:

BUFSP (8704)



The DEFINE CLUSTER Command

The SHAREOPTIONS Parameter

This parameter specifies how a VSAM dataset can be shared across the regions and across the system.

Across the system is thru ISC (Inter System Communication)

Format : SHAREOPTIONS (CR value CS value)

Values :-Default :- SHAREOPTIONS(1 3)

- Cross-region : concurrent access on a standalone system

Value can be one of 1,2,3,4

DISP coded as SHR

- Cross-system : sharing between different systems

Value can be either 3 or 4.



The DEFINE CLUSTER Command

The SHAREOPTIONS Parameter (Contd.)

Value

- Shareoption 1
 - means that either one user can update or many users can read the data set.
- Shareoption 2
 - allows multiple users to read a data set at the same time that one user is updating it.
- Shareoption 3
 - allows any number of user to read the data set and also allows multiple users to update at same time. This share option does not ensure read or write integrity.
- Shareoption 4
 - has same problems associates with it as shareoption 3. However, buffers used for direct processing are refreshed each time an I/O is performed on the data set.



The DEFINE CLUSTER Command

SHAREOPTIONS Parameter...contd

For cross-region sharing, each batch job must have DISP=SHR coded in its JCL.
DISP=OLD assumes exclusive control and nullifies sharing options.

For cross-system sharing, DISP=OLD is ignored.

You cannot force exclusive control across systems by coding DISP=OLD.



The DEFINE CLUSTER Command

Concept of ERASE/NOERASE

ERASE/NOERASE:

- The ERASE parameter instructs VSAM to overwrite sensitive data component with binary zeros when the cluster is deleted.
- NOERASE is the default and means that the deleted cluster is not to be overwritten with binary zeros.
 - Format: ERASE | NOERASE
 - Abbreviation: ERAS | NERAS
 - Used for: KSDS, ESDS, RRDS
 - Required: No, because of default.



The DEFINE CLUSTER Command

Concept of ERASE/NOERASE

Default: NOERASE

Example:

```
NOERASE  
ERASE
```



Summary

In this lesson, you have learnt:

- Functional-IDCAMS Commands:
- BUILDINDEX
- REPRO
- PRINT
- DELETE
- VERIFY
- ALTER
- LISTCAT
- Modal
- SET
- FORMAT OF IDCAMS COMMAND
- IDCAMS return codes



Review Question

Question 1: The ____ parameter assigns one or more storage volumes to your dataset.

- Option 1: CYL
- Option 2: VOL
- Option 3: DISK

Question 2: DEFINE AIX command tells IDCA to create and name a VSAM cluster.

- True/ False

Question 3: IDCAMS stores the condition codes in variables named ____ and ____.



VSAM (Virtual Storage Access Method)

Lesson 04: Using LISTCAT



Lesson Objectives

In this lesson, you will learn to:

- Execute LISTCAT from TSO prompt



4.1: LISTCAT

Introduction to LISTCAT

The LISTCAT command is provided to list the information from catalog entries. LISTCAT's basic function is to list information about VSAM and NONVSAM objects. It is frequently necessary to list fields from the catalog entry for a VSAM object to diagnose problems.



LISTCAT Format

Format:

- LISTCAT ENTRIES (entry name) options
- LISTCAT LEVEL (level) options
- entry name: The generic name of each entry to list.
- level: The level of qualification at which to begin listing.

Abbreviation:

- LISTC ENT
- LIST LVL

Used for: KSDS, ESDS, RRDS and non-VSAM

Required: No, because of default.



LISTCAT Format

Default: All entries are listed if just LISTCAT is coded.

Example:

```
LISTCAT ENTRIES(da0001t.lib.ksds.cluster)
LISTCAT LEVEL(da0001t.lib.ksds. *)
LISTCAT LEVEL(da0001t.*.lib. *)
```



Entry Name Parameter

Entry Name signifies the generic name of each entry to list.

- For example: A.B.C would list the A.B.C entry only

The Entries parameter requires you to specify each level of qualification either implicitly or explicitly, using asterisk as a wild card character in place of one or more levels.



LEVEL Parameter

The syntax of LEVEL parameter does not require that you account for every qualifier.

- If the LEVEL qualifier is specified, then VSAM assumes that qualifier to be the high level qualifier and lists every entry with that qualifier.
- The asterisk can be coded as a wild character to specify all names for a level.
- The asterisk need not be coded at the last level.
- This is because level, by definition, lists all lower levels.
- Hence, coding asterisk at the last does not make sense.



LEVEL Parameter

The level of qualification is followed to begin listing:

- For example: A.B would list all entries that have A.B as a level of qualification.

An asterisk can also be coded as a wild card character:

- For example: A.*.C
 - The asterisk cannot be coded as the lowest level A.B.*
 - The above implies the same as A.B.



Examples of LISTCAT

Example 1:

```
LISTCAT                                -  
    ENT(A2000.KSDS.CLUSTER)          -  
    ALL
```

The ALL parameter requests all catalog information and has no logical connection to entries.



Example of LISTCAT

```
(A) Passport.zws - PASSPORT
File Edit View Communication Options Transfer Macro Help

Menu List Mode Functions Utilities Help

ISPf Command Shell
Enter TSO or Workstation commands below:

==> LISTCAT ENT(PADMAJA)ALL

Place cursor on choice and press enter to Retrieve command

=> IND$FILE GET 'IGY.SIGYCOMP(IGYCRCTL)'
=> IND$FILE GET 'IGY.SIGYCOMP(IGYCRCTL)' ASCII CRLF
=> IND$FILE GET 'DSRP035.VANDANA.DB2SORCE(EXCUR)' ASCII CRLF
=>
=>
=>
=>
=>
=>
IKJ58600I QUALIFIERS FOR DATA SET DSRP035.PADMAJA ARE
IKJ58600I COBOL      EMP4      EMP8      JCL
IKJ58600I LOADLIB   MYMYEP2   OUT1      SQLIN
***
Tn  ↑ R 24 C 6

Connected to 220.227.139.118:23 CAP NUM 24, 6
```

Capgemini Public



Examples of LISTCAT

Example 2:

```
LISTCAT                                -  
    ENT(A2000.LIB.KSDS.*)            -  
    ALL
```

The above example will include entries such as A2000.LIB.KSDS.BACKUP since the asterisk is used as a wild card character.

Examples of LISTCAT



Example 3:

LISTCAT	-
ENT(A2000.KSDS.CLUSTER)	-
CLUSTER	-
ALL	

Example 4:

LISTCAT	-
ENT(A2000.LIB.KSDS.*)	-
CLUSTER	-
ALL	

The cluster parameter in the above examples restricts the information to the base cluster only.



4.2: Various LISTCAT Options

LISTCAT Options

The following options are available for LISTCAT:

- HISTORY or HIST
- VOLUME or VOL
- ALLOCATION or ALLOC
- ALL



LISTCAT Options

LISTCAT(HISTORY):

- It lists information for the dataset including the following:
 - Name
 - Type of entry
 - Creation and Expiration date



LISTCAT Options

LISTCAT(VOLUME):

- It lists the device type and one or more volume serial numbers where the dataset resides.
- History information is listed as well.

LISTCAT(ALLOCATION):

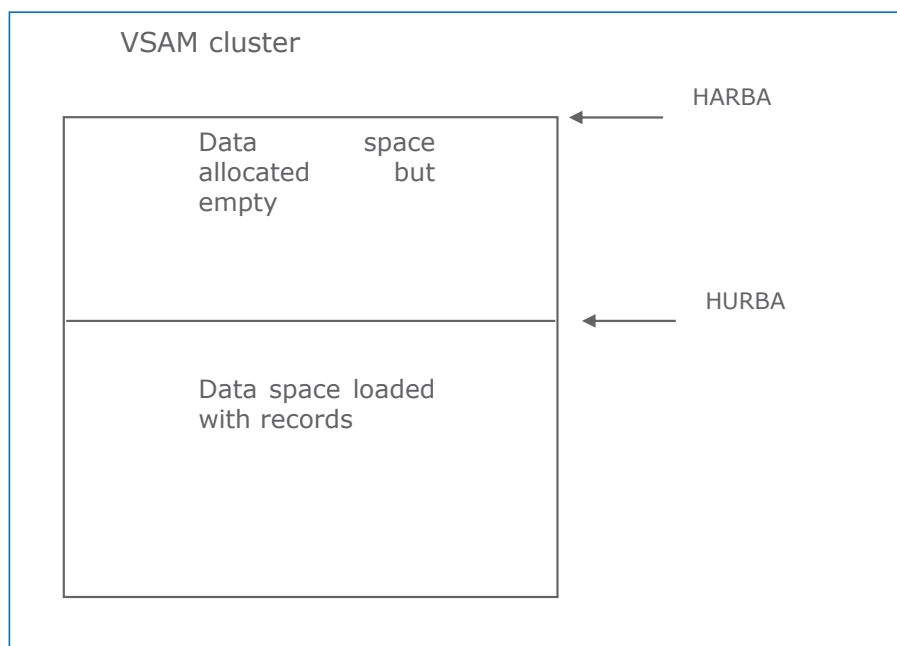
- It lists information for the dataset including the following:
- Information about space allocation.
 - The above information is about data and index component.
 - HISTORY and VOLUME information is displayed as well.



4.3: LISTCAT from TSO Prompt

To Execute LISTCAT from TSO Prompt

LISTCAT ENTRIES (LIB.KSDS.CLUSTER) ALL





Summary

In this lesson, you have learnt:

- To execute LISTCAT from TSO prompt



Review Questions

Question 1: Identify the valid LISTCAT options:

- Option 1: HISTORY or HIST
- Option 2: VOLUME or VOL
- Option 3: ALLOCATION or ALLOC
- Option 4: All of the above

Question 2: LISTCAT's basic function is to list information about VSAM and NONVSAM objects

- True/ False

Question 3: ____ is the highest byte that can be used.



VSAM (Virtual Storage Access Method)

Lesson 05: Creating Alternate Indexes



Lesson Objectives

On completion of this lesson, you will be able to:

- Define and build Alternate Indexes
- Illustrate steps to create an Alternate Index
- Use the DEFINE ALTERNATEINDEX command
- Use the DEFINE PATH command
- Use the BLDINDEX command



5.1. Overview

Why Alternate Index (AIX)?

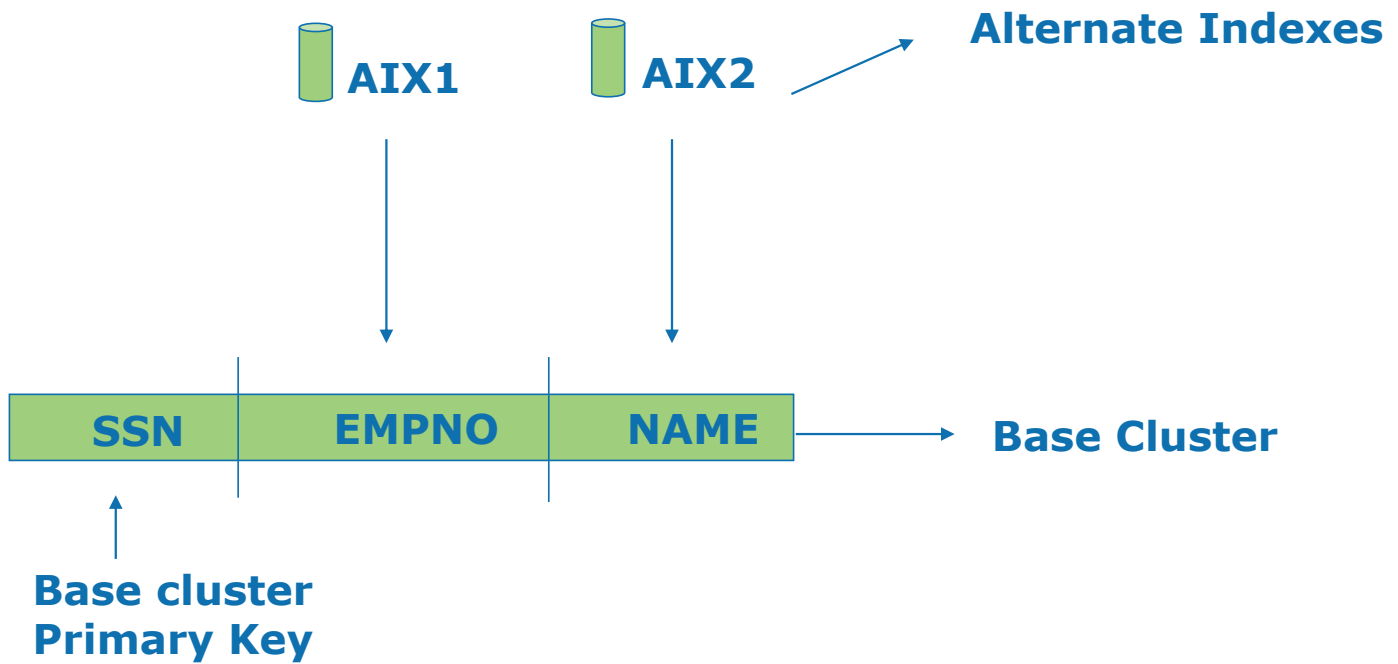
AIXs enable the logical records of an ESDS or of a KSDS (in this context called a base cluster) to be accessed sequentially and directly by more than one key field.

