Cobol VSAM

VSAM permet 3 types d'accès :

- Séquentiel
- Direct
- Dynamique

Un peu de vocabulaire :

- *Enregistrement logique :* unité de traitement
- *Control Interval (CI)* : unité de transfert physique en VSAM
- *Control Area (CA)* : un ensemble de CI. C'est l'unité d'allocation. Définir un fichier VSAM revient à allouer un nombre entier de CA.

IDCAMS

IDCAMS est un programme utilitaire permettant de maintenir un catalogue de fichier VSAM. **AMS** pour Access Method Service. Avec IDCAMS, on pourra définir un cluster, un aix, un path, supprimer un cluster.

ESDS : Entry Sequenced Data Set

L'ESDS en VSAM correspond au fichier séquentiel.

- Ajout à la fin du fichier seulement.
- L'accès est séquentiel.
- On ne peut pas supprimer d'enregistrement.
- Il est possible de modifier un enregistrement sans en changer la longueur.

Le `SELECT` pour un ESDS :

```
SELECT [OPTIONAL] fich-name ASSIGN TO AS-ddname
    ORGANIZATION IS SEQUENTIAL
    ACCESS MODE IS SEQUENTIAL
    .
```

Chargement du fichier

Le fichier doit être vide avant l'ouverture en OUTPUT sauf si le cluster a été défini avec `REUSE` (sinon, code 3000 à l'exécution).

```
OPEN OUTPUT fich-name
```

Le write crée un enregistrement à la suite des précédents.

```
WRITE masque
```

Lecture du fichier

Le fichier qui a été chargé doit être ouvert en INPUT

```
OPEN INPUT fich-name
```

La lecture d'un enregistrement se fait avec un `READ`

```
READ fich-name
    AT END MOVE "1" TO EOF
END-READ
```

Ajout dans le fichier

Le fichier chargé doit être ouvert en `EXTEND`

```
OPEN EXTEND fich-name
```

Un `WRITE` ajoutera un enregistrement à la fin.

Pour un fichier non chargé, l'ouverture en extend est possible que si `OPTIONAL` est spécifié lors du `SELECT`

Modification d'un enregistrement

Le fichier chargé doit être ouvert en `I-O`.

```
OPEN I-O fich-name
```

La modification se fait avec :

```
REWRITE masque
```

`REWRITE` modifie le dernier enregistrement qui a été lu. La longueur de l'enregistrement ne peut pas changer !

RRDS : Relative Record Data Set

- Les enregistrements sont identifiés par un RRN (Relative Record Number).
- Le RRN ne fait pas partie de l'enregistrement.
- Des trous sont possibles.
- Les enregistrements sont de longueur fixe.

Chargement du fichier

```
SELECT [OPTIONAL] fich ASSIGN TO dd-name
    ORGANIZATION IS RELATIVE
    ACCESS MODE IS {SEQUENTIAL | DYNAMIC | RANDOM}
    RELATIVE KEY IS id1
    .
```

- id1 doit être déclaré en WORKING-STORAGE SECTION.

WRITE

```
WRITE masque
    [INVALID KEY proc1]
    [NOT INVALID KEY proc2]
[END-WRITE]
```

- INVALID KEY si le RRN est invalide, ou déjà existant.

Chargement du fichier

- Le mode d'accès est sequential ou random.
- Le fichier vide doit être ouvert en output.
- `OPEN OUTPUT fich`
- En séquentiel : Le RRN est inutile, les enregistrements ont automatiquement les numéros 1, 2, 3...
- En random : Le RRN est obligatoire et doit être affecté avant le WRITE

Ouverture en extend

- Le fichier déjà chargé (sauf si optional) doit être ouvert en extend.
- `OPEN EXTEND fich`
- L'access mode doit être sequential.
- Un WRITE ajoute un enregistrement à la fin, le RRN associé sera incrémenté de 1.

Lecture du fichier

- Le fichier déjà chargé (sauf si optional) doit être ouvert en INPUT.
- `OPEN INPUT fich`
- Les 3 modes d'accès sont possibles.
- Le format du READ dépend du mode d'accès.

