Chapter 27: Indexed Sequential Access Method (ISAM)

Author's Note: *This chapter is copied almost verbatim from the material in Chapter 20 of the textbook by Peter Abel. It is used by permission.*

Indexed sequential access method (ISAM) is available in many variations on microcomputers, minicomputers, and mainframes, although the preferred method under **DOS/VS** and **OS/VS** is the newer VSAM. (ELB: It is a fact that VSAM, to be discussed in the next chapter, can perform all of the services of ISAM. For this reason, many instructors prefer not to cover ISAM, but skip to VSAM. ISAM is included for historical reasons.)

A significant way in which ISAM (and other nonsequential file organization methods) differs from sequential organization is that the record keys in an indexed file must be unique; this is a system requirement, not just a programming practice. Consequently, an indexed file is typically a master file. Also, there is a clear difference between updating a sequential file and updating an indexed file. When you update a sequential file, you rewrite the entire file; this practice leaves the original file as a convenient backup in case the job must be rerun. When you update an indexed file, the system rewrites records in the file directly in place, thereby providing no automatic backup file. To create a backup, you periodically copy the file onto another device.

The flexibility of indexed sequential access method is realized at some cost in both storage space and accessing time. First, the system requires various levels of indexes to help locate records in the file. Second, the system stores new, added records in special reserved overflow areas. Check that your system supports ISAM before attempting to use it.

CHARACTERISTICS OF INDEXED SEQUENTIAL FILES

ISAM initially stores records sequentially and permits both sequential and random processing. The features that provide this flexibility are indexes to locate a correct cylinder and track and keys to locate a record on a track.

Keys

A key is a record control field such as customer number or stock number. Records in an indexed file are in sequence by key to permit sequential processing and to aid in locating records randomly, and blocks are formatted with keys. That is, ISAM writes each block immediately preceded by the highest key within the block, namely, the key of the last or only record in the block. The key is usually also embedded within each data record, as normal.

Unblocked Records

This is the layout of keys for unblocked records:

Key 201		Record 201	Key 205		Record 205	Key 206		Record 206

The records could represent, for example, customer numbers, and the keys could be for customer numbers 201, *20S,* and 206. In this example, the key is 3 characters long and the data record is the conventional size. Under unblocked format, a key precedes each block containing one record.

Blocked Records
This is the layout of keys for blocked records based on the preceding unblocked example:

Key 206		Record 201	Record 205	Record 206

Under blocked format, the key for the last record in the block, 206, precedes the block.

ISAM automatically handles this use of keys, and when you perform a read operation, the system delivers the block, not the separate key, to main storage.

Indexes
To facilitate locating records randomly, ISAM maintains three levels of indexes on disk: track index, cylinder index, and an optional master index.

Track index. When ISAM creates a file, it stores a track index in track 0 of each cylinder that the file uses. The track index contains the highest key number for each track on the cylinder. For example, if track 4 on cylinder 12 contains records with keys 201,205,206, and 208, the track index contains an entry for key 208 and a reference to cylinder 12, track 4. If a disk device has ten tracks per cylinder, there are ten key entries for each track index, in ascending sequence.

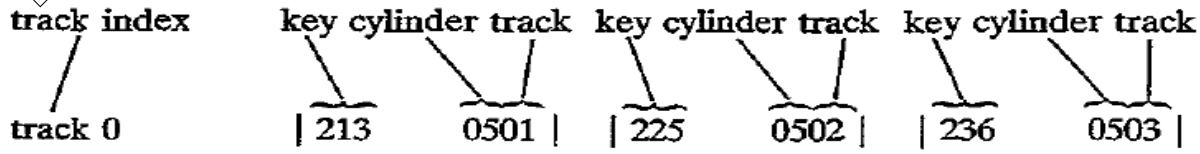
Cylinder index. When ISAM creates a file, it stores a cylinder index on a separate cylinder containing the highest key for each cylinder. For example, if the file is stored on six cylinders, the cylinder index contains six entries.

