



VSAM - COBOL

VSAM File Organisation	1
File Access Modes	2
In the ENVIRONMENT Division	5
In the PROCEDURE Division	6
1. The OPEN Statement	6
2. The READ Statement	7
3. The START Statement	9
4. The WRITE Statement	10
5. The REWRITE Statement	11
6. The DELETE Statement	13
7. The CLOSE Statement	13

Appendix - Status Key Values for VSAM files





VSAM - COBOL

VSAM File Organisation

There are three types of file organisation you can use with VSAM.

VSAM sequential file organisation -

Also referred to as VSAM ESDS (Entry Sequenced Data Set) organisation.

In such a data set the records are stored in the order in which they were entered. The order of the records is fixed. Records in sequential files can only be accessed (read or written) sequentially.

VSAM indexed file organisation -

Also referred to as VSAM KSDS (Key Sequence Data Set) organisation.

In a KSDS the records are ordered according to the collating sequence of a prime key field defined in each record. This key field consists of one or more consecutive characters within the records and uniquely identifies each record. A prime key for a record might be, for example, an employee number or an invoice number. In your COBOL program, you specify this key through the clause:

RECORD KEY IS data-name

where data-name is the name of the key field as you defined it in the record description entry in the Data Division. The data set can be accessed sequentially, i.e. in primary key sequence, or directly by specifying the key of the record required.

You can also specify one or more alternate keys to use for retrieving records. Using alternate keys you can access the file to read records in some sequence other than the prime key sequence. For example, you could access the file through employee department rather than through employee number. Alternate keys need not be unique, more than one record will be accessed, given a department number as a key. This is permitted if alternate keys are specified as allowing duplicates.

VSAM relative-record file organisation -

Also referred to as VSAM RRDS (Relative-Record Data Set) organisation.

A RRDS is a series of fixed length slots in storage into which you place records. The relative key identifies the record - the relative key being the relative record number of the slot that the record occupies.

File Access Modes

You can access records in VSAM KSDS and RRDS files in three ways, sequentially, randomly or dynamically. (Records in an ESDS file can be accessed sequentially).

1. Sequential Access -

For indexed files, records are accessed in the order of the key field selected (either primary or alternate).

For relative files, records are accessed in the order of the relative record numbers.

2. Random Access -

For indexed files, records are accessed according to the value that you place in the key field.

For relative files, records are accessed according to the value that you place in the relative key.

3. Dynamic Access

Dynamic access is a mixture of sequential and random access within the same program. Using dynamic access, you can write one program to perform both sequential and random processing, accessing some records in sequential orders and others by their keys. As this method allows the greatest flexibility it is usually coded.

Example

Suppose you had a KSDS of employee records and the hourly wage formed the record key. Also suppose your program was interested in those employees earning £7 and £9 per hour and those earning £15 per hour and above.

To do this, retrieve the first record randomly based on the key of 0700. Then begin reading sequentially until the wage field exceeds 0990. Then switch back to random read, this time based on a key of 1500 and then switch back to reading sequentially until you reach the end of the file.

In the ENVIRONMENT Division

For an ESDS:

```

SELECT file-name
  ASSIGN TO UT-AS-ddname
  ORGANIZATION IS SEQUENTIAL
  ACCESS MODE IS SEQUENTIAL
  FILE-STATUS IS data-name-1 data-name-2.

```

Notes:

- (1) The ORGANIZATION and ACCESS clauses are optional as the default is SEQUENTIAL.
- (2) The PASSWORD clause is not used within Boots.
- (3) The FILE STATUS clause monitors the execution of each input-output request for the file. This clause is optional but when it is specified, the system moves a value into data-name-1 after each input-output request that explicitly or implicitly refers to this file. This field is defined in the Data Division (although not on the File Section) as a two-byte alphanumeric item and must not be the object of an OCCURS DEPENDING ON clause.
- (4) Data-name-2 can be coded in VS COBOL II programs to access native VSAM return code information to the COBOL program. This field must be defined as a group item of 6 bytes in the Working-Storage Section or Linkage Section of the Data Division.

The first 2 bytes of data-name-2 contain the VSAM **return code** in binary notation.

