#SHAREorg

SHARE
Technology · Connections · Results

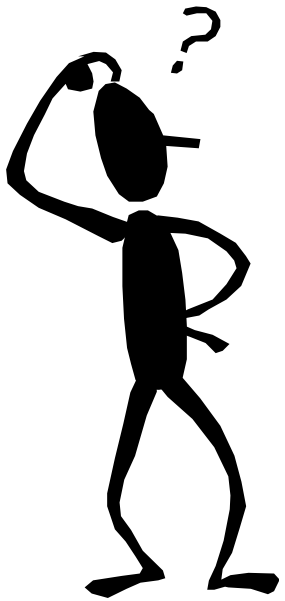
Introduction to VSAM

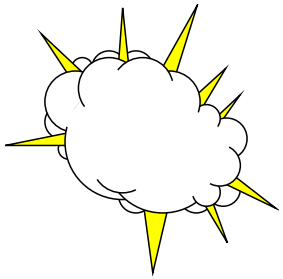
Session 12994

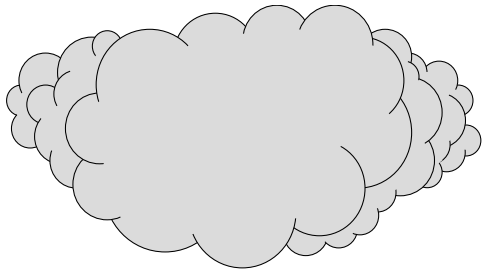
Presented by
Michael E. Friske

SHARE in San Francisco 2013

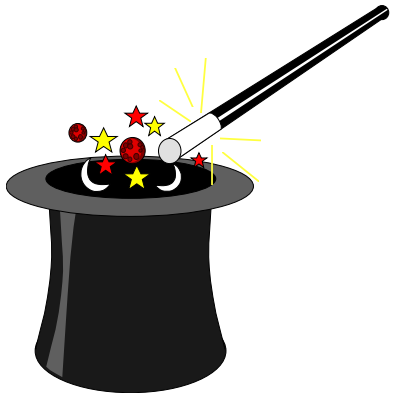
Exactly What Is VSAM?



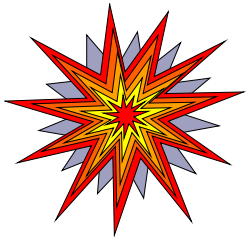




Is it a mysterious black cloud?



Is it magic?



Does VSAM Confuse, Frustrate, or Overwhelm You?

VSAM - The Acronym

V irtual

S torage

A ccess

M ethod

Types of VSAM Data Sets

- ESDS - Entry-Sequenced Data Set
- RRDS - Relative Record Data Set
- KSDS - Keyed-Sequenced Data Set
- LDS - Linear Data Set

Entry-Sequenced Data Set

- Sequences records by the order they are entered
- Supports variable length records
- Records are only added to the end of the data set
- Records can be updated, but not lengthened
- Existing records cannot be deleted, only marked as “inactive”
- Alternate index can be defined over an ESDS
- Used for applications that require only sequential access (direct access by RBA only)

Relative Record Data Sets

- Records can be either fixed-length or variable-length, but must be specified on DEFINE
- Records are stored in “slots” for fixed-length RRDSs
- Spanned records are not allowed
- Best for applications that require direct access only

Keyed-Sequenced Data Set

- Records are stored in ascending, collating sequence by key
- Records can be accessed with sequential, skip-sequential, or direct access easily with high-level languages
- Alternate indexes can be defined over a KSDS

Linear Data Sets

- LDS's can be accessed using:
  - VSAM
  - Data-In-Virtual (DIV), if the CI size is 4096
  - Window Services, if the CI size is 4096
- VSAM does not have any concept of a "record" in an LDS
- DB2 is currently the biggest user of Linear Data Sets
- Many program products use Linear Data Sets

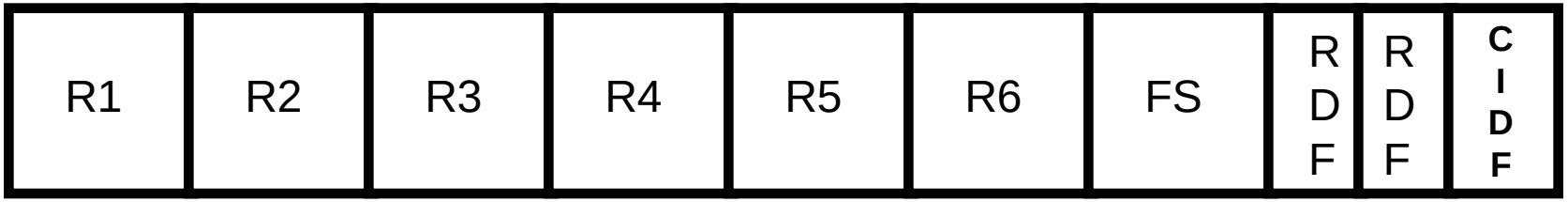
Control Interval (CI)

- Control Intervals (CI) in VSAM are equivalent to blocks for non-VSAM data sets
- Control Intervals contain:
  - Logical records
  - Free space
  - Control information fields
- The smallest unit of transfer between a disk and the CPU is a Control Interval

Control Interval Size

- For NONSPANNED, the CI size must be at least 7 bytes larger than the max record size
- CI size can be 512 to 8192 bytes in 512 byte increments or 8KB to 32KB in 2KB increments
- For Linear data sets, the CI size can be 4096 to 32768 in 4096 increments
- VSAM will adjust if a valid size is not specified
- Usually different sizes for DATA and INDEX component