This eliminates the need to store the same data in different sequences in multiple data sets for the purposes of various applications.



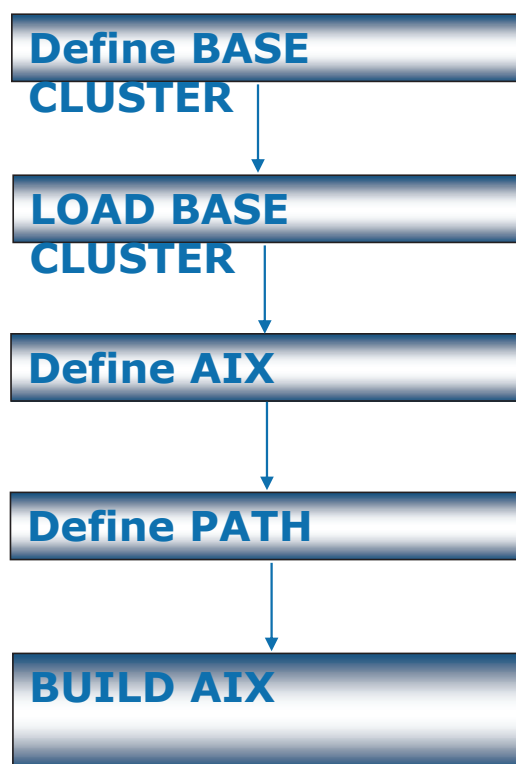
5.1. Overview

Why Alternate Index (AIX)? (Contd.)



5.1. Overview

AIX Flow Diagram





5.1. Overview

Alternate Index (Cont..)

other fields of a record →

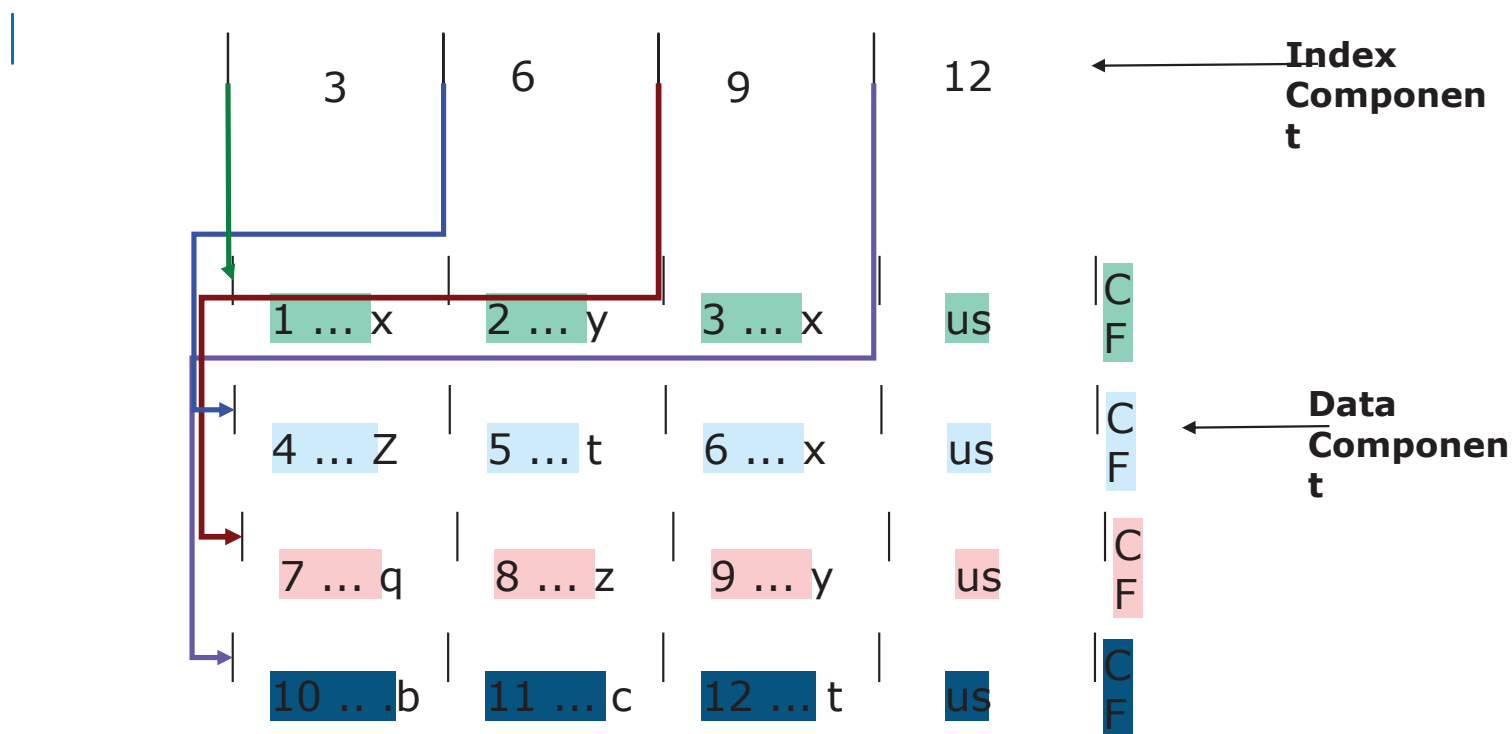
	Book No.	Author
1	...	x
2	...	y
3	...	x
4	...	z
5	...	t
6	...	x
7	...	q
8	...	z
9	...	y
10	...	b
11	...	c
12	...	t

Data set having
'Book No' as a
Primary Key



5.1. Overview

Base Cluster (KSDS)





5.1. Overview

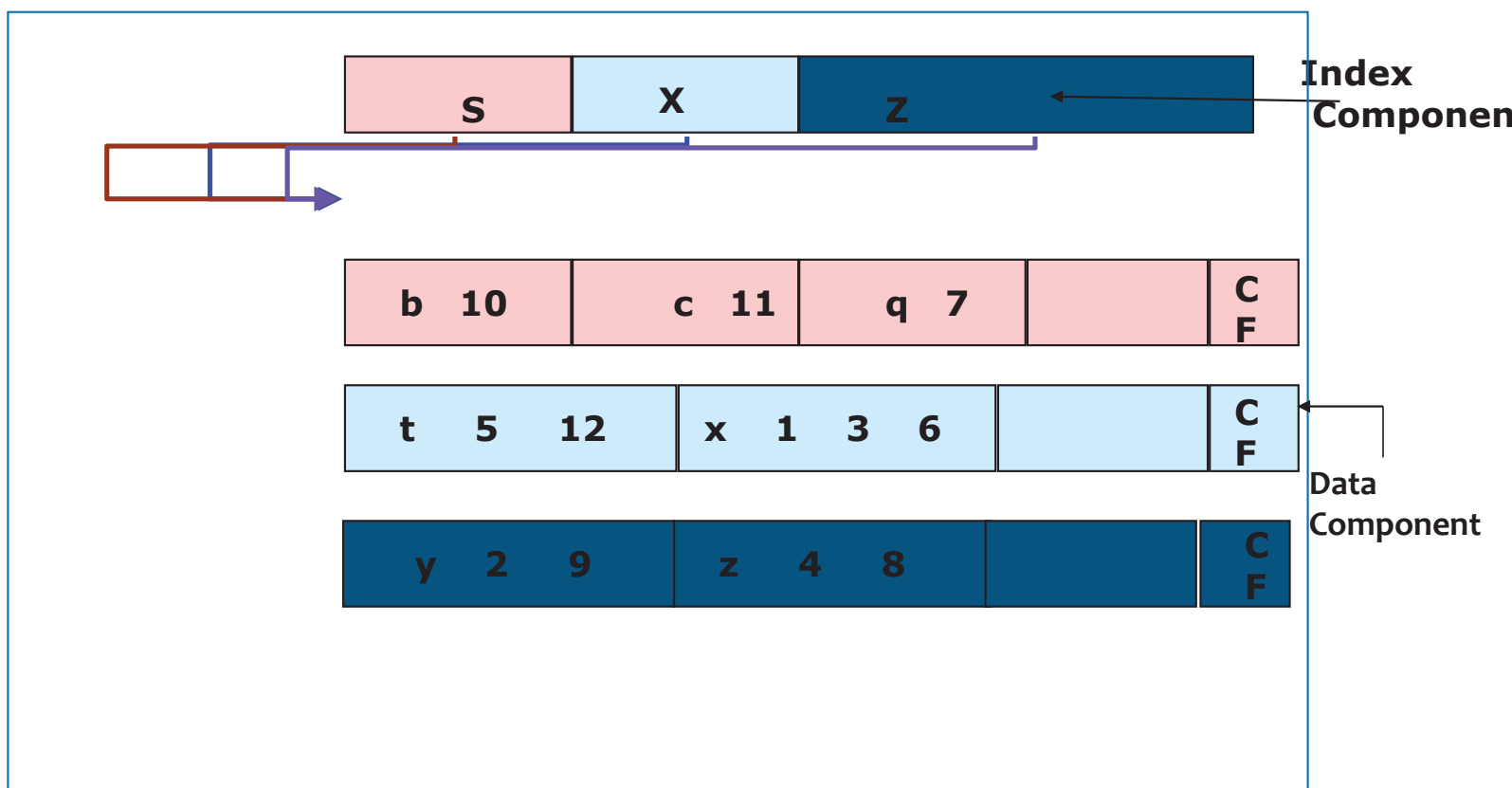
Alternate Index (Cont..)

Author		Book No.
b	<div></div>	... 10
c		... 11
q		... 7
t		... 5, 12
x		... 1, 3, 6
y		... 2, 9
z		... 4, 8



5.1. Overview

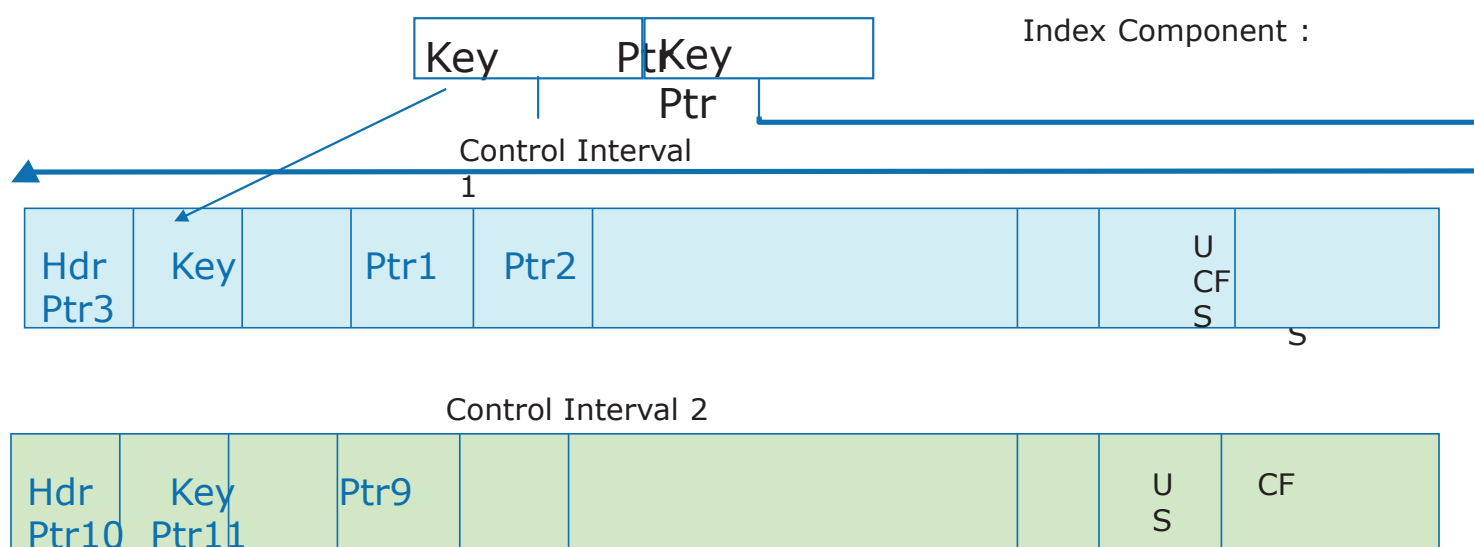
Non-Unique Alternate Key





5.1. Overview

Alternate Index (Cont...)



The pointers point to a control interval in the data component of the base cluster that contains the record.



5.1. Overview

The Alternate Index

Alternate index may be defined on one or more than one alternate keys, that is, Fields other than the primary keys.

Alternate key(s) need not be unique.

Separate cluster is defined.

A path is also defined for connection between base cluster and alternate index cluster.

Each alternate index itself is a KSDS.

It consists of both a data component and an index component.

Alternate Index upgradation is by default.



5.1. Overview

The Alternate Index

The alternate index does not support a reusable base cluster.

It greatly reduces data redundancy.

It may lead to performance degradation.

It builds the ALT. index after the base cluster has been both defined and loaded with at least 1 record.

AIX on ESDS is supported only for assembler or online CICS.



5.1. Steps to create an Alternate Indexes

Creating an Alternate Index

Creating an alternate index requires three separate steps:

- Define the alternate index using the IDCAMS DEFINE AIX command.
- Define an alternate index path using the IDCAMS DEFINE PATH command.
- Build the alternate index and populate it with records using the IDCAMS BLDINDEX command.