It can take a value of 0, 8, or 12.

The next 2 bytes contain the VSAM **function code** in binary notation. The value for this code is defined as 1, 2, 3, 4, or 5.

The last 2 bytes of data-name-2 contain the VSAM **feedback code** again in binary. The code value is 0 through to 255.

The Function Code and Feedback Code are set only if the Return Code is set to non-zero. If they are set when the Return Code is zero, the contents of the fields are not reliable.

Example

```

SELECT VSAM-MASTER ASSIGN TO UT-AS-VSMAS
FILE STATUS IS STATUS-CODE VSAM-RETURN-CODE.

```

WORKING-STORAGE SECTION.

```

01 STATUS-CODE                PIC XX.
01 VSAM-RETURN-CODE.
   05 VSAM-R15-RETURN-CODE    PIC S99 COMP.
   05 VSAM-FUNCTION-CODE      PIC S9 COMP.
   05 VSAM-FEEDBACK-CODE     PIC S999 COMP.

```

In the ENVIRONMENT Division

For a KSDS:

```
SELECT file-name
  ASSIGN TO UT-ddname
  ORGANIZATION IS INDEXED
  ACCESS MODE IS SEQUENTIAL / RANDOM / DYNAMIC
  RECORD KEY IS data-name-1
  ALTERNATIVE RECORD KEY IS data-name-2
  WITH DUPLICATES
  FILE-STATUS IS data-name-3 data-name-4
```

Notes:

(1) The RECORD KEY clause specifies the data item within the record that is the prime record key for the indexed file. The value contained in this data item must be unique among records in the file. Data-name-1 must be described as an alphanumeric item within the record description. An IBM extension allows data-name-1 to be defined to be numeric, numeric-edited, alphanumeric-edited or alphabetical though the key is still treated as an alphanumeric item for the input and output statements for the named file.

(2) The same rules apply to the ALTERNATE RECORD KEY clause as in note 1. Data-name-2 must not be at the same position on the record as the primary index key or any other alternate record key field.

(3) If the DUPLICATE clause is specified, the values contained in the ALTERNATE RECORD KEY data item may be duplicated within any records in the file. In sequential access, the records with duplicate keys are retrieved in the order in which they were placed in the file. In random access, only the first record written of a series of records with duplicate keys can be retrieved.

Example

```
SELECT MASTER-FILE ASSIGN TO UT-KSDSMAST
  ORGANIZATION IS INDEXED
  ACCESS MODE IS DYNAMIC
  RECORD KEY IS MAT-EMP-NO
  ALTERNATE RECORD KEY IS MAST-EMP-DEPT
  WITH DUPLICATES
  FILE STATUS IS STATE-CODE VSAM-RETURN-CODE.
```

In the ENVIRONMENT Division

For RRDS:

```
SELECT file-name
      ASSIGN TO UT-ddname
      ORGANIZATION IS RELATIVE
      ACCESS MODE IS SEQUENTIAL / RANDOM / DYNAMIC
      RELATIVE KEY IS data-name-1
      FILE-STATUS IS data-name-2 data-name-3.
```

Notes :

- (1) The RELATIVE KEY clause identifies a data-name that specifies the relative record number for a specific logical record within the relative file. Data-name-1 must be defined as an unsigned integer data item and must not be included in the record description entry for the data set. That is the relative key is not part of the record.
- (2) When ACCESS MODE IS SEQUENTIAL the Relative Key data field need not be specified unless the START statement (see page 10) is to be used. When the START statement is used the contents of data-name-1 are used to determine where processing of the data set is to begin. If a value is put into the Relative Key field, and a START statement is not used, then the value is ignored and processing begins with the first record in the file.
- (3) For RANDOM or DYNAMIC access the Relative Key field must be specified and a value inserted as it is used to obtain the record with that relative record number.

In the Procedure Division

1. The *OPEN* Statement:

OPEN INPUT / OUTPUT / I-O / EXTEND file-name-1 file-name-2 ...