Master index. An optional master index facilitates locating an appropriate cylinder index. This index is recommended if the entries in the cylinder index exceed four cylinders, which is a very large file.

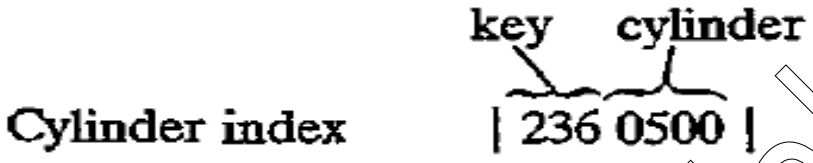
PROCESSING AN INDEXED FILE
Consider a small indexed file containing 14 records on cylinder 5, with tracks 1 and 2 containing five records and track 3 containing four. This area is known as the ***prime data area***. Track 1, for example, contains records with keys 205, 206, 208, 210, and 213. Assume that records are suitably blocked.

Track	Data Records on Cylinder 5
1	205 206 208 210 213
2	214 219 220 222 225
3	226 227 230 236 unused

Track 0 of cylinder 5 contains the track index, with an entry indicating the high key for each track. The track index entries specify that the highest keys on cylinder 5, tracks 1, 2, and 3 are 213, 225, and 236, respectively:



The cylinder index contains an entry for each cylinder that contains data, indicating the high key for each cylinder. In this case, the only index entry is a key 236 on cylinder 5 (the track number is not important in this index):



As an example of processing, a program has to locate randomly a record with key **227**. The read statement directs the system to perform the following steps:

1. Check the cylinder index (assuming no master index), comparing key **227** against its first entry, 236. Since **227** is lower, the required record should be on cylinder 5.
2. Access the track index in cylinder 5, track 0, comparing key 227 successively against each entry: 213 (high), 225 (high), and 236 (low). According to the entry for 236, the required record should be on cylinder 5, track 3.
3. Check the keys on track 3; find key 227 and deliver the record to the program's input area. If the key and the record do not exist, ISAM signals an error.

As you can see, locating a record randomly involves a .number of additional processing steps, although little extra programming effort is required. Even more processing steps are involved if a new record has to be added. If ISAM has to insert the record within the file, it may have to "bump" a record into an overflow area.

Overflow Areas

When a program first creates a file, ISAM stores the records sequentially in a prime data area. If you subsequently add a new record, ISAM stores it in an overflow area and maintains links to point to it. (ELB: In what follows, the reader should recall that a **disk cylinder** can be defined functionally as a set of records that the read/write heads can access without being physically moved. Track–to–track head motion is time consuming, taking about 10 msec.)

There are two types of overflow areas: cylinder and independent:

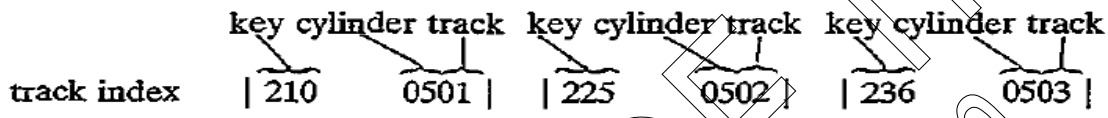
1. For a ***cylinder overflow area***, each cylinder has its own overflow track area. ISAM reserves tracks on the same cylinder as the prime data for all of its overflow records stored on a specific cylinder. The advantage of cylinder overflow is that less disk seek time is required to locate records on a different cylinder. The disadvantage is an uneven distribution of overflow records: Some of the overflow cylinders may contain many records, whereas other overflow cylinders may contain few or none.
2. For an ***independent overflow area***, ISAM reserves a number of separate cylinders for all overflow records in the file. The advantage is that the distribution of overflow records is unimportant. The disadvantage is in the additional access time to locate records in the overflow area.

A system may adopt both types: the cylinder overflow area for initial overflows and the independent overflow area in case cylinder overflow areas overflow.