The above three steps need to be repeated for each alternate index built over the base cluster.



5.1. The DEFINE ALTERNATEINDEX Command

The DEFINE ALTERNATEINDEX Command

Define the Alternate Index Cluster using the IDCAMS DEFINE AIX command.

Format	DEFINE ALTERNATEINDEX
Abbrevia tion	DEF AIX
Use	Alternate index over a KSDS, ESDS
Required	Yes, if an alternate index is defined.
Default	None
Example	DEFINE AIX



5.1. The DEFINE ALTERNATEINDEX Command

The NAME Parameter

NAME is a mandatory parameter that names the alternate index.

Format	NAME (<i>index cluster name</i>)
Abbreviation	None
Use	Alternate index over a KSDS, ESDS
Required	Yes, if an alternate index is defined.
Default	None
Example	NAME (da0001t.lib.ksds.ename.aix)



5.1. The DEFINE ALTERNATEINDEX Command

The VOLUMES Parameter

The VOLUMES parameter specifies the volume (or volumes) where the alternate index is to be stored.

Format	VOLUMES (<i>vol ser</i>) or VOLUMES (<i>vol ser</i> ... <i>vol ser</i>) <i>Vol ser</i>: The six-digit volume serial number of a volume.
Abbreviation	VOL
Use	Alternate index over a KSDS, ESDS.
Required	Yes, if an alternate index is defined.
Default	None
Example	VOLUMES (BS3011) VOLUMES (BS3011 BS3040 BS3042)



5.1. The DEFINE ALTERNATEINDEX Command

The RELATE PARAMETER

RELATE establishes the relationship between the base cluster and the alternate index using the base cluster name. It is both necessary and unique to the DEFINE AIX command.

Format	RELATE (<i>base cluster name</i>)
Abbreviation	REL
Use	Alternate index over a KSDS, ESDS
Required	Yes, if an alternate index is defined.
Default	None
Example	RELATE (da0001t.lib.ksds.cluster)



5.1. The DEFINE ALTERNATEINDEX Command

The UPGRADE Parameter

Format	<u>UPGRADE</u> NOUPGRADE
Abbreviation	<u>UPG</u> NUPG
Use	Alternate index over a KSDS, ESDS
Required	No, because of default.
Default	UPGRADE
Example	UPGRADE NOUPGRADE



5.1. The DEFINE ALTERNATEINDEX Command

The UPGRADE Parameter (Contd.)

Changes made to the base cluster reflect automatically in all the alternates indexes provided that you have specified the UPGRADE option in each of alternate indexes (the DEFINE AIX command) of the base cluster.

For that, the application program has to open the base cluster.

The set of alternate indexes VSAM that updates automatically is called Upgrade set.



5.1. The DEFINE ALTERNATEINDEX Command

The KEYS Parameter

This parameter defines the length and offset of the alternate index key.

Format	KEYS (<i>length offset</i>) <i>length</i> : length in bytes of the alternate key <i>offset</i> : The <i>offset</i> is the key’s displacement (in bytes) from the beginning of the record.
Abbreviation	None
Use	Alternate index over a KSDS, ESDS.
Required	No, because of default.
Default	KEYS (64 0)
Example	KEYS (30 4)



5.1. The DEFINE ALTERNATEINDEX Command

The RECORDSIZE Parameter

The RECORDSIZE parameter specifies the average and maximum of each alternate index record.

Format	RECORDSIZE (<i>avg max</i>)
Abbreviation	RECSZ
Use	Alternate index over a KSDS, ESDS
Required	No
Default	RECORDSIZE (4086 32600)
Example	RECORDSIZE (80 120)



5.1. The DEFINE ALTERNATEINDEX Command

The KSDS Unique Alternate Index

Each UNIQUE alternate index record built over KSDS consists of the following:

- Five bytes of VSAM housekeeping information
- The alternate index key
- The single primary key pointer to which the alternate index resolves

KSDS :

ALT-INDEX-KEY

PRM-KEY

HOUSE KEEPING	SOC_SEC_NUM	EMP_NUM
5	9	8

**DATA
COMPONENT :**

RECORDSIZE(22 22)

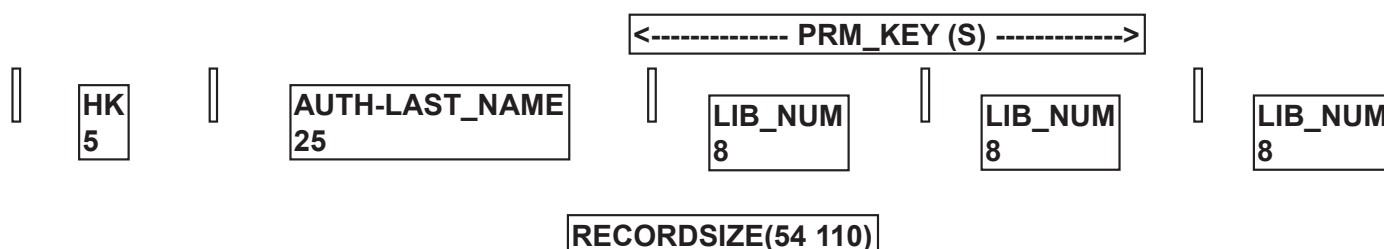


5.1. The DEFINE ALTERNATEINDEX Command

The KSDS Unique Alternate Index

Each Non-UNIQUE alternate index record built over KSDS consists of the following.

- Five bytes of VSAM housekeeping information
- The alternate index key
- One or more primary key pointers to which the alternate index resolves (For example, avg 3, max 10)





5.1. The DEFINE ALTERNATEINDEX Command

The ESDS Alternate Index

Each alternate index record built over ESDS consists of the following:

- Five bytes of VSAM housekeeping information
- Timestamp (12 bytes)
- Relative Byte Address of the record (4 bytes)

ESDS :

- **Occupies Less Space**
- **RBA-Access is Faster**



HK
5

TIMESTAMP
12

RBA
4



5.1. The DEFINE ALTERNATEINDEX Command

The FREESPACE Parameter

FREESPACE Allocates some percentage of both the control interval and control area for planned free space.

It can be used for adding new records.

FREESPACE applies only to the data component.



5.1. The DEFINE ALTERNATEINDEX Command

The FREESPACE Parameter (Contd.)

Format	FREESPACE (ci% ca%) ci%: Percentage of control interval to leave free for expansion ca%: Percentage of control area to leave free for expansion
Abbreviation	FSPC
Use	Alternate index over KSDS, ESDS
Required	No, because of default
Default	FREESPACE (0 0)
Example	FREESPACE (20 10)



5.2. Other DEFINE AIX Parameters

Other Parameters for DEFINE AIX Command

You can also add the following parameters to your DEFINE AIX command. These parameters have been described in the DEFINE CLUSTER command.

- BUFFERSPACE (bytes)
- CISZ (bytes)
- ERASE | NOERASE
- REUSE | NOREUSE
- SHAREOPTIONS
- UNIQUEKEY | NONUNIQUEKEY



5.2. Other DEFINE AIX Parameters

DEFINE AIX: Example

```
//STEP1 EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT =*
//SYSIN DD *
    DEFINE AIX (NAME (DA0001T.LIB.KSDS.EMPNAME.AIX) -
                VOLUMES (BS3013)
                -
                RELATE
(DA0001T.LIB.KSDS.CLUSTER) -
                UPGRADE
                SHR (1 3)
                NOERASE
                NOREUSE
                KEYS (30 8)
                NONUNIQUEKEY
                FREESPACE (20 10)
                )
    DATA (NAME (DA000A1T.LIB.KSDS.EMPNAME.DATA) -
           TRACKS (10 1)
           RECORDSIZE (75 125)
           )
    INDEX (NAME (DA0001T.LIB.KSDS.EMPNAME.INDEX))
/*
```



5.2. Overview

Path

A path is defined for each alternate index.

Path is a separate catalog entry.

Path is a logical connection through the ALT. Index to the base cluster.

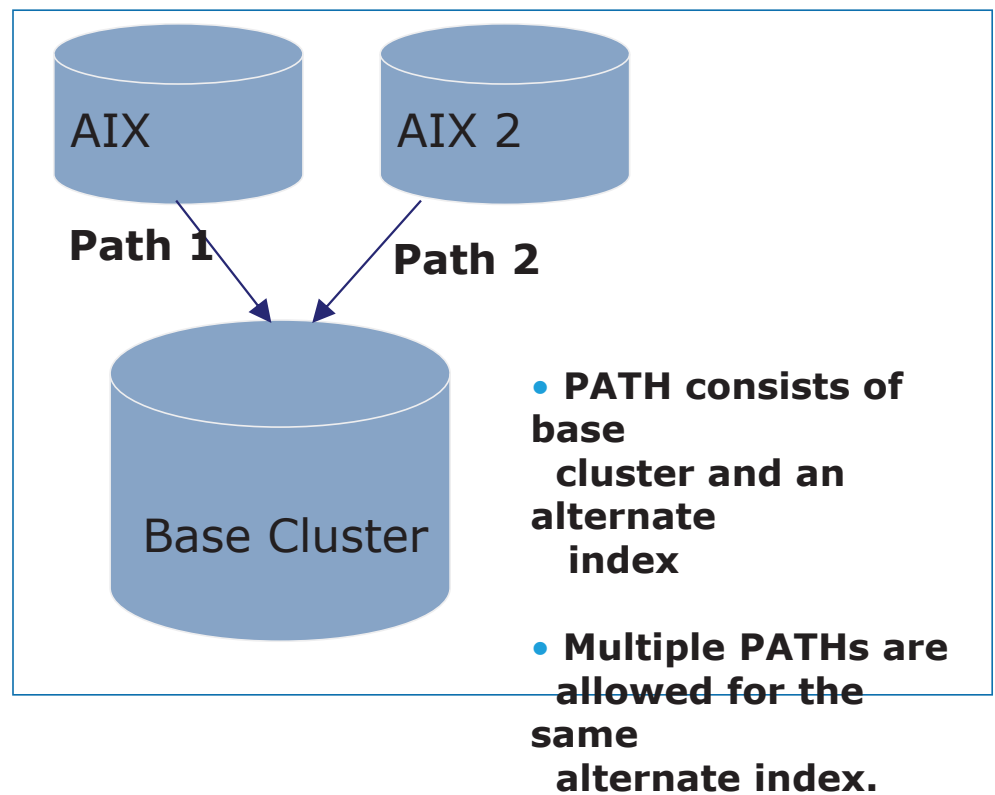
Pathname is the dataset name in JCL (DSN=PATHNAME).



5.2. Overview

Path

The DEFINE PATH command defines a path directly over an AIX and its related base cluster.





5.2. The DEFINE Path Command

The DEFINE PATH Command

A path is considered a VSAM object that does not contain any records. The path name is specified in the JCL for applications that access records via the alternate index.

Format	DEFINE PATH (NAME (<i>path name</i>) parameters)
Abbreviation	DEF PATH
Use	Alternate index over a KSDS, ESDS
Required	Yes, for an alternate index
Default	None
Example	DEFINE PATH (NAME (da0001t.lib.ksds.ename.path))



Capgemini Internal



5.2. The PATHENTRY Parameter

PATHENTRY Parameter

The PATHENTRY parameter specifies the alternate index cluster name. This establishes the path entry between alternate index cluster and the path.

Format	DEFINE PATH (NAME (<i>path name</i>) parameters)
Abbreviation	DEF PATH
Use	Alternate index over a KSDS, ESDS
Required	Yes, for an alternate index
Default	None
Example	DEFINE PATH (NAME (da0001t.lib.ksds.ename.path))



5.2. The UPDATE Parameter

The UPDATE Parameter

Format	<u>UPDATE</u> NOUPDATE
Abbreviation	<u>UPD</u> NUPD
Use	Alternate index over a KSDS, ESDS
Required	Yes, for an alternate index.
Default	UPDATE
Example	NOUPDATE



5.2. The UPDATE Parameter

The UPDATE Parameter

Changes made to the base cluster would be reflected automatically in all other alternates indexes provided that you have specified UPDATE in the DEFINE PATH and the UPGRADE in DEFINE AIX commands.

For this, the application program has to open the alternate index path.



5.2. The UPDATE Parameter

Define Path - Example

```
//STEP1      EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT =*
//SYSIN      DD *
    DEFINE PATH (NAME (DA0001T.LIB.KSDS.EMPNAME.PATH)
-
    PATHENTRY (DA0001T.LIB.KSDS.EMPNAME.AIX)
-
    UPDATE
-
    )
/*
//
```



5.2. Building The Alternate Index

The BLDINDEX Command

Note :

To populate the data component of ALT. index with records

VSAM constructs the index component of ALT. Index as it does for any KSDS.

Assume absolute control of base cluster by DISP=OLD.

Output dataset can be ALT. index cluster or pathname.



5.2. Building The Alternate Index

The BLDINDEX Command

Format	BLDINDEX INFILE (<i>ddname</i>) OUTFILE (<i>ddname</i>) INTERNALSORT EXTERNALSORT INFILE and OUTFILE to DD statements BLDINDEX INDATASET (<i>dataset name</i>) OUTDATASET (<i>dataset name</i>) INTERNALSORT EXTERNALSORT
---------------	---



Capgemini Internal



5.2. Building The Alternate Index

The BLDINDEX Command

Abbreviation	<u>BIX</u>
Use	Alternate index over a KSDS.
Required	Yes, for an alternate index.
Default	None
Example	NOUPDATE



Capgemini Internal



5.2. Building The Alternate Index

The BLDINDEX Command(Cont...)