Anatomy of a Control Interval

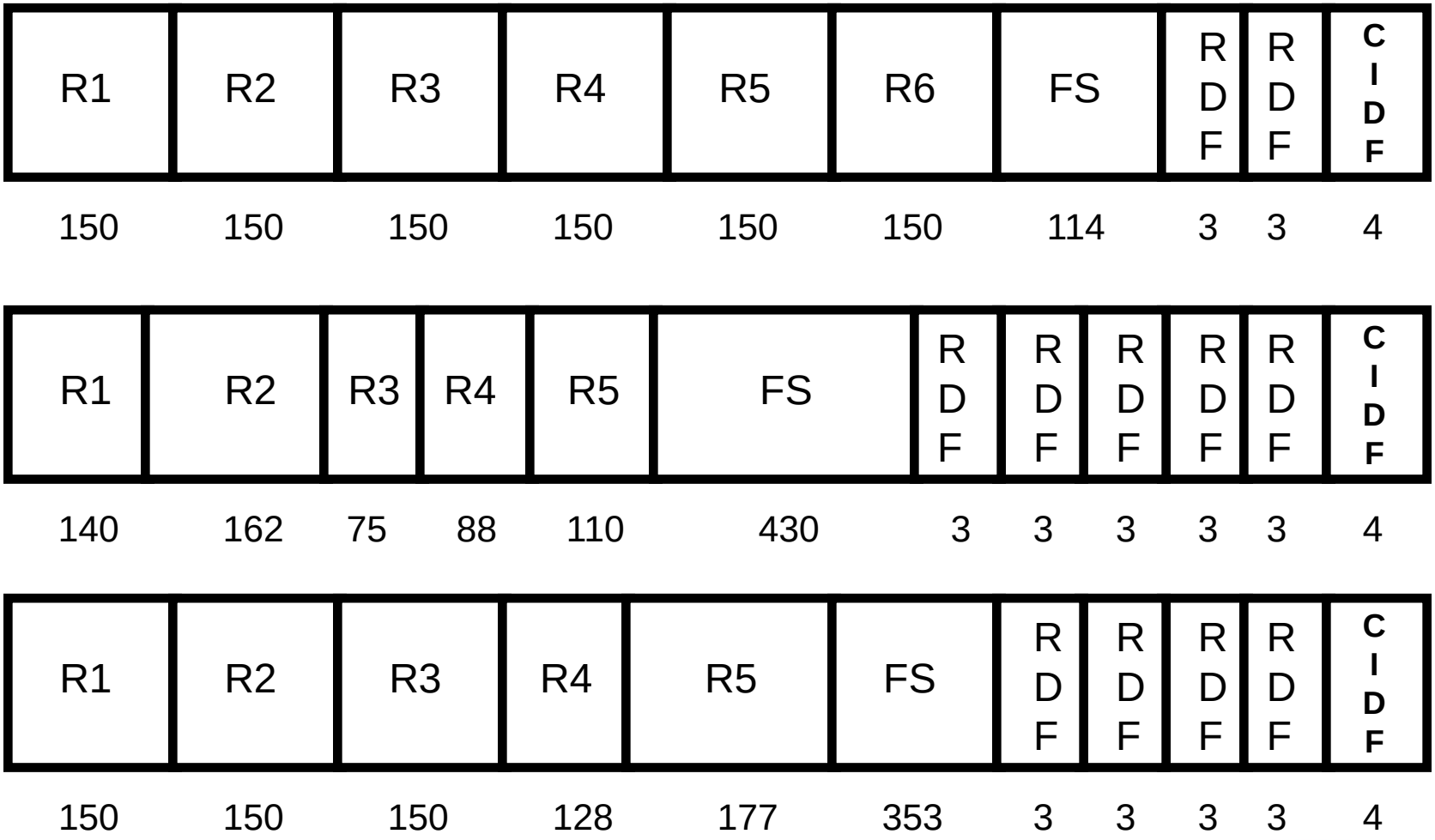


- Rn - Records
- FS - Free space
- CIDF - Control Interval Definition Fields indicate where the free space begins and how much is available
- RDF - Record Definition Fields describe the length of records in the and how many adjacent records are the same length

Control Interval Metadata

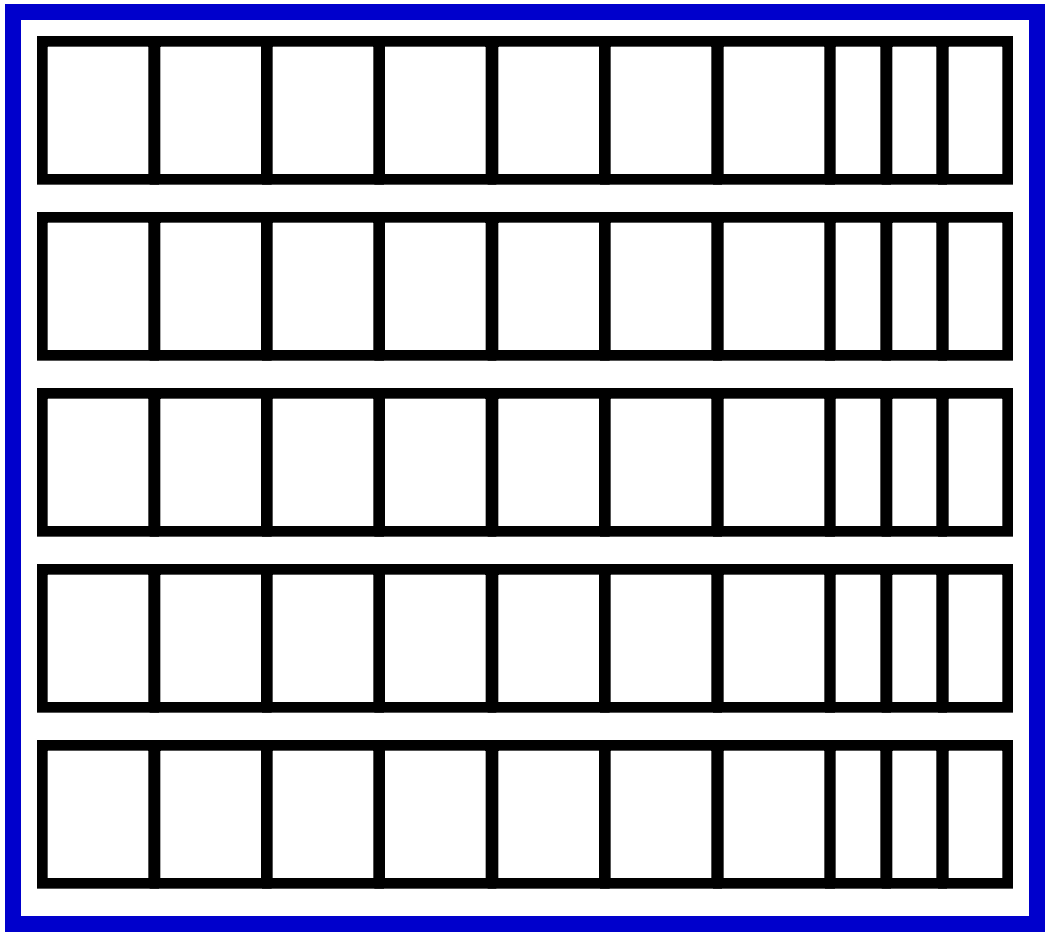
- CIDF - Control Interval Definition Fields are 4 bytes long
  - 2 bytes – The displacement from the beginning of the CI to the beginning of the unused space
  - 2 bytes – The length of the unused space
- RDF - Record Definition Fields are 3 bytes long and describe the length of records and how many adjacent records are the same length
  - 1 byte – RDF type
  - 2 bytes – Record length or how many adjacent records