In our most recent example, adding a record with key 209 causes ISAM to bump record 213 from track 1 into an overflow area, move 210 in its place, and insert 209 in the place vacated by 210. The following assumes a cylinder overflow area in track 9:

Track	Data Records on Cylinder 5	Comments
1	205 206 208 209 210	Prime Data Area
2	214 219 220 222 225	
3	226 227 230 236 unused	
…		
9	213	Overflow Area

The track index now becomes 210, with a pointer (not shown) to key 213 in the overflow area.



Reorganizing an Indexed File

Because a large number of records in overflow areas cause inefficient processing, an installation can use a program periodically to rewrite or reorganize the file. The program simply reads the records sequentially and writes them into another disk area. ISAM automatically follows its indexes for the input file and delivers the records sequentially from the prime and overflow areas. It stores all the output records sequentially in the new prime data area and automatically creates new indexes. At this time, the program may drop records coded for deletion.

PROCESSING DOS INDEXED SEQUENTIAL FILES

Since ISAM automatically handles indexes and overflow areas, little added programming effort is involved in the use of indexed files. There are four approaches to processing:

1 ***Load or Extend***. The initial creation of an ISAM file is known as loading. Once a file is loaded, you may extend it by storing higher-key records at the end of the file.

2 ***Adding Records***. New records have keys that do not currently exist on the file. You have to insert or add these records within the file.

3. ***Random Retrieval***. To update an ISAM file with data (such as sales and payments on customer records), you use the key to locate the master record randomly and rewrite the updated record.

4. ***Sequential Processing***. If you have many records to update and the new transactions are in sequence, you can sequentially read, change, and rewrite the ISAM master.

Load or Extend a DOS ISAM File

Loading a file creates it for the first time, and extending involves storing records at the end. Input records must be in ascending sequence by a predetermined key, and all keys must be unique. For load and extend, you code the usual OPEN and CLOSE to activate and deactivate the file. Figure 27-1 uses sequential input records to load an ISAM file named DISKIS. The new macros for this purpose are SETFL, WRITE, ENDFL, and DTFIS.

The prototypes for these macros are as follows:

Name	Operation	Operand
[label]	SETFL	filename
[label]	WRITE	filename, NEW_KEY
[label]	ENDFL	filename

Let's examine the imperative macros and the DTFIS file definition macro.

SETFL (**Set F**ile **L**oad Model. SETFL initializes an ISAM file by preformatting the last track of each track index. The operand references the DTFIS name of the ISAM file to be loaded. In Fig. 27–1, SETFL immediately follows the OPEN macro.

WRITE. The WRITE macro loads a record into the ISAM file. Operand 1 is your DTFIS filename, and operand 2 is the word NEWKEY. You store the key and data area in a work area (named ISAMOUT in Fig. 27–1). DTFIS knows this area through the entry WORKL=ISAMOUT. For the WRITE statement, ISAM checks that the new key is in ascending sequence. ISAM then transfers the key and data area to an I/O area (named IOARISAM in the figure and known to DTFIS by IOAREAL=IOARISAM).
Here ISAM constructs the count area:

ENDFL (**End F**ile **L**oad Model. After all records are written and before the CLOSE, ENDFL writes the last data block (if any), an end-of-file record, and any required index entries.

```
 1                PRINT ON,NODATA,NOGEN
 2 PROG20A        START
 3                BALR  3,0
 4                USING *,3
 5                OPEN  DISKSD,DISKIS
14                SETFL DISKIS                    SET ISAM LIMITS
20                TM    DISKISC,B'10011000'       ANY SETFL ERRORS?
21                BO    R10ERR                       YES - ERROR ROUTINE
22                GET   DISKSD,SDISKIN            GET 1ST SEQ'L RECORD

29 *                    M A I N  P R O C E S S I N G
30 A10LOOP       MVC   ISKEYNO,ACCTIN            SET UP KEY NUMBER
31               MVC   ISRECORD,SDISKIN          SET UP ISAM DISK RECORD
32               WRITE DISKIS,NEWKEY             WRITE ISAM RECORD
37               TM    DISKISC,B'11111110'       ANY WRITE ERRORS?
38               BO    R10ERR                       YES - ERROR ROUTINE
39               GET   DISKSD,SDISKIN            GET NEXT SEQ'L RECORD
45               B     A10LOOP                      NO  - CONTINUE

47 *                    E N D - O F - F I L E
48 A90END        ENDFL DISKIS                    END ISAM FILE LIMITS
60               TM    DISKISC,B'11000001'       ANY ENDFL ERRORS?
61               BO    R10ERR                       YES - ERROR ROUTINE
62               CLOSE DISKSD,DISKIS             NORMAL TERMINATION
71               EOJ

75 *                    D I S K  E R R O R  R O U T I N E S
76 R10ERR        EQU   *                         DISK ERROR
77 *              .                              RECOVERY ROUTINES
78 *              .
79               CLOSE DISKSD,DISKIS             ABNORMAL TERMINATION
88               EOJ
```

```
 92 *                D E C L A R A T I V E S
 93 DISKSD    DTFSD BLKSIZE=360,                SEQUENTIAL DISK INPUT    +
                    DEVADDR=SYS015,                                      +
                    EOFADDR=A90END,                                      +
                    DEVICE=3340,                                         +
                    IOAREA1=IOARSD1,                                     +
                    RECFORM=FIXBLK,                                      +
                    RECSIZE=90,                                          +
                    TYPEFLE=INPUT,                                       +
                    WORKA=YES
154 IOARSD1  DS     CL360                     SEQ'L DISK BUFFER-1

156 SDISKIN  DS     0CL90                     SEQ'L DISK INPUT AREA
157 ACCTIN   DS     CL06                      *    KEY
158          DS     CL84                      *    REST OF RECORD

160 DISKIS   DTFIS CYLOFL=1,                  INDEXED SEQ'L LOAD       +
                    DEVICE=3340,                                         +
                    DSKXTNT=2,                                           +
                    IOAREAL=IOARISAM,                                    +
                    IOROUT=LOAD,                                         +
                    KEYLEN=6,                                            +
                    KEYLOC=1,                                            +
                    NRECDS=3,                                            +
                    RECFORM=FIXBLK,                                      +
                    RECSIZE=90,                                          +
                    VERIFY=YES,                                          +
                    WORKL=ISAMOUT
209 IOARISAM DS     CL284                     ISAM BUFFER AREA

211 ISAMOUT  DS     0CL96                     ISAM WORKAREA
212 ISKEYNO  DS     CL06                      *    KEY LOCATION
213 ISRECORD DS     CL90                      *    DATA AREA

215          LTORG
216                =C'$$BOPEN '
217                =C'$$BSETFL'
218                =C'$$BENDFL'
219                =C'$$BCLOSE'
220                =A(DISKIS)
221                =A(DISKSD)
222                =A(SDISKIN)
223          END   PROG20A
```

Figure 27–1 Program: Loading a DOS ISAM File

The DTFIS Macro

The maximum length for an ISAM filename is 7 [characters]. In Fig. 27–1, the DTFIS entries for the file being loaded are as follows:

CYLOFL= 1 gives the number of tracks on each cylinder to be reserved for each cylinder overflow area (if any).

DEVICE= 3340 is the disk device containing the prime data area or overflow area.

DSKXTNT= 2 provides the number of extents that the file uses: one for each data extent and one for each index area and independent overflow area extent. The program in Fig. 27–1 has one extent for the prime data area and one for the cylinder index.

IOAREAL=IOARISAM provides the name of the ISAM I/O load area. The symbolic name, IOARISAM, references the OS buffer area. For loading blocked records, you calculate the field length as Count area (8) + key length (6) + block length (90 x 3) = 284

IOROUT=LOAD tells the assembler that the program is to load an ISAM file.

KEYLEN=6 gives the length of each record's key.

KEYLOC= 1 tells ISAM the starting location of the key in the record, where 1 is the first position.

NRECDS=3 provides the number of records per block.

RECFORM=FIXBLK indicates fixed, blocked record format.

RECSIZE =90 gives the length of each record.

VERIFY= YES tells the system to check the parity of each record as it is written.