The BLDINDEX command does the following:

- The data component of the base cluster is read sequentially and pairs of key pointers are extracted.
- These pairs consist of the alternate key field and its corresponding primary key field.
- VSAM creates a temporary file with these records.
- This temporary file is sorted in ascending alternate key sequence.



```
//STEP1 EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT =*
//DD1 DD DSN=DA0001T.LIB.KSDS.CLUSTER,
// DISP=OLD
//IDCUT1 DD UNIT=SYSDA,SPACE=(TRK, (2, 1))
//IDCUT2 DD UNIT=SYSDA,SPACE=(TRK, (2, 1))
// SYSIN DD *
BLDINDEX
-
INFILE (DD1)
-
OUTDATASET (da0001t.lib.ksds.empname.aix)
-
INTERNALSORT
/*
//
```



5.2. Building The Alternate Index

The INFILE/INDATASET Parameter

The INFILE parameter specifies a DD statement that refers to the input dataset. The INDATASET parameter names the dataset.

Format	INFILE (<i>ddname</i>) INDATASET (<i>dataset name</i>)
Abbreviation	<u>BIX</u>
Use	IFILE IDS
Required	Yes, you must code one or the other.
Default	None
Example	INFILE (dd1) INDATASET (da0001t.lib.ksds.cluster)



5.2. Building The Alternate Index

The OUTFILE/OUTDATASET Parameter

The OUTFILE parameter specifies a DD statement that refers to the output dataset. The OUTDATASET parameter names the dataset.

Format	OUTFILE (<i>ddname</i>) OUTDATASET (<i>dataset name</i>)
Abbreviation	OFFILE ODS
Use	KSDS, ESDS.
Required	Yes, you must code one or the other.
Default	None
Example	OUTFILE (dd2) OUTDATASET (da0001t.lib.ksds.empname.aix)



5.2. Building The Alternate Index

Internalsort vs Externalsort

Internalsort is the default.

In case, there is no enough space for virtual storage to perform an internal sort, VSAM performs an external sort using the two workfiles.

The datasets would be used only if VSAM cannot perform an internal sort due to some reason.

Externalsort can be used only if the temporary files are specified.



DEFINE AIX Example

```
//STEP1          EXEC PGM=IDCAMS
//SYSIN          DD *
    DEFINE AIX -
    (NAME(DA0001T.LIB.KSDS.EMPNAME.AIX) -
    VOLUMES (BS3013) -
    RELATE(DA0001T.LIB.KSDS.CLUSTER) -
    UPGRADE -
    KEYS(25 9) -
    FREESPACE(20 10) -
    SHAREOPTIONS(1) -
    NONUNIQUEKEY) ) -
```



Define AIX Example (Cont....)

```
DATA(NAME(DA000A1T.LIB.KSDS.EMPNAME.DATA)-  
TRACKS(10 1)                                -  
RECORDSIZE(70 110)                          -  
      CISZ(4096)                             -  
      )                                       -  
INDEX(NAME(DA0001T.LIB.KSDS.EMPNAME.INDEX))  
/*
```



```
//STEP1 EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT =*
//SYSIN DD *
    DEFINE PATH -
    NAME(DA0001T.LIB.KSDS.EMPNAME.PATH) -
    PATHENTRY(DA0001T.LIB.KSDS.EMPNAME.AIX) -
    UPDATE -
    )
/*
//
```



BUILD THE ALTERNATE INDEX Example

JCL:

```
                //STEP1                EXEC PG=IDCAMS
                //SYSPRINT DD SYSOUT =*
//DD1           DD DSN=DA0001T.LIB.KSDS.CLUSTER,
                //                      DISP=OLD
//IDCUT1        DD UNIT=SYSDA,SPACE=(TRK, (2, 1))
//IDCUT2        DD UNIT=SYSDA,SPACE=(TRK, (2, 1))
                // SYSIN                DD *
BLDINDEX                      -
INFILE(DD1)                   -
OUTDATASET(DA0001T.LIB.KSDS.AUTHNAME.AIX) -
                INTERNALSORT
                /*
```


Summary



Defining and building an alternate indexes

- Steps to create an Alternate Indexes
- The DEFINE ALTERNATEINDEX Command
- DEFINE PATH Command
- BLDINDEX



Review Question

Question 1. _____parameter establishes the relationship between the base cluster and the alternate index via the use of the base cluster name

- UPGRADE
- PATHENTRY
- RELATE

Question 2. UPDATE specifies that the records are to be updated automatically whenever the records in the base cluster are updated.

- True/ False?



VSAM (Virtual Storage Access Method)

Lesson 06: COBOL VSAM Considerations



Lesson Objectives

COBOL and VSAM: What's New, What's Different?

- ENVIRONMENT and DATA DIVISION entries for VSAM files.
- PROCEDURE DIVISION statements for VSAM files
- VSAM ERROR processing.



6.1: COBOL VSAM Considerations

ENVIRONMENT and DATA DIVISION Entries for VSAM Files

In the Input-Output Section of the Environment Division, you code a SELECT statement.

In the File Section of the Data Division, you code an FD entry and your file's record description.

The SELECT Statement



Format of SELECT statement for a VSAM dataset:

ENVIRONMENT DIVISION.

INPUT-OUTPUT SECTION.

FILE-CONTROL.

 SELECT file ASSIGN TO DDNAME / AS-DDNAME

 ORGANIZATION IS SEQUENTIAL/INDEXED/RELATIVE

 ACCESS MODE IS SEQUENTIAL/INDEXED/DYNAMIC

 RECORD KEY IS primary Key Dataname

 ALTERNATE KEY IS Alternate Key Dataname [With
Duplicates]

 FILE STATUS IS status-key



The SELECT Statement (contd..)

Format of SELECT statement for a VSAM dataset:

ENVIRONMENT DIVISION.

INPUT-OUTPUT SECTION.

FILE-CONTROL.

 SELECT file ASSIGN TO DDNAME / AS-DDNAME

 ORGANIZATION IS SEQUENTIAL/INDEXED/RELATIVE

 ACCESS MODE IS SEQUENTIAL/INDEXED/DYNAMIC

 RECORD KEY IS primary Key Dataname

 ALTERNATE KEY IS Alternate Key Dataname [With
Duplicates]

 FILE STATUS IS status-key



6.1: COBOL VSAM Considerations

Examples: SELECT Statement for VSAM files

Code Snippet 1:

```
SELECT INVTRAN ASSIGN TO AS-  
INVTRAN  
ORGANIZATION IS SEQUENTIAL  
ACCESS MODE IS SEQUENTIAL  
FILE STATUS IS INVTRAN-STATUS
```




6.1: COBOL VSAM Considerations

Examples: SELECT Statement for VSAM files

Code Snippet 2:

```
SELECT INVMASST ASSIGN TO INVMASST  
ORGANIZATION IS RELATIVE  
ACCESS MODE IS RANDOM  
RELATIVE KEY IS ITEM-NUMBER  
FILE STATUS IS INVMASST-STATUS
```

```
SELECT CUSTMAST ASSIGN TO CUSTMAST  
ORGANIZATION IS INDEXED  
ACCESS MODE IS SEQUENTIAL  
RECORD KEY IS CM-CUSTOMER-NUMBER  
FILE STATUS IS CUSTMAST-STATUS.
```



6.1: COBOL VSAM Considerations

File Section Entries for VSAM Files

Should have the record structure.

If KSDS then key field must match with length and position of KEYS parameter in DEFINE CLUSTER information.



6.1: COBOL VSAM Considerations

Procedure Division Statements

Use the OPEN statement to connect to your programs to a VSAM file.

Then, use READ, WRITE, REWRITE, DELETE, and START statements to process the file's records.

READ statement is used for both sequential as well random processing with its two different forms.

Finally, use CLOSE statement to disconnect your programs from the VSAM file.



Alternate Index Processing

In JCL there must be a DD statement for base cluster and one or more DD statements for alternate index path name:

```
//LIBMAST DD DSN=DA0001T.LIB.KSDS.CLUSTER,DISP=SHR  
//LIBMAST1 DD  
DSN=DA0001T.LIB.KSDS.NAME.PATH,DISP=SHR  
//LIBMAST2 DD  
DSN=DA0001T.LIB.KSDS.DEPT.PATH,DISP=SHR
```



6.2: Alternate Index Processing

COBOL SELECT Clause for AIX

FD: Should have record description with a primary and alternate key dataname.

Key of reference: READ filename.

KEY IS: Primary or alternate key dataname.

```
SELECT  file  ASSIGN TO  LIBMAST  
        RECORD KEY IS .....  
        ALTERNATE KEY IS .....  
                        [WITH DUPLICATES]
```



6.3: VSAM I/O Error Processing

VSAM I/O ERROR Processing

I/O error handling is one vital area where VSAM dataset processing differs from non-VSAM dataset processing.

When processing non-VSAM datasets, most programmers code their application programs to ignore errors, because the access method would abend the program if a serious I/O error occurs.

This is not the case when processing VSAM dataset.



6.3: VSAM I/O Error Processing

VSAM I/O ERROR Processing (contd..)

I/O error handling is one vital area where VSAM dataset processing differs from non-VSAM dataset processing.

When processing non-VSAM datasets, most programmers code their application programs to ignore errors, because the access method would abend the program if a serious I/O error occurs.

This is not the case when processing VSAM datasets.



The COBOL FILE STATUS Key

VSAM places program control in the hands of the programmer, not the O/S.

For this reason, it is important to check the COBOL status key designated in the FILE STATUS clause after every I/O operation.

For some error keys you need toabend the program immediately; for others you can just display the key, record, and an informative message and continue processing.

Summary



COBOL and VSAM:

- What's New, What's Different:
 - ENVIRONMENT and DATA DIVISION entries for VSAM files
 - PROCEDURE DIVISION statements for VSAM files
 - VSAM ERROR processing



Review Question

Question 1: For a relative-record data set, specify _____ for organization clause of SELECT statement.

- Option 1: Indexed
- Option 2: Sequentially
- Option 3: Relative



Review Question

Question 2: A combination of the ORGANIZATION and ACCESS clauses makes it clear that although the file has indexed organization, you are going to access it sequentially

- Option 1: True
- Option 2: False

Question 3: While processing KSDS _____, we access its records in the ascending order of the key field.



VSAM (Virtual Storage Access Method)

Lesson 07: Reorganizing VSAM datasets



Lesson Objectives

In this lesson, you will learn:

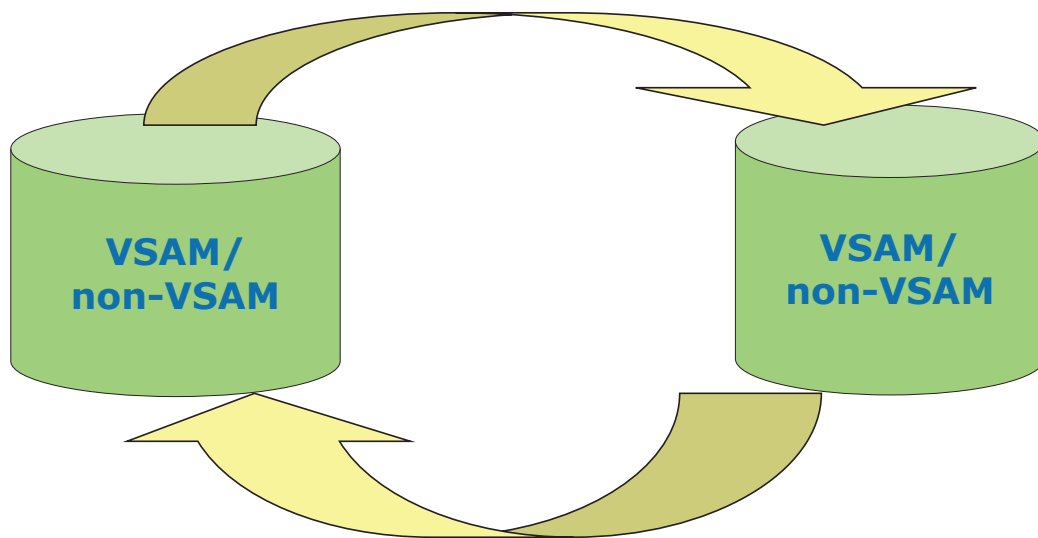
- REPRO Command
- Backing up VSAM Datasets



7.1. REPRO command

Repro

REPRO is an all-purpose load and backup utility command.
It is VSAM's version of the old IBM warhorse, IEBEGENER.





Functions Of Repro

REPRO performs three basic functions:

- Loads empty VSAM cluster with records. The data and index components (for KSDS) are built automatically.
- Creates backup of a VSAM dataset on a physical sequential dataset.
- Merge data from two VSAM datasets.



Advantages Of Repro

The following things can be done using Repro:

- An ISAM dataset can be converted to a VSAM format.
- A non-VSAM dataset can be copied to a Physical sequential to a or a partitioned dataset.
- Records can be copied from one type of VSAM dataset to another type of VSAM dataset, for example KSDS to ESDS.
- In case of KSDS, data and index components are built automatically.



