ARQUIVOS VSAM

O VSAM é um método de acesso de alta performance para uso com dispositivos de acesso direto em ambiente de acesso virtual

O VSAM provê rápida pesquisa e armazenamento de dados, facilidade de uso incluindo comandos de JOB CONTROL simplificados, proteção contra acesso não autorizado, controle central de funções de manuseio de dados.

O VSAM suporta arquivos indexados, seqüenciais e acesso direto.

TIPOS DE ORGANIZAÇÃO

ORGANIZAÇÃO SEQÜENCIAL

Nesta organização, os registros são armazenados consecutivamente na seqüência em que são criados. Uma vez estabelecida, esta seqüência não pode ser alterada, podendo, portanto, o arquivo ser estendido. Os registros podem ser de tamanho fixo ou variável, não havendo KEYS (chaves).

ORGANIZAÇÃO INDEXADA

Nesta organização, cada registro tem uma ou mais chaves incluídas, cada uma das quais é associada a um índice. Cada índice provê um caminho lógico para o registro de dados, de acordo com o conteúdo das KEYS incluídas nos registros. Os registro podem ser de tamanho fixo ou variável.

Cada registro num arquivo indexado precisa ter uma KEY primária incluída. Quando são incluídos, atualizados ou deletados registros, eles são identificados unicamente pêlos valores de suas KEYS primárias. Assim, o valor de cada KEY primária precisa ser único e não pode ser alterado quando o registro é atualizado. Cada registro num arquivo indexado pode conter uma ou mais KEYS alternadas incluídas, possibilitando um caminho alternativo de acesso aos registros, os valores contidos numa KEY alternada não precisam ser únicos.

ORGANIZAÇÃO RELATIVA

Com esta organização, cada registro no arquivo é identificado pelo seu numero relativo de registro. O arquivo pode ser imaginado como uma série de áreas, cada uma das quais pode conter um registro lógico. Cada uma destas áreas é identificada por um número relativo de registro, através do qual é feito o armazenamento ou a pesquisa do registro.

TIPOS DE ARQUIVOS

O VSAM suporta três tipos de organização : Entry Sequence Data Set (ESDS), Key Sequence Data Set (KSDS) e Relative Record Data Set (RRDS). Seus equivalentes em COBOL, são, respectivamente : organização seqüencial, indexada e relativa.

CARACTERÍSTICAS DO ARQUIVO ESDS - ENTRY-SEQUENCED DATA SET

- ACESSO SEQÜENCIAL
- REGISTROS CARREGADOS NA ORDEM DE ENTRADA
- TODAS AS INSERÇÕES SÃO FEITAS NO FIM DO ARQUIVO
- AS ATUALIZAÇÕES PODEM SER FEITAS SEM ALTERAÇÃO NO TAMANHO DO REGISTRO
- NÃO PODE SER FEITAS EXCLUSÕES
- NÃO PRECISA SER RESERVADO FREE SPACE

CARACTERÍSTICAS DO ARQUIVO KSDS - KEY-SEQUENCED DATA SET

- ORDENADO SEQUENCIALMENTE POR CHAVE
- NÃO PODE HAVER CHAVE PRIMÁRIA DUPLICADA
- ACESSO PODE SER PELA CHAVE PRIMÁRIA OU CHAVES ALTERNADAS
- CHAVE TAMANHO FIXO
- REGISTROS PODEM SER INCLUIDOS, ALTERADOS E EXCLUIDOS
- PODE SER RESERVADO FREE SPACE

CARACTERÍSTICAS DO ARQUIVO RRDS - RELATIVE RECORD DATA SETS

- ACESSO AOS REGISTROS POR UM NUMERO RELATIVO
- NÃO UTILIZA INDICE
- NÃO ADMITE CHAVES ALTERNADAS
- SÓ PARA REGISTROS DE TAMANHO FIXO
- REGISTROS PODEM SER INCLUIDOS, ALTERADOS E EXCLUIDOS

ENVIRONMENT DIVISION - INPUT-OUTPUT SECTION

Como feito na definição dos arquivos seqüenciais, esta seção destina-se a configuração dos arquivos KSDS.

Neste parágrafo devemos especificar a cláusula SELECT para arquivos INDEXADOS (KSDS), da seguinte forma:

Formato:

```
SELECT nome-arquivo ASSIGN TO nome-externo
        ORGANIZATION IS tipo-de-organização
        ACCESS MODE IS tipo-de-acesso
        RECORD KEY IS key-principal
        [ALTERNATE RECORD KEY IS key-alternada[WITH DUPLICATES]]
        [FILE STATUS IS campo-status].
```

Regras:

- ***Nome-arquivo*** – sintaxe idêntica aos arquivos seqüenciais.
- ***Nome-externo*** - sintaxe idêntica aos arquivos seqüenciais.
- **ORGANIZATION:**
  - **INDEXED:** para arquivos seqüenciais indexados.
  - **RELATIVE:** para arquivos relativos.
- **ACCESS MODE:**
  - **INDEXED:** para arquivos seqüenciais indexados.
  - **SEQUENTIAL** – O arquivo será lido sequencialmente.
  - **RANDOM** – O arquivo será lido aleatoriamente.
  - **DYNAMIC** – O arquivo será lido e atualizado.

- **RECORD KEY:** Campo do registro escolhido para identificar a chave de gravação e acesso principal do arquivo. O arquivo só pode ter uma RECORD KEY. A variável referenciada como RECORD KEY deve obrigatoriamente ser declarada como parte do registro na File Section.

- **ALTERNATE RECORD KEY:** Qualquer outro campo do registro usado para pesquisa de registros. Pode-se acrescentar a esta cláusula a declaração **WITH DUPLICATES**, que indica que a pesquisa por esta chave pode conduzir a mais de um registro. O arquivo pode ter várias chaves alternativas. A chave alternativa deve ser usada com cuidado porque aumenta o processamento nas atualizações por manter mais índices no arquivo. A variável referenciada como ALTERNATE RECORD KEY deve obrigatoriamente ser declarada como parte do registro na File Section.

DATA DIVISION - FILE SECTION – FD (FILE DESCRIPTION)

Como já vimos, a FILE SECTION é usada para detalhar o conteúdo dos registros dos arquivos que o programa irá ler/gravar. Na INPUT-OUTPUT SECTION (ENVIRONMENT DIVISION), cada arquivo a ser tratado no programa tem uma instrução SELECT especificando e definindo um nome para o arquivo. Na FILE SECTION precisamos agora detalhar cada um destes arquivos. Isto é feito usando o parágrafo padrão FD – FILE DESCRIPTION.

Formato:

```
FD   nome-do-arquivo.
01     nome-do-registro.
         03   key-principal PIC
         [03       key-alternada-1 ... PIC]
```

A regra para definição da estrutura do arquivo segue o mesmo padrão dos arquivos seqüenciais. **Deve ser observado que é obrigatório a definição da key-principal e das key-alternadas no registro (caso existam).**

PROCEDURE DIVISION

OPEN

Formato:

```
OPEN [INPUT | OUTPUT | I-O] Nome-arquivo1 ...
```

Regras:

1. **INPUT:** Permite abrir o arquivo apenas para operações de leitura.

2. **OUTPUT:** Permite abrir o arquivo para operações de gravação. Esta operação pode ser especificada quando o arquivo estiver sendo criado. Um arquivo VSAM somente pode ser aberto omo output uma única vez.Esta opção não permite comandos de leitura no arquivo.

3. **I-O:** Permite abrir um arquivo para ambas as operações de leitura e gravação.

Exemplo:

```
         OPEN   INPUT   CADAGENCIA.
         OPEN   I-O   CADBANCOS,   CADFILIAIS.
```

READ

Formato:

```
READ nome-arquivo
         NEXT
         [INTO nome-area-working]
         [AT END instrução-imperativa]
         [NOT AT END instrução imperativa]
[END-READ].
```

Regras:

1. **NEXT**: Para arquivo VSAM Indexado a opção NEXT é obrigatória.

2. **INTO:** O registro lido é movido para uma área de trabalho, de acordo com as regras da instrução MOVE.

3. **AT END:** Se a condição AT END for especificada, a instrução-imperativa do após o AT END será executada.

4. **NOT AT END:** Se esta condição existir serão executadas as instruções após a leitura do registro.

Exemplo:

```
LER-ARQ-SEQENT.
PERFORM UNTIL WS-FIM = "N"
   READ SEQENT01 NEXT INTO WS-REGISTRO-SEQENT01
                AT END MOVE "S" TO WS-FIM
   END-READ
END-PERFORM.
```

WRITE

A instrução WRITE grava um novo registro num arquivo VSAM.

Formato:

```
WRITE nome-de-registro-1 [FROM identificador-1]
      [INVALID KEY instrução-imperativa-1]
[END-WRITE].
```

Regras:

1. O arquivo VSAM associado à instrução WRITE deve ser aberto no modo OUTPUT ou I-O.

2. **NOME-DE-REGISTRO-1**: Deve ser o nome de um registro lógico (nível 01) da FD na DATA DIVISION.

3. **FROM identificador-1**: Antes do WRITE, o conteúdo de identificador-1 é movido para nome-de-registro-1. Depois da execução com sucesso da instrução WRITE, o registro continua disponível no identificador-1.

4. **INVALID KEY**: Uma condição INVALID KEY ocorre quando:

- Já existe no arquivo um registro com a mesma RECORD KEY primaria do registro a ser gravado.

- Já existe no arquivo um registro com a mesma ALTERNATE RECORD KEY para a qual WITH DUPLICATES não foi especificada.

Exemplo:

```
WRITE REGISTRO-ARQUIVO FROM REGISTRO-AUXILIAR
      INVALID KEY
```

```
                PERFORM ROTINA-ERRO-GRAVAÇÃO
        END-WRITE.
```

ACESSO DIRETO EM ARQUIVOS VSAM

Formato:

```
READ nome-arquivo
        [INTO nome-area-working]

         [KEY IS [chave-principal | chave-alternada]]
         [INVALID KEY instrução-imperativa]
[END-READ]
```

Regras:

1. A chave-principal ou chave-alternada devem conter um valor antes de se executar o comando READ.

2. **INTO:** O registro lido é movido para uma área de trabalho, de acordo com as regras da instrução MOVE

3. **KEY IS:** O NOME-CHAVE deve identificar uma chave de registro associada ao NOME-ARQUIVO (chave primaria do registro ou qualquer chave alternada). Quando a opção KEY IS não é declarada a leitura é feita pela chave primaria do arquivo.

4. **INVALID KEY:** Quando a opção INVALID KEY for especificada a instrução será executada quando a chave de leitura, movida antes do comando READ, não for encontrada.

Exemplo:

```
SELECT  CADENT01   ASSIGN    TO  CADENT01
          ORGANIZATION     IS    INDEXED
          ACCESS  MODE     IS    RANDOM
          RECORD KEY       IS    CAD-RECKEY
          FILE STATUS      IS    WS-STATUS.
……..
MOVE   WS-RECKEY   TO   CAD-RECKEY
READ   CADENT01
        INVALID KEY PERFORM ROTINA-ARQUIVO-ERRO
END-READ.
```

ALTERAÇÃO/REGRAVAÇÃO DE ARQUIVOS VSAM

REWRITE

A instrução REWRITE regrava o conteúdo de um registro já existente em um arquivo VSAM.

Formato:

```
REWRITE nome-de-registro-1 [FROM identificador-1]
        [INVALID KEY instrução-imperativa-1]
[END-REWRITE].
```

Importante:

- **VSAM INDEXED FILE: O dado da RECORD KEY definida para o arquivo não pode ser alterado.** Os valores da ALTERNATE RECORD KEY no registro podem ser diferentes daquele no registro a ser atualizado. O sistema assegura que acessos posteriores ao registro possam ser baseados em qualquer das chaves do registro. Se uma condição INVALID KEY ocorrer é porque a execução da instrução REWRITE não obteve sucesso na operação de atualização. Nesta situação o registro no arquivo não foi afetado.

Regras:

1. O arquivo VSAM associado à instrução REWRITE deve ser aberto no modo I-O e um comando de leitura (READ KEY IS) deve ter sido executado antes do comando REWRITE.

2. **NOME-DE-REGISTRO-1**: Deve ser o nome de um registro lógico (nível 01) da FD na DATA DIVISION.

3. **FROM identificador-1**: Antes do REWRITE o conteúdo de identificador-1 é movido para nome-registro-1. Depois da execução com sucesso da instrução REWRITE, o registro continua disponível no identificador-1.

4. **INVALID KEY**: Uma condição INVALID KEY ocorre quando:

    - O modo de acesso é seqüencial e o valor contido na RECORD KEY primaria do registro a ser regravado não é igual ao valor da RECORD KEY primaria do último registro lido do arquivo.

    - O valor contido na ALTERNATE RECORD KEY (para a qual WITH DUPLICATES não foi especificado) é igual a uma chave já existente no arquivo.

Exemplo:

```
REWRITE  REGISTRO-ARQUIVO FROM REGISTRO-AUXILIAR
                  INVALID KEY
                  PERFORM ROTINA-ERRO-REGRAVAÇÃO
END-REWRITE.
```

EXCLUSÃO DE REGISTROS EM ARQUIVOS VSAM

DELETE

A instrução DELETE exclui um registro de um arquivo indexado ou relativo**.**

Formato:

```
DELETE nome-de-arquivo
        [INVALID KEY instrução-imperativa]
[END-DELETE] ]
```

Regras:

1. **NOME-DE-ARQUIVO:** Deve estar definido na DATA DIVISION e deve ser de um arquivo VSAM.
   - Nome-de-arquivo deve ser aberto como I-O.

2. **INVALID KEY:** Quando a instrução DELETE é executada, o VSAM tenta remover o registro identificado pelo conteúdo da RECORD KEY. Se o arquivo não contiver o registro, uma condição INVALID KEY irá ocorrer e instrução-imperativa será executada.

Exemplos:

```
MOVE ‘A001’ TO CODIGO-AGENCIA
DELETE CADAGENCIA
   INVALID KEY
            MOVE "N" TO WS-FLAG-DELETOU
   END-DELETE.
```

FILE STATUS

Campo com o código de retorno dos comandos executados sobre o arquivo. Devem ter formato PIC XX e os valores retornados estão na tabela abaixo:

STATUS	DESCRIÇÃO
'00'	'SUCCESSFUL COMPLETION'
'02'	'DUPLICATE KEY, NON UNIQ. ALT INDX'
'04'	'READ, WRONG LENGTH RECORD'
'05'	'OPEN, FILE NOT PRESENT'
'07'	'CLOSE OPTION INCOMPAT FILE DEVICE OPEN IMPLIES TAPE; TAPE NOT USED'
'10'	'END OF FILE'
'14'	'RRN > RELATIVE KEY DATA'
'20'	'INVALID KEY VSAM KSDS OR RRDS'
'21'	'SEQUENCE ERROR, ON WRITE OR CHANGING KEY ON REWRITE'
'22'	'DUPLICATE KEY'
'23'	'RECORD OR FILE NOT FOUND'
'24'	'BOUNDARY VIOLATION. WRITE PAST END OF KSDS RECORD. COBOL 370: REL: REC# TOO BIG. OUT OF SPACE ON KSDS/RRDS FILE'
'30'	'PERMANENT DATA ERROR. DATA CHECK, PARITY CHK, HARDW'
'34'	'BOUNDARY VIOLATION. WRITE PAST END OF ESDS RECORD OR NO SPACE TO ADD KSDS/RRDS RECORD. OUT OF SPACE ON SEQUENTIAL FILE'
'35'	'35' 'OPEN, FILE NOT PRESENT'
'37'	'OPEN MODE INCOMPAT WITH DEVICE'
'38'	'OPENING FILE CLOSED WITH LOCK'
'39'	'OPEN, FILE ATTRIB CONFLICTING'
'41'	'OPEN, FILE IS OPEN'
'42'	'CLOSE, FILE IS CLOSED'
'43'	'DELETE OR REWRITE & NO GOOD READ FIRST'
'44'	'BOUNDARY VIOLATION/REWRITE REC TOO BIG'
'46'	'SEQUENTIAL READ WITHOUT POSITIONING'
'47'	'READING FILE NOT OPEN AS INPUT/IO/EXTEND'
'48'	'WRITE WITHOUT OPEN IO'
'49'	'DELETE OR REWRITE WITHOUT OPEN IO'
'90'	'UNKNOWN'
'91'	'VSAM - PASSWORD FAILURE'
'92'	'LOGIC ERROR/OPENING AN OPEN FILE OR READING OUTPUT FILE OR WRITE INPUT FILE OR DEL/REW BUT NO PRIOR READ '
'93'	'VSAM - VIRTSTOR. RESOURCE NOT AVAILABLE'
'94'	'VSAM - SEQUENTIAL READ AFTER END OF FILE OR NO CURRENT REC POINTER FOR SEQ'
'95'	'VSAM - INVALID FILE INFORMATION OR OPEN OUTPUT (LOAD) WITH FILE THAT NEVER CONTAINED DATA'
'96'	'VSAM - MISSING DD STATEMENT IN JCL'
'97'	'VSAM - OPEN OK, FILE INTEGRITY VERIFIED FILE SHOULD BE OK'
OTHER	'UNKNOWN REASON'