WORKL=ISAMOUT gives the name of your load work area, which is a OS defined elsewhere in the program. For blocked records, you calculate the field length as
Key length (6) + data area (90 X 3) = 284 [ELB. I say 276]

For unblocked records, you would calculate the field length as
Count area (8) + key length + "sequence link field" (10) + record length.

Status Condition
On execution, ISAM macros may generate error conditions, which you may test. After each I/O operation, ISAM places its status in a one-byte field, named "filenameC". For example, if your DTFIS name is DISKIS, ISAM calls the status byte DISKISC. Following is a list of the 8 bits in filenameC tbat the system may set when loading an ISAM file:

BIT	LOAD STATUS ERROR CONDITION
0	Any uncorrectable disk error except wrong length record.
1	Wrong length record detected on output.
2	The prime data area is full.
3	SETFL has detected a full cylinder index.
4	SETFL has detected a full master index.
5	Duplicate record – the current key is the same as the one previously loaded.
6	Sequence error – the current key is lower than the one previously loaded.
7	The prime data area is full, and ENDFL has no place to store the end-of-file record.

The program in Fig. 27–1 uses TM operations to test DISKIS after execution of the macros SETFL, WRITE, and ENDFL. After SETFL, for example, TM tests whether bits 0, 3, and 4 are on. If any of the conditions exist, the program executes an error routine (not coded) where the program may isolate the error and issue an error message.

The job control commands also vary. First, the DLBL job entry for "codes" contains ISC, meaning indexed sequential create, and second, there is an EXTENT command for both the cylinder index and the data area.

Random Retrieval of an ISAM File
The main purpose of organizing a file as indexed sequential is to facilitate the random accessing of records. For this, there are a number of special coding requirements. The program in Fig. 27–2 randomly retrieves records in the file created in Figure 27–1. The program reads a file of modification records in random sequence, with changes to the ISAM master file. For each modification record, the program uses the account number (key) to locate the correct ISAM record, correct it, and then update the record on the ISAM file.

ISAM Macros for Random Retrieval
The new macros for random retrieval are:

Name	Operation	Operand
[label]	READ	filename, KEY
[label]	WAITF	filename
[label]	WRITE	filename, KEY

READ causes ISAM to access a required record from the file. Operand 1 contains the DTFIS filename, and operand 2 contains the word KEY. You store the key in the field referenced by the DTFIS entry KEYARG. In Fig. 27–2, KEYARG=KEYNO. For each modification record processed, the program transfers the account key number to KEYNO.

WAITF allows completion of a READ or WRITE operation before another is attempted. Since a random retrieval reads and rewrites the same record, you must ensure that the operation is finished. Code WAITF anywhere following a READ or WRITE and preceding the next READ or WRITE.

WRITE rewrites an ISAM record. Operand 1 is the DTFIS filename, and operand 2 is the word KEY, which refers to your entry in KEYARG.

The DTFIS Macro
In Fig. 27–2, the DTFIS entries for the random retrieval include these:

IOAREAR=IOARISAM provides the name of the ISAM I/O retrieval area. The symbolic name, IOARISAM, references the DS retrieval area for unblocked records. For blocked records, the buffer size is given by: Record length (including keys) x blocking factor

For unblocked records, the buffer size is given by:
Key length + "sequence link field" (10) + record length

TYPEFLE= RANDOM means that the system is to retrieve records randomly by key. Other entries are SEQNIL for sequential and RANSEQ for both random and sequential.

WORKR=ISAMOUT gives the name of your retrieval work area.

Status Condition
The status byte for add and retrieve is different from load. The following is a list of the 8 bits in finenameC that the system may set.