Disadvantages of Repro

REPRO has the following disadvantages:

- Catalog information is not copied with the data.
- Prior DELETE, redefinition is required before loading the cluster; unless REUSE is specified in the DEFINE CLUSTER command.



Repro Format

Format: REPRO
INFILE (ddname) | INDATASET (dsname) -
OUTFILE (ddname) | OUTDATASET (dsname) -
optional parameters

Abbreviation: None

Use : KSDS, ESDS. RRDS

Required: Yes.

Default: None

Example: REPRO
INFILE (dd2) -
OUTDATASET (da0001t.lib.ksds.empname.aix)-



Limiting Input and Output Records

The following parameters can be used to limit the input for the Repro command:

- FROMKEY – TOKEY
- FROMADDRESS- TOADDRESS
- FROMNUMBER-TONUMBER
- SKIP,COUNT



7.2. The FROMKEY and TOKEY parameters

FROMKEY-TOKEY

FROMKEY specifies the key of the input record at which the reading begins.

TOKEY specifies the key of the last input record to write.

- You can code either or both parameters.
 - Format: FROMKEY (key)
 - TOKEY (key)



FROMKEY-TOKEY (Cont...)

Abbreviation	:FKEY, TKEY
Use	:KSDS, ISAM
Required	:No
Default record	:First record in dataset for FROMKEY and last in dataset for TOKEY.
Example	:FROMKEY (PQR) TOKEY (XYZ)



7.3. The FROMADDRESS and TOADDRESS parameters

FROMADDRESS-TOADDRESS

FROMADDRESS specifies the Relative Byte Address (RBA) value of the key of the input record at which copy to begin.

TOADDRESS specifies the Relative Byte Address (RBA) value of the key of the last input record to be copied.

You can code either or both parameters.



FROMADDRESS-TOADDRESS(FORMAT)

Format: FROMADDRESS (address)
TOADDRESS (address)

Address: The RBA address of the first or last record to copy

Abbreviation: FADDR, TADDR

Use: KSDS, ESDS

Required: No

Default: First record in dataset for FROMADDRESS and last record in dataset for TOADDRESS.

Example: FROMADDRESS (6250)
TOADDRESS (12684)



7.4. The FROMNUMBER and TONUMBER parameters

FROMNUMBER-TONUMBER

FROMNUMBER specifies the relative record number of the first RRDS record to copy.

TONUMBER specifies the relative record number of the last record to copy.

You can code either or both parameters.



FROMNUMBER-TONUMBER(FORMAT)

Format: FROMNUMBER(relative number)

TONUMBER(relative number)

relative number: The relative record number of the first or
last record to copy. The first record is
0.

number

Abbreviation: FNUM, TNUM

Use: RRDS

Required: No

Default: First record in dataset for FROMNUMBER and last record in
dataset for TO NUMBER.

FROMNUMBER (10)

TONUMBER (1000)

Example:



7.5. The SKIP and COUNT parameters

SKIP AND COUNT PARAMETERS

SKIP specifies the number of input records to skip before beginning to copy.

COUNT specifies the number of output records to skip to copy.

You can code either or both parameters.



SKIP AND COUNT EXAMPLES

Format: SKIP (number)

COUNT (number)

Number: The number of records to skip or copy.

Abbreviation: None

Use: KSDS, ESDS, RRDS, non-VSAM datasets

Required: No

Default: SKIP defaults to the first record in dataset
and COUNT defaults to all the records in
dataset.

the

Example: SKIP (10)

COUNT (1000)



7.6. Backing up VSAM Datasets

Backing up a VSAM dataset

It is good to take a backup of VSAM datasets on a regular basis.
Backing up a VSAM dataset involves only one step.



Backing up a VSAM dataset(Cont...)

```
//JOBNAME DA0001TA...
//STEP10 EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT = *
//DD2 DD DSN=DA0001T.KSDS.BACKUP(+1),
// DISP=(NEW,CATLG,DELETE),UNIT=TAPE,
// VOL=SER=32970,
// DCB=(RECFM=FB,LRECL=80,BLKSIZE=0)
//SYSIN DD *
        REPRO
        INDATASET (DA0001T.LIB.KSDS.CLUSTER) -
        OUTFILE (DD2)

/*
//
```



Restoring and Rebuilding a VSAM dataset

Backing up a VSAM dataset involves only one step.

Restoring and rebuilding it from backup involves one step with three parts.

DELETE-DEFINE-REPRO sequence is required to restore the cluster in case of KSDS.

- Delete the original cluster using IDCAMS DELETE command.
- Redefine the cluster using IDCAMS DEFINE CLUSTER command.
- Load the empty cluster with data using the IDCAMS REPRO command.

DELETE-DEFINE-REPRO



```
//DD1 DD DSN=DA0001T.LIB.KSDS.BACKUP(0),  
//                               DISP=OLD, UNIT=TAPE  
//SYSIN DD *  
DELETE DA0001T.LIB.KSDS.CLUSTER  
DEFINE CLUSTER (NAME(DA0001T.LIB.KSDS.CLUSTER)      -  
                INDEXED                             -  
                KEYS(4 0)                             -  
                RECORDSIZE(80 80)                       -  
                VOLUMES(BS3013)                         -  
                )                                       -  
                DATA(NAME(DA0001T.LIB.KSDS.DATA)) -  
INDEX(NAME(DA0001T.LIB.KSDS.INDEX))
```

DELETE-DEFINE-REPRO(Cont...)



REPRO

-

INFILE(DD1)

-

OUTDATASET(DA0001T.LIB.KSDS.CLUSTER)

/*



DELETE-DEFINE-REPRO(Effects)

The dataset is reorganized, that is the control interval and control area splits are eliminated.

Free space is redistributed throughout the dataset, as specified in the FREESPACE parameter.

Primary index is rebuilt, however the DELETE command deletes the base cluster as well as its indexes.

Hence alternate indexes have to be redefined.



The REUSE parameter with REPRO

If the VSAM cluster has been originally defined with the REUSE parameter, you can specify REUSE again with the REPRO command when restoring it.

```
//DA0001TA          JOB
//STEP1             EXEC PGM=IDCAMS
//SYSPRINT          DD SYSOUT = *
//DD1               DD *
123456789123456789
AAAAAAAAABBBBBBBCCCC
/*//DD2             DD DSN=DA0001T.ESDS.CLUSTER,DISP=SHR
//SYSIN             DD *
    REPRO                -
    INFILE (DD1)         -
    OUTFILE (DD2)        -
    REUSE                /*
```



Delete Command

Use the DELETE command to delete both VSAM and non-VSAM objects.

Format:	DELETE entry name
Entry name:	The name of the entry to be deleted
Abbreviation:	DEL
Use:	KSDS, ESDS, RRDS, non-VSAM
Required:	No
Default:	None
Example:	DELETE da0001t.lib.ksds.cluster



Optional Parameters

Aix : Requests that only the alternate index of that name be deleted.

Cluster: Requests that only the cluster of that name be deleted.

NONVSAM: Requests that only the non-VSAM dataset of that name be deleted.

PATH: Requests that only the path of that name be deleted.

GDG: Requests that only the Generation data group of that name be deleted.

ERASE|NOERASE: ERASE Specifies that the deleted item is to be overwritten with binary zeros.

FORCE|NOFORCE: FORCE allows an item to be deleted even if it is non-empty.

PURGE|NOPURGE: PURGE allows an item to be deleted even if its expiration date has not arrived.



Optional Parameters(Cont...)

SCRATCH|NOSCRATCH: SCRATCH scratches the non-VSAM datasets. NOSCRATCH just un-catalogs the dataset.

Unless ERASE parameter is specified VSAM performs a logical delete on the selected object.

In such a case, only the catalog entry is removed.

The data still exists, but it can now be overwritten by new data.

If the cluster is already defined or altered with the NOERASE option and the ERASE parameter is specified at the time of deletion, the ERASE parameter is ignored.



Reorganization of a KSDS

Need for reorganization

- Frequent adds and updates to a KSDS in batch and on-line (CICS) environments can cause CI and CA splits. Although the data set is still logically in collating sequence on the prime key after such splits, the records are physically out of sequence.
- While you can still access these records randomly and sequentially, access time increases. These I/O delays can cause response time problems in CICS/VS and other online systems and also increase the run time of batch jobs. In order to put the KSDS records back into physical sequence, it is necessary to reorganize the data set.
- Remember that reorganization is needed only for KSDS and not for an ESDS or an RRDS.
- Usually, if the total number of CA splits exceeds 15 percent of the total number of CA's in the data set, it is best to reorganize it.
- The LISTCAT command can be used to determine the number of CI and CA splits in a data set.



Reorganization of a KSDS

Reorganization is a four step procedure

- 1. Make a backup of the KSDS into a PS using REPRO command.
- 2. Delete the KSDS with the AMS DELETE command.
- 3. Define the KSDS using DEFINE CLUSTER command.
- 4. Reload the KSDS from the backup file using the REPRO command.



Export / Import

Commands can be used for backup and recovery . You can export a dataset, alternate index or a catalog to a different system.

EXPORT / IMPORT has several advantages compared to REPRO

- Catalog information is exported along with data
- Cluster deletion and redefinition not required during import as input dataset already contains catalog information
- Easily ported on other systems as catalog information available with data
- Like REPRO KSDS datasets are reorganized however three steps of REPRO are replaced by one

Disadvantages:

- Exported data cannot be processed until Imported
- Can be used only for VSAM dataset



Export

EXPORT FORMAT :

```
EXPORT entryname | password  
      OUTFILE (ddname) |  
      OUTDATASET (dsname)  
      Optional parameters
```

Example

```
EXPORT MFCVT01.KSDS.CLUSTE  
      OUTFILE(DD2)
```

The output dataset from an EXPORT must always be a sequential dataset (usually on a tape)

Export



Optional parameters are

- INHIBITSOURCE / NOINHIBITSOURCE
 - NOINHIBITSOURCE : Data set can be updated after backup
 - INHIBITSOURCE : Cannot be updated after backup
- INHIBITTARGET / NOINHIBITTARGET

NOINHIBITTARGET : Specifies that the data set will be available for processing after it is restored

- INHIBITTARGET : Not available
- TEMPORARY / PERMANENT

Import



IMPORT Format :

```
IMPORT          -  
INFILE (ddname) | INDATASET (dsname)-  
OUTFILE (ddname) | OUTDATASET (dsname) -  
Optional parameters:
```

Example :

```
IMPORT INFILE (DD2)-  
OUTDATASET(MFCVT01.KSDS.CLUSTER)  
Imports only EXPORTED dataset
```



Summary

In this lesson, you have learnt:

- REPRO command
- REPRO considerations
- REPRO Command
- Limiting Input and Output Records
- The FROMKEY and TOKEY parameters
- The FROMADDRESS and TOADDRESS parameters
- The FROMNUMBER and TONUMBER parameters
- The SKIP and COUNT parameters

Backing up VSAM Datasets



Review Question

Question 1: _____ specifies the relative record number of the first RRDS record to copy.

- Option 1: FROMNUMBER
- Option 2: FROMADDRESS
- Option 3: FROMKEY

Question 2: REPRO loads empty VSAM cluster with records.

- True/False?

Question 3: _____ specifies the number of output. records to skip to copy.



VSAM (Virtual Storage Access Method)

Lesson 08: VERIFY, PRINT, DELETE, ALTER Command



Lesson Objectives

The VERIFY command
The DELETE command
The PRINT command
The ALTER command

Capgemini Public



8.1: VERIFY Command

The VERIFY Command

Verifies and updates catalog with information from the physical end of the data in the cluster via HURBA.

Verifies that catalog HURBA field stores true values from the control block HURBA field in the data component.

VERIFY DATASET (DA0001T.LIB.KSDS.CLUSTER)

Can be issued from a TSO or within a JCL statement either the FILE or DATASET parameter.



8.1: VERIFY Command

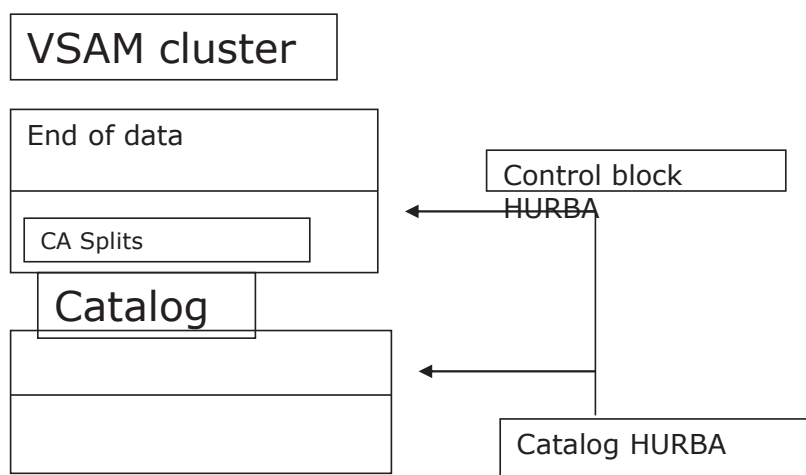
Example

```
//VERIFY JOB 'JAY  
MOSELEY',CLASS=A,MSGLEVEL=(1,1),MSGCLASS=A  
//IDCAMS EXEC PGM=IDCAMS,REGION=4096K  
//SYSPRINT DD SYSOUT=A  
//SYSIN DD *  
    VERIFY DATASET(MVS801.STUDENT.FILE)  
/*  
//
```



8.1: VERIFY Command

Relationship Between HURBA in Catalog and Cluster





8.2: DELETE Command

DELETE Command

Used to delete both VSAM and non-VSAM objects:

Format	:	DELETE entry name
Entry name	:	The name of the entry to delete.
Abbreviation	:	DEL
Use	:	KSDS, ESDE, RRDS, non-VSAM
Required	:	No
Default	:	None
Example	:	DELETE da000 1t.lib.ksds.cluster



8.2: DELETE Command

Optional Parameters

Following are parameters:

- Aix:
 - Requests deletion of the alternate index of only that name
- Cluster:
 - Requests deletion of the cluster of only that name
- NONVSAM:
 - Requests that only the non-VSAM dataset of that name be deleted
- PATH:
 - Requests that only the path of that name be deleted
- GDG:
 - Requests deletion of the Generation data group of that name only



8.2: DELETE Command

Optional Parameters (Cont...)