Notes:

- (1) INPUT permits opening the file for input operations.
- (2) OUTPUT permits opening the file for output operations. This is specified when a file is being created. If OUTPUT is specified for a file that contains records, the file will be replaced by the new data.
- (3) I-O permits opening the file for input and output operations. The I-O option may be specified only for the files assigned to direct access devices.
- (4) EXTEND permits opening the file for output operations. The EXTEND phrase is only allowed for sequential access files. When an OPEN EXTEND statement is executed, the file is prepared for the addition of records immediately following the last record in the file. (The record with the highest prime record key value is considered the last record.) Subsequent WRITE statements add records as if the file were opened OUTPUT.
- (5) After the OPEN statement you should check the "Status Code" for successful completion of the OPEN. If it has successfully executed then the Current Record Pointer is set to the first record in the data set. In the case of KSDS this will be the record with the lowest primary index key and for a RRDS it is set to 1. If the data set is empty then the Current Record Pointer will be set to indicate end of file in the case of a sequential read being executed.

2. The READ Statement

For a sequential READ

```
READ file-name NEXT RECORD
      INTO identifier-1
      AT END statement-1
END-READ
```

Notes:

- (1) The sequential READ must be used for all data sets in sequential access mode.
- (2) The NEXT RECORD is the next in the logical sequence of records. For a KSDS this is the ascending value of the current Key of Reference (i.e. the primary index key or the alternate index key being used). For RRDS the sequence is the ascending value of the Relative Record numbers for the records that exist in the data set. The word NEXT can be omitted for files that have sequential access mode.
- (3) Before the READ statement is executed, the Current Record Pointer must be set by a successful OPEN, START, or READ statement.
- (4) If the Current Record Pointer indicates that no next logical record exists, the following occur in the order specified:
 - (a) The value '10' is placed into the "Status-Code" associated with the file-name to indicate the AT END condition.
 - (b) If the AT END clause has been specified control is transferred to statement-1 in the clause.
- (5) If the FILE STATUS clause has been used for the data set then the "Status-Code" field is updated when the READ statement has been executed.



For a Direct READ:

```
READ    file-name RECORD
        INTO identifier-1
        KEY IS data-name-1
        INVALID KEY statement-1
END-READ
```

Notes:

(1) Direct retrieval is used for KSDS and RRDS with ACCESS MODE specified as RANDOM or DYNAMIC. As stated previously, DYNAMIC access allows both direct access and sequential access to the data set and because of this greater flexibility this is used at Boots.

(2) The KEY IS clause is specified only for a KSDS. Data-name-1 must identify a record key associated with file-name - either the prime record key or any alternate record key.

(3) The value of the key of the KSDS record required must be moved to the key field before the READ statement is issued. Then when the READ statement is executed it causes this value to be compared with the value of the corresponding key data item in the file records, until the first record having an equal value is found. The Current Record Pointer is then positioned to this record, which is then made available. If no match is found then the INVALID KEY condition exists. In this case a value of '23' is returned to the "Status CODE" field and control passes to statement-1.

(4) If the KEY IS phrase is not specified, the prime RECORD KEY or the last key field that was used by a READ becomes the key of reference for this request.

(5) For the RRDS, execution of the direct READ statement sets the Current Record Pointer to the record whose relative record number is contained in the RELATIVE KEY data item specified in the SELECT clause, and makes the record available. If the file does not contain such a record, the INVALID KEY condition exists. The KEY IS clause may not be specified for a RRDS.

3. The **START** Statement:

This statement provides a means of positioning the Current Record Pointer within a KSDS or RRDS for subsequent sequential record retrieval.

```
START file-name  
      KEY IS operand data-name-1  
      INVALID KEY statement-1  
END START
```

where operand is: =,>,< ,NOT<,>=,etc

Notes:

(1) When the KEY IS clause is specified, the Current Record Pointer is positioned at the logical record in the file whose key field satisfies the comparison. When the KEY IS clause is not specified, KEY IS EQUAL (to the prime record key or relative key) is implied.

(2) The comparison made during the execution of a START statement is between the current value in data-name-1 and the corresponding key field in the file's records.