BIT ADD AND RETRIEVE STATUS ERROR CONDITION

0 Any uncorrectable disk error except wrong length record.

1 Wrong length record detected on output.

2 End–of–file during sequential retrieval (not an error).

3 The requested record is not in the file.

4 The ID given to SETFL for SEQNTL is outside the prime data limits.

5 Duplicate record – an attempt to add a record that already exists in the file.

6. The cylinder overflow area is full.

7 A retrieval operation is trying to process an overflow record.

```
 1          PRINT ON,NODATA,NOGEN
 3 PROG20B  START
 4          BALR  3,0
 5          USING *,3
 6          OPEN  FILEIN,DISKIS
15          GET   FILEIN,RECDIN             READ 1ST INPUT RECORD

22 *                M A I N  P R O C E S S I N G
23 A10LOOP  MVC   ISKEYNO,ACCTIN            SET UP KEY NUMBER
24          READ  DISKIS,KEY                READ ISAM RANDOMLY
29          TM    DISKISC,B'11010101'       ANY READ ERROR?
30          BO    R10ERR                        YES - ERROR ROUTINE
31          WAITF DISKIS                    COMPLETE READ OPERATION
36          MVC   ACCTDKO,ACCTIN            MOVE FIELDS
37          MVC   NAMEDKO,NAMEIN            *  TO DISK
38          MVC   ADDRDKO,ADDRIN            *  WORKAREA
39          PACK  BALNDKO,BALNIN            *
40          MVC   DATEDKO,DATEIN            *
42          WRITE DISKIS,KEY                WRITE NEW ISAM RECORD
47          TM    DISKISC,B'11000000'       ANY WRITE ERROR?
48          BO    R10ERR                    *  YES - ERROR ROUTINE
49          GET   FILEIN,RECDIN             READ NEXT INPUT RECORD
55          B     A10LOOP                   *  NO  - CONTINUE
57 *                E N D - O F - F I L E
58 A90END   CLOSE FILEIN,DISKIS             TERMINATION
67          EOJ

71 *                D I S K  E R R O R  R O U T I N E S
72 R10ERR   EQU   *                         DISK ERROR
73 *                                        RECOVERY ROUTINES
74          B     A90END

76 *                D E C L A R A T I V E S
77 FILEIN   DTFCD DEVADDR=SYSIPT,           INPUT FILE              +
                  IOAREA1=IOARIN1,                                  +
                  BLKSIZE=80,                                       +
                  DEVICE=2540,                                      +
                  EOFADDR=A90END,                                   +
                  TYPEFLE=INPUT,                                    +
                  WORKA=YES
101 IOARIN1 DS    CL80                      INPUT BUFFER 1

103 RECDIN  DS    0CL80                     INPUT AREA:
104 CODEIN  DS    CL02                      *  RECORD CODE '01'
105 ACCTIN  DS    CL06                      *  ACCOUNT NO.
106 NAMEIN  DS    CL20                      *  NAME
107 ADDRIN  DS    CL40                      *  ADDRESS
108 BALNIN  DS    ZL06'0000.00'             *  BALANCE
109 DATEIN  DS    CL06'DDMMYY'              *  DATE

111 DISKIS  DTFIS CYLOFL=1,                 ISAM RANDOM RETRIEVAL   +
                  DEVICE=3340,                                      +
                  DSKXTNT=2,                                        +
                  IOAREAR=IOARISAM,                                 +
                  IOROUT=RETRVE,                                    +
                  KEYARG=ISKEYNO,                                   +
                  KEYLEN=6,                                         +
                  KEYLOC=1,                                         +
                  NRECDS=3,                                         +
                  RECFORM=FIXBLK,                                   +
                  RECSIZE=90,                                       +
                  TYPEFLE=RANDOM,                                   +
                  VERIFY=YES,                                       +
                  WORKR=ISAMOUT
193 IOARISAM DS   CL270                     ISAM BUFFER AREA

195 ISAMOUT DS    0CL90                     ISAM WORKAREA:
196 ISKEYNO DS    CL06                      *  KEY AREA
197 ACCTDKO DS    CL06                      *  ACCOUNT NO.
198 NAMEDKO DS    CL20                      *  NAME
199 ADDRDKO DS    CL40                      *  ADDRESS
200 BALNDKO DS    PL04                      *  BALANCE
201 DATEDKO DS    CL06                      *  DATE
202         DC    CL14' '                   *  RESERVED

204         LTORG
205               =C'$$BOPEN '
206               =C'$$BCLOSE'
207               =A(FILEIN)
208               =A(RECDIN)
209               =A(DISKIS)
210         END   PROG20B
```

Figure 27–2 Program: Random retrieval of a DOS ISAM file

The program in Fig. 27–2 uses TM operations to test DISKIS after execution of the macros READ and WRITE. Once again, the program would isolate the error and issue a message.