ERASE|NOERASE:

- Specifies that the deleted item is to be overwritten with binary zeros.

FORCE|NOFORCE:

- Allows an item to be deleted even if it is non-empty.

PURGE|NOPURGE:

- Allows an item to be deleted even if its expiration date has not arrived.

SCRATCH|NOSCRATCH:

- Scratches non-VSAM datasets.

NOSCRATCH just un-catalogs the dataset.



8.2: DELETE Command

Optional Parameters (Cont...)

Unless ERASE parameter is specified VSAM performs a logical delete on the selected object.

- In such a case only the catalog entry is removed.
- Data still exists but it can now be overwritten by new data.

If the cluster is already defined or altered with the NOERASE option and the ERASE parameter is specified at the time of deletion, the ERASE parameter is ignored.



8.3: PRINT Command

PRINT Command

Used to view the contents of the dataset.

- Format:
 - PRINT INDATASET (entry name) options
 - PRINT INFILE (ddname)
- Entry name: The name of the entry to print.
- ddname: The name of a DD statement.

Abbreviation:

PRINT IDS (entry name)
PRINT IFILE (ddname)

Capgemini Public



8.3: PRINT Command

PRINT Command (Contd..)

Use: KSDS, ESDS, RRDS, non-VSAM

Required: No

Default: None

Example:

```
PRINT da0001t.lib.ksds.cluster  
PRINT INDATASET (da0001t.lib.ksds.cluster)  
PRINT INFILE (dd1)
```




8.3: PRINT Command

PRINT Command (Contd..)

Default output destination for PRINT is SYSPRINT.

- Following options are allowed:

CHARACTER | DUMP | HEX or CHAR:

- Specifies the format in which to print.

FROMADDRESS, [TOADDRESS]:

- Specifies relative byte address at which to begin and end printing.

FROMKEY,[TOKEY]:

- Specifies the first key to print and the key of last record to print.



8.3: PRINT Command

Print Command (Options)

FROMNUMBER, [TONUMBER]:

Specifies the relative record number of the first and the last record to print.



8.3: PRINT Command

OUTFILE Parameter

To direct printed output other than to SYSPRINT, for example, dataset, add the OUTFILE parameter.

Format:

- OUTFILE (ddname)
- ddname:
 - Name of a DD statement that specifies where the output is to be written.

Abbreviation: OFILE

- Use: KSDS, ESDS, RRDS, non-VSAM



8.3: PRINT Command

OUTFILE Parameter (Contd...)

Required: No, because of default.

Default: OUTFILE (SYSPRINT).

Example:

```
PRINT da0001t.lib.ksds.cluster
-
      OUTFILE (dd1)
PRINT INDATASET
(da0001t.lib.ksds.cluster)      -
      OUTFILE (dd1)
```

Capgemini Public



8.4: ALTER Command

ALTER Command

Changes certain dataset attributes.

Format of Alter command is as follows:

- Format: ALTER entry name/password parameters
- Entry name: The name of the entry to alter
- Abbreviation: None
- Use: KSDS, ESDS, RRDS, non-VSAM
- Required: No
- Default: None



8.4: ALTER Command

ALTER Command (Contd...)

Example:

```
ALTER da0001t.lib.ksds.cluster  
-  
FSPC (30 20)
```

Can be used to change certain attributes of a previously defined VSAM object.

- Objects include:
 - Base cluster and its data or index component
 - An alternate index cluster and its data or index component.



8.4: ALTER Command

Alterable Attributes

Following is a list of common attributes that are completely alterable:

- Add / Remove Volumes
- Change Keys and Uniqueness
- Change Record Size
- Change Upgrade Option
- Change % of FREESPACE etc.

Capgemini Public



8.4: ALTER Command

Alterable Attributes

ADDVOLUMES (volumes)

AUTHORIZATION (entry string)

BUFFERSPACE (size)

ERASE | NOERASE

FREESPACE (ci%, ca%)

MASTERPW (password)



8.4: ALTER Command

Alterable Attributes (Contd...)

NEWNAME (newname)
READPW (password)
SCRATCH | NOSCRATCH (for GDG)
SHAREOPTIONS (cross region cross system)
UPGRADE | NOUPGRADE
UPDATE | NOUPDATE
UPDATEPW (password)



8.4: ALTER Command

Alterable Attributes (Contd...)

Following attributes are alterable only for empty clusters:

- KEYS (length offset)
- RECORDSIZE (avg max)
- UNIQUEKEY | NONUNIQUEKEY



8.4: ALTER Command

TO(date)

Value for TO can be specified in one of the two formats:

- Using a two-byte year and a three-byte day of the year.
 - Example 1: 92015 Assumes January 15,1992.
 - Example 2: 98365 Assumes December 31,1998.
- Using a four-byte year and a three byte day of the year.
 - Example 1: 1992015 Assumes January 15,1992.
 - Example 2: 2001001 Assumes January 1,2001.



8.4: ALTER Command

For(days)

For(9998) directs VSAM to retain the dataset for 9,998 days.

For(9999), the highest value allowed ,directs VSAM to retain the dataset indefinitely.



8.4: ALTER Command

Alter Examples

Example: Altering Name of a Dataset

```
ALTER          DA0001T.LIB.KSDS.CLUSTER      -  
NEWNAME (DA0001T.MY.CLUSTER)
```

Example: Adding Additional Volumes

```
ALTER          DA0001T.LIB.KSDS.CLUSTER      -  
ADDVOLUMES (BS3001 BS3005)
```



8.4: ALTER Command

Unalterable Attributes

Following attributes are cannot be altered:

- CISZ
- REUSE/NOREUSE

In case these parameters are to be altered, the cluster has to be deleted and redefined with the new attributes.

Lab



Lab – To Do assignments on various commands

Capgemini Public



Summary

- **The VERIFY command**
- **The DELETE command**
- **The PRINT command**
- **The ALTER command**



Review Question

Question 1: The DELETE command can be used to delete_____.

- Option 1: non-VSAM objects
- Option 2: VSAM objects
- Option 3: None of the above

Question 2: VERIFY verifies that the catalog HURBA field stores the true values from the control block HURBA field in the data component.

- Option 1: True
- Option 2: False



VSAM (Virtual Storage Access Method)

Lesson 09: Generation Data Groups



Lesson Objectives

Creation of GDGs

Modifying Features of GDG

Deleting GDG Index

Adding a Dataset to a GDG

Deleting GDG Index and Datasets



9.1: Generation Data Groups

Overview

A Generation Data Group (GDG) is a group of related datasets, usually created in chronological order.

Simplifies data management:

- Use a GDG to control creation and retention of this type of dataset group.

Before you create the first generation of a GDG, perform the following steps:

- First, the GDG base, which contains rules to manage the group, is defined using AMS.
- Second, create a model for DCB parameters of individual datasets to be created.



9.1: Generation Data Groups

Overview (contd..)

Generations can continue until a specified limit is reached.

Limit specifies total number of generations that can exist at any one time.

Once limit is reached, the oldest generation is deleted.

A GDG (referred to as Generation Data Set) is used when JCL must remain unchanged and yet the name of the dataset must be different for different executions.

MVS uses the generation data group's catalog entry to keep track of relative generation numbers.

As a result, GDGs must be cataloged.

- Each generation dataset that is a part of the group must be cataloged too.



9.1: Generation Data Groups

Overview (contd..)

When you create a generation data group's catalog entry, specify the number of generations that should be maintained.

- For example, you can specify five generations including the current generation.

During each processing cycle, the new version of the file becomes the current version.



9.2: Specifications to Create a GDG

Specifications to Create a GDG

Name of the GDG.

Number of generations that are to be retained. (Limit)

Action when number of generations to be retained reaches the limit.
(EMPTY/NOEMPTY)



9.3: Features of GDG

Features of GDG

All datasets within a GDG will have the same name.

Generation number of a dataset within a GDG is automatically assigned by OS when created.

Datasets within a GDG can be referenced by their relative generation number.

Generation 0 always references current generation.

Creating GDG



JCL for creating index

```
//STEP1      EXEC      PGM=IDCAMS
```

```
//SYSIN      DD
```

```
    DEFINE GDG                                -  
        (NAME(DA0001T.ACCOUNTS.MONTHLY) -  
        LIMIT(5)                                -  
        EMPTY                                -  
        SCRATCH                                -  
        )
```

```
/*
```

```
//
```



Modifying GDG

Use the ALTER command:

```
//STEP1 EXEC PGM=IDCAMS
//SYSIN DD
        ALTER DA0001T.ACCOUNTS.MONTHLY -
                NOSCRATCH -
                EMPTY

/*
```

When NOSCRATCH is specified the GDG is UNCATALOGED.

Generations are visible through the 3.4 option and can be accessed by specifying unit and volser.



Deleting GDG

Can be deleted by the DELETE parameter of IDCAMS.

Results in an error on reference to any generation datasets of the GDG.

```
/STEP1 EXEC PGM=IDCAMS
//SYSIN DD
        DELETE DA0001T.ACCOUNTS.MONTHLY GDG
/*
//
```



Deleting GDG Index and Datasets

FORCE parameter in the DELETE statement of IDCAMS can be used .

This deletes the GDG as well as all the generations.

The default is NOFORCE in which case a GDG cannot be deleted unless it empty. i.e. it has members.

- Example:

```
/STEP1 EXEC PGM=IDCAMS
```

```
//SYSIN DD
```

```
DELETE DA0001T.ACCOUNTS.MONTHLY
```

```
-
```

```
GDG
```

```
-
```

```
FORCE
```

```
-
```

```
/*
```

```
//
```



Deleting GDG Index and Datasets (contd..)

file.c1(+1)	Next Generation
file.c1(0)	Current Generation
file.c1(-1)	Previous Generations
file.c1(-2)	
file.c1(-3)	

- **There are three previous generation.**
- **Generations are numbered relative to the current generation, file.c1(0).**
- **Relative generation numbers are adjusted when each processing cycle completes, so that the current generation is always referred to as relative generation 0.**



Deleting GDG Index and Datasets (contd..)

Any number can theoretically be used inside the parentheses between -254 and +354, with 0 and +1 the most common by far.

Note that negative numbers do not work the same way as the positive.

Example:

- file.c1(-1) simply means go one back from the latest file.c1(+2). System find the latest entry in the catalog, adds the number which appears in the parenthesis, +2 to the number that follows the "G" (GnnnnVnn) and generates a new name.

MVS uses "Absolute Generation Numbers" in the form GnnnnV00 to identify each generation dataset uniquely. GnnnnV00 represents the chronological sequence number of the generation, beginning with G0000.



Deleting GDG Index and Datasets (contd..)

V00 is a version number, which lets you maintain more than one version of a generation. Each time a new generation dataset is created, MVS adds one to the sequence number.

Let us have a look at the catalog entries of GDG.

ENTRIES IN THE CATALOG

```
-----  
DSN                      VOL          UNIT            
-----          -----          -----  
DA0001T.MASTER.DAY.G0001V00 BS3007          3390  
DA0001T.MASTER.DAY.G0002V00 BS3007          3390  
DA0001T.MASTER.DAY.G0003V00 BS3007          3390
```



Deleting GDG Index and Datasets (contd..)

DA0001T.MASTER.DAY.G0003V00

Remark:

DA0001T.MASTER.DAY is often referred to as GDG base.

DA0001T.MASTER.DAY(0) and DA0001T.MASTER.DAY(+1)

Known as "relative" names.

DA0001T.MASTER.DAY.G0003V00 as "absolute" names.

Absolute names are actual names of the datasets.



Deleting GDG Index and Datasets (contd..)

<Filename>.GnnnnV00

- The first portion of the dataset name filename is provided by the user and can be 35 characters. The last portion (GnnnnVnnnn) is provided by the system and is always eight characters long:
 - Example: DA0001T.MASTER.DAY.G0003V00
- Note: Since the catalog is the main tool used by the system to translate relative to absolute names, it is mandatory to catalog new generations of GDC.