How Record Lengths Affect the Number of RDFs



Control Areas (CA)

- Groups of Control Intervals (CI)
- Contiguous space on disk
- Maximum size for non-striped data sets is 1 cylinder
- CA size for striped data sets is # stripes in tracks



Control Area

Size of Control Area

- The Control Area is the smaller of your primary and secondary allocation up to a maximum of 1 cylinder for non-striped data sets

- Examples:
  - TRACKS(10 2) = 2 track CA
  - CYLINDERS(10 2) = 1 cylinder CA
  - MEGABYTES(50 10) = 1 cylinder CA
  - TRACKS(100 20) = 1 cylinder CA
  - TRACKS(1 100) = 1 track CA

Advantages & Disadvantages of a Large CA

- Advantages
  - Less frequent CA splits
  - Less index records
  - More CI's can be read into storage at one time
- Disadvantages
  - More data has to be moved to do a CA split
  - More real storage and buffers are tied up during sequential operations

Defining VSAM Data Sets Using IDCAMS

```
//STEP010  EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=*
//SYSIN    DD *
  DEFINE CLUSTER( -
      NAME(A.B.C) CYLINDERS(50,10) SPEED -
      RECORDSIZE(150,476) INDEXED KEY(18,0)
      SHAREOPTIONS(1,3)) -
    DATA(CONTROLINTERVALSIZE(4096))
```

```
//STEP010  EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=*
//SYSIN    DD *
  DEFINE CLUSTER( -
      NAME(A.B.C) DATACLASS(VS150KS))
```

Defining VSAM Data Sets Using JCL

```
//NEWVS   DD DSN=A.B.C,DISP=(NEW,CATLG),
//   SPACE=(CYL,(50,10)),RECORG=KS,
//   LRECL=150,KEY=18,KEYOFF=0

//NEWVS   DD DSN=A.B.C,DISP=(NEW,CATLG),
//   DATACLAS=VS150KS
```

SMS Constructs for VSAM

- DATACLASS(dcname)
  - Data set attributes
  - System Managed Buffering
  - RLS CF caching and buffering options
  - CA Reclaim
- MANAGEMENTCLASS(mcname)
  - Retention requirements
  - Backup requirements
  - Migration requirements
- STORAGECLASS(scname)
  - Striping
  - RLS Cache Set
  - RLS Lock Set