Accès séquentiel Les enregistrements sont obtenus en ordre croissant du RRN. Si un RRN est prévu, il est automatiquement affecté au numéro d'enregistrement.

```
READ fich [INTO]
    AT END ...
END-READ
```

Accès séquentiel avec START

- START se positionne sur un enregistrement
- Il ne réalise pas de lecture
- Le RRN doit être préalablement affecté à la valeur qui nous interesse
- En cas de réussite du start, il est possible de parcourir séquentiellement à partir de cette position.

```
START fich
    [KEY is {= | > | >= | NOT < } id1 ]
    [INVALID KEY proc1]
    [NOT INVALID KEY proc2]
[END-START]
```

Accès en aléatoire

- Le RRN doit être affectée avant le READ.

```
READ fich [INTO]
    [INVALID KEY proc1]
    [NOT INVALID KEY proc2]
[END-READ]
```

Accès en dynamique

- La lecture peut être directe ou séquentielle.
- Pratique pour se positionner avec un START, ou READ direct, puis lire séquentiellement.

```
READ fich next [INTO ...]
    AT END proc1
END-READ
```

Mise à jour du fichier

- Le fichier chargé doit être ouvert en I-O
- `OPEN I-O fich`
- Les 3 modes d'accès sont possibles
- Les 5 instructions sont possibles : READ, WRITE, REWRITE, DELETE, START, sauf WRITE en séquentiel.

DELETE

```
DELETE fich2
    [INVALID KEY proc1]
    [NOT INVALID KEY proc2]
[END-DELETE]
```

- En accès séquentiel, le DELETE supprime le dernier enregistrement lu, donc pas d'INVALID KEY.
- En accès aléatoire / dynamique, le DELETE supprime l'enregistrement correspondant à la valeur du RRN. INVALID KEY est souhaitable. Il faut donc prévoir la valeur du RRN.

REWRITE

```
REWRITE fich [FROM...]
    [INVALID KEY proc1]
    [NOT INVALID KEY proc2]
[END-REWRITE]
```

- En accès séquentiel, REWRITE modifie le dernier enregistrement lu, donc pas d'INVALID KEY.
- En accès aléatoire / dynamique, REWRITE modifie l'enregistrement correspondant à la valeur du RRN. INVALID KEY est souhaitable. Il faut donc prévoir la valeur du RRN.

KSDS : Keyed Sequential Data Set

- Un fichier à organisation séquentielle indexé possède un index défini à partir d'une clé primaire : la record key. Cette fois ci, la clé fait partie de l'enregistrement.

- Un KSDS peut également posséder un ou plusieurs index secondaires (AIX), définis sur des clés secondaire (à valeur unique ou non).
- Les 3 modes d'accès sont possibles

SELECT

```
SELECT [OPTIONAL fich ASSIGN TO dd-name
    ORGANIZATION IS {SEQUENTIAL | DYNAMIC | RANDOM}
    RECORD KEY IS id1
    [ALTERNATE RECORD KEY IS id2 [with duplicates]]
    .
```

Chargement du fichier

- Le mode d'accès est sequential ou random.
- Le fichier vide est ouvert en OUTPUT.
- `OPEN OUTPUT fich`
- SEQUENTIAL : Les valeurs de la record key doivent arriver en ordre strictement croissant sinon INVALID KEY.
- RANDOM : Ordre quelquonque, INVALID KEY si doublons.

Ouverture en EXTEND

- Le fichier déjà chargé (sauf si optional) est ouvert en extend.
- `OPEN EXTEND fich`
- Le mode d'accès doit être séquentiel.
- Un write ajoute un enregistrement à la fin, la record key doit donc être plus grande que la record key du dernier enregistrement sinon INVALID KEY.

Lecture

- Le fichier déjà chargé (sauf optional) est ouvert en INPUT.
- `OPEN INPUT fich`
- Les 3 modes d'accès sont possibles
- La lecture se fait par un READ dont le format dépend du mode d'accès.

Accès séquentiel

```
READ fich [NEXT] [INTO ...]
    AT END proc1
END-READ
```

Accès séquentiel avec START Le principe est le même que pour le RRDS. Sauf que c'est la record key qui doit être préalablement affecté.

Accès random Le principe est le même que pour le RRDS. Sauf que c'est la record key qui doit être préalablement affecté.

Accès dynamique Le principe est le même que pour le RRDS. Sauf que c'est la record key qui doit être préalablement affecté.

Mise à jour du fichier

- Le fichier déjà chargé doit être en I-O.
- `OPEN I-O fich`
- Les 3 modes d'accès sont possibles
- Les 5 instructions sont possibles : READ, WRITE, REWRITE, DELETE, START, sauf WRITE en séquentiel.
- LE DELETE et REWRITE suivent les mêmes règles que pour le RRDS.

AIX : Alternate Index

Un AIX permet d'accéder à des record par une autre clé que la primaire. La clé secondaire peut être unique ou non. Il faut le préciser dans l'IDCAMS.