DFSMS Basics: Data Set Fundamentals Get to Know Your Data Sets!

Neal Bohling and Tom Reed
DFSMS Defect Support @ IBM

August 7, 2014
Session Number 16119

#SHAREorg

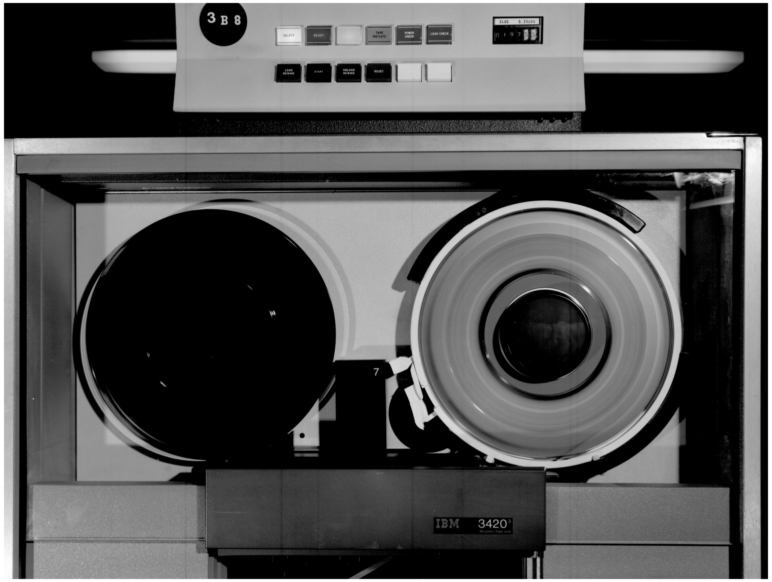
SHARE in Pittsburgh 2014

What's your experience level?

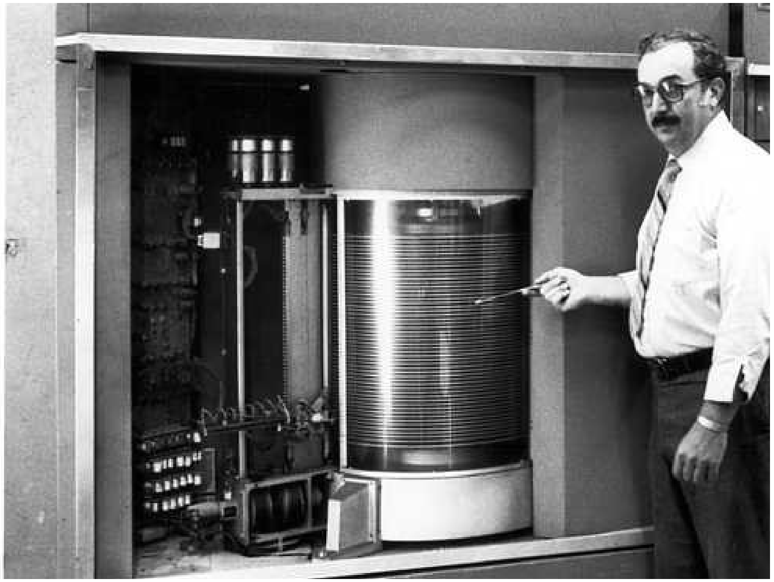
IBM 7094
1965
~500KHz

Background

To fully understand z/OS Data Sets,
let's look at how data is stored



Tape
(Sequential)



Disk
(Direct)

DASD Structure

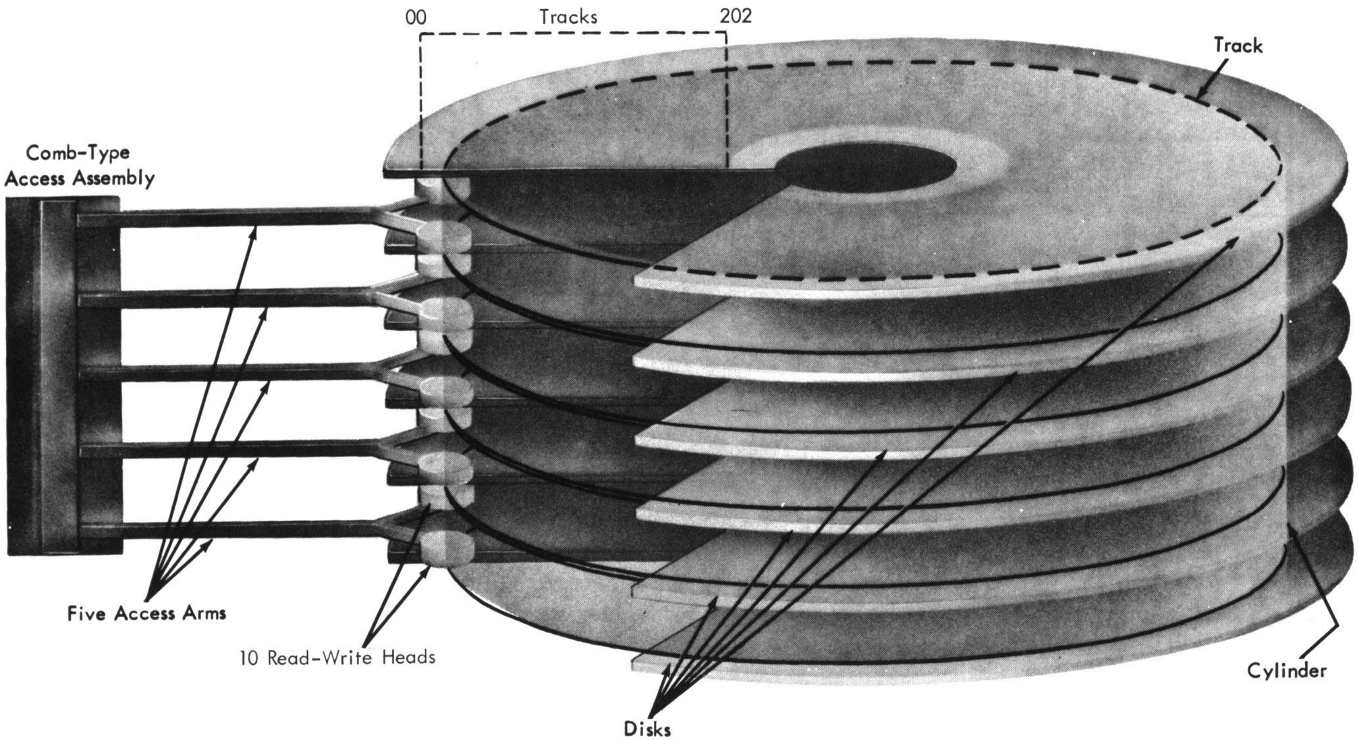


Figure 7. 2311 access mechanism

Modern Devices are Modeled after this architecture:
1 Track = 56664 Bytes
1 Cylinder = 15 Tracks

Data Sets

- Volumes provide a stream of data.