Specifying the Type of VSAM Data Set

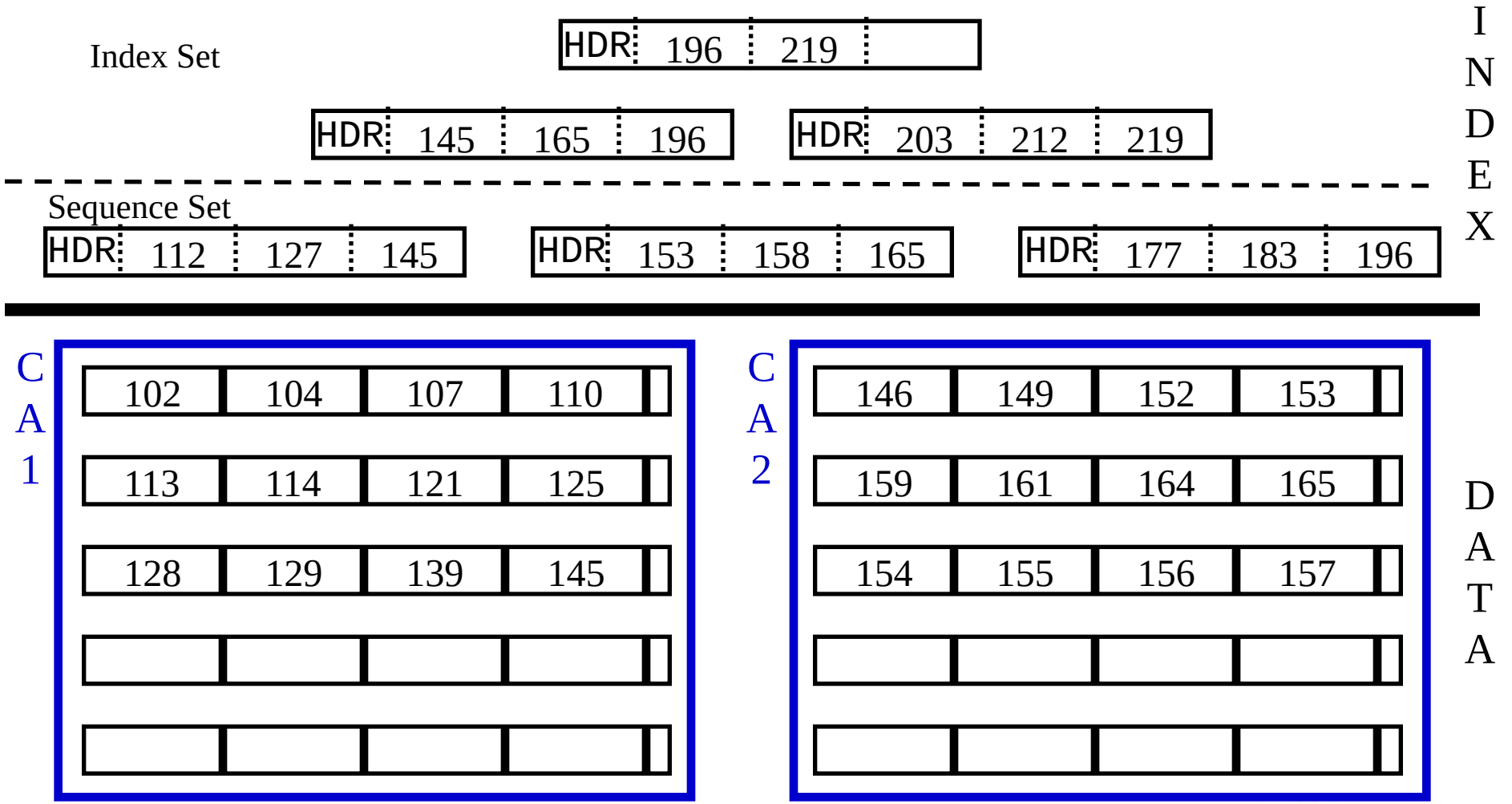
With IDCAMS

- INDEXED - KSDS
- NONINDEXED - ESDS
- NUMBERED - RRDS
- LINEAR - LDS

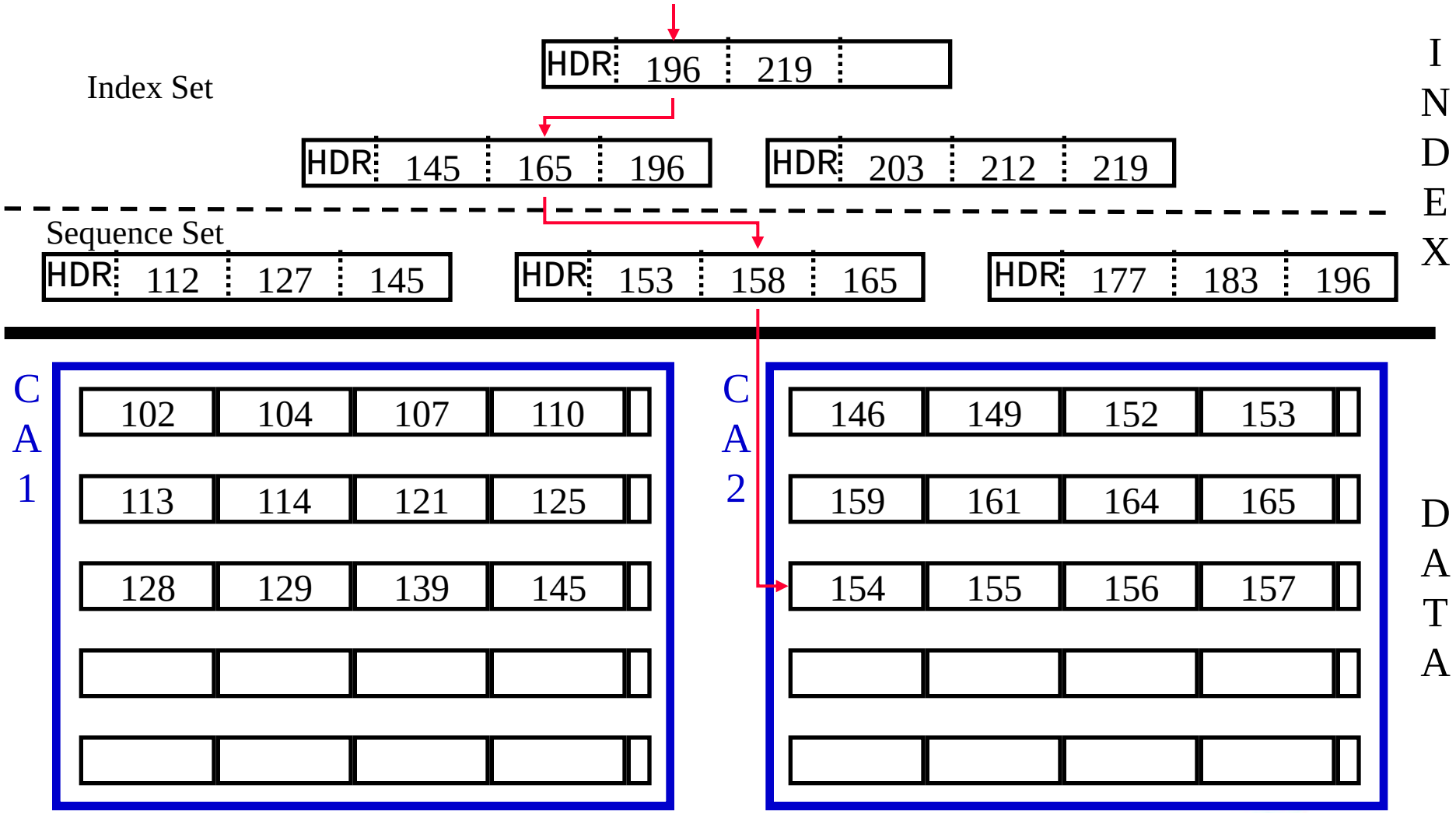
With JCL

- RECORG=KS - KSDS
- RECORG=ES - ESDS
- RECORG=RR - RRDS
- RECORG=LS - LDS

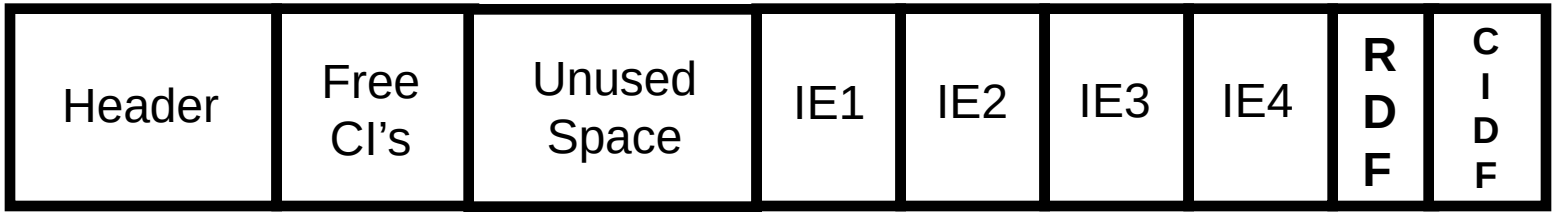
Structure of a KSDS



Locating Record with Key=156



KSDS Index Record



- Index record length is CISIZE - 7
- Header is 24 bytes long
- Index entries contain compressed keys + 3 bytes to describe:
  - F: Number of characters eliminated from the front
  - L: Number of characters left after compression
  - P: Vertical pointer

KSDS Index

- There is only 1 index record per control interval (so Length of CI = Length of Index Record - 7)
- CIDF = Length of the index record – (CIDF + RDF)
- There is only 1 RDF for each CI in the INDEX component

Index Record Correlation to DATA Component

- There is an Index Entry for each Control Interval in the DATA component
- An Index record cannot be larger than the CI size for the INDEX component
- Each INDEX record has to be large enough to point to all Control Intervals in a Control Area in the DATA component

KSDS Key Compression

- Front compression eliminates the leading characters on the key that are the same as the preceding key in the index.
- Rear compression eliminates the insignificant values from the end of the key. The current index key is compared to the next data key, and the characters to the right of the first character that is unequal to the corresponding character in the following key are eliminated.

