



# *Support stagiaire*

## *COBOL MODULE 3 Gestion de fichiers*





# Sommaire

<b>LE FILE STATUS .....</b>	<b>6</b>
1. <b>DEFINITION.....</b>	<b>6</b>
<b>LES DECLARATIVES .....</b>	<b>8</b>
1. <b>DEFINITION.....</b>	<b>8</b>
2. <b>INSTRUCTIONS. ....</b>	<b>9</b>
<b>L'ORGANISATION SEQUENTIELLE INDEXEE.....</b>	<b>10</b>
1. <b>DEFINITION.....</b>	<b>10</b>
1.1. <b>ACCES REALISABLES SUR UN FICHIER SEQUENTIEL INDEXE .....</b>	<b>10</b>
1.2. <b>STRUCTURE D'UN FICHIER SEQUENTIEL INDEXE.....</b>	<b>11</b>
2. <b>MECANISME DE FONCTIONNEMENT D'UN FICHIER SEQUENTIEL INDEXE.....</b>	<b>14</b>
2.1. <b>PHASE D'ALLOCATION. ....</b>	<b>15</b>
2.2. <b>CHARGEMENT INITIAL.....</b>	<b>17</b>
2.3. <b>INSERTION (CAS STANDARD).....</b>	<b>18</b>
2.4. <b>INSERTION (CAS C.I SPLIT).....</b>	<b>19</b>
2.5. <b>INSERTION (CAS C.A SPLIT) .....</b>	<b>20</b>
3. <b>LE POINTEUR D'ARTICLE COURANT .....</b>	<b>21</b>
3.1. <b>UTILISATION.....</b>	<b>21</b>
3.2. <b>POSITIONNEMENT ET INCREMENTATION.....</b>	<b>21</b>
3.3. <b>DESTRUCTION DU POSITIONNEMENT.....</b>	<b>21</b>
4. <b>INSTRUCTIONS D' ACCES AUX FICHIERS SEQUENTIELS INDEXES .....</b>	<b>22</b>
4.1. <b>RECAPITULATIF DES OPERATIONS ENTREE-SORTIE. ....</b>	<b>26</b>
5. <b>LES CLES GENERIQUES OU CLES TRONQUEES .....</b>	<b>27</b>
5.1. <b>REDECOUPAGE DE LA CLE.....</b>	<b>27</b>
6. <b>LES CLES SECONDAIRES .....</b>	<b>29</b>
6.1. <b>DEFINITION. ....</b>	<b>29</b>
6.2. <b>SITUATION DU POINTEUR SELON LA DEMANDE D'ACCES SELECTIF OU DE POSITIONNEMENT REALISEE. ....</b>	<b>30</b>
6.3. <b>ELEMENTS UTILISES LORS DES ACCES .....</b>	<b>31</b>
6.4. <b>DEFINITION ET INSTRUCTIONS D' ACCES AUX FICHIERS SEQUENTIELS INDEXES PAR LE CHEMIN DES CLES SECONDAIRES .....</b>	<b>32</b>
6.5. <b>EXEMPLE DE LECTURE PAR LA CLE PRINCIPALE OU SECONDAIRE.....</b>	<b>36</b>
6.6. <b>MODIFICATION DES DONNEES.....</b>	<b>38</b>
<b>L'ORGANISATION RELATIVE .....</b>	<b>40</b>
1. <b>DEFINITION.....</b>	<b>40</b>
2. <b>ACCES AUX ENREGISTREMENTS. ....</b>	<b>41</b>
3. <b>GESTION DES DOUBLES.....</b>	<b>42</b>

3.1.	EXEMPLE DE GESTION DES DOUBLES .....	43
4.	GESTION DES ANNULATIONS .....	45
5.	DEFINITION ET INSTRUCTIONS D'ACCES AUX FICHIERS RELATIFS.....	47
	<b>EXERCICE CI CA.....</b>	<b>52</b>
1.	Exercice sur la mise à jour d'un fichier séquentiel indexé .....	52
	<b>TP CLÉS PRIMAIRES.....</b>	<b>57</b>
1.	Sujet.....	57
1.1.	ORGANIGRAMME.....	57
1.2.	LE TRAITEMENT.....	58
1.3.	LES FICHIERS .....	60
2.	LES FICHIERS AVANT LE TRAITEMENT .....	61
3.	LES EDITIONS.....	63
	<b>EXERCICE CLÉS SECONDAIRES.....</b>	<b>65</b>
1.	Exercice D'apPLICATION SUR LES cles secondaires.....	65
	<b>TP CLÉS SECONDAIRES .....</b>	<b>69</b>
1.	Sujet.....	69
1.1.	ORGANIGRAMME.....	69
1.2.	LE TRAITEMENT.....	69
1.3.	LES FICHIERS .....	71
2.	LES EDITIONS.....	72
	<b>TP ORGANISATION RELATIVE.....</b>	<b>73</b>
1.	SUJET .....	73
1.1.	ORGANIGRAMME.....	73
1.2.	TRAITEMENT.....	73
1.3.	LES FICHIERS .....	74
2.	LES EDITIONS.....	75

# LE FILE STATUS

---

## 1. DEFINITION.

---

Le file status est une zone qui est alimentée d'un code alphanumérique sur deux caractères par la méthode d'accès à chaque opération d'entrée-sortie sur le support magnétique auquel il est lié.

L'utilisation du file status permet donc de détecter des anomalies lors du traitement d'un fichier.

La clause file status est spécifiée dans la rubrique **FILE-CONTROL**.

exemple :     **FILE-CONTROL.**  
                  **SELECT Nf ASSIGN TO Nx**  
                  **FILE STATUS IS FS-fich.**

Le file status est décrit en **WORKING-STORAGE SECTION** et se décompose en deux éléments, **STATUS KEY 1** et **STATUS KEY 2**.

Le **STATUS KEY 1** (sur 1 caractère) donne une indication générale sur le déroulement de l'opération d'entrée-sortie.