(3) For a KSDS, data-name-1 can be any of the following:

- The prime RECORD KEY
- Any ALTERNATE RECORD KEY
- An alphanumeric data item which re-defines the record key for the file.

(4) When the START statement is successful, the RECORD KEY with which data-name-1 is associated becomes the key of reference for subsequent READ statements.

(5) For a RRDS, when the KEY IS clause is specified, data-name-1 must be the relative key.

(6) If the FILE STATUS clause is specified in the FILE-CONTROL entry then the associated "Status Code" is updated when the START statement is executed.

(7) If the comparison is not satisfied by any record in the file, an INVALID KEY condition exists and if specified statement-1 is executed.

4. The WRITE Statement

For an ESDS

```
WRITE record-name
      FROM identifier-1
END-WRITE
```

For a KSDS and a RRDS

```
WRITE record-name
      FROM identifier-1
      INVALID KEY statement-1
END-WRITE
```

Notes:

(1) For the WRITE statement to be successful the data set must have been OPENED for OUTPUT, I-O, or EXTEND.

(2) For a KSDS, before the WRITE statement is executed, the required value must be placed in the prime record key or alternate record key field. When ACCESS IS SEQUENTIAL, records must be released in ascending order of the record key values. When ACCESS IS RANDOM or ACCESS IS DYNAMIC, records may be released in any programmer-specified order.

(3) For a KSDS, an INVALID KEY condition is caused by any of the following:

- ACCESS IS SEQUENTIAL and the file is opened for OUTPUT, and the value of the prime record key is not greater than that of the previous record.
- The file is opened I-O and the value of the prime record key equals that of an already existing record.
- An attempt is made to write beyond the externally defined boundaries of the file.

When an INVALID KEY condition is recognised, WRITE statement execution is unsuccessful and the contents of the record are unaffected.

(4) For a RRDS, when ACCESS IS SEQUENTIAL, the first record released has relative record number 1, the second record released has a relative record number of 2 and so on. If ACCESS IS RANDOM or ACCESS IS DYNAMIC the RELATIVE KEY must contain the desired relative record number for the record before the WRITE statement is issued..

(5) The Current Record Pointer is not affected by the execution of a WRITE statement.

5. The REWRITE Statement:

This statement logically replaces an existing record in a direct-access file.

```
REWRITE  record-name-1
          FROM identifier-1
          INVALID KEY statement-1
END-REWRITE
```

Notes:

- (a) To use the REWRITE statement the data set must have been opened for I-O.
- (b) For files in the sequential access mode, the input/output statement executed for this file must be a successfully executed READ statement. When the REWRITE statement is executed, the record retrieved by the READ statement is replaced.
- (c) For a KSDS:
 - when the access mode is sequential, the record to be replaced is specified by the value contained in the prime record key. When the REWRITE statement is executed, this value must equal the value of the prime record key data item in the last record READ from this file.
 - when the access mode is random or dynamic, the record to be replaced is specified by the value contained in the prime record key.
 - values of alternate record keys in the rewritten record may differ from those in the record being replaced.
 - an INVALID KEY condition exists when:
 - the access mode is sequential and the value contained in the prime record key of the record to be replaced does not equal the value of the prime record key data item of the last retrieved record from the file, or
 - the value contained in the prime record key does not equal that of any record in the file, or
 - the value of an alternative record key data item for which "duplicates" is not specified is equal to that of a record already in the file.



(d) For a RRDS

- when the mode is sequential, the INVALID KEY clause may not be specified.
- when the access mode is random or dynamic, the record to be replaced is specified on in the RELATIVE KEY data item. If the file does not contain the record specified, an invalid key condition exists and, if specified, statement-1 is executed.

(e) The Current Record Pointer is not affected by the execution of the REWRITE statement.