Unusable CI's / Wasted Space

- A KSDS with keys that do not compress well and an INDEX CI size that is too small will not be able to use all of the CI's in the DATA component
- A KSDS with CI's that cannot be addressed in the DATA component will require more disk space to store data records
- The EXAMINE command can be used to identify these situations

EXAMINE INDEXTEST Messages

- IDC11773I nnnn KEYS PROCESSED ON INDEX LEVEL y, AVERAGE KEY LENGTH: keylen
- IDC11774I CURRENT INDEX CISIZE IS cisize, RECOMMENDED MINIMUM INDEX CISIZE IS newcisize
- IDC11775I nnnn DATA COMPONENT CIS ARE ESTIMATED TO BE UNREACHABLE

Index CI Size

- VSAM calculates the minimum CI size for the INDEX component assuming the key will compress to 1/3 the original size
- If the user specifies a CI size smaller than the minimum value VSAM calculates, VSAM will override the user specified CI size
- Check to make sure the value used is sufficient for your data set

KSDS CI Reclaim

- When all records in a CI have been deleted, that CI becomes available for new records with any key that can be loaded into that CA
- If CA reclaim is turned on (system level and for the data set), the CI can be reused for any key once the CA has been reclaimed

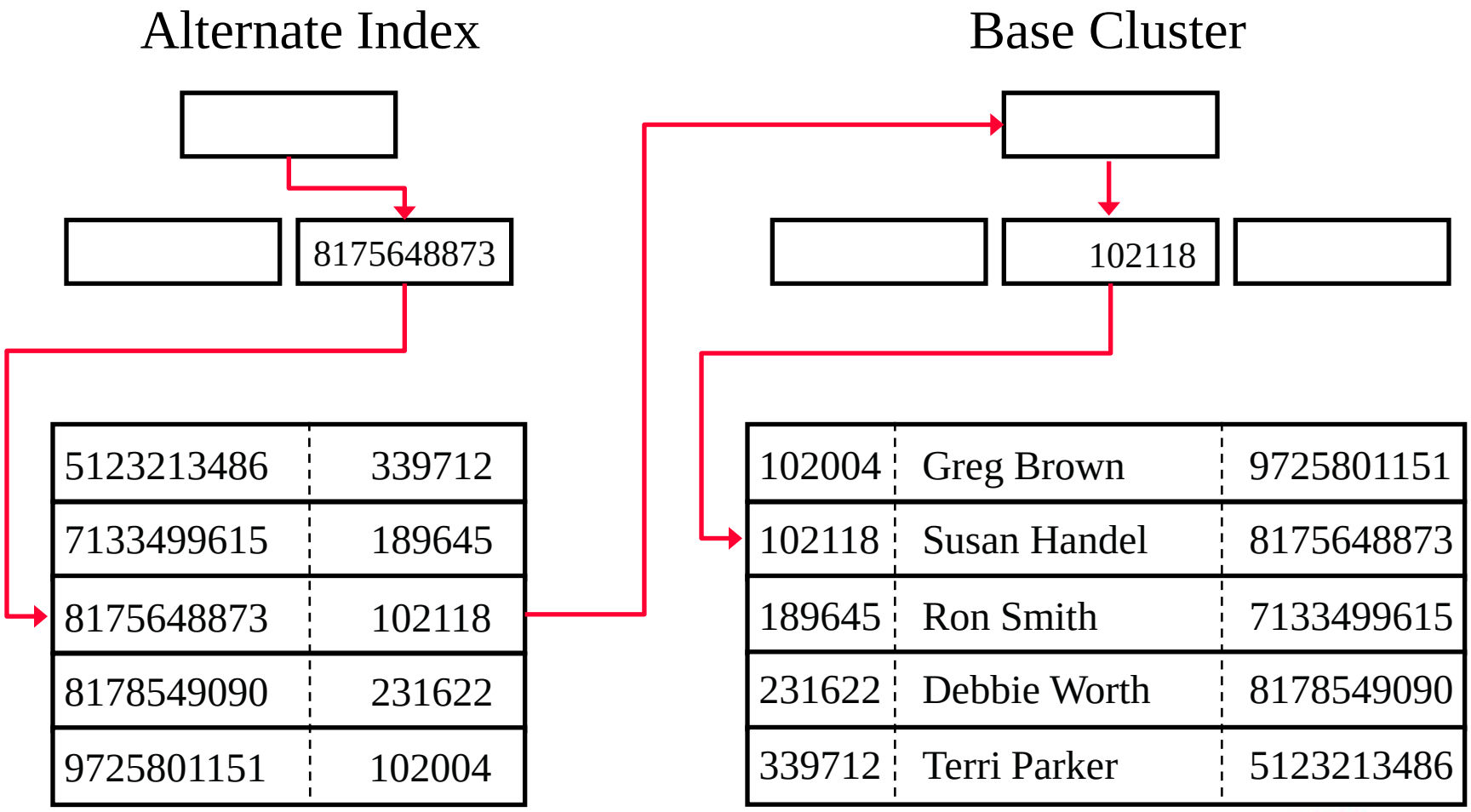
KSDS CA Reclaim

- When CA reclaim is not turned on for a data set, a CA can only be reused for records that fit within the key structure for that CA once all of the records in the CA have been deleted
- When CA reclaim is turned on for a data set, a CA can be reused for any records once all of the records in the CA have been deleted