- 0**     L'opération s'est correctement déroulée.
- 1**     Fin de fichier
- 2**     Clé invalide
- 3**     Erreur permanente
- 4**     Erreur logique
- 9**     Erreur système

Status	key	SIGNIFICATION
1	2	
0	0	L'opération s'est bien déroulée.
0	1	Après une lecture la valeur de la clé courante est identique à celle de l'article suivant.
0	2	Après une écriture ou une réécriture l'article crée un doublon pour au moins une clé secondaire pour laquelle les clés en doubles sont autorisées.
0	4	Après une lecture la longueur de l'article lu n'est pas conforme aux attributs définis pour ce fichier.
0	5	Après une ouverture, le fichier n'a pas été alloué avant l'exécution de l'ordre d'ouverture.
0	7	Erreur lors d'une fermeture sur un fichier bande.
1	0	Fin de fichier.
1	4	Erreur lors d'une lecture sur un fichier relatif.
2	1	Erreur de séquence lors d'un accès séquentiel sur un fichier séquentiel indexé.
2	2	Clé dupliquée après une écriture ou une réécriture (principale ou secondaire si clés en doubles non autorisées).
2	3	Clé non trouvée.
2	4	Tentative d'écriture en dehors des limites d'un fichier séquentiel indexé ou relatif.
3	0	Pas d'information disponible sur l'opération d'entrée-sortie.
3	4	Tentative de lecture en dehors des limites d'un fichier séquentiel.
3	5	Ouverture en mode entrée-sortie ou ajout sur un fichier non déclaré.
3	7	Ouverture tentée sur un fichier incompatible avec le mode d'ouverture.
3	8	Ouverture tentée sur un fichier fermé verrouillé.
3	9	Conflit entre les attributs du fichier externe et ceux spécifiés dans le programme.
4	0	Tentative d'ouverture d'un fichier déjà ouvert.
4	2	Tentative de fermeture d'un fichier fermé.
4	3	Tentative de lecture séquentielle à partir d'un pointeur détruit.
4	4	Tentative d'écriture ou réécriture d'un article de longueur différente.
4	6	Tentative de lecture séquentielle suite à une précédente anomalie en lecture.
4	7	Tentative de lecture ou de positionnement sur un fichier non ouvert en entrée ou en entrée-sortie.
4	8	Tentative d'écriture sur un fichier non ouvert en sortie, entrée-sortie ou ajout.
4	9	Tentative d'effacement ou de réécriture sur un fichier non ouvert en entrée-sortie.

# *LES DECLARATIVES*

---

## **1. DEFINITION**

---

L'usage des déclaratives spécifie des séquences d'instructions écrites par l'utilisateur pour le traitement des erreurs d'entrée-sortie qui s'ajoutent à celles du système.

Les procédures indiquées sont exécutées lorsque les locutions INVALID KEY ou AT END ne sont pas spécifiées.

Après l'exécution de la déclarative, le contrôle est rendu à l'instruction qui suit celle ayant provoqué le débranchement.

L'usage des déclaratives n'est pas limité aux fichiers organisés en séquentiel-indexé.

Il ne peut y avoir qu'une seule section pour un fichier donné.

---

## **NOTES**



**2. INSTRUCTIONS.**

PROCEDURE DIVISION.  
 DECLARATIVES.  
 nom SECTION.

USE AFTER STANDARD	[	EXCEPTION	PROCEDURE	[	Nom de fichier
		ERROR			INPUT
					OUTPUT .
					I-O
					EXTEND

Nom de paragraphe.  
 Instructions COBOL.  
 END DECLARATIVES.  
 PROG SECTION.  
 DEBUTPROG.

NOTES

# *L'ORGANISATION SEQUENTIELLE INDEXEE*

---

## 1. DEFINITION.

---

La méthode d'accès "séquentiel indexé" permet de lire les données stockées sur un fichier à partir d'**une information identifiant les enregistrements**.

Cette information s'appelle la **CLE D'ACCES**. Il s'agit d'une zone **faisant partie** de la description **de l'enregistrement** et permettant de l'identifier par rapport à tous les autres enregistrements du fichier. Le contenu de cette zone devra donc être **unique pour un enregistrement** par rapport aux autres enregistrements du fichier.

La clé d'accès qui peut être définie à n'importe quel endroit de l'enregistrement doit avoir une position de départ et **une longueur fixe** pour tous les enregistrements du fichier. Dans un enregistrement de longueur variable, la clé d'accès appartiendra toujours à la partie fixe.

### 1.1. ACCES REALISABLES SUR UN FICHIER SEQUENTIEL INDEXE

- SEQUENTIEL
- DIRECT
- DYNAMIQUE (SEQUENTIEL + INDEXE)

## 1.2. STRUCTURE D'UN FICHIER SEQUENTIEL INDEXE

La structure d'un fichier séquentiel indexé est constituée de deux éléments :

- la **partie INDEX**
- la **partie DATA**

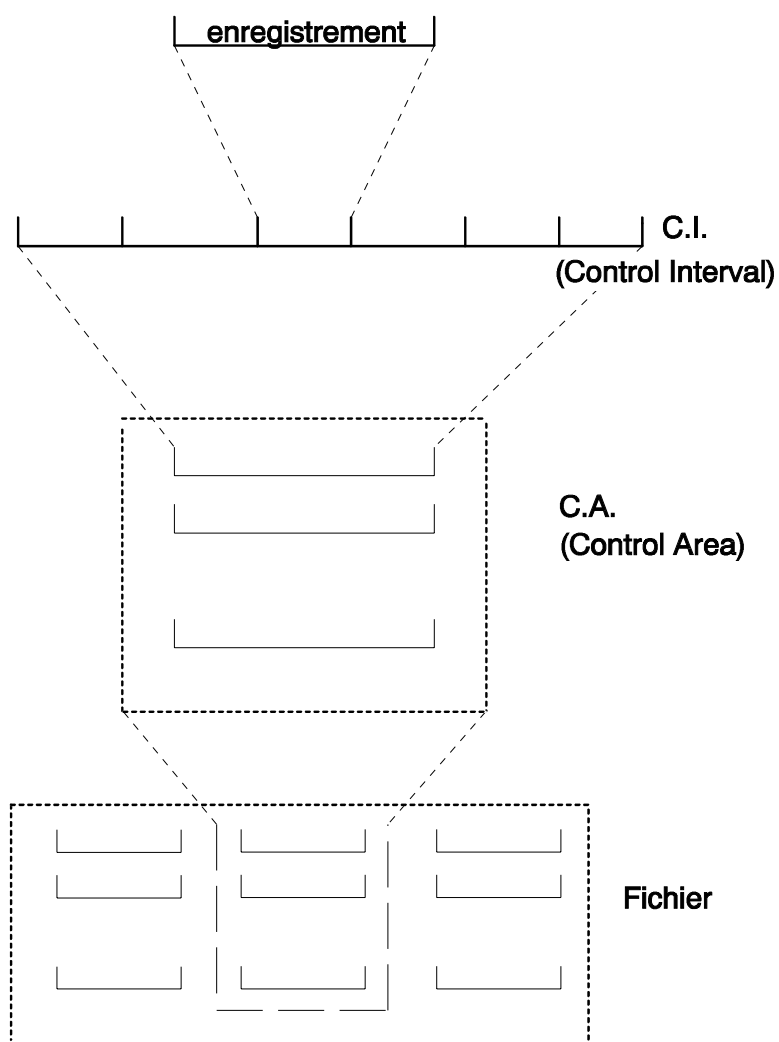
### LA PARTIE "INDEX"