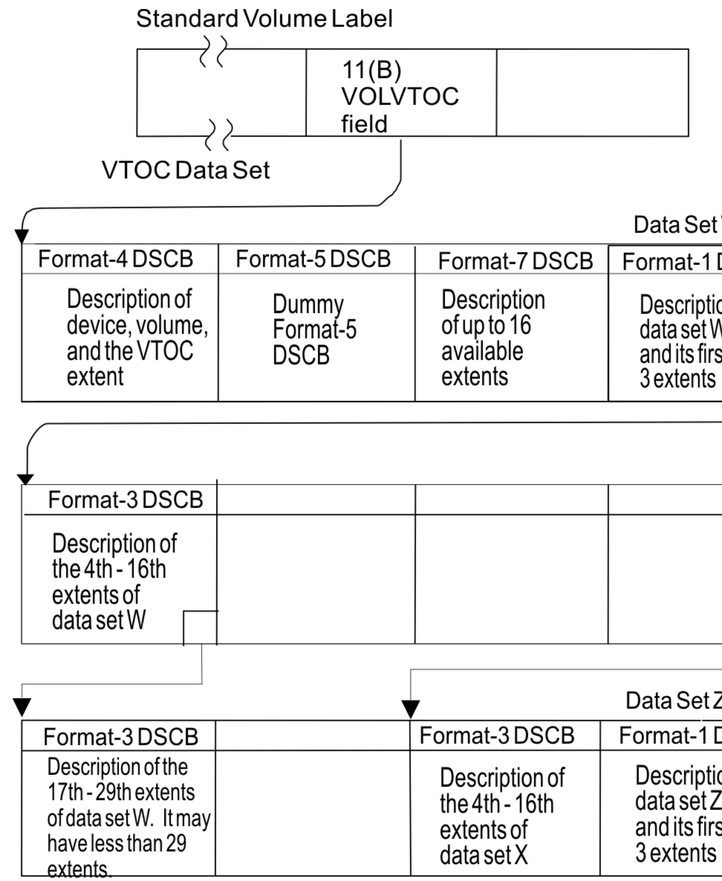
```
110010011100001011010100110010011110001011100011110010001100010111000010110001011110001011100011110000111101011011010100
101011111100100111000111100100111010101110001111100001111010110110101001101011111000001110101011110100011001001110000101
010100110010011110001011100011110010001100010111000010110001011110001011100011110000111101011011010100110101111110010011
0001111001001110101011100011111000011110101101101010011010111111000001110101011110100011001001110000101101010011001001111
0010111000111100100011000101110000101100010111100010111000111100001111010110110101001101011111110010011100011110010011101
101110001111100001111010110110101001101011111100000111010101111010001100100111000010110101001100100111100010111000111100 1
001100010111000010110001011110001011100011110000111101011011010100110101111110010011100011110010011101010111000111110000
1110101101101010011010111111000001110101011110100011001001110000101101010011001001111000101110001111001000110001011100001
```

- We can reference by various methods..
  - DASD by CCHHRR – Cylinder, Head, Record
  - Relative Track (converts to Cyl, Head)
  - Relative Byte (which converts to CCHHRR)

- But where do files begin and end?

Volume Table of Contents

- A **Volume** is a logically defined disk or tape
  - Can be a real disk
  - Usually is a virtual disk
- **Volume Label** (at Cyl0,Trk0)
  - Points to VTOC
- **Volume Table of Contents** (VTOC)
  - Comprised of DSCBs
    - 10 types – Format 0-9
    - For more info, check out DFSMSdfp Advanced Services: https://ibm.biz/BdF49T
  - Maps out the entire drive
    - Drive info (FMT4)
    - Free space (FMT5/7)
    - Data sets (FMT1/8/3)
- Now we can find any data set on the volume



Data Set Names

SYS1.PROCLIB.#BACKUP1

HLQ

2nd Level Qualifier

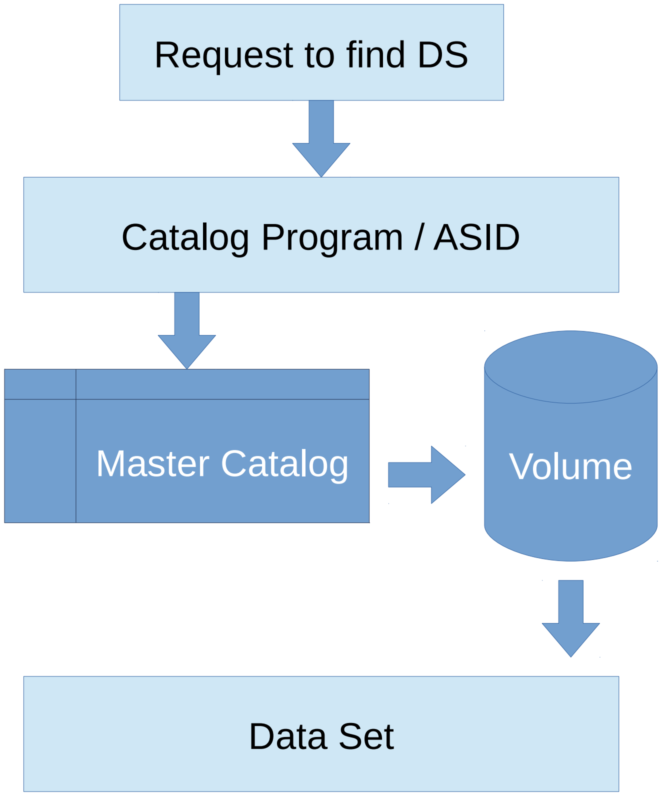
LLQ

- Rules:
  - Up to 22 qualifiers, each at least 1 character long
  - Qualifiers separated by period (.)
  - Up to 44 total characters long
  - First character must be A-Z or National (#,$,@)
  - Remaining can be alphanumeric, national, or hyphens

Catalog

- A catalog is a data set that keeps track of other data sets
- Ties DSN to Volume
- Managed by the CATALOG address space
- Includes MASTER and USER catalogs

- Now we can find any data set in the system
- With shared catalogs, any data set in the plex



Data Set Organization

Data sets are still just streams of bytes with no structure.

Finding information within that stream is difficult.

Solution: Data Set Organization

Defines how the data set is structured internally..

Two Main Types, with sub-types

- **Non-VSAM**
  - Direct
  - Sequential
  - Partitioned
- **VSAM**
  - KSDS
  - ESDS
  - Linear
  - RRDS

Blocking

- Data Streams are logically divided into BLOCKS which are further divided into RECORDS
- This is to reduce the number of I/Os

Data Stream on a disk (1 track)			
Block 1		Block 2	
Record 1	Record 2	Record 3	Record 4

- Track Length: set by device (3390 is 56664 bytes/track)
- Block Size (BLKSIZE): Set by user or calculated automatically
- Record Size (LRECL): Set by the user

Non-VSAM

Direct and Sequential

Direct Organization

- Blocks are arranged by their control number
- No records, blocks are organized by the application
- Accessed via the BDAM access method
- Works like a hashtable – space is divided into even blocks
- Because not every entry may be used, some space may be wasted.
- Reads and writes are for whole blocks at a time