Using an Alternate Index

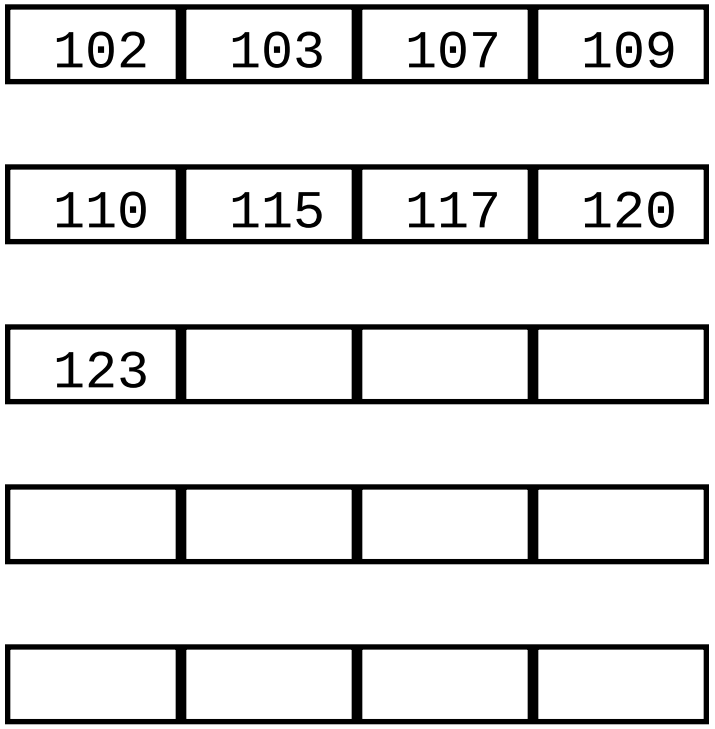
- Provides another way to access data in a VSAM KSDS or an ESDS
- Allows access back to the base cluster through a PATH definition
- Supports UPGRADE so that when a record accessed through the base cluster is updated the corresponding record in the AIX is also updated