Cet espace est utilisé par la méthode d'accès pour localiser les enregistrements dans la partie DATA à partir de la clé d'accès sur laquelle la demande d'E/S est effectuée.

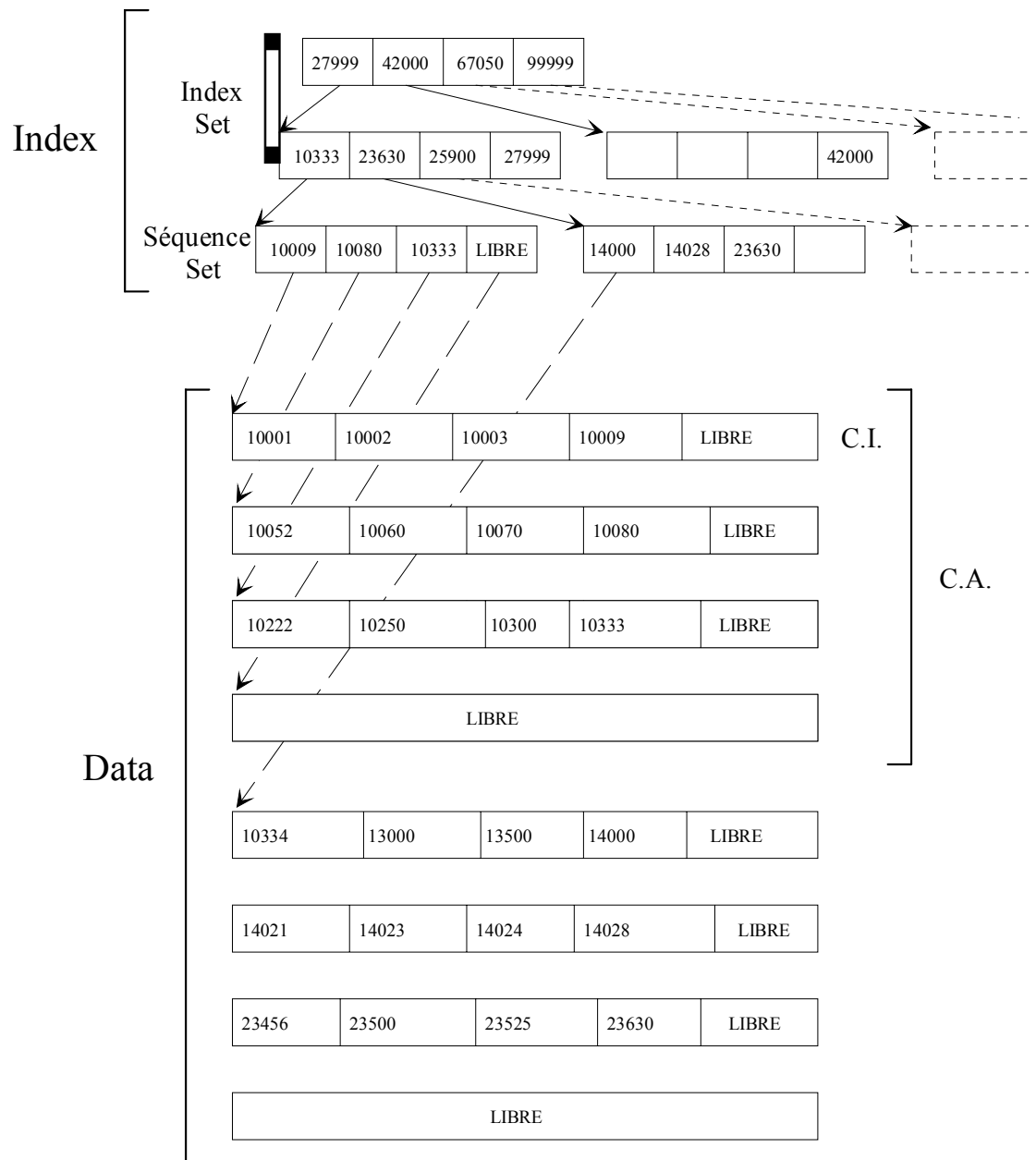
### LA PARTIE "DATA"

Cet espace contient les données telles qu'elles ont été formatées dans le programme.

NOTES



*Un CI est un regroupement de clés d'index d'enregistrements, un CA est un regroupement de CI (BULL).*



---

## 2. MECANISME DE FONCTIONNEMENT D'UN FICHER SEQUENTIEL INDEXE

---

Les différentes étapes

- Phase d'allocation.
- Phase de chargement initial.
- Phase de mise à jour.

Insertion "cas standard"

Insertion avec "CI SPLIT"

Insertion avec "CA SPLIT"

---

NOTES

**2.1. PHASE D'ALLOCATION.**

Réalisée à l'aide d'un utilitaire fourni par le constructeur cette étape permet de définir les caractéristiques du fichier et de réserver les ressources qui lui seront nécessaires.

Principaux paramètres d'allocation.

Nom du fichier

Volume

Espace à réserver

Taille des enregistrements

Position et longueur de la clé

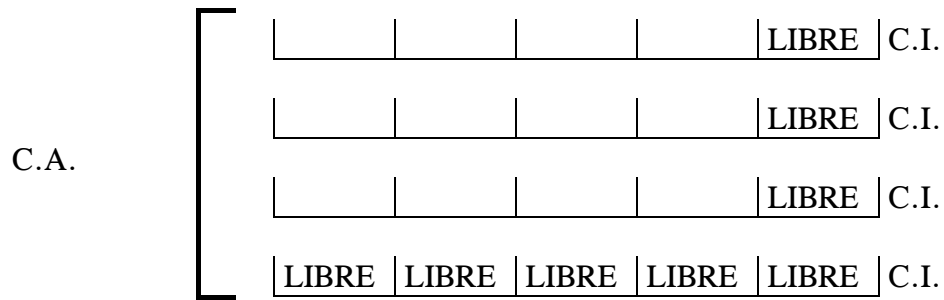
Espace libre par CI, CA

**NOTES**

2.1.1. EXEMPLE :

Les enregistrements de ce fichier sont de longueur fixe et la taille des CI permet d'en contenir 5.