CONTROL	BLOCK [BLKSIZE]
1	DATASTRING ONE,
2	[empty]
3	IN A GALAXY FAR, FAR AWAY
4	42

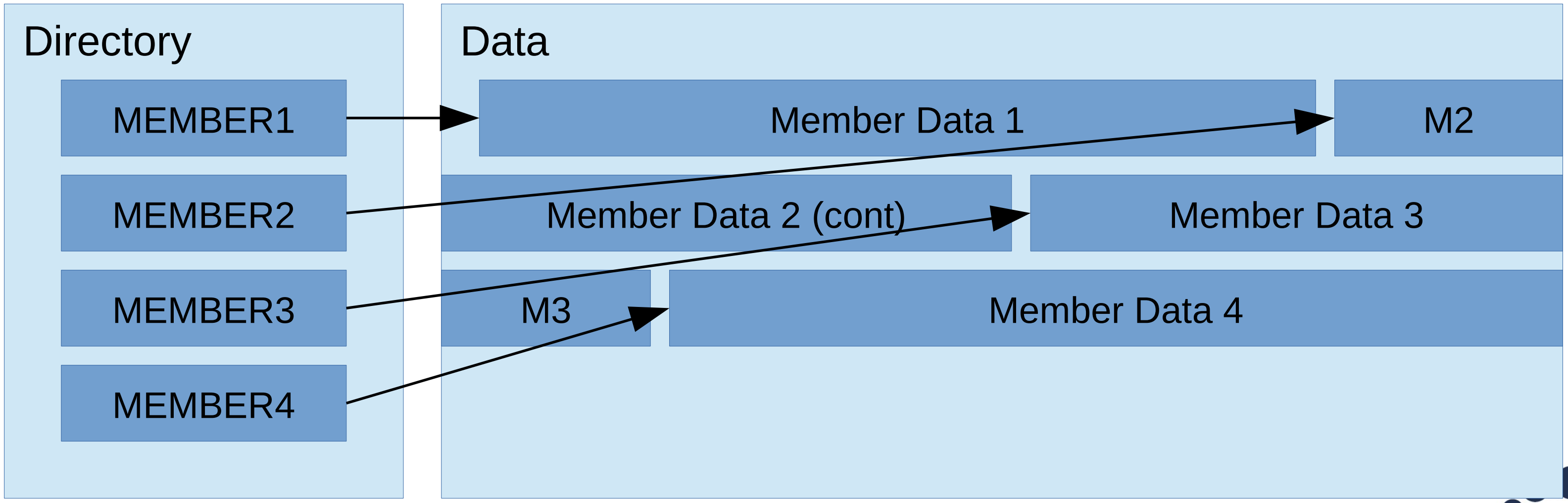
Sequential Organization

- One of the most common organizations you'll see
- Data is split into blocks, which are split into records
- Records are arranged in the order they are written
- To add new records, you either:
  - Rewrite the whole file
  - Add to the end
- Accessed via the BSAM or QSAM access method
- Also comes in LARGE and EXTENDED formats

Block 1	Block 2	Block 3	Block 4

Partitioned Data Sets (PDS)

- Data is divided up into members
- Members are stored sequentially
- Members have unique names (1-8 characters)
- Directory Entries at the beginning of the data set links member names to data locations



PDS Limitations

- Limited to one volume
- Prone to fragmentation:
  - When a member is deleted, the directory entry is removed, but the space remains unused leading to fragmentation
  - Adding records to a member remove the old member and re-writes it at the end
  - Adding new members add to the end
- Eventually, a REORG is needed to rebuild the data set and reclaim space
- Sharing can cause problems:
  - Only one user can update at a time, but not enforced

PDS Extended (PDSE)

- **Works relatively interchangeably with PDS**
- Internal structure is different
- Advantages:
  - Can reuse space (no more fragmentation)
  - Can extend as needed (still limited to one volume)
  - Directory is indexed, lowering seek time
  - Members can be shared
- Can store program objects or data, but not both
- Comes in Version1 and Version2
- For more information, see Tom's presentations:
  - 16126: PDSE Best Practices (Monday)
  - 16127: PDSE Version 2 Member Generations (Wed)

Generation Data Groups

- Not a data set organization, but a catalog construct
- Groups of data sets organized by number (G0000V00)
- Allows easy tracking of multiple generations/version
- Must be non-VSAM, and must be CATALOGed

GDG: MY.DATA
Current entry (4)

MY.DATA(+1)

MY.DATA(-2)

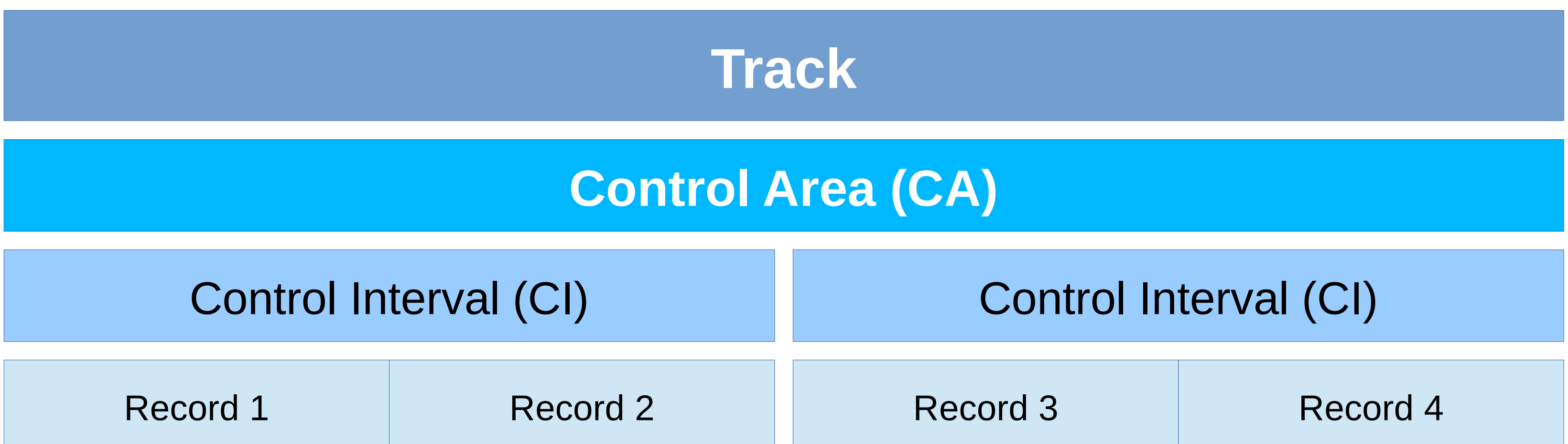
MY.DATA.G0005V00
MY.DATA.G0004V00
MY.DATA.G0003V00
MY.DATA.G0002V00
MY.DATA.G0001V00
MY.DATA.G0000V00