Accessing Records with an Alternate Index

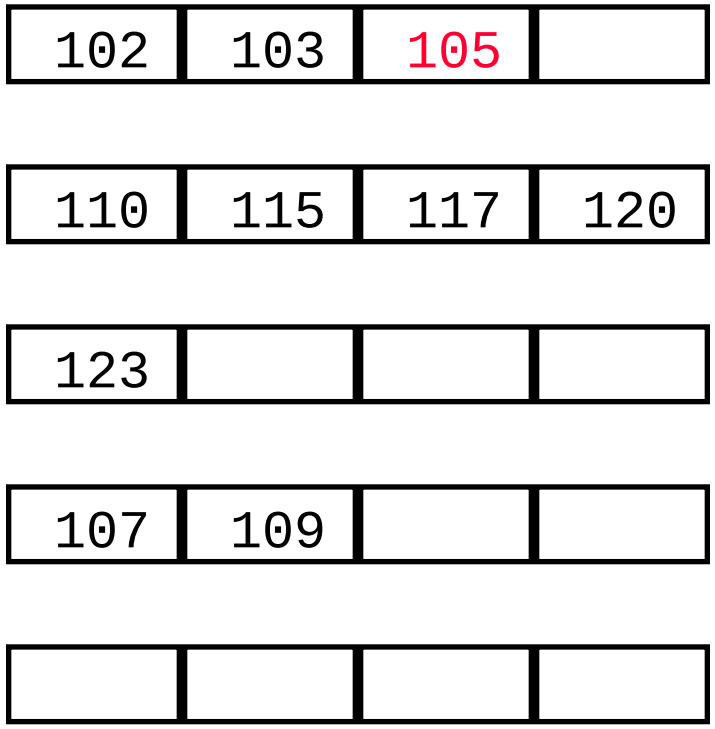


CI Split Processing

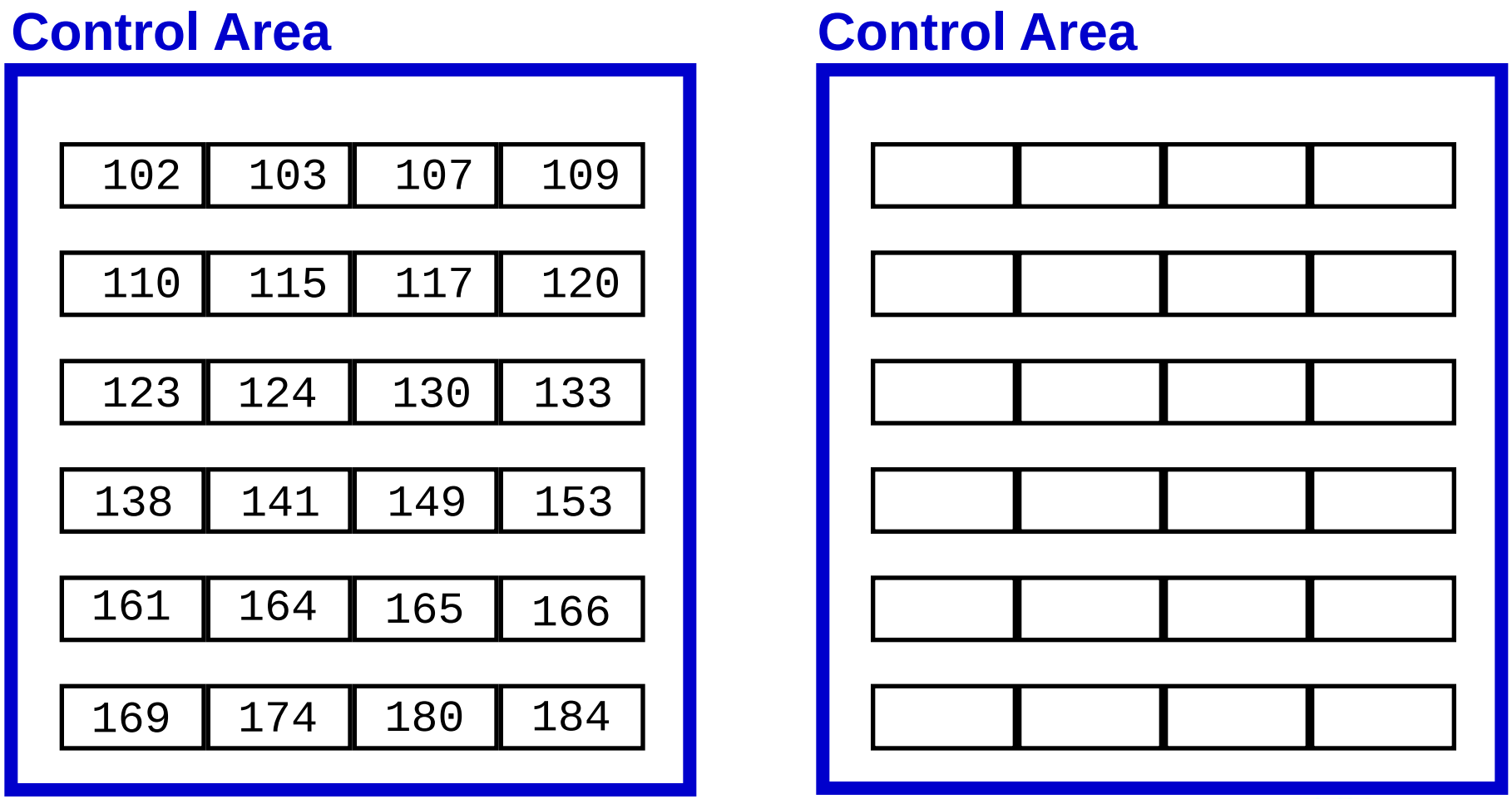
Before Insert of Record 105



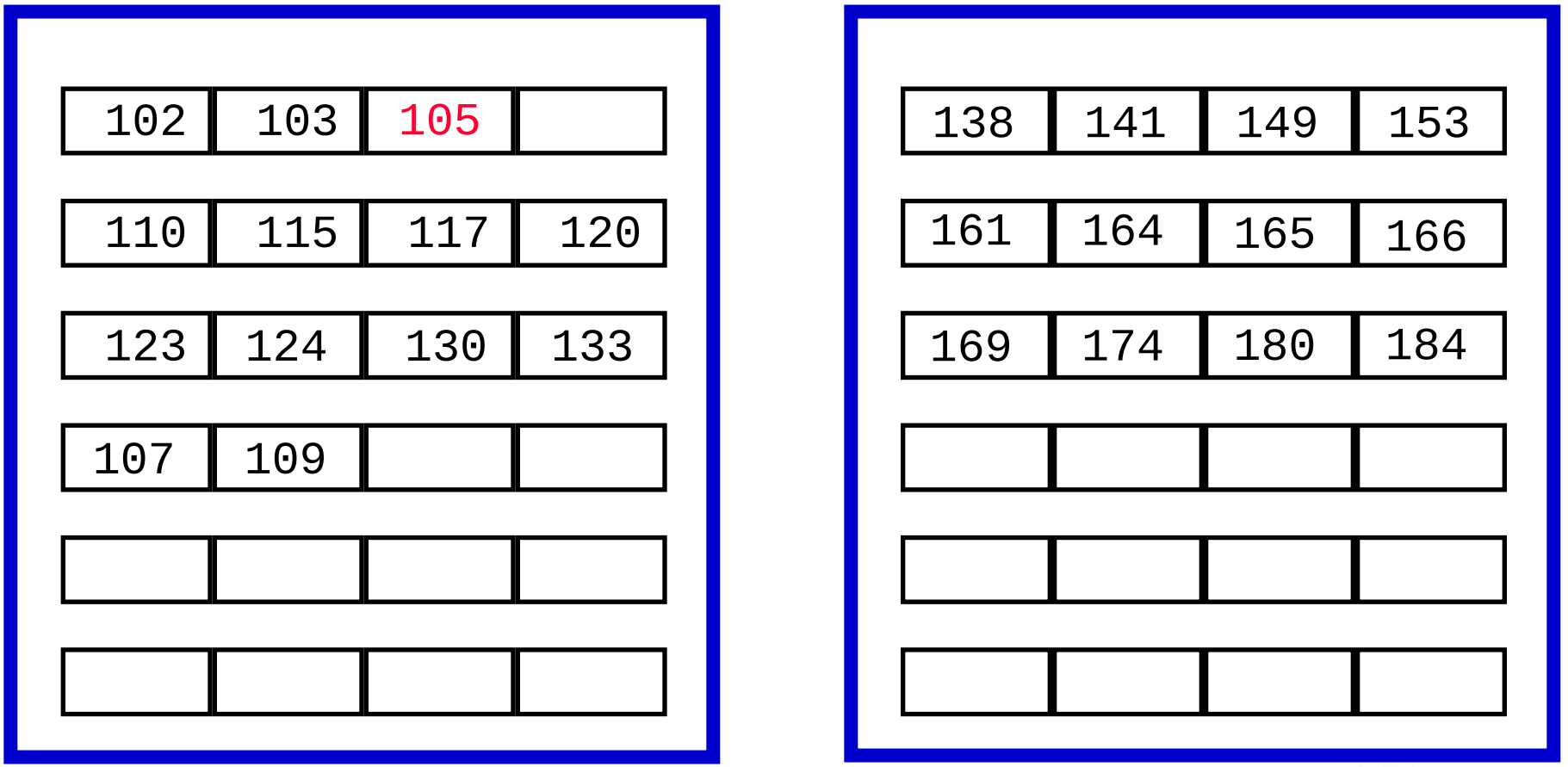
After Insert of Record 105



CA Split Example – Insert Record 105



CA Split Example – After CA Split



CI & CA Splits Can Be Good

- The existence of CI and CA splits does not cause performance problems
- Splits create free space right in the areas of the data set where the inserts are being done
- Reorganizing a data set removes this free space and can cause the data set some performance problems until the splits can be done again
- Data sets should only be reorganized to reclaim disk space

Two Types of Split Processing

- The type of split that occurs is determined by the strategy specified at open time
  - NIS – Normal Insert Strategy
  - SIS – Sequential Insert Strategy
- Specifying the strategy in Assembler
  - MACRF=NIS|SIS
- Specifying the strategy in COBOL
  - ACCESS MODE IS RANDOM|DYNAMIC

When to Specify NIS or SIS

- ACCESS IS RANDOM should be used to when record insertion is random
- ACCESS IS DYNAMIC should be used for mass sequential insertion

Mass Sequential Insertion Tests

- Test #1 – ACCESS IS RANDOM and insert 30,000 records in sequential order
  - CI Splits – 2,003
  - CA Splits – 28
  - EXCP's – 32,717
  - File size grew from 34 cylinders to 62 cylinders
- Test #2 – ACCESS IS DYNAMIC and same 30,000 records as Test #1
  - CI Splits – 1
  - CA Split – 11
  - EXCP's – 3,511
  - File size grew from 34 cylinders to 44 cylinders