6. The DELETE Statement:

This statement removes a record from a KSDS or RRDS file. For indexed files, the space is then immediately available for a new record. For relative files, the space is then available for new record with the same relative key value.

```
DELETE file-name RECORD
      INVALID KEY statement-1
END-DELETE
```

Notes:

- (a) To use the DELETE statement the data set must be opened in I-O mode.
- (b) After successful execution of a DELETE statement, the record is removed from the file and can no longer be accessed.
- (c) The Current Pointer Record is not affected by execution of the DELETE statement.
- (d) For a file in sequential access mode, the last previous input/output statement must be a successfully executed READ statement. When the DELETE statement is executed the system removes the record retrieved by the read statement. The INVALID KEY clause must not be specified for a file in this access mode.
- (e) For a file in random or dynamic access mode, when the DELETE statement is executed, the system removes the record identified by the contents of the prime record key for a KSDS, or the relative key field for a RRDS. If the file does not contain such a record, an INVALID key condition exists.

7. The CLOSE Statement

```
CLOSE file-name-1 WITH LOCK
```

Notes:

- (a) If the WITH LOCK clause is specified the compiler ensures that this file cannot be opened again during this execution of the object program.



Appendix A - Status Key Values for VSAM Files

Language Elements Changed from OS/VS COBOL

Figure 5 (page 2 of 2). Status Key Values - QSAM Files

VS COBOL II Release 3 Status Keys	OS/VS COBOL Status Keys	Meaning
91	91	VSAM password failure
92	92	Logic error
93	93	VSAM resource not available
94	94	No file position indicator for VSAM sequential rest
95	95	Invalid or incomplete VSAM file information
96	96	No file identification (No DD statement for this VSAM file)

Figure 6 (page 1 of 2). Status Key Values - QSAM Files

VS COBOL II Release 3 Status Keys	OS/VS COBOL Status Keys	Meaning
00	00	Successful completion
02	02	Duplicate key, and DUPLICATES specified. Successful completion
04	00	Wrong length record. Successful completion
05	00	Optional file not present. Successful completion
10	10	At END (no next logical record) Successful completion
14	(undefined)	On sequential READ for relative file, size of relative record number too large for relative key
20	20	Invalid key for a VSAM indexed or relative file
21	21	Invalid key for a VSAM indexed or relative file; sequence error
22	22	Invalid key for a VSAM indexed or relative file; duplicate key and duplicates not allowed
23	23	Invalid key for a VSAM indexed or relative file; no record found



Figure 6 (page 2 of 2). Status Key Values - QSAM Files

VS COBOL II Release 3 Status Keys	OS/VS COBOL Status Keys	Meaning
24	24	Invalid key for a VSAM indexed or relative file; attempt to write beyond file boundaries VS COBOL II only: for a WRITE to a relative file, size of relative record number too large for relative key
30	30	Permanent error
35	93 96	Nonoptional file not present
37	90	Attempt to open a file not on a mass storage device
39	95	Conflict of fixed file attributes; OPEN fails
41	92	OPEN attempted for a file in OPEN mode
42	92	OPEN attempted for a file not in OPEN mode
43	92	REWRITE attempted when last I/O statement was not READ or DELETE
46	92	Sequential READ attempted with no valid next record
47	92	READ attempted when file not in OPEN INPUT or I-O mode
48	92	WRITE attempted when file not in OPEN OUTPUT, I-O, or EXTEND mode
49	92	DELETE or REWRITE attempted when file not in OPEN I-O mode
90	90	Other errors with no further information
91	91	VSAM password failure
93	93	VSAM resource not available
94	94	Under CMPR2: No file position indicator for VSAM sequential request
95	95	Invalid or incomplete VSAM file information
96	96	No file identification (no DD statement for this VSAM file)
97	97	OPEN statement execution successful; file integrity verified



VSAM - COBOL Interface

This course consists of working through a student folder for the first unit only for the VSAM-Cobol material. A summary of the course which includes a video is attached. The student notes are self-explanatory - pages 1-4 outline how to use the material.

Notes:

Do make notes of questions you may have and these can be tackled after the course. Take photocopies of material you regard as relevant but please do not write on the student folder.

At Boots only key sequenced VSAM datasets (KSDS) are used. As a result, only the first unit is to be studied. Additional notes on Cobol-VSAM are included.

Related courses include: VSAM concepts and facilities, VSAM and Easytrieve Plus, MVS/ESA. Further details may be obtained by contacting the Training department.

Finally,

Please complete the attached evaluation form to help us review the quality of the courses we arrange.