Sequential Reading of an ISAM File
Sequential reading of an ISAM file involves the use of the SETL, GET, and ESETL macros. SETL (**Set L**ow) establishes the starting point of the first record to be processed. Its options include these:

- Set the starting point at the first record in the file:

```
SETL filename,BOF
```

- Set the starting point at the record with the key in the field defined by the DTFIS KEYARG entry: .

```
SETL filename,KEY
```

- Set the starting point at the first record within a specified group. For example, the KEYARG field could contain "B480000" to indicate all records with key beginning with B48:

```
SETL filename,GKEY
```

The ESETL macro terminates sequential mode and is coded as **`ESETL filename`**.

DTFIS entries include these:

IOAREAS=buffername for the name of the buffer area. You calculate the buffer size just as you do for random retrieval.

IOROUT=RETRVE to indicate sequential retrieval.

TYPEFLE=SEQNTL or **RANDOM** for sequential or random retrieval.

KEYLOC=n to indicate the first byte of the key in a record; if processing begins with a specified key or group of keys and records are blocked.

To delete a record, you may reserve a byte in the record and store a code in it. A practice is to use the first byte to match OS requirements. Subsequently, your program may test for the code when retrieving records and when reorganizing the file.

PROCESSING OS INDEXED SEQUENTIAL FILES
Processing ISAM files under OS is similar to DOS processing, except that QISAM (**Q**ueued **I**ndexed **S**equential **A**ccess **M**ethod) is used for sequential processing and BISAM (**B**asic **I**ndexed **S**equential **A**ccess **M**ethod) is used for random processing.

The Delete Flag
Under OS, the practice is to reserve the first byte of each record with a delete flag, defined with a blank when you create the file. You also code OPTCD=L in the DCB macro or the DD command. When you want to delete a record, store X'FF' in this byte. QISAM subsequently will not be able to retrieve the record. QISAM automatically drops the record when the file is reorganized. Let's examine some features of OS ISAM processing.

Load an ISAM File

The OS imperative macros concerned with loading an ISAM file are the conventional OPEN, PUT, and CLOSE. DCB entries are as follows:

DDNAME	Name of the data set.
DSORG	Set to "IS" for **I**ndexed **S**equential.
MACRF	(PM) for move mode or (PL) for locate mode.
BLKSIZE	Length of each block.
CYLDFL	Number of overflow tracks per cylinder.
KEYLEN	Length of the key area.
LRECL	Length of each record.
NTM	Number of tracks for the master index, if any.
OPTCD	Options required, such as MYLU in any sequence: M establishes a master index (or omit M). Y controls use of cylinder overflow areas. I controls use of an independent area. L is a delete flag to cause bypassing records with X'FF' in the first byte. U (for fixed length only) establishes the track index in main storage**.**
RECFM	Record format for fixed/variable and unblocked/blocked: F, FB, V, and VB.
RKP	Relative location of the first byte of the key field, where 0 is the first location. (For variable-length records, the value is 4 or greater.)

Sequential Retrieval and Update

Under OS, sequential retrieval and update involve the OPEN, SETL, GET, PUTX, ESETL, and CLOSE macros. Once the data has been created with standard labels, many DCB entries are no longer required. DDNAME and DSORG=IS are still used, and the following entries are available:

MACRF=(entry)	The entries are (GM) or (GL) for input (PM) or (PL) for output (GM,SK,PU) if read and rewrite in place, where S is use of SETL, K is key or key class, and PU is use of PUTX macro
EODAD=eofaddress	Used for input, if reading to end-of-file.
SYNAD=address	Requests optional error checking.