Lors de la phase d'allocation, le paramètre d'espace libre demandait 20 % au niveau des CI et 25 % au niveau des CA.





## 2.2. CHARGEMENT INITIAL

Fichier séquentiel trié utilisé pour le chargement initial.

010	015	020	040	099	100	150	175	190	200
-----	-----	-----	-----	-----	-----	-----	-----	-----	-----

C.A. APRES CHARGEMENT	[	010	015	020	040	LIBRE	C.I.
		099	100	150	175	LIBRE	C.I.
		190	200			LIBRE	C.I.
		LIBRE	LIBRE	LIBRE	LIBRE	LIBRE	C.I.

INDEX	040	1	175	2	200	3		C.I.
-------	-----	---	-----	---	-----	---	--	------

NOTES

**2.3. INSERTION (CAS STANDARD)**

Lors de cette phase, la méthode d'accès utilise l'espace libre défini pour les C.I lors de la phase d'allocation.

INDEX	040 1	175 2	200 3		C.I.
C.A. AVANT	010	015	020	040	C.I.
	099	100	150	175	C.I.
	190	200			C.I.
					C.I.

INSERTION DES ENREGISTREMENTS

025
101

INDEX	040 1	175 2	200 3		C.I.
C.A. APRES	010	015	020	<b>025</b>	040   C.I.
	099	100	<b>101</b>	150	175   C.I.
	190	200			C.I.
					C.I.

**2.4. INSERTION (CAS C.I SPLIT)**

Lors de cette phase, la méthode d'accès utilise l'espace libre défini pour les C.A lors de la phase d'allocation.

INDEX	040 1	175 2	200 3			C.I.
C.A. AVANT	010	015	020	025	040	C.I.
	099	100	101	150	175	C.I.
	190	200				C.I.
						C.I.

INSERTION DE L'ENREGISTREMENT

| 012 |

INDEX	015 1	040 4	175 2	200 3	C.I.	
C.A. APRES	010	<b>012</b>	015			C.I.
	099	100	101	150	175	C.I.
	190	200				C.I.
	020	025	040			C.I.

2.5. INSERTION (CAS C.A SPLIT)

Lors de cette phase, la méthode d'accès doit étendre les ressources allouées au fichier.

INDEX	015 1	040 4	175 2	200 3		
C.A. AVANT	010	012	015			C.I.
	099	100	101	150	175	C.I.
	190	200				C.I.
	020	025	040			C.I.

INSERTION DE L'ENREGISTREMENT 168

040 1 | 200 2

INDEX

	015 1	040 4				101 1	175 3	200 2	
C.A. APRES	010	012	015			099	100	101	
						190	200		
						150	<b>168</b>	175	
	020	025	040						

NOTES

---

### 3. LE POINTEUR D'ARTICLE COURANT

---

#### 3.1. UTILISATION.

Ce pointeur est utilisé par l'instruction de lecture séquentielle et se trouve positionné sur l'enregistrement qui sera mis à la disposition de l'utilisateur lors de chaque demande d'accès de ce type.

#### 3.2. POSITIONNEMENT ET INCREMENTATION.

Trois instructions positionnent ce pointeur :

- L'ouverture
- La lecture sélective
- Le positionnement

Une 4ème utilise et incrémente ce positionnement :

- La lecture séquentielle

#### 3.3. DESTRUCTION DU POSITIONNEMENT.

Si lors de l'exécution d'une des instructions de positionnement, on se trouve en condition de clé invalide, le pointeur est perdu.

Aucune demande de lecture séquentielle ne pourra être satisfaite et le FILE STATUS sera à une valeur différente de zéro, tant qu'un repositionnement n'aura pas été effectué avec une des instructions appropriées.

---

**4. INSTRUCTIONS D' ACCES AUX FICHIERS SEQUENTIELS INDEXES**

---

**↪ DEFINITION**

FILE-CONTROL.  
SELECT Nf ASSIGN TO Nx  
ORGANIZATION INDEXED

ACCESS MODE [ SEQUENTIAL  
RANDOM  
DYNAMIC

RECORD KEY ref1  
FILE STATUS ref2.

**↪ INSTRUCTIONS****□ OUVERTURE DES FICHIERS**

OPEN [ INPUT  
OUTPUT Nf  
I-O  
EXTEND

---

**NOTES**

**☐ ACCES SELECTIF**

```
READ Nf [ INTO ref ] [ INVALID KEY... ]  
[ NOT INVALID KEY... ]  
[ END-READ ]
```

Exemple :

```
MOVE '15' TO Ref1  
READ Nf INVALID KEY...
```

Lecture de l'enregistrement dont la clé est égale à '15' si celui-ci existe sinon, exécution de la séquence INVALID KEY.

**☐ ACCES SEQUENTIEL**

```
READ Nf NEXT [ INTO ref ] [ AT END ... ]  
[ NOT AT END ... ]  
[ END-READ ]
```

Exemple :

```
READ Nf NEXT AT END ...
```

Lecture de l'enregistrement sur lequel est positionné le pointeur d'article courant.

## □ POSITIONNEMENT

```
START Nf KEY [ =
              >
              NOT < ou Ref [ INVALID KEY ]
              > =       [ NOT INVALID KEY ]
[END-START]
```

Exemple :

```
MOVE '15' TO Ref1
START Nf KEY > = Ref1 INVALID KEY ...
```

Positionne le pointeur sur l'enregistrement dont la clé est égale à '15' si celui-ci existe, sinon sur l'enregistrement de clé immédiatement supérieure.

## □ CREATION D'ENREGISTREMENT

```
WRITE Ne [ FROM ref ] [ INVALID KEY ... ]
          [ NOT INVALID KEY... ]
[END-WRITE]
```

Exemple :

```
MOVE '15' TO Ref1
WRITE Ne INVALID KEY ...
```

Création d'un enregistrement de clé '15' si cette clé n'existe pas dans le fichier sinon, exécution de la séquence INVALID KEY.