VSAM

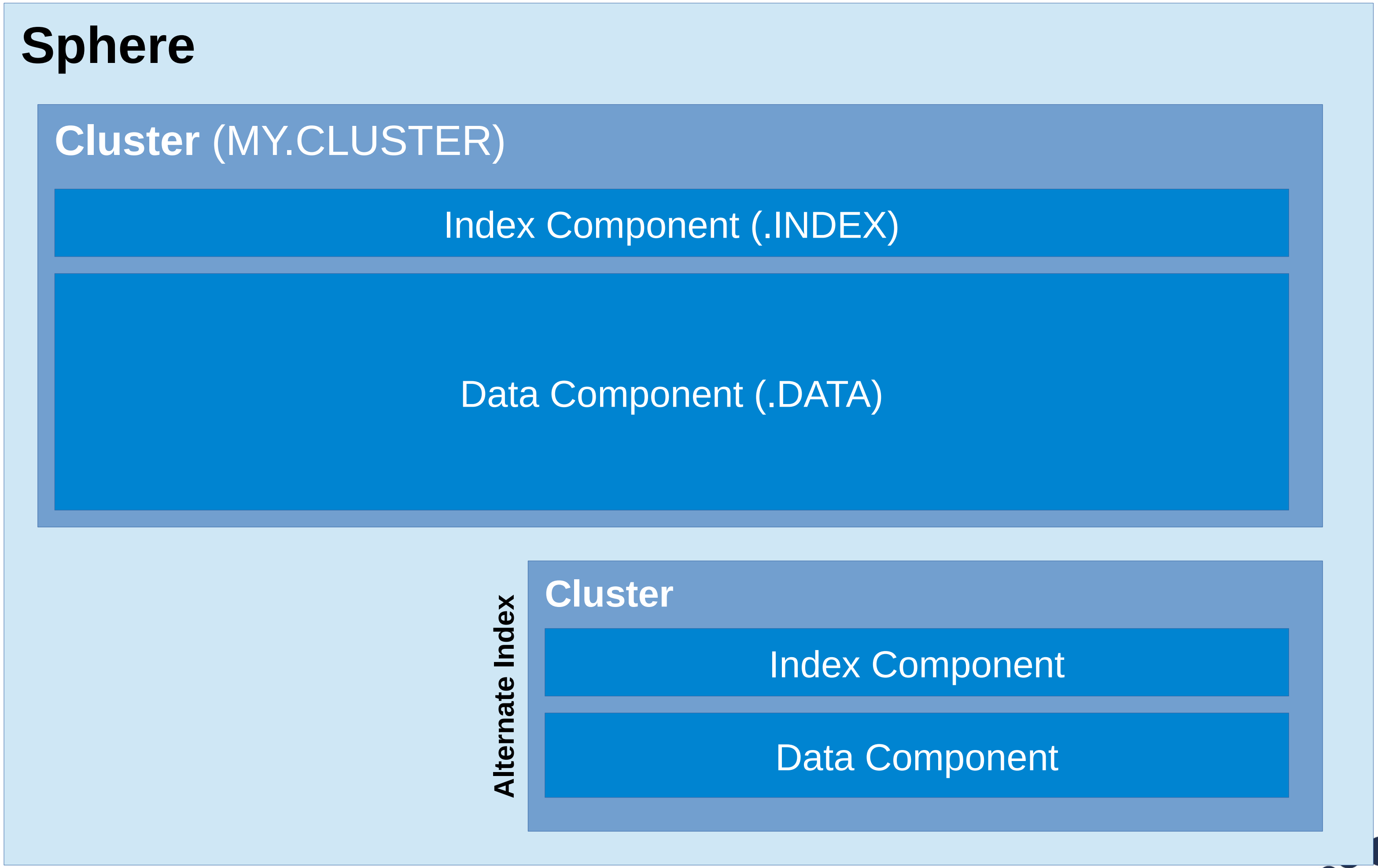
Virtual Sequential Access Method

Virtual Sequential Access Method (VSAM)

- Four sub-types:
  - Key-Sequenced Data Sets (**KSDS**)
  - Entry-Sequenced Data Sets (**ESDS**)
  - Relative Record Data Sets (**RRDS**)
  - Linear Data Sets (**LDS**)
- Instead of blocks, VSAM uses Control Intervals and Areas