The SETL macro. SETL (**Set L**ow Address) establishes the first sequential record to be processed anywhere within the data set. The general format is the following:

Name	Operation	Operand
[label]	SETL	dcb–name, start–position, address

The start–position operand has the following options:

B Begin with the first record in the data set. (Omit operand 3 for B or BD.)

K Begin with the record with the key in the operand 3 address.

KC Begin with the first record of the ***key class*** in operand 3. A key class is any group of keys beginning with a common value, such as all keys **`H48xxxx`**. If the first record is "deleted," begin with the first non–deleted record.

I Begin with the record at the actual device address in operand 3.

BD, KD, KDH, KCD, and ID cause retrieval of only the data portion of a record. Here are some examples of SETL to set the first record in a file named DISKIS, using a 6–character key:

• Begin with the first record in the data set:

```
SETL DISKIS,B
```

• Begin with the record with the key 012644:

```
SETL DISKIS,K,KEYADD1
```

• Begin with the first record of the key class 012:

```
SETL DISKIS,KC,KEYADD2


KEYADD1  DC   C'012644'          6-character key
KEYADD2  DC   C'012',XL3'00'     3-character key class
```

The ESETL macro, used as ESETL DCB–name, terminates sequential retrieval. If there is more than one SETL, ESETL must precede each one.

The program in Fig. 27–3 reads an ISAM file sequentially and inserts a delete code in any record that is more than five years old. The TIME macro delivers the standard date from the communication region as packed **`OOyyddd+`**, and the date in the record (positions 26-28) is in the same format. The PUTX macro rewrites an obsolete record with a delete byte in the first position.

```
PROG20C  START
         SAVE  (14,12)
         BALR  3,0
         USING *,3
         ST    13,SAVEAREA+4
         LA    13,SAVEAREA

         OPEN  (ISFILE)
         SETL  ISFILE,B                 START 1ST RECORD OF DATA SET
         TIME
         ST    1,TODAY
         SP    TODAY,=P'5000'           CALC DATE 5 YEARS AGO
         GET   ISFILE                   GET 1ST RECORD

A10LOOP  CP    26(3,1),TODAY            5 YEARS OR OLDER?
         BNL   A20                      *    NO  - BYPASS
         MVI   0(1),X'FF'               *    YES - SET DELETE CODE
         PUTX  ISFILE                   *    RE-WRITE RECORD
A20      GET   ISFILE                   GET NEXT
         B     A10LOOP                  LOOP

A90EOF   ESETL ISFILE                   END-OF-FILE
         CLOSE (ISFILE)
         L     13,SAVEAREA+4
         RETURN (14,12)

SAVEAREA DS    18F
TODAY    DS    F                        TODAY'S DATE: 00YYDDD+
IOAREA   DS    CL100                    DISK IO AREA

ISFILE   DCB   DDNAME=INDEXDD,                                  +
               DSORG=IS,                                        +
               EODAD=A90EOF,                                    +
               MACRF=(GL,S,PU)

         LTORG
         END   PROG20C
```

Figure 27–3 Program: Sequential retrieval of an OS ISAM file

KEY POINTS

- The indexed system writes a key preceding each block of records. The key is that of the highest record in the block.
- The track index, cylinder index, and master index help the system locate records randomly.
- The track index is in track 0 of each cylinder of the file and contains the highest key number for each track of the cylinder.
- The cylinder index is on a separate cylinder and contains the key number of the highest record on the cylinder.
- The optional master index is recommended if the cylinder index exceeds four cylinders in size.
- The master index facilitates locating keys in the cylinder index, the cylinder index facilitates locating keys in the track index, and the track index facilitates locating the track containing the required record.
- ISAM creates a file sequentially in a prime data area. Subsequent additions of higher keys append to the end, and additions of lower keys cause records to bump into an overflow area.
- Cylinder overflow areas reserve tracks on a cylinder for all overflows in that cylinder. This method reduces disk access time.
- Independent overflow areas reserve separate cylinders for overflows from the entire file. This method helps if there is an uneven distribution of overflow records – that is, many overflow records in some cylinders and few or none in others.