Deleting GDG Index and Datasets (contd..)



```
// IN      DD  DSN=DA0001T.MASTER,  
DISP=SHR  
// OUT DD  DSN=DA0001T.MASTER.DAY(+1),  
//        DISP= (NEW,CATLG,DELETE),  
//        UNIT=3390, VOL=SER=BS0006,  
//        SPACE= (CYL,(10,5),RLSE),  
//        DCB=(PROD.GDGMOD,  
//        BLKSIZE=23440,LRECL=80,RECFM=FB)
```



DA0002T.MASTER.DAY(0)

---> Relative

Name

DA0002T.MASTER.DAY.G00001V00 --> Absolute

Name

Defining a GDG

```
// Step1          EXEC  PGM=IDCAMS
```

```
// SYSPRINT       DD  SYSOUT = *
```

```
// SYSIN          DD  *
```

```
                DEFINE
```

```
GDG(NAME(DA0001T.MASTER.DAY-
```

```
        LIMIT(5)          -
```

```
        SCRATCH           -
```

```
        EMPTY)
```

```
/*
```

```
//
```



Deleting GDG Index and Datasets (contd..)

NAME parameter identifies the GDG base.

LIMIT parameter identifies the maximum number of generations available at any point in time.

- SCRATCH/NOSCRATCH parameter:
 - When you specify SCRATCH in the DEFINE GDG, the oldest generation is deleted and the catalog entry is also removed.
 - However, NOSCRATCH, which is default, only removes the catalog entry but does not delete the dataset.



Deleting GDG Index and Datasets (contd..)

EMPTY/NOEMPTY parameter

When you specify EMPTY in the DEFINE GDG, all previous generations get uncataloged (or deleted depending on SCRATCH/NOSCRATCH) when the limit is exceeded.

NOEMPTY, which is default, only removes the oldest generation.



Deleting GDG Index and Datasets (contd..)

Following code contains 1 job with 2 steps:

```
//DA0003TA JOB
//UPDATE      EXEC PGM=PAY3200
//OLDMAST DD
DSN=DA0001T.MMA2.PAY.MAST(0),DISP=OLD
//NEWMAST DD  DSN=DA0001T.MMA2.PAY.MAST(+1),
//                               DISP=
(NEW,CATLG),UNIT=3300,
//                               VOL=SER=BS3001,
//
DCB=(LRECL=80,BLKSIZE=1600)
//PAYTRAN DD
DSN=DA0001T.MMA2.PAY.TRAN,DISP=OLD
//PAYLIST DD  SYSOUT=*
```

Deleting GDG Index and Datasets (contd..)



```
//REPORT    EXEC PGM=PAY3300  
//PAYMAST   DD  
DSN=DA0001T.MMA2.PAY.MAST(+1),DISP=OLD  
//PAYRPT    DD SYSOUT=*
```



Deleting GDG Index and Datasets (contd..)

Following code contains 2 jobs:

```
//DA0003TA      JOB
//UPDATE EXEC PGM=PAY3200
//OLDMAST       DD
DSN=DA0001T.MMA2.PAY.MAST(0),DISP=OLD
//NEWMAST       DD
DSN=DA0001T.MMA2.PAY.MAST(+1),
//              DISP=
(NEW,CATLG,DELETE),UNIT=3390,
//              VOL=SER=BS3001,
//              DCB=(LRECL=80,BLKSIZE=1600)
//PAYTRAN       DD
```


Deleting GDG Index and Datasets (contd..)



```
DSN=DA0001T.MMA2.PAY.TRAN,DISP=OLD
//PAYLIST DD  SYSOUT=*
//*
```

```
JOB2                JOB .....
//REPORT EXEC PGM=PAY3300
//PAYMAST          DD
DSN=DA0001T.MMA2.PAY.MAST(0),DISP=OLD
//PAYRPT DD  SYSOUT=*
//
```



9.4: Characteristics of GDGs

Characteristics of GDGs

Changing the Characteristics of GDG:

- It sometimes becomes necessary to change the characteristics of a GDG such as SCRATCH, or EMPTY, after you create several generations.
- Use the ALTER command of IDCAMS to change easily NOSCRATCH to SCRATCH and EMPTY to NOEMPTY (or vice versa).
- Example:

```
//STEP1 EXEC PGM=IDCAMS
//SYSIN DD
        ALTER DA0001T.MASTER.DAY      -
        NOSCRATCH                      -
        EMPTY
/*
//
```



Characteristics of GDGs (contd..)

Delete GDG Index:

- Use DELETE parameter of IDCAMS.

```
/STEP1 EXEC PGM=IDCAMS
//SYSIN DD
        DELETE DA0001T.MASTER.DAY GDG FORCE
/*
//
```



Characteristics of GDGs (contd..)

Deleting GDG Index and Datasets

FORCE parameter in the DELETE statement of IDCAMS can be used if GDG is not empty

Example:

```
/STEP1 EXEC PGM=IDCAMS
//SYSIN DD
        DELETE DA0001T.MASTER.PAY      -
                                GDG      -
                                FORCE
/*
//
```



9.5: Advantages of GDGs

Advantages of GDGs

Record Keeping is the responsibility of the Operating System, and not of the programmer.

Convenient way to relate data sets together and automatic deletions of outdated data sets.



Summary

Creation of GDGs

Modifying Features of GDG

Deleting GDG Index

Adding a Dataset to a GDG

Deleting GDG Index and Datasets



Review Question

Question 1: The DISP parameter must be set to ____ for all new generation data sets.

- Option 1: UNCATLG
- Option 2: CATLG
- Option 3: EMPTY

Question 2: Generations can continue until a specified limit is reached.

- True/False

Question 3: _____specifies number of generations belonging to GDG.

Virtual Storage Access Method

Lab Book

Document Revision History

Date	Revision No.	Author	Summary of Changes
03-Nov-09	2.0	Rajita Dhumal	Revamped
04-Nov-09	2.1	Ummeaiman Diwanji	Quality Review, Transfer to new template.
30-June-2011	3.0	Rajita Dhumal	Revamped
30-Aug-2012	3.1	Rajita Dhumal	Revamped after Assignment Review

Table of Contents

<i>Document Revision History</i>	<i>2</i>
<i>Table of Contents</i>	<i>3</i>
<i>Getting Started</i>	<i>4</i>
<i>Overview.....</i>	<i>4</i>
<i>Setup Checklist for VSAM</i>	<i>4</i>
<i>Pre-requisites:</i>	<i>4</i>
<i>Requirements</i>	<i>4</i>
<i>Lab 1. Executing IDCAMS (on IBM Mainframes)</i>	<i>6</i>
<i>1.1: Create KSDS Cluster.....</i>	<i>6</i>
<i>Lab 2. Development Assignments: Set 1</i>	<i>9</i>
<i>Lab 3. Analysis, Enhancement and Debugging Assignments: Set 2</i>	<i>12</i>
<i>Problem Statement 1: Refer to VSAM02-MAST-KSDS dataset. The given code</i>	
<i>list a catalog and verify a data set</i>	<i>12</i>
<i>Lab 4. Analysis, Enhancement and Debugging Assignments: Set 3</i>	<i>14</i>
<i>Appendices</i>	<i>15</i>
<i>Appendix A: Table of Figures.....</i>	<i>15</i>

Getting Started

Overview

This Lab book is a guided tour for Virtual Storage Access Method (VSAM). It contains solved examples and “To Do” assignments. Follow the steps provided in the solved examples and then work out the ‘To Do’ assignments given.

Setup Checklist for VSAM

Here is what is expected on your machine in order for the lab to work.

- *Hardware:*
- *Software:*

Pre-requisites:

It is expected that participants know the following:

- Good knowledge of COBOL, MVS, JCL
- Using the ISPF editor
- Various options available on ISPF such as dataset allocation, copy, member list of PDS, how to compress PDS, etc.

Requirements

You need to create four PDS, one each for storing COBOL source code, JCL code, IDCAMS, and loadlib. The convention followed for naming the PDS is as follows:

IBMID.NAME.X where IBMID is IBM login id used by the participant, NAME is name of the participant and X is either COBOL/SOURCE or JCL or LOADLIB.

For example, the participant using IBM login id DA0001T and name as PAI has three PDS as follows:

- *DA0001T.PAI.COBOL:* To store COBOL source programs as PDS members.
- *DA0001T.PAI.JCL:* To store JCL's as PDS members.
- *DA0001T.PAI.LOADLIB:* To store load modules of programs as PDS members.

Note: If these PDSs do not exist, then these are to be created. Kindly refer to the MVS lab book for the same.

Lab 1. Executing IDCAMS (on IBM Mainframes)

Goals	<ul style="list-style-type: none"> At the end of this lab session, you will be able to create VSAM cluster
Time	15 Minutes

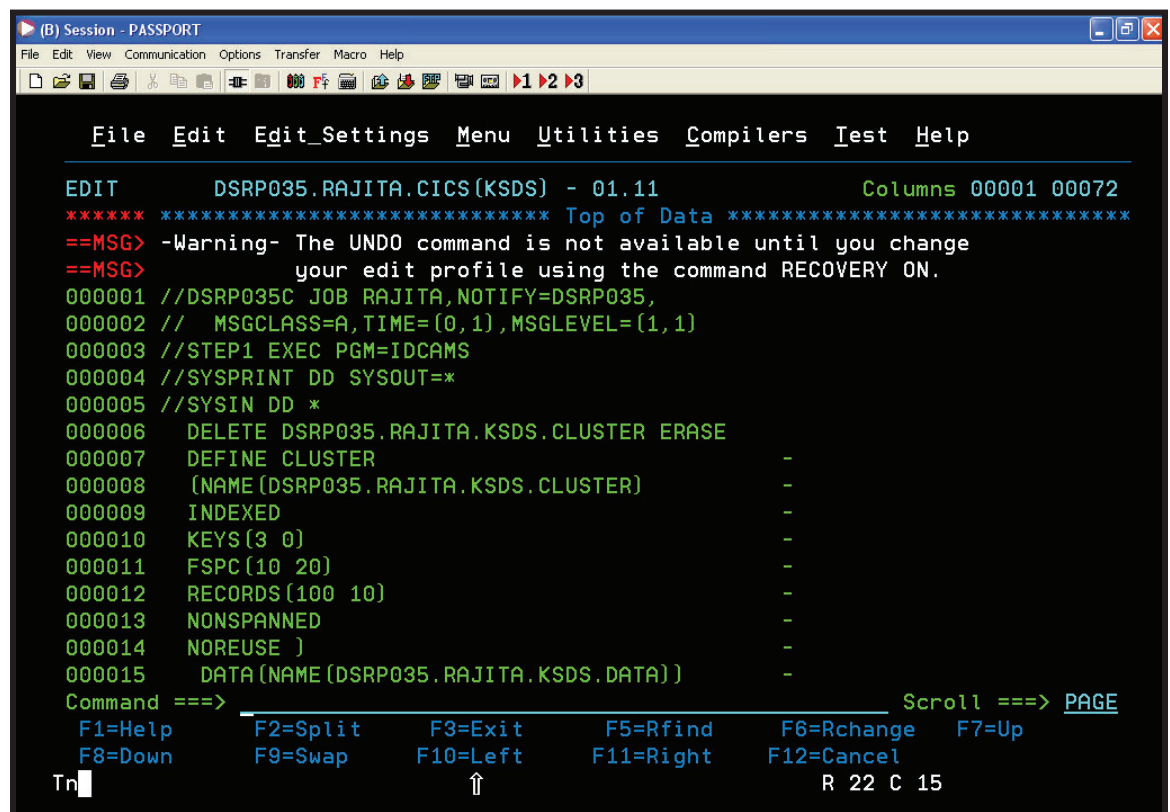
In this exercise, you will try out various IDCAMS commands.

1.1: Create KSDS Cluster

Problem Statement:

The following is the IDCAMS command to create a KSDS cluster.

Solution:



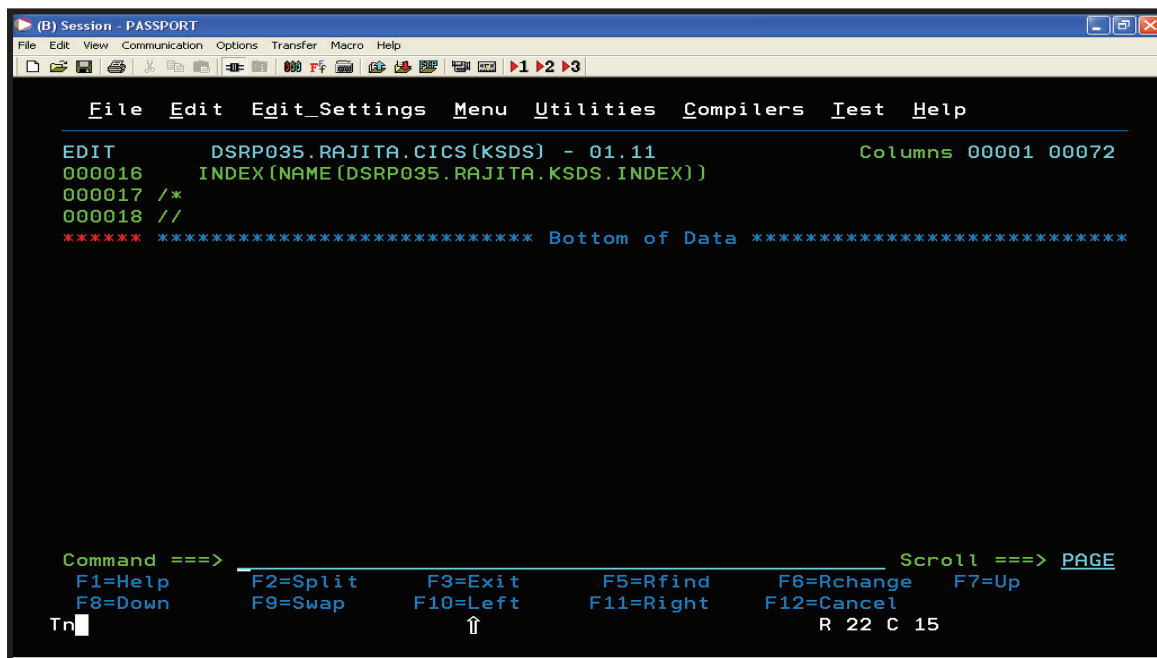
```

(B) Session - PASSPORT
File Edit View Communication Options Transfer Macro Help

File Edit Edit_Settings Menu Utilities Compilers Test Help

EDIT      DSRP035.RAJITA.CICS(KSDS) - 01.11          Columns 00001 00072
***** ***** Top of Data *****
==MSG> -Warning- The UNDO command is not available until you change
==MSG>          your edit profile using the command RECOVERY ON.
000001 //DSRP035C JOB RAJITA,NOTIFY=DSRP035,
000002 //  MSGCLASS=A,TIME=(0,1),MSGLEVEL=(1,1)
000003 //STEP1 EXEC PGM=IDCAMS
000004 //SYSPRINT DD SYSOUT=*
000005 //SYSIN DD *
000006 DELETE DSRP035.RAJITA.KSDS.CLUSTER ERASE
000007 DEFINE CLUSTER
000008 (NAME(DSRP035.RAJITA.KSDS.CLUSTER)
000009 INDEXED
000010 KEYS(3 0)
000011 FSPC(10 20)
000012 RECORDS(100 10)
000013 NONSPANNED
000014 NOREUSE )
000015 DATA(NAME(DSRP035.RAJITA.KSDS.DATA))
Command ==>
F1=Help      F2=Split      F3=Exit      F5=Rfind      F6=Rchange    F7=Up
F8=Down      F9=Swap       F10=Left     F11=Right     F12=Cancel
Tn          ↑
R 22 C 15
  
```

Figure 1: KSDS Source Code



```

(B) Session - PASSPORT
File Edit View Communication Options Transfer Macro Help

File Edit Edit Settings Menu Utilities Compilers Test Help

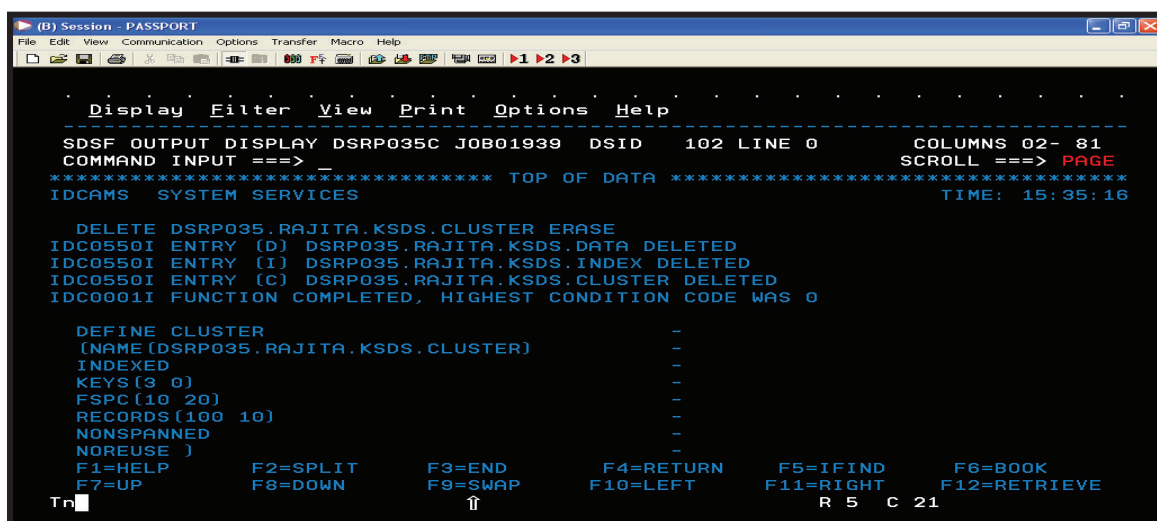
EDIT      DSRP035.RAJITA.CICS(KSDS) - 01.11      Columns 00001 00072
000016    INDEX(NAME(DSRP035.RAJITA.KSDS.INDEX))
000017    /*
000018    //
***** ***** Bottom of Data *****

Command ==>
F1=Help    F2=Split    F3=Exit    F5=Rfind    F6=Rchange    F7=Up
F8=Down    F9=Swap     F10=Left   F11=Right   F12=Cancel
Tn         ↑
R 22 C 15
  
```

Figure 2: KSDS Source Code (contd...)

Step 2: Once you have keyed in the code, check for JCL errors. To do this, type “SUB” .

Step 3: Type “START SD;ST” .The output is displayed as shown below:



```

(B) Session - PASSPORT
File Edit View Communication Options Transfer Macro Help

Display Filter View Print Options Help

SDSF OUTPUT DISPLAY DSRP035C J0801939 DSID 102 LINE 0      COLUMNS 02- 81
COMMAND INPUT ==> _      SCROLL ==> PAGE
***** ***** TOP OF DATA *****
IDCAMS  SYSTEM SERVICES      TIME: 15:35:16

DELETE DSRP035.RAJITA.KSDS.CLUSTER ERASE
IDC0550I ENTRY (D) DSRP035.RAJITA.KSDS.DATA DELETED
IDC0550I ENTRY (I) DSRP035.RAJITA.KSDS.INDEX DELETED
IDC0550I ENTRY (C) DSRP035.RAJITA.KSDS.CLUSTER DELETED
IDC0001I FUNCTION COMPLETED, HIGHEST CONDITION CODE WAS 0

DEFINE CLUSTER
(NAME(DSRP035.RAJITA.KSDS.CLUSTER)
INDEXED
KEYS(3 0)
FSPC(10 20)
RECORDS(100 10)
NONSPANNED
NOREUSE )
F1=HELP    F2=SPLIT    F3=END    F4=RETURN    F5=IFIND    F6=BOOK
F7=UP      F8=DOWN     F9=SWAP   F10=LEFT    F11=RIGHT   F12=RETRIEVE
Tn         ↑
R 5 C 21
  
```


Lab 2. Development Assignments: Set 1

Goals	<ul style="list-style-type: none"> At the end of this lab session, you will be understanding various IDCAMS command along with parameters.
Time	5 Hours

- Define a reusable ESDS cluster. The followings parameters have to be coded:
 - Space allocation has to be in units of RECORDS
 - Control interval size of 4K
 - The records are of fixed length and average and maximum size is 80 bytes
 - Specify **SHR (1 3)**-Multiple read or single write

Specify all the requisite parameters.
- Populate the ESDS cluster via a COBOL program.. The input file EMPLOYEE has the following structure.

```

01 Employee-Record.
   05 Employee-Number          PIC    X(04).
   05 Employee-Name.
      10 Emp-First-Name        PIC    X(15).
      10 Emp-Last-Name         PIC    X(15).
   05 Employee-Dept            PIC    X(04).
   05 Employee-Salary          PIC    9(06)V9(02).

```

- Use LISTCAT command form either the TSO or write a JCL to list the catalog and other dataset information for the ESDS. Thoroughly, analysis the output of the LISTCAT.
- Write a JCL (use the PRINT command) to view the contents of the ESDS cluster.
- Define a non-reusable KSDS cluster. The followings parameters have to be coded:
 - Space allocation has to be in units of RECORDS.
 - Control interval size of 4K.
 - The records are of fixed length and average and maximum size is 80 bytes.
 - Specify **SHR (1 3)**-Multiple read or single write.
 - Specify **KEYS** parameter.

- Specify **FREESPACE** parameter-20% of freespace within control interval and 10% of free control interval within control area.

Specify all the requisite parameters.

6. Populate the above-created KSDS cluster via a COBOL program. Using the same EMPLOYEE file as the input file.
7. Use LISTCAT command from either the TSO or write a JCL to list the catalog and other dataset information for the KSDS. Thoroughly, analysis the output of the LISTCAT.
8. Write a JCL (Use the PRINT command) to view the contents of the KSDS cluster.
9. Define and AIX on the KSDS cluster you have created above.
 - Space allocation has to be in units of RECORDS.
 - Control interval size of 4K.
 - Compute the average and maximum number of records for the RECSZ parameter.
 - Specify **SHR (1 3)**-Multiple read or single write.
 - Specify **KEYS** parameter.
 - Specify **FREESPACE** parameter-20% of freespace within control interval and 10% of free control interval within control area.
 - The AIX is part of the base cluster's UPGRADE set.
 - Alternate index has duplicates.
 - Relate it to the base cluster.

It is recommended that most of the parameters specified for the alternate index is identical to that of the base cluster although it is not mandatory to have the same.

10. On successful creation of the AIX, proceed to create a PATH.
 - Establish the relationship between the PATH and the AIX you have created.
 - Changes made to the base cluster will be reflected automatically in the AIX.
11. Populate the AIX you have created via the BUILDINDEX command.
12. Write a JCL (Use the PRINT command) to view the contents of the base cluster, AIX and the PATH.
13. Write a COBOL program to display all the employees belonging to a [specific](#) department using an alternate index. Use VERIFY command to ensure the data integrity of the dataset.
14. Use an ALTER command to change the name of the base cluster.
15. Use an ALTER command to change the name of the base cluster back to its original name.
16. Use an ALTER command to change the FREESPACE parameter.

17. Write a JCL to create a GDG with the following specifications:
 - Empty the GDG when the limit of five is reached.
 - Scratch the dataset.
18. Use a REPRO command to take the backup of the KSDS cluster.
20. Use a DELETE command to erase all the dataset you have created.

Lab 3. Analysis, Enhancement and Debugging

Assignments: Set 2

Goals	<ul style="list-style-type: none"> At the end of this lab session, you will be understand various IDCAMS command along with parameters.
Time	2 Hr

Problem Statement 1: Refer to VSAM02-MAST-KSDS dataset. The given code list a catalog and verify a data set

```
LISTCAT ENTRY (AJK006) ALL
VERIFY FILE (AJKJCL6)
LISTCAT ENTRY (AJK006) ALL
```

Modify the above code so as to list a catalog and print a data set if the last condition code is 0, but want to list its catalog entry before and after a VERIFY command if the last condition code is greater than 0.

Problem Statement 2: Refer the given code. In this example, an alternate index is defined. An example for DEFINE CLUSTER illustrates the definition of the alternate index's base cluster, EXAMPLE.KSDS2. A subsequent example illustrates the definition of a path, EXAMPLE.PATH, that allows you to process the base cluster's data records using the alternate key to locate them. The alternate index, path, and base cluster are defined in the same catalog, USERRCAT.

```
//DEFAIX1 JOB ...
//STEP1 EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=A
//SYSIN DD \
DEFINE ALTERNATEINDEX -
(NAME(EXAMPLE.AIX) -
RELATE(EXAMPLE.KSDS2) ) -
CATALOG(USERRCAT/USERUPPW)
/
```

Do modify the code as per the requirements given below:

- The alternate key field is the first three bytes of each data record.
- The alternate index's records are variable-length, with an average size of 40 bytes and a maximum size of 50 bytes.
- The alternate index is to reside on volume VSER01.

- 3 cylinders are allocated for the alternate index's space. When the alternate index is extended, it is to be extended in increments of 1 cylinder.
- The alternate key value might be the same for two or more data records in the base cluster.
- The alternate index is to be opened by VSAM and upgraded each time the base cluster is opened for processing.

Problem Statement 3: Refer the given code.

```
//DEFCLU5 JOB ...  
//STEP1 EXEC PGM=IDCAMS  
//SYSPRINT DD SYSOUT=A  
//SYSIN DD \  
DEFINE CLUSTER -  
(NAME(EXAMPLE.ESDS2) -  
RECORDSIZE(2500 3000) -  
VOLUMES(VSER03) -  
CYLINDERS(2 1) -  
FREESPACE(20 10) )  
/
```

Modify the above cluster in such a way that

- Each time the cluster is opened, its high-used RBA can be reset to zero. The cluster is suballocated (that is, it can reside in a VSAM data space with other VSAM objects).
- Data records can cross control interval boundaries.
- The cluster is entry sequenced.

Problem Statement 4:

Refer to the given code file named 3.4.

The referenced source program has to be analyzed and have to be made error free.

All the errors found have to be documented. Write the test cases and test your programs against all valid test cases. During the peer to peer review, document the observation/findings in the program.

Lab 4. Analysis, Enhancement and Debugging Assignments: Set 3

Goals	• Solving Analysis, Enhancement and Debugging Assignments.
Time	4 Hours

4.1: Refer to folder MPST0PG01

The referenced source program has to be analyzed and have to me made error free.

All the errors found have to be documented. Write the test cases and test your programs against all valid test cases. During the peer to peer review, document the observation/findings in the program.

4.2: Refer to folder MPST0PG02

The referenced source program has to be analyzed and have to me made error free.

All the errors found have to be documented. Write the test cases and test your programs against all valid test cases. During the peer to peer review, document the observation/findings in the program.

Appendices

Appendix A: Table of Figures

Figure 1: KSDS Source Code.....	6
Figure 2: KSDS Source Code (contd....)	7