Random Insertion Tests

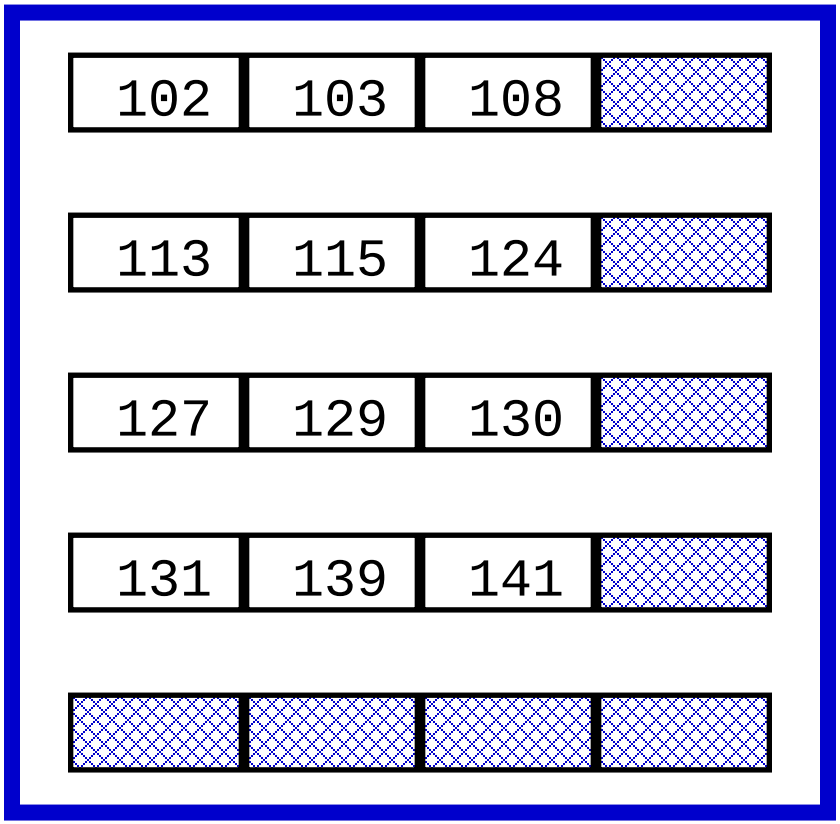
- Test #1 – ACCESS IS DYNAMIC and insert 10,000 records in random order
  - CI Splits – 4,176
  - CA Splits – 154
  - EXCP's – 40,587
  - File size grew from 34 cylinders to 187 cylinders
- Test #2 – ACCESS IS RANDOM and same 10,000 records as Test #1
  - CI Splits – 3,576
  - CA Split – 20
  - EXCP's – 20,545
  - File size grew from 34 cylinders to 54 cylinders

Some Splits Are Bad

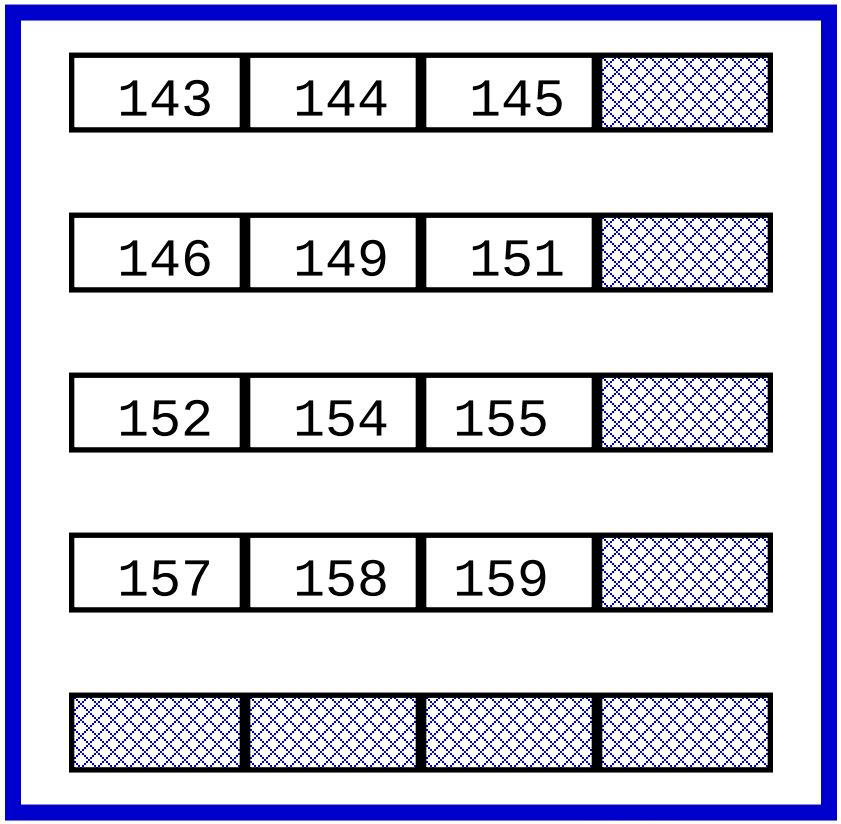
- Specifying the wrong insert strategy could cause excessive splits and increase CPU utilization and job elapse time
- Unnecessary CI and/or CA splits can increase response time for online transactions

FREESPACE(25 20)

Control Area



Control Area



FREESPACE Is Not “Free”

VSAM KSDS:
DATA CISZ = 4096
CI/CA = 180
FREESPACE(15,20)

Total bytes in CA = 4096 * 180 = 737,280

CA Free Space = (180 * .2) * 4096 = 147,456

CI Free Space = 4096 * .15 * (180 * .8) = 88,560

Percentage of each CA reserved for Free Space =
(147,456 + 88,560) / 737,280 = 32%

SHAREOPTIONS Parameter

- SHAREOPTIONS(n x)
  - “n” is the option for cross-region sharing requirements
  - “x” is the option for cross-system sharing requirements
- SHAREOPTIONS is ignored for VSAM data sets opened for Record Level Sharing (RLS)

Cross-region Sharing

- 1 - any number of users for READ OR one user for UPDATE; VSAM ensures complete integrity
- 2 - any number of users for READ AND one user for UPDATE; VSAM ensures only write integrity
- 3 - the data set can be fully shared, but it is the user's responsibility to maintain integrity
- 4 - same as option 3, but VSAM will refresh buffers for direct requests

Cross-system Sharing

- 3 - the data set can be fully shared, but the user is responsible for maintaining the integrity of the data set
- 4 - the same as option 3, but VSAM will refresh the buffers for each direct processing request

VSAM Features for Extended Format Only

- Compression
- VSAM partial space release
- Enhanced multi-volume allocation support
- Extended Addressability
- Striping
- System-managed Buffering (SMB)

VSAM Manuals

- DFSMS Using Data Sets
- DFSMS Access Method Services (AMS)
- VSAM Demystified