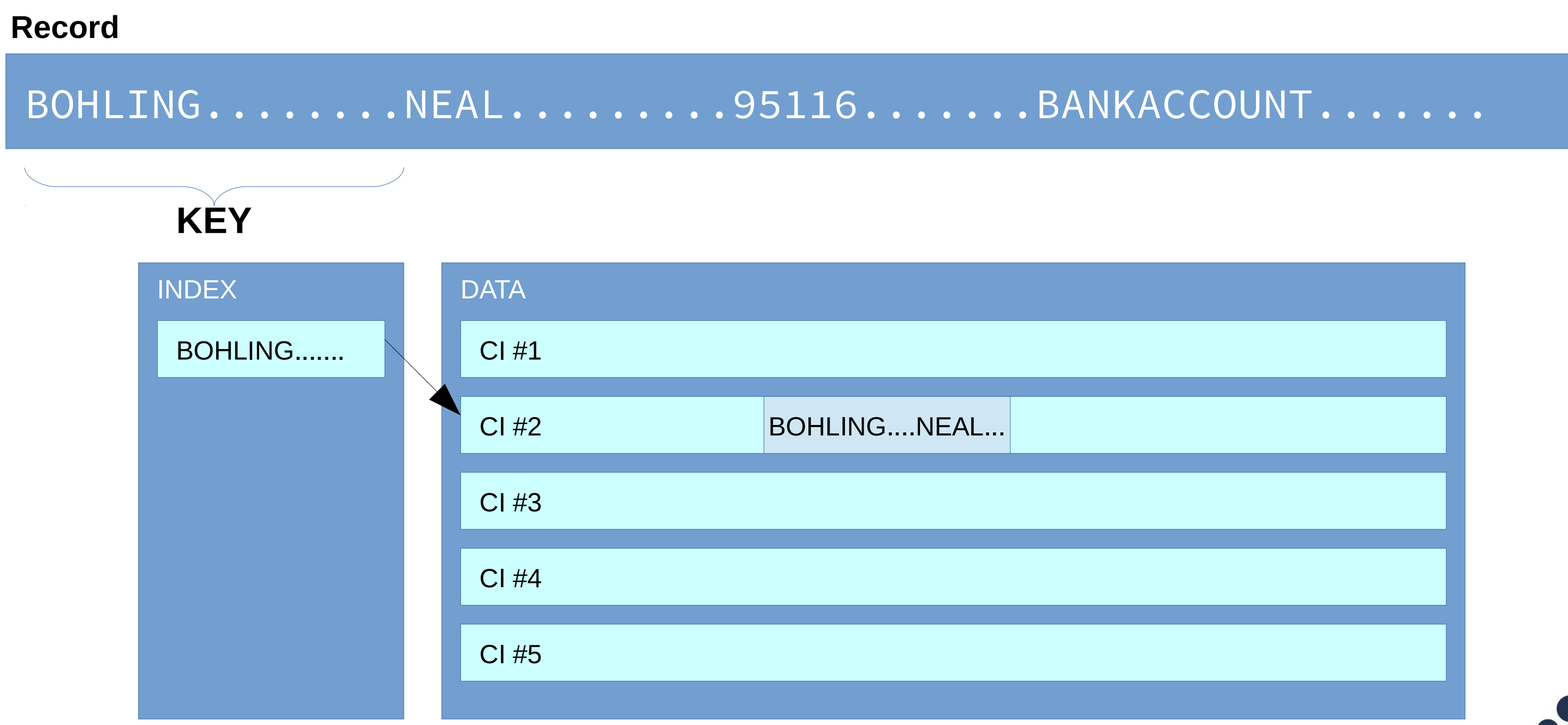


VSAM Terms

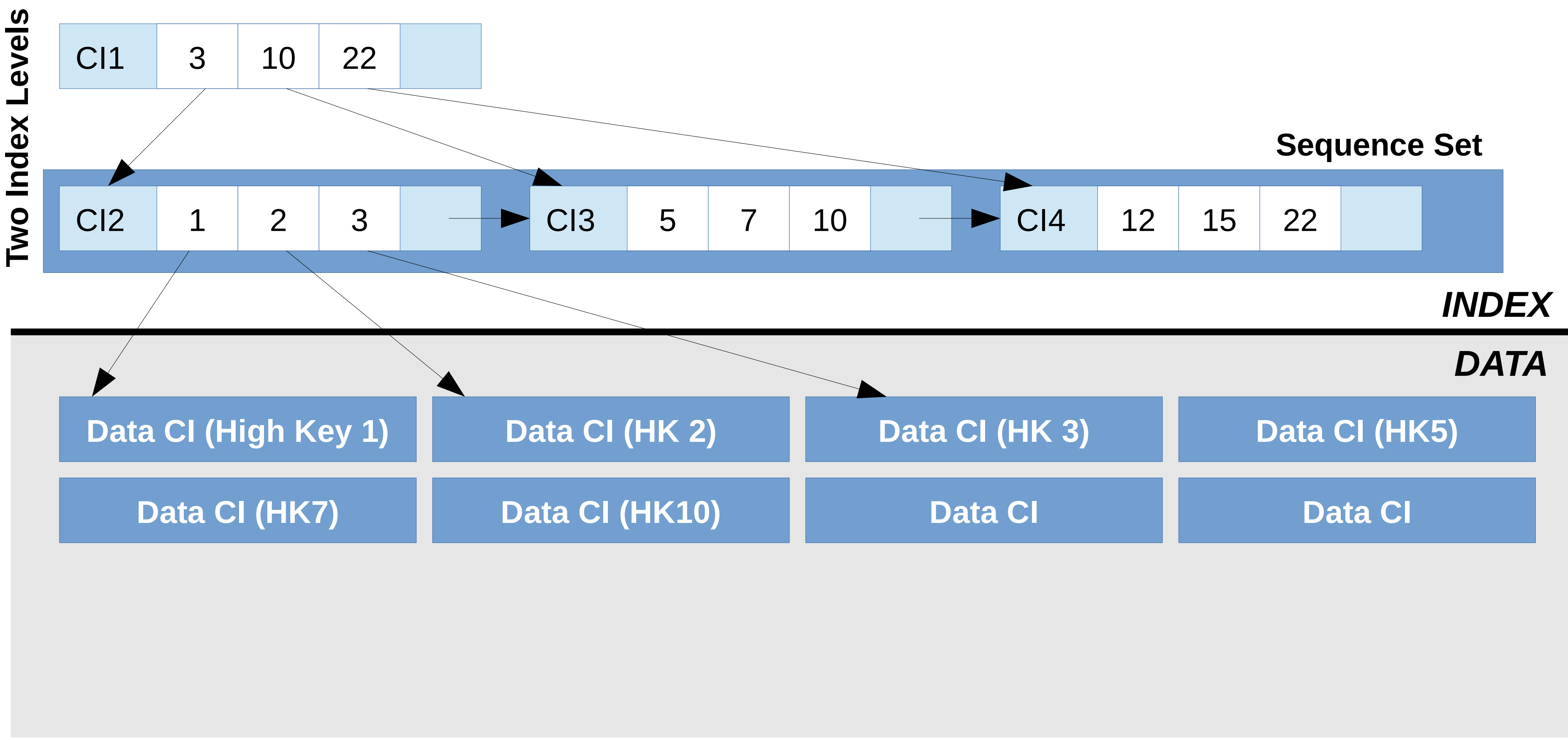

Sphere
Cluster (MY.CLUSTER)
Index Component (.INDEX)
Data Component (.DATA)
Alternate Index
Cluster
Index Component
Data Component

Key Sequenced Data Set

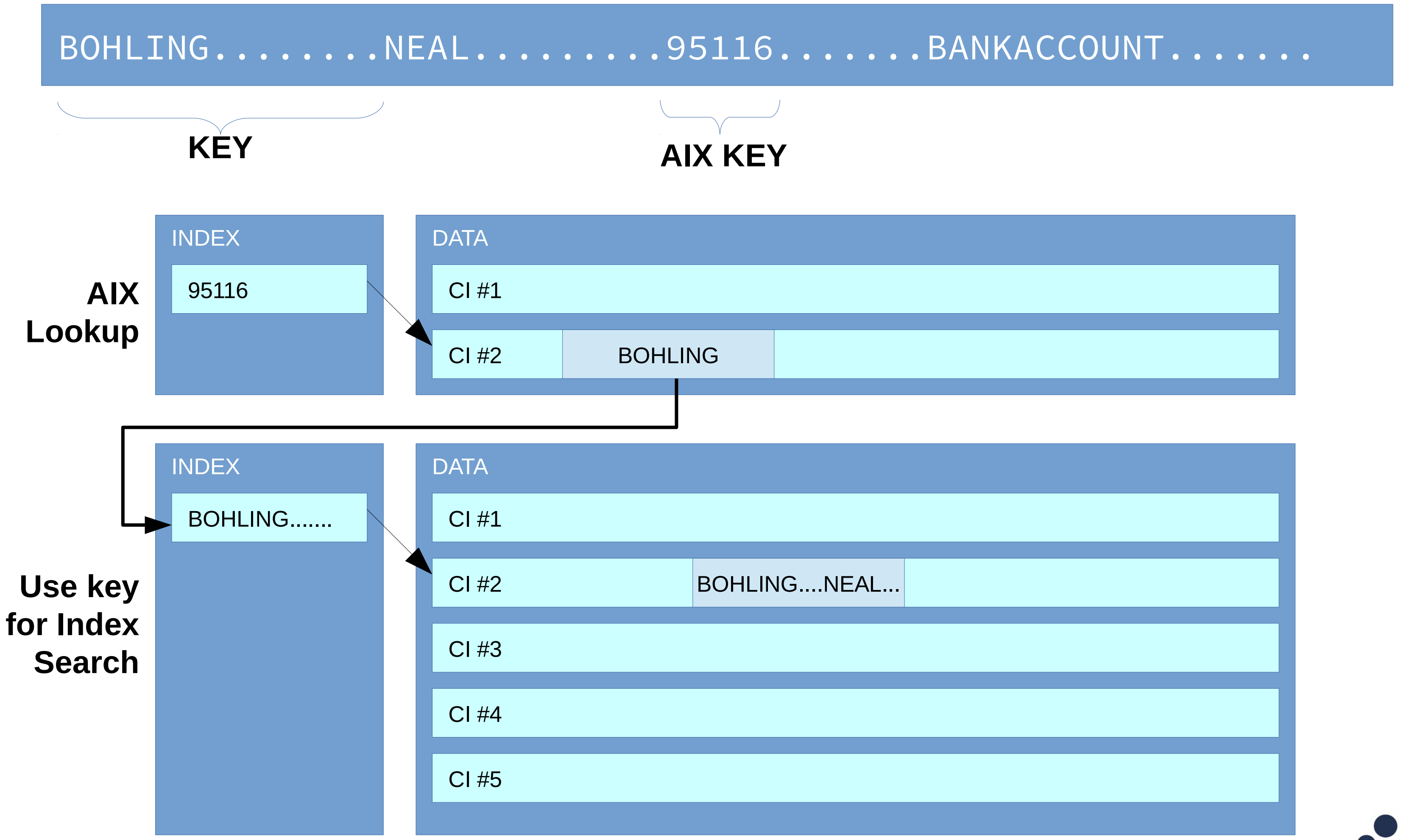
- Records contain a KEY and DATA
- That KEY is used to create an index
- The index provides direct access to any record