Ne nécessite pas une lecture préalable.



**❑ MODIFICATION D'ENREGISTREMENT**

```
REWRITE Ne [ FROM ref ] [ INVALID KEY ]  
[ NOT INVALID KEY ]  
[ END-REWRITE ]
```

Exemple :

```
MOVE '15' TO Ref1  
READ Nf INVALID KEY ...  
Modifications  
REWRITE Ne INVALID KEY ...
```

Modifie les données de l'enregistrement dont la clé est égale à '15' s'il existe sinon, exécution de la séquence INVALID KEY.

Ne permet pas la modification de la clé, si la clé doit être modifiée faire une annulation puis une création.

**❑ SUPPRESSION D'ENREGISTREMENT**

```
DELETE Nf [ INVALID KEY ]  
[ NOT INVALID KEY ]  
[ END-DELETE ]
```

Exemple :

```
MOVE '15' TO Ref1  
DELETE Nf INVALID KEY ...
```

Annule l'enregistrement de clé '15' s'il existe sinon, exécution de la séquence INVALID KEY.

Ne nécessite pas une lecture préalable.

**❑ FERMETURE DES FICHIERS**

```
CLOSE Nf [ WITH LOCK ]
```

**4.1. RECAPITULATIF DES OPERATIONS ENTREE-SORTIE.**

TYPE OPEN ACCES MODE	INPUT	OUTPUT	I-O	EXTEND
SEQUENTIAL	READ NEXT AT END START	WRITE	READ NEXT AT END START REWRITE DELETE	WRITE
RANDOM	READ INVALID KEY	WRITE	READ INVALID KEY WRITE REWRITE DELETE	Néant
DYNAMIC	READ INVALID KEY READ NEXT AT END START	WRITE	READ INVALID KEY READ NEXT AT END WRITE REWRITE DELETE START	Néant

NOTES

---

## 5. LES CLES GENERIQUES OU CLES TRONQUEES

---

Le principe repose sur les règles suivantes :

Redécoupage de la zone référencée dans la clause RECORD KEY et utilisation de cette ou de ces zones redécoupées **uniquement dans l'instruction START**.

**Seuls les caractères de droite de la clé peuvent être tronqués** et de ce fait le premier caractère de gauche de la clé tronquée doit correspondre au premier caractère de gauche de la clé d'origine.

### 5.1. REDECOUPAGE DE LA CLE

#### *DEFINITION SANS CLE TRONQUEE*

FD FICHER.

01 ENR.

02 CLE PIC X(13).

02 FILLER PIC X(67).

#### *DEFINITION AVEC CLE TRONQUEE*

FD FICHER.

01 ENR.

02 CLE.

04 P1A3.

06 P1 PIC X.

06 P2A3 PIC XX.

04 FILLER PIC X(10).

02 FILLER PIC X(67).

La zone CLE est celle référencée dans la clause RECORD KEY.

C'est elle qui sera utilisée lors d'un START sur clé complète.

Les zones P1A3 ou P1 pourront être utilisées pour un positionnement avec clé tronquée puisqu'elles correspondent aux caractères de gauche de la clé complète CLE.

Exemple de positionnement avec clé tronquée.

MOVE '2' TO P1.

START Nf KEY = P1 INVALID KEY ...      Positionnement sur le premier  
enregistrement dont la première position  
de la clé est égale à 2.

MOVE '192' TO P1A3.

START Nf KEY = P1A3 INVALID KEY ...      Positionnement sur le premier  
enregistrement dont les positions 1 à 3 sont  
égales à 192.

---

**6. LES CLES SECONDAIRES**

---

L'utilisation des clés secondaires permet à l'utilisateur d'extraire les enregistrements à partir d'une autre information que la clé principale.

**6.1. DEFINITION.**

Le fichier des clés secondaires peut être défini au moment de la phase d'allocation (UFAS, Bull), ou à n'importe quel moment de la vie du fichier principal (VSAM, IBM).

On ne peut créer plus de 15 fichiers secondaires. Leur gestion est identique à celle du fichier principal, toutefois la clé secondaire contrairement à la clé principale n'est pas obligatoirement unique.

Un fichier secondaire est ordonné logiquement d'après sa clé.

Il est impossible de consulter ou modifier les enregistrements d'un fichier secondaire, ses enregistrements n'établissant qu'une passerelle entre le fichier secondaire et le fichier principal.

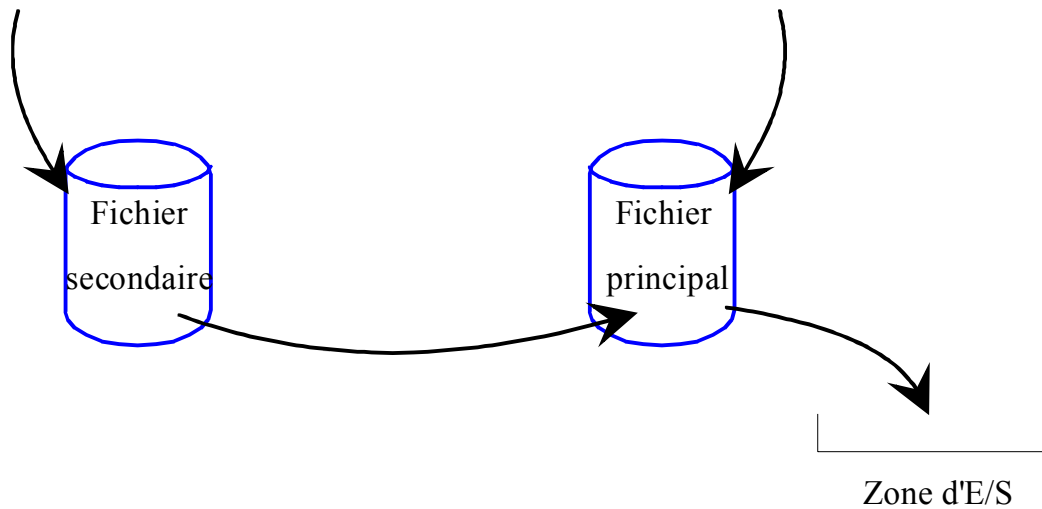
A chaque modification du fichier principal, les fichiers secondaires concernés sont mis à jour.

**6.2. SITUATION DU POINTEUR SELON LA DEMANDE D'ACCES SELECTIF OU DE POSITIONNEMENT REALISEE.**

**ACCES SELECTIF SUR CLE SECONDAIRE    ACCES SELECTIF SUR CLE PRINCIPALE**

Lecture d'un engt du fichier principal  
Positionnement sur fichier secondaire

Lecture d'un engt du fichier principal  
Positionnement sur fichier principal



Lors de la demande de lecture séquentielle, la séquence exécutée, sera différente selon le fichier sur lequel le pointeur est positionné au début de la demande d'E/S :

POSITIONNEMENT DU POINTEUR	ACTION		
Fichier principal	Lecture de l'engt F.Ppal pointé	Incrément PTR dans F.Ppal	
Fichier secondaire	Extraction clé pointée	Lecture engt correspondant dans F.Ppal	Incrément PTR dans F.Second.

**6.3. ELEMENTS UTILISES LORS DES ACCES**

	ORIGINE DES INFOS PERMETTANT LA LECTURE DE L'ENREGISTREMENT			POINTEUR	
	PTR	CLE Ppale	CLE Second	AVANT	APRES
READ sélectif ou START sur CLE PRINCIPALE (READ nf IK) ou (START nf IK)		<b>X</b>		<b>FP</b> <b>FS</b>	<b>FP</b>
READ sélectif ou START sur CLE SECONDAIRE (READ nf KEY IS DEST) ou (START nf KEY = DEST)			<b>X</b>	<b>FP</b> <b>FS</b>	<b>FS</b>
READ séquentiel (READ nf NEXT AT END)	<b>X</b>			<b>FP</b> <b>FS</b>	<b>FP</b> <b>FS</b>

NOTES

#### 6.4. DEFINITION ET INSTRUCTIONS D' ACCES AUX FICHIERS SEQUENTIELS INDEXES PAR LE CHEMIN DES CLES SECONDAIRES

##### ↳ DEFINITION

FILE-CONTROL.  
SELECT Nf ASSIGN TO Nx  
ORGANIZATION INDEXED

ACCESS MODE [ SEQUENTIAL  
RANDOM  
DYNAMIC

RECORD KEY ref1  
ALTERNATE RECORD KEY Ref3 WITH DUPLICATES  
FILE STATUS ref2.

NOTES



## ☞ INSTRUCTIONS

L'accès par le chemin des clés secondaires n'est possible que pour les opérations de consultation ou de positionnement sur le fichier.

Dans tous les cas l'utilisateur recevra un enregistrement du fichier principal.

### ☐ OUVERTURE DES FICHIERS

OPEN	[	INPUT	
		OUTPUT	Nf
		I-O	
		EXTEND	
	]		

### ☐ ACCES SELECTIF

READ Nf KEY IS Clé secondaire [ INVALID KEY .....]  
[ NOT INVALID KEY....]

[ END-READ ]

Appel à la méthode d'accès après avoir servi la zone définie en ALTERNATE RECORD KEY avec la valeur de la clé à rechercher dans le fichier de clés secondaires.

Exemple :

MOVE 'LC' to DEST

READ Nf KEY IS DEST INVALID KEY .....

Si la clé secondaire existe, lecture de l'enregistrement du fichier Principal correspondant à la 1ère clé principale associée à cette clé secondaire.

Sinon exécution de la séquence INVALID KEY.

❑ POSITIONNEMENT.

START Nf KEY	[	=		
		>	Clé secondaire [ INVALID KEY ... ]	
		NOT <	ou	[ NOT INVALID KEY ... ]
		> =		

[ END-START ]

Appel à la méthode d'accès après avoir servi la zone définie en ALTERNATE RECORD KEY avec la valeur de la clé secondaire sur laquelle un positionnement est demandé.

Exemple :

MOVE 'LC' to DEST

START Nf KEY = DEST INVALID KEY .....

Si la clé secondaire existe, le positionnement est effectué sur la 1ère clé principale associée à cette clé secondaire.

Sinon exécution de la séquence INVALID KEY.

### ↳ LES CLES GENERIQUES OU CLES TRONQUEES.

Le principe, identique à celui indiqué lors de la présentation des clés principales permet d'obtenir un enregistrement du fichier principal à partir d'une clé secondaire qui n'est pas connue en totalité.

Rappel des règles :

- Redécoupage de la clé secondaire dans la FD décrivant le fichier.
- Référence à la clé tronquée dans l'instruction START.

### □ ACCES SEQUENTIEL.

```
READ Nf NEXT [ AT END .....]  
              [ NOT AT END ...]  
[ END-READ ]
```

Le format du READ séquentiel est unique et la méthode d'accès utilise lors de son exécution le pointeur d'article courant.

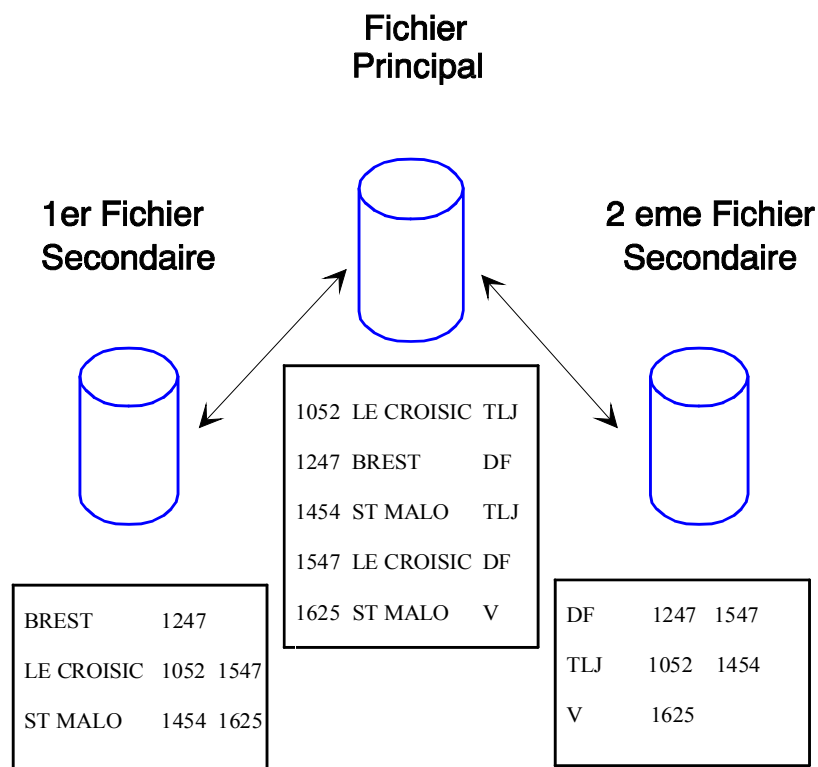
Ce pointeur unique se déplace du fichier principal vers le ou les fichiers secondaires lors de l'exécution des instructions de positionnement ou de lecture sélective.

### 6.5. EXEMPLE DE LECTURE PAR LA CLE PRINCIPALE OU SECONDAIRE

Le fichier présenté ici contient des informations concernant des trains.

Les données sont les suivantes :

- Numéro du train
- Destination
- Jour de circulation



1052	LC	TLJ	1247	B	DF
------	----	-----	------	---	----

B	1247
---	------

1454	SM	TLJ	1547	LC	DF
------	----	-----	------	----	----

LC	1052	1547
----	------	------

1625	SM	V			
------	----	---	--	--	--

SM	1454	1625
----	------	------

OPEN I-O FICHER  
 MOVE "1547" TO NUM  
 READ Nf INVALID KEY...  
 MOVE "SM" TO DEST  
 READ Nf KEY IS DEST IK...  
 READ Nf NEXT AT END  
 READ Nf NEXT AT END  
 (At end)  
 MOVE "1247" TO NUM  
 READ Nf INVALID KEY...

ARTICLE LU	POSITION	POINTEUR(*)
	1052	FP
1547 LC DF	1625	FP
1454 SM TLJ	1625	FS
1625 SM V	AT END	(FS)
1247 B DF	1454	FP

(\*) FP = fichier principal    FS = fichier secondaire

## 6.6. MODIFICATION DES DONNEES

Les créations, modifications ou suppressions de données sont effectuées sur le fichier principal et répercutées sur le ou les fichiers secondaires.

De ce fait il sera toujours nécessaire pour réaliser les instructions WRITE REWRITE et DELETE de connaître la CLE PRINCIPALE ou de l'acquérir par la voie des clés secondaires.

### 6.6.1. EXEMPLE DE MODIFICATION A PARTIR D'UNE CLE SECONDAIRE

1052	LC	TLJ	1247	B	DF	B	1247
1454	SM	TLJ	1547	LC	DF	LC	1052 1547
1625	SM	V				SM	1454 1625

Ce programme doit supprimer tous les enregistrements dont la clé secondaire est égale à 'LC'.

ARTICLE LU	POSITION	POINTEUR
OPEN I-O FICHER MOVE 'LC' TO DEST START Nf KEY = DEST INVALID KEY .... END-START READ Nf NEXT AT END... END-READ PERFORM UNTIL DEST NOT = 'LC' OR fin fichier DELETE Nf INVALID KEY... END-DELETE READ Nf NEXT AT END... END-READ END-PERFORM	1052	FP (*)
	1052	FS (*)
1052 LC TLJ	1547	FS (*)
	1547 LC DF	1454 FS (*)
puis 1454 SM TLJ	1625	FS (*)

- (\*) FP = fichier principal  
FS = fichier secondaire

# *L'ORGANISATION RELATIVE*

---

## **1. DEFINITION.**

---

L'organisation de fichier relative permet la gestion d'enregistrements de longueur FIXE selon les mêmes principes que ceux de gestion de table mais en faisant référence à un numéro d'enregistrement pour accéder aux données.

Toute la différence réside dans le fait que les données sont stockées sur support externe et implique donc des opérations d'entrée/sortie.

Le choix de cette organisation de fichier sera nécessaire lorsque la masse d'informations sera supérieure aux limites autorisées pour les tables mais surtout quand les données sont modifiables ou accessibles par plusieurs utilisateurs.

---

NOTES



---

## 2. ACCES AUX ENREGISTREMENTS.

---

L'accès aux données du fichier en lecture ou écriture pourra se faire de manière séquentielle ou sélective en faisant référence au numéro d'enregistrement à accéder.

**ACCES SEQUENTIEL :**

- **EN LECTURE :**

La méthode d'accès délivre les enregistrements dans l'ordre où ils sont rangés dans le fichier, du plus petit numéro d'enregistrement au plus grand, sans transmettre les enregistrements vides.

- **EN ECRITURE :**

Les données sont stockées séquentiellement depuis le premier numéro d'enregistrement, dans l'ordre où elles sont fournies à la méthode d'accès.

**ACCES SELECTIF :**

- **EN LECTURE :**

La méthode d'accès délivre les données stockées dans l'enregistrement dont le numéro est indiqué dans la relative key, si celui-ci est servi.

- **EN ECRITURE :**

L'enregistrement est stocké dans le numéro d'enregistrement indiqué dans la relative key, s'il n'est pas déjà utilisé.

---

**3. GESTION DES DOUBLES.**

---

La principale difficulté dans la manipulation des fichiers relatifs réside dans la gestion des données devant être stockées dans un numéro d'enregistrement déjà servi.

En effet, il est très fréquemment nécessaire de devoir utiliser un algorithme pour déterminer le numéro d'enregistrement où ranger les données. Si bien étudié soit-il, celui-ci présente tous les risques de délivrer même de manière exceptionnelle un numéro en double.

La méthode d'accès ne prenant pas en charge la gestion de ces doubles il appartient à l'utilisateur de rechercher une technique de gestion pour le stockage de ces enregistrements.

---

**NOTES**

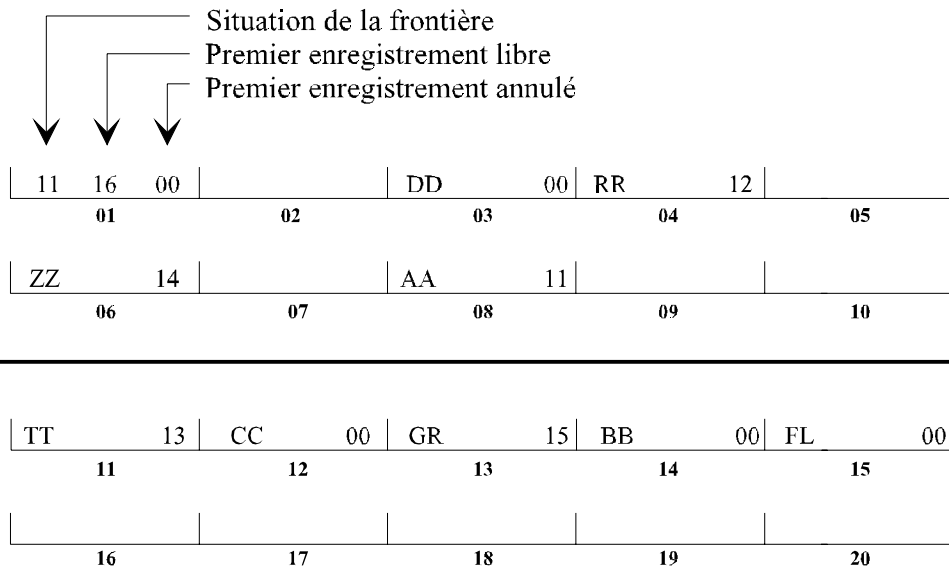
**3.1. EXEMPLE DE GESTION DES DOUBLES.**

## 3.1.1. STOCKAGE DES DONNEES

DONNEES		NUMERO D'ENRGT
ZZ	A	6
DD	L	3
AA	G	8
RR	O	4
TT	R	8
CC	I	4
GR	T	8
BB	H	6
FL	M	8
	E	

NOTES

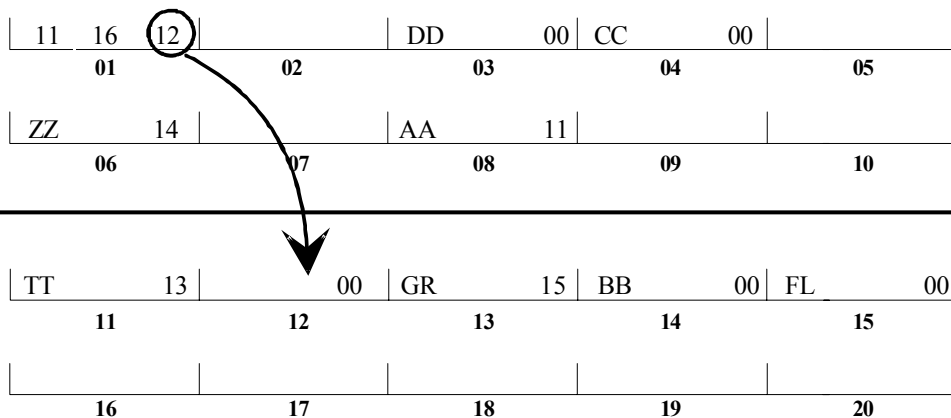
ENREGISTREMENT DE GESTION :



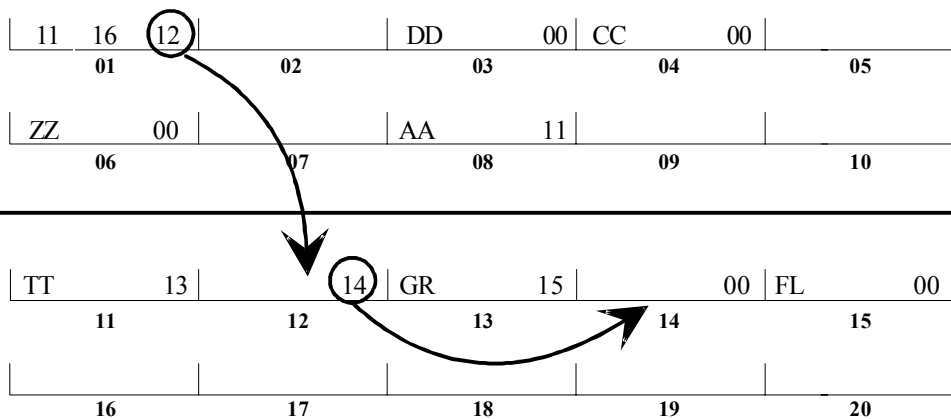
NOTES

**4. GESTION DES ANNULATIONS**

➤ Annulation de l'enregistrement RR (avant la 'frontière')

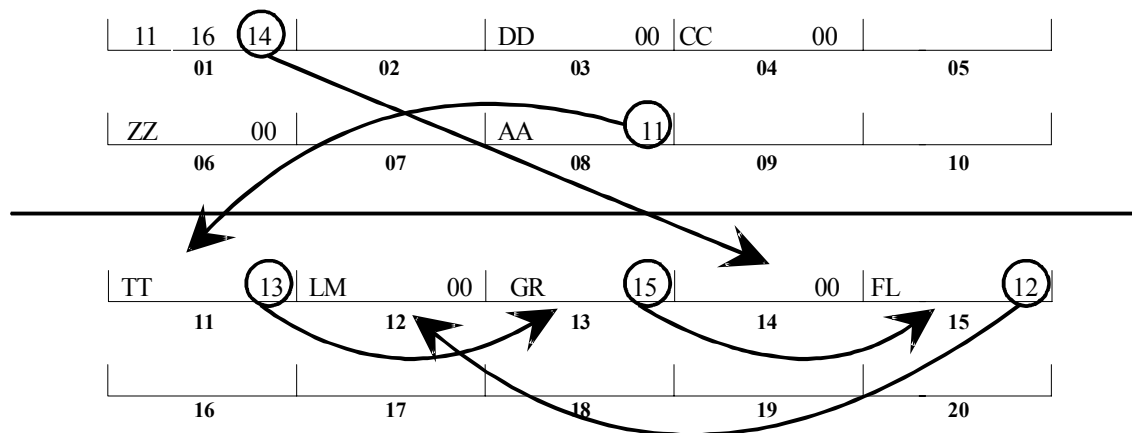


➤ Annulation de l'enregistrement BB (après la 'frontière')



- Création de l'enregistrement LM ( numéro 8 en sortie d'algorithme)

Récupération des enregistrements devenus disponibles lors des annulations



NOTES

---

**5. DEFINITION ET INSTRUCTIONS D'ACCES AUX FICHIERS RELATIFS**

---

↪ **DEFINITION**

FILE-CONTROL.

SELECT Nf ASSIGN TO Nx  
ORGANIZATION RELATIVE

ACCESS MODE [ SEQUENTIAL  
RANDOