UNIT 1

Introduction to VSAM

Introduction to VSAM

- Objectives
- VSAM overview
- What is VSAM?
- VSAM terminology
- Data set types
- Schematic representation Data Set Types

Objectives

- In this chapter we will build the foundation for the rest of the book.
- The concepts and terminology associated with VSAM data sets are reviewed.
- How a VSAM data set is different from other data set types.
- The various types of VSAM data sets, and what makes them unique, are discussed, along with an explanation of how data is stored and accessed in a VSAM data set.

VSAM Overview

- In the early 1970s, IBM introduced a collection of three data set organizations sequential, indexed, and direct-access, together with the access methods and utilities to be used on the mainframe operating systems.

- This collection of data set organizations is called the Virtual Storage Access Method (VSAM).

- The word *virtual* relates to the fact that VSAM was introduced at approximately the same time as the initial IBM virtual storage operating systems OS/VS1 and OS/VS2.

- VSAM was developed to replace the Indexed Sequential Access Method (ISAM), which is a much older technology. ISAM has major processing overheads, which IBM wanted to improve.

What is VSAM?

- VSAM is one of several access methods that define the technique by which data is stored and retrieved. It is a GET/PUT interface used to transfer data from a direct access storage device (DASD) to an application program.

- VSAM does not support data stored on tape. VSAM is used to organize and access data, and maintain information about this data that is stored or referenced in a catalog.

- VSAM data sets must be cataloged in an integrated catalog facility (ICF) structure. Records are arranged by an index key or by relative byte addressing.

- VSAM uses direct or sequential processing of fixed and variable length records on DASD.

- There are two major parts to VSAM: catalog management and record management.

Catalog Management

- The catalog, which contains information about the data sets, can be an ICF or a VSAM catalog. DFSMS/MVS deals only with ICF catalogs.

- All VSAM data sets must be defined in an ICF catalog. In this book, we will not go into detail on catalog management.

Record management

The purpose of record management is to maintain records in a VSAM data set for an application or a system program. Today, VSAM supports five data set organizations:

- Key-sequenced data set (KSDS)
- Entry-sequenced data set (ESDS)
- Fixed-length relative record data set (RRDS)
- Variable-length relative record data set (VRRDS)
- Linear data set (LDS)

The primary difference between the VSAM data set organizations is the way in which their records are stored and accessed.

VSAM Terminology

Before we discuss VSAM data set organizations in detail, we need to review some terms that will be used throughout the book.

- ❑ Logical record
- ❑ Control interval
- ❑ Control area

VSAM Terminology (Continued)

Logical record

- A logical record is a unit of information used to store data in a VSAM data set. The terms ***logical record*** and ***record*** are used interchangeably in this book.

- Logical records in VSAM data sets are stored differently than logical records in non-VSAM data sets.

Control Interval

- Logical records are contained in a control interval (CI).

- The fundamental building block of every component of a VSAM data set is the control interval. A control interval is the unit of information that VSAM transfers between the storage device and the processor. One CI can be made of one or more physical blocks of DASD.

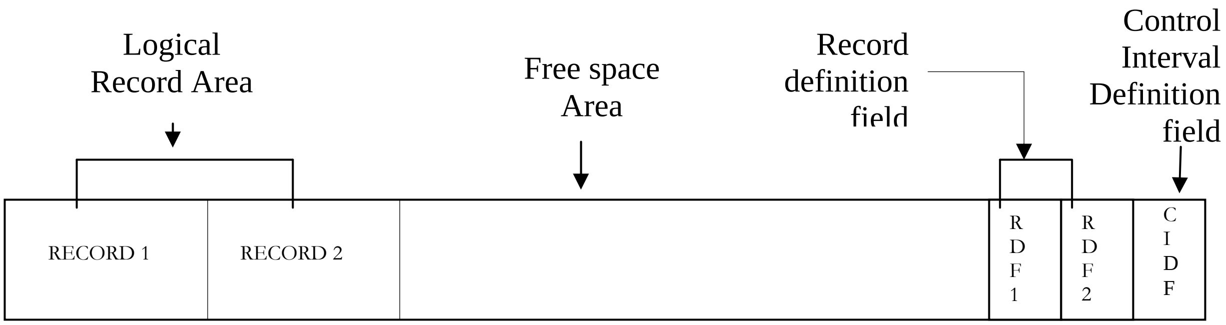


Figure 1- 1

A CI consists (Figure 1-1)

- ***Logical records*** stored from beginning to end
- ***Unused space***, referred to as free space, for data records to be inserted into or lengthened
- The maximum size of a CI is 32k and the minimum size is 512 bytes.
- ***Control information***, which is made up of two types of fields; one control interval definition field (CIDF) per CI, and one or more record definition fields (RDF) per logical record.
  - ***CIDF*** is a 4-byte field. It contains information about the length of data in the CI and the amount and location of free space.
  - ***RDF*** is a 3-byte field. It describes the length of records and how many adjacent records are of the same length.

VSAM Terminology (Continued)

Control area

- A control area (CA) is two or more CIs put together into fixed-length contiguous areas of DASD. A VSAM data set is composed of one or more CAs.

- The number of CIs in a CA is fixed by VSAM. The CA size is implicitly defined when you specify the size of a data set at data set definition.

- The maximum size of a CA is one cylinder and the minimum size is one track.

- Figure 1-2 shows the structure of a CA.

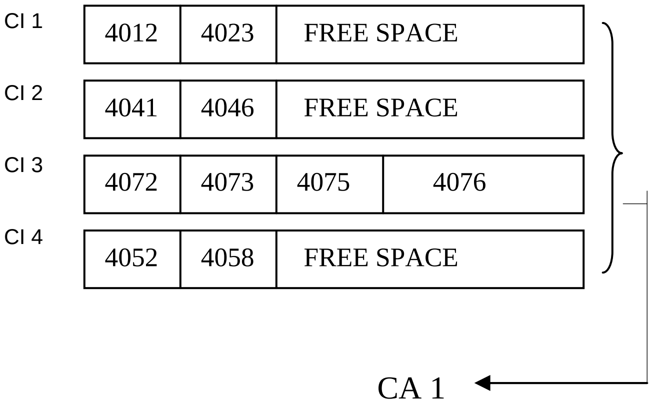


Figure 1-2

Data set Types

VSAM supports five data set organizations:

- Entry-sequenced.
- Key-sequenced.
- Fixed-length.
- Variable-length relative record.
- Linear.

Entry sequenced data set (ESDS)

- An ESDS is comparable to a sequential non-VSAM data set in the sense that key field in the logical record sequences records by the order of their entry in the data set, rather than. This could be fixed or variable length record.

- All new records are placed at the end of the data set. Existing records can never be deleted. If the application wants to delete a record, it must flag that record as inactive. As far as VSAM is concerned, the record is not deleted. It is the responsibility of the application program to identify that record as invalid.

- Records can be updated, but without length change. To change the length of a record, you must either store it at the end of the data set as a new record, or override an existing record of the same length that you have flagged as inactive.

A record can be accessed sequentially or directly by its RBA:

- *Sequential processing*: VSAM automatically retrieves records in stored sequence. Sequential processing can be started from the beginning or somewhere in the middle of a data set. If processing is to begin in the middle of a data set, positioning is necessary before sequential processing can be performed.

- *Direct processing*: When a record is loaded or added, VSAM indicates its RBA. To retrieve records directly, you must supply the RBA for the record as a search argument. Although an ESDS does not contain an index component, you can build an alternate index to keep track of these RBAs.

Empty spaces in the CI are referred to as unused space because they can never be used. This is a result of CI internal fragmentation (spanned is only for logical records greater than the CI).

You specify ESDS organization using the IDCAMS DEFINE command and specifying the NONINDEXED parameter.

Keyed sequenced data set (KSDS)

- In a KSDS, records are placed in the data set in ascending collating sequence by key. The key contains a unique value that determines the record's collating position in the data set. The key must be in the same position in each record.

- The key data must be contiguous and each record's key must be unique. After it is specified, the value of the key cannot be altered, but the entire record can be deleted. When a new record is added to the data set, it is inserted in its collating sequence by key. This could be fixed or variable length record.

There are three methods by which to access a KSDS. These are sequential, direct, or skip-sequential.

- *Sequential access* is used to load a KSDS, and to retrieve, update, add and delete records in an existing data set. VSAM uses the index to access data records in ascending or descending sequence by key. When retrieving records, you do not need to specify key values because VSAM automatically obtains the next logical record in sequence. The sequence set is used to find the next logical CI. Sequential access allows you to avoid searching the index more than once. Sequential is faster than direct for accessing multiple data records in ascending key order.

- *Direct access* is used to retrieve, update, add and delete records in an existing data set. You need to supply a key value for each record to be processed. You can supply the full key or a generic key. The generic key is the high order portion of a full key. For example, you might want to retrieve all records whose keys begin with XY (where XY is the generic key), regardless of the full key value. VSAM searches the index from the highest-level index set CI to the sequence set for a record to be accessed. Vertical pointers in the sequence set CI are used to access the data CA containing the record. Direct access saves you a lot of overhead by not retrieving the entire data set sequentially to process a small percentage of the total number of records.

- *Skip-sequential access* is used to retrieve, update, add and delete records in an existing data set. VSAM retrieves selected records, but in ascending sequence of key values.

You specify the KSDS organization using the IDCAMS DEFINE command with the INDEXED parameter.

Relative record data set (RRDS)

- An RRDS consists of a number of pre-formatted fixed-length slots. Each slot has a unique relative record number, and ascending relative record number sequences the slots.

- Each fixed length logical record occupies a slot, and is stored and retrieved by the relative record number of that slot. The position of a data record is fixed and its relative record number cannot change.

- Because the slot can either contain data or be empty, a data record can be inserted or deleted without affecting the position of other data records in the RRDS. The RDF shows whether the slot is occupied or empty. Free space is not provided because the entire data set is divided into fixed-length slots.

Typical RRDS processing

The application program inputs the relative record number of the target record and VSAM is able to find its location quickly using a formula that takes into consideration the geometry of the DASD device. The relative record number is always used as a search argument. An RRDS can be processed sequentially, directly or skip-sequentially.

- RRDS sequential processing is treated the same way as ESDS sequential processing. Empty slots are automatically skipped by VSAM.

- An RRDS can be processed directly by supplying the relative record number as a key. VSAM calculates the RBA and accesses the appropriate record or slot. RRDS direct address processing by supplying the RBA is not supported.

- Skip-sequential processing is treated like an RRDS direct processing request, but the position is maintained. Records must be in ascending sequence.

You specify the RRDS organization using the IDCAMS DEFINE command with the NUMBERED option.

Variable relative record data set (VRRDS)

- A VRRDS is similar to a fixed-length RRDS, except that it contains variable-length records. Each record has a unique relative record number, and is placed in ascending relative record number order. Each record is stored and retrieved using its relative record number. VRRDS has no slots.

- The relative record number of a record cannot change. When that record is erased, the relative record number can be reused for a new record.

- You can specify free space for inserting records and increasing the length of a record. VRRDS is a KSDS processed as an RRDS, so an index will be created.

- You specify the VRRDS organization with the IDCAMS DEFINE command with the NUMBERED option and variable length record.

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 control interval size of 4096 bytes.

- An LDS has no imbedded control information in its CI, which is, no RDFs and CIDFs. All LDS bytes are data bytes. Logical records must be blocked and deblocked by the application program, but records do not exist from the point of view of VSAM. Like the ESDS and RRDS, an LDS contains a data component only.

- An LDS can only be defined using ICF catalogs. IDCAMS is used to define an LDS but it is accessed using a Data-In-Virtual (DIV) macro. An LDS is sometimes referred to as a DIV object.

- You specify the LDS organization with the IDCAMS DEFINE command specifying the LINEAR parameter.

Data-in-Virtual (DIV) is an optional and unique buffering technique used for LDS data sets. Application programs can use DIV to “map” an LDS data set or a portion of a data set into an address space, a data space, or a hyperspace. Data is read into central storage through the paging algorithms only when that data block is actually referenced. During RSM page steal processing, only changed pages are written to auxiliary storage. Unchanged pages are discarded since they can be retrieved again from the permanent data set. DIV is designed to improve the performance of applications that process large files non-sequentially in an unpredictable pattern. It reduces the number of I/O operations that are traditionally associated with data retrieval. Likely candidates are large arrays or table files.

Schematic representation of Data Set Types

LDS

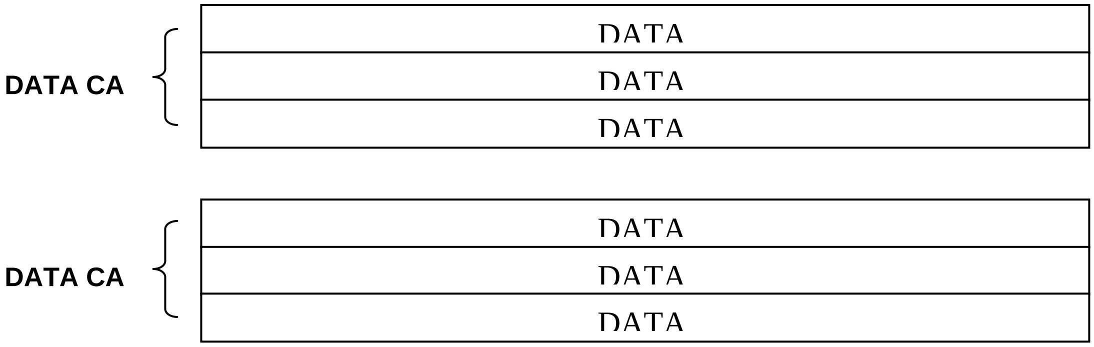


Figure: 1- 3

RRDS

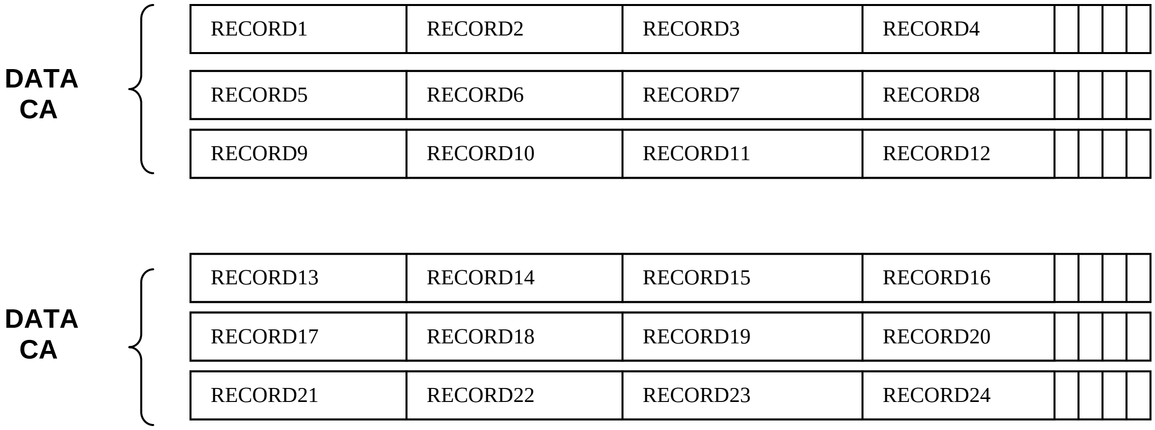


Figure: 1- 4

Schematic representation of Data Set Types (Continued)

ESDS

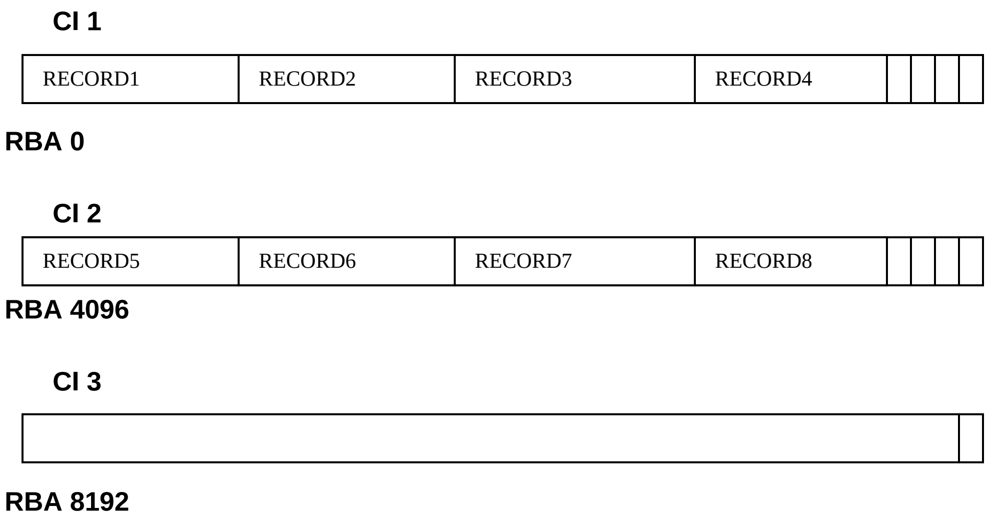


Figure: 1- 5

KSDS

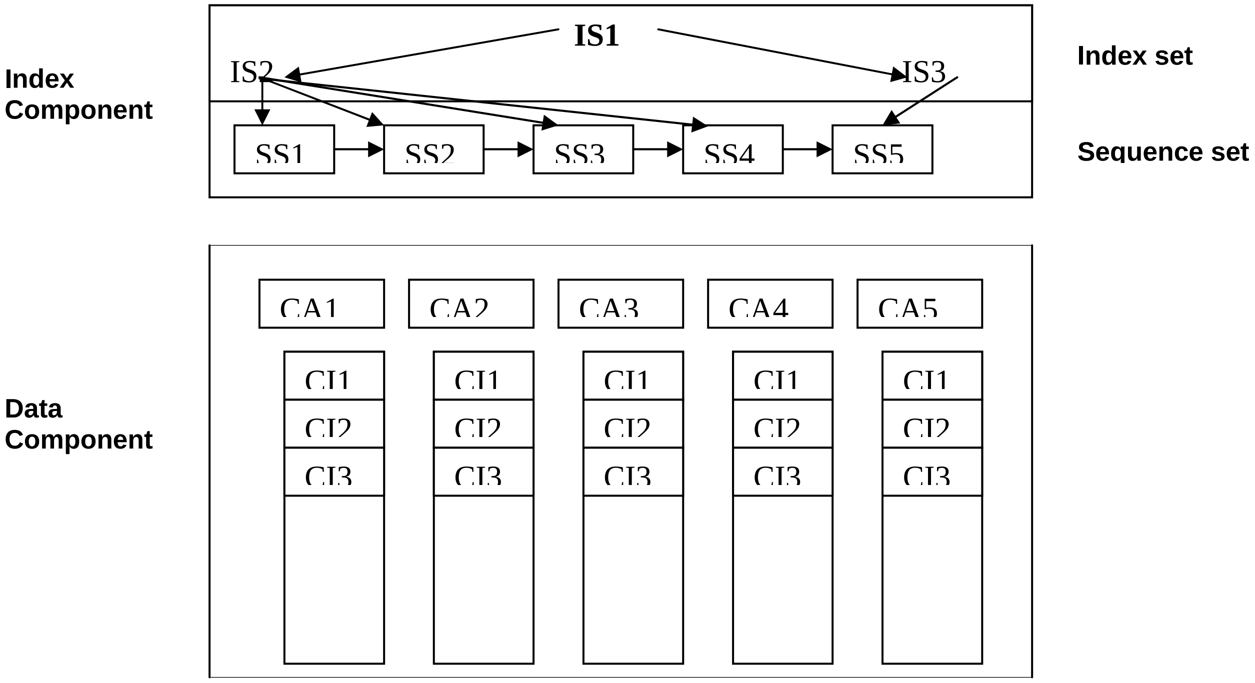
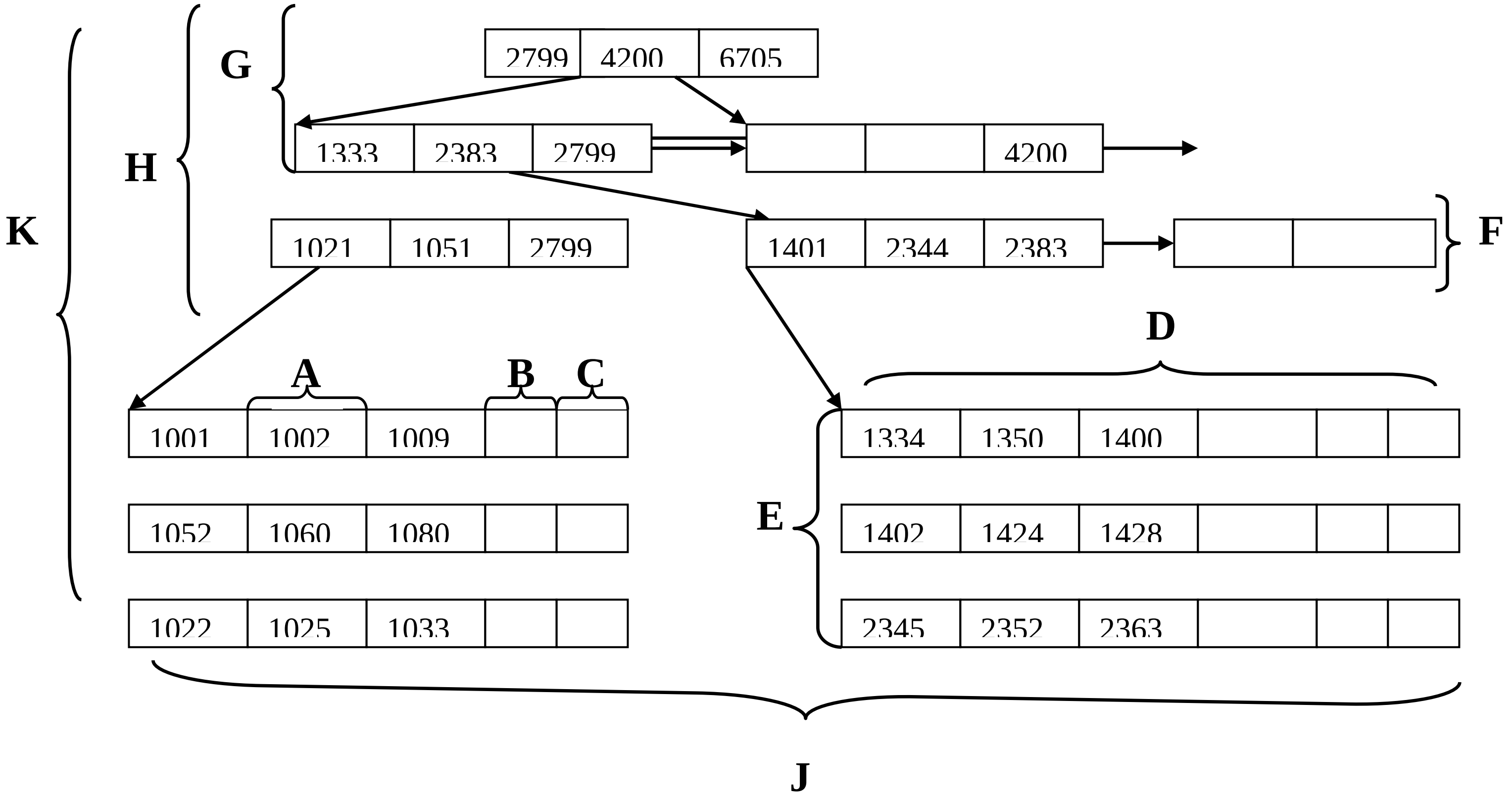


Figure: 1- 6

Unit 1 Exercises

1. Data Structure Review



1. __________ Control Interval (CI)
2. __________ Free Space or Unused Bytes.
3. __________ Logical Record
4. __________ Control Area (CA)
5. __________ Sequence Set
6. __________ Index Set
7. __________ RDF/CIDF
8. __________ Index Component
9. __________ Data Component
10. __________ Cluster.

2. Circle One

1. Which VSAM cluster support only fixed length records?
   A. LDS
   B. RRDS
   C. ESDS
   D. KSDS

2. VSAM control intervals containing data records have at least one RDF and one CIDF.
   A. TRUE
   B. FALSE

3. FREESPACE can be specified for which type of cluster?
   A. LDS
   B. RRDS
   C. ESDS
   D. KRDS

4. In which cluster type are records always added at the end of the data set?
   A. LDS
   B. RRDS
   C. ESDS
   D. KSDS

5. For a KSDS, the larger the data CA size, the smaller the number of index CIs.
   A. TRUE
   B. FALSE.