KSDS Index Structure

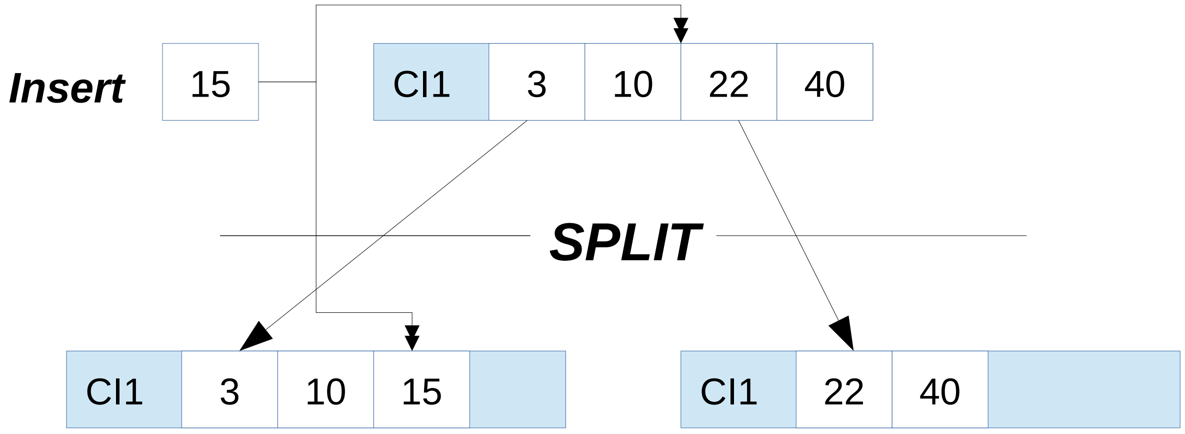


Alternate Indexes



VSAM Splits

- Split is when an INSERT won't fit in a CI
- About half of the data is moved to a new CI

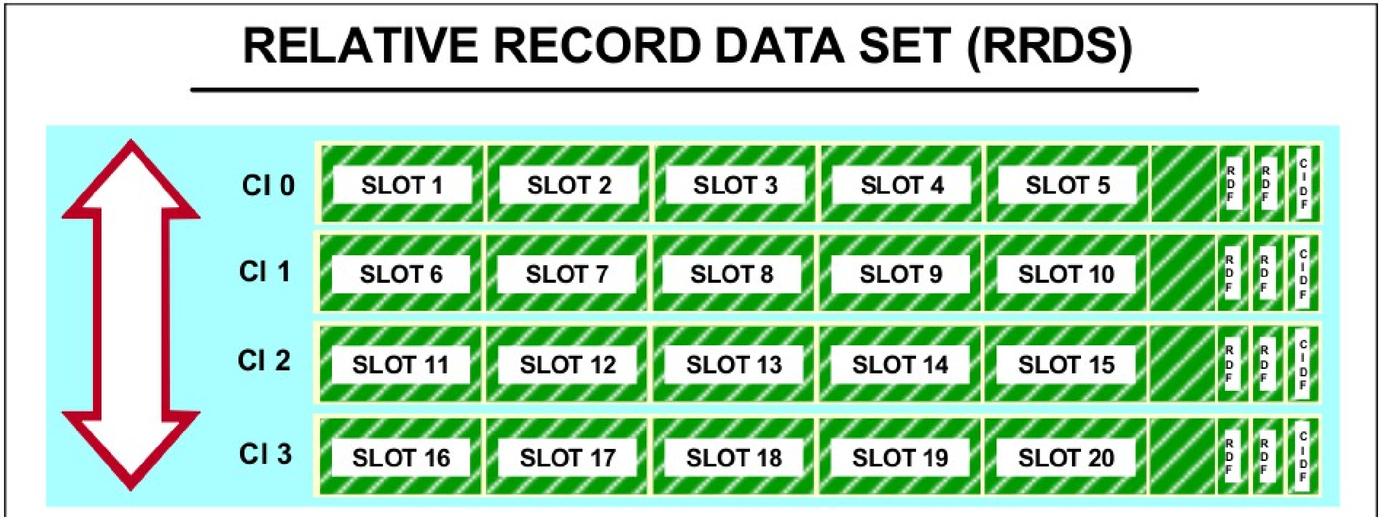


Entry Sequenced Data Set

- No INDEX
- Records in order they were added
- Always add to the end
  - No such thing as “delete”, only flagged “inactive”
  - Empty spaces can never be used
- Access sequentially, or directly using RBA
- You can use an AIX to link keys to RBA
- Good for logs, bank transaction history, etc

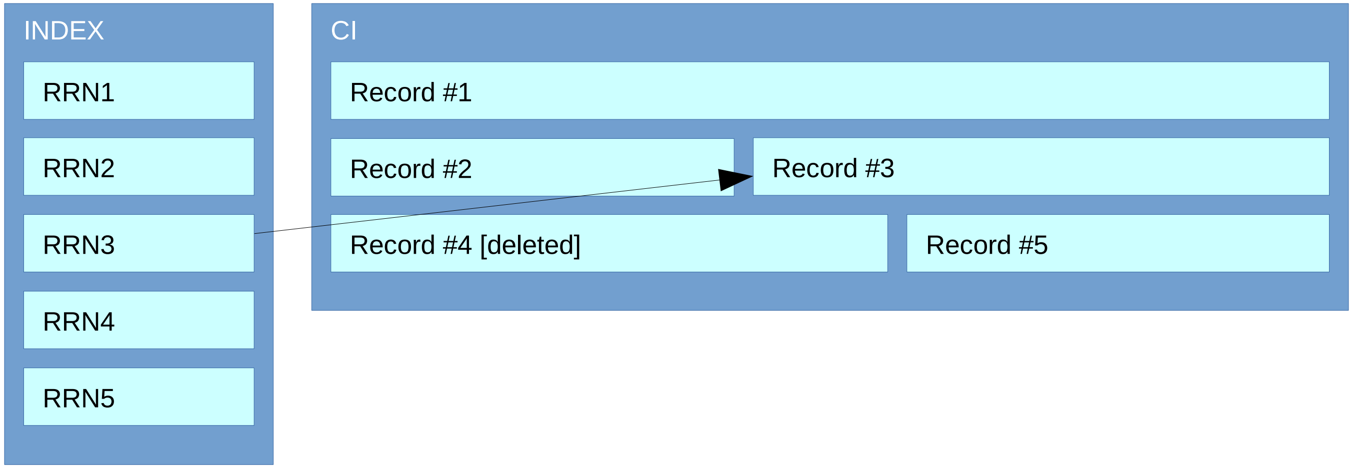
Relative Record Data Set

- Pre-formatted fixed-length records
  - Sequenced by relative number
  - Slots can be used or unused (may have high fragmentation)
  - Insert and access are by RRN (relative record number)
  - Allows direct and sequential access
  - Think Hash table



Variable Relative Record Data Set

- Similar to RRDS, but uses variable-length records
- Records are stored in relative number order
- Similar to a KSDS:
  - Has an index that correspond RRN to RBA (CCHH)
  - Uses SPLITs when inserting



Linear Data Set

- Byte-addressable storage (GET byte 15674)
- CI Size is multiple of 4096
- Essentially, a non-VSAM file with VSAM facilities
- Allows Data In Virtual
  - Reads a 4k page into storage
  - Lets the program access it as if it were memory

HFS and z/FS

- HFS - Hierarchical File System
  - Used by UNIX to store directory structures
  - Single-volume sequential data
  - From z/OS, it looks similar to a PDS
  - Unix system “mounts” them – think ISO file

- Z/FS – Newer version of HFS (z/OS 1.7)
  - Better performance
  - Uses VSAM Linear DS

- References:
  - *z/OS Unix System Service Planning,* https://ibm.biz/BdF43v
  - *DFSMS Using Data Sets,* https://ibm.biz/BdF43m

Common Uses

TYPE	Common Use
Direct	Data, not that common
Sequential	Everything – logs, EREP, notes
PDS	LOADLIBs, JCL collections, CLIST libraries
PDSE	Same as PDS
KSDS	DATA, such as bank records
ESDS	Transaction logs
RRDS	Data
Linear	DB2 tables, SMS SCDS
HFS/zFS	Unix directory structures

Limitations

- All format have limitations:

Limitation	Direct/Seq	VSAM	PDS	PDSE	Unix
Max Size	65,535 tracks	4GB	65,535 TRK	-	-
Max Extents	16/vol	123/255	16	123	123/255
Max Volumes	59	59	1	1	1
Sharing Integ	No	Some	No	Some	No
Fragmentation	Yes/No	Yes	Yes	Yes	No

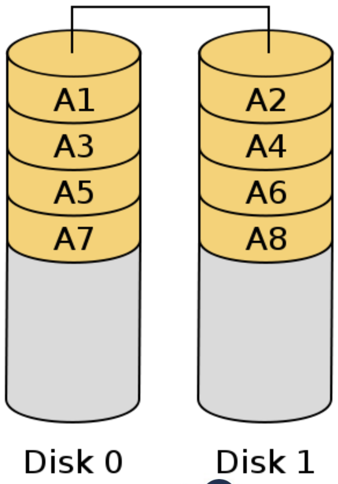
Extended Format

- Extended Format relieves some of the limitations
- Logically the same format
- Stored differently on the hardware to exploit hardware and software facilities of SMS
- Must be SMS managed
- Enabled through allocation or data class parameter

- Allows some extra features:
  - Compression
  - Data Striping
  - Extended Addressing (larger files)
  - VSAM Allocation and Buffering

Extended Format Features

- **Compression:**
  - Reduces space to store data
  - Enabled via Data Class compaction attribute
  - Works with Sequential and KSDS
  - Session 16130, 16138, 15709 all talk more about it
- **Striping:**
  - Distributes Data blocks across multiple volumes
  - Allows higher throughput rate
  - Works with VSAM and Sequential data sets
  - Controlled by storage class parameters



Extended Format Features

- **Extended Addressing:**
  - VSAM can grow to 4GB * CI Size (128TB for 32K CI)
  - Sequential DS can have 123 extends per vol over 59 vols
  - PDS, PDSE, Direct do not change

- **VSAM Allocation / Buffering**
  - Partial Release
  - System-Managed Buffering
  - Note: Catalogs cannot be extended format

Creating Data Sets

Parameter Definitions

Defining Data Set Attributes

- Defining Non-VSAM
  - **JCL**
  - ISPF Panels
  - Dynamic Allocation
- Defining VSAM
  - **IDCAMS**
  - JCL
  - Dynamic Allocation

Parameter are roughly the same between utilities.
We'll focus on the Attribute

Data Set Attributes

- **Space Units**
  - Defines which construct you'll use to allocate space
  - Possible values:
    - CYL – cylinders (=15 tracks, ~830KB)
    - TRK – tracks (=56,664 bytes)
    - BLKS – blocks
    - KB, MB, BYTES
    - Records
- **Average Record Unit (AVGREC)**
  - Used primarily in Data Class
  - Defines a multiple for Primary and Secondary space
  - Possible values: U (unit), K (1024), M (104856)

Data Set Attributes

- **Primary Space**
  - How much space to allocate in the first extent
  - Specified in whole numbers

- **Secondary Space**
  - Space to allocate when primary is exhausted

- **Directory Blocks:**
  - Number of block to allocate for the PDS or PDSE directory
  - Set to 0 (or leave blank) for non-PDS

Data Set Attributes

- Space attribute example (JCL):

SPACE=(TRK,(50,3,100),RLSE)

Units **Primary** **Directory Blocks**

Secondary

Non-VSAM Attributes

- **Data Set Organization (DSORG)**
  - PS – Physical Sequential
  - PO – Partitioned
  - DA – Direct
  - For absolute addressing, add a “U”, such as “PSU”

- **Record Format (RECFM)**
  - Specifies characteristics of the records

Byte 1	Byte 2
F – Fixed Length	B – Blocked Records
V – Variable Length	S – Spanned Records
U – Undefined length	BS – Blocked Spanned

  - Example: FB, VB, VBS, U

Data Set Attributes

- **Logical Record Length (LRECL/RECORDSIZE)**
  - Specifies the length, in bytes, of the records
  - If variable length (VB), specifies the maximum length
  - Has no effect for RECFM=U

- **Block Size (BLKSIZE)**
  - Defines the size of the blocks to be used
  - Specify 0 to use System-Determined Block Size
  - Must be a multiple of LRECL

Non-VSAM Attributes

- **Data Set Type (DSNTYPE)**
  - Defines the type of data set you are creating
  - Possible values:
    - LIBRARY – Partitioned Data Set Extended (PDSE)
    - PDS – Partitioned Data Set
    - HFS – Hierarchical File System
    - LARGE – Creates a large-format sequential
    - EXTREQ – Extended format, required
    - EXTPREF – Extended format, preferred
    - BASIC – Basic format sequential
    - Blank – Sequential or PDS, depending on Directory field
  - DSNTYPE relies on other parms, and does not override

VSAM Attributes

- **RECORDSIZE**
  - Same as LRECL
  - Defines the size of the records
- **CONTROLINTERVALSIZE**
  - Defines the CI Size
  - Similar to BLKSIZE
- **FREESPACE**
  - Defines how much space is left in the CI for insert/expansion
- **SPANNED/NONSPANNED**
  - Defines whether records are spanned (similar to S in FBS)

VSAM Attributes

- **KEYS**(length offset)
  - Defines the length and offset of the VSAM key for KSDS
  - Example: KEYS(14,0)

```
BOHLING........NEAL.........95116.......BANKACCOUNT.......
```

KEY

  - Example 2: KEYS(5,28)

```
BOHLING........NEAL.........95116.......BANKACCOUNT.......
```

KEY

VSAM Attributes

- **INDEXED / LINEAR / NONINDEXED / NUMBERED:**
  - Keyword that defines the VSAM file type
  - INDEXED – KDSD
  - LINEAR – linear
  - NONINDEXED – ESDS
  - NUMBERED – RRDS
- Many, many more...
  - See *DFSMS Access Method Services Commands* (SC23-6846-01)

Examples of Data Set Definitions

- JCL DD Card:

```
//OUTFILE  DD  DSN=SYS1.TRACE.T,
//             SPACE=(CYL,(50,3),RLSE),DISP=(,CATLG,DELETE),
//             LRECL=1024,BLKSIZE=6144,RECFM=FB,DSORG=PS,
//             UNIT=3390,VOL=SER=DUMPD1
```

- IDCAMS Statements

```
DEFINE CLUSTER( NAME(MY.VSAM.FILE) -
         CYLINDERS(10 5) INDEXED VOLUMES(*) -
         RECORDSIZE(1024 1024) KEYS(16 0)
         CONTROLINTERVALSIZE(32768) ) -
       INDEX( NAME(MY.VSAM.FILE.INDEX) ) -
       DATA( NAME(MY.VSAM.FILE.DATA) )
```

Utilities and Subsystems

Intro to Managing Data

Utilities

- IDCAMS
  - Works with VSAM data sets
  - Can do define, copy, delete
  - Also interacts with catalog information
- IEBCOPY
  - Copy PDS and PDSE
  - Convert between PDS and PDSE
- IEBGENER
  - Copy sequential files
- IEFBR14
  - Does nothing, but can use DD cards to manage data sets

Utilities

- DFSMSdss (DSS)
  - Very powerful data movement tool
  - Does copy / backup / data conversion
- ISPF Option 3 panels
  - Panel-driven utilities to create / delete / manage data sets
  - Also has an editor that can edit sequential data sets
- ISMF Data Set Panels
  - Allows edit / delete / rename / etc
  - Allows you to save data sets lists

Introduction to Managing Data

- Storage Management Subsystem (SMS)
  - Runs under the SMS ASID
  - Helps storage administrators manage data sets
  - Includes classes to simplify allocation and define attributes:
    - DATA CLASS – defines JCL parms for default use
    - STORAGE CLASS – defines accessibility and performance requirements
    - MANAGEMENT CLASS – defines retention and management
    - STORAGE GROUP – defines which group of volumes
  - Automatic Class Selection (ACS) routines assign classes based on user-written logic
    - *Come to session 16115 for Hand-On ACS writing (Fri @ 10a)*
  - SMS-managed = has a storage class assigned

Subsystems Continued

- **SMSPDSE / SMSPDSE1**
  - Address spaces needed to use PDSE
  - Manage serialization on PDSE across the plex
  - Enables system-wide buffering
  - See sessions 16126 (Monday @ 4:15) for further info

- **SMSVSAM**
  - Enables VSAM Record Level Sharing
  - Enables cross-plex serialization at the record level
  - Also has enhanced buffering and caching capabilities
  - Not required for VSAM, but has benefits in cross-plex environments
  - See sessions 16124 and 16125 (Mon/Tues) for more info

Subsystems

- **Hierarchical Storage Management (HSM)**
  - Very powerful storage management tool
  - Migrates / recalls data sets based on Retention parameters
  - Create and maintains backups
  - Allows you to keep important data on DASD and roll old or less-used data to backup storage
  - Retention values are set by Management Class (SMS)
  - HSM control data sets are VSAM data sets
  - For more, see sessions:
- **CATALOG**
  - Provides the interface into the catalog data sets
  - Runs in ASID CATALOG

Summary

- Two basic types of data sets – VSAM / NON-VSAM
- Many sub-varieties
  - PDS/PDSE
  - Sequential/Direct
  - Fixed/Variable/Spanned Block
- Data set attributes define type and options
- Attributes reflect devices and format
- There are several utilities to help create / manage data
- Several subsystems can be enabled to assist and enable additional features

DFSMS Basics:
Data Set Fundamentals
Get to Know Your Data Sets!

Neal Bohling and Tom Reed
DFSMS Defect Support @ IBM

August 7, 2014
Session Number 16119

Notices & Disclaimers

Product information and data has been reviewed for accuracy as of the date of initial publication. Product information and data is subject to change without notice. This document could include technical inaccuracies or typographical errors. IBM may make improvements and/or changes in the products and/or programs described herein at any time without notice.

References in this document to IBM products, programs, or services does not imply that IBM intends to make such products, programs or services available in all countries in which IBM operates or does business. Consult your local IBM representative or IBM Business Partner for information about the product and services available in your area.

Any reference to an IBM Program Product in this document is not intended to state or imply that only that program product may be used. Any functionally equivalent program, that does not infringe IBM's intellectually property rights, may be used instead. It is the user's responsibility to evaluate and verify the operation of any non-IBM product, program or service.

THE INFORMATION PROVIDED IN THIS DOCUMENT IS DISTRIBUTED "AS IS"WITHOUT ANY WARRANTY, EITHER EXPRESS OR IMPLIED. IBM EXPRESSLY DISCLAIMS ANY WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR INFRINGEMENT. IBM shall have no responsibility to update this information. IBM products are warranted according to the terms and conditions of the agreements (e.g., IBM Customer Agreement, Statement of Limited Warranty, International Program License Agreement, etc.) under which they are provided. IBM is not responsible for the performance or interoperability of any non-IBM products discussed herein.

Notices & Disclaimers

The performance data contained herein was obtained in a controlled, isolated environment. Actual results that may be obtained in other operating environments may vary significantly. While IBM has reviewed each item for accuracy in a specific situation, there is no guarantee that the same or similar results will be obtained elsewhere.

The responsibility for use of this information or the implementation of any of these techniques is a customer responsibility and depends on the customer's or user's ability to evaluate and integrate them into their operating environment. Customers or users attempting to adapt these techniques to their own environments do so at their own risk. IN NO EVENT SHALL IBM BE LIABLE FOR ANY DAMAGE ARISING FROM THE USE OF THIS INFORMATION, INCLUDING BUT NOT LIMITED TO, LOSS OF DATA, BUSINESS INTERRUPTION, LOSS OF PROFIT OR LOSS OF OPPORTUNITY.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not necessarily tested those products in connection with this publication and cannot confirm the accuracy of performance, compatibility or another claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

The provision of the information contained herein is not intended to, and does not, grant any right or license under any IBM patents or copyrights. Inquiries regarding patent or copyright licenses should be made, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

Any statements regarding IBM's future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only.

Trademarks

DFSMSdfp, DFSMSdss, DFSMShsm, DFSMSrmm, IBM, IMS, MVS, MVS/DFP, MVS/ESA, MVS/SP, MVS/XA, OS/390, SANergy, and SP are trademarks of International Business Machines Corporation in the United States, other countries, or both.

AIX, CICS, DB2, DFSMS/MVS, Parallel Sysplex, OS/390, S/390, Seascape, and z/OS are registered trademarks of International Business Machines Corporation in the United States, other countries, or both.

Domino, Lotus, Lotus Notes, Notes, and SmartSuite are trademarks or registered trademarks of Lotus Development Corporation. Tivoli, TME, Tivoli Enterprise are trademarks of Tivoli Systems Inc. in the United States and/or other countries.

Java and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both. UNIX is a registered trademark in the United States and other countries licensed exclusively through The Open Group.

Other company, product, and service names may be trademarks or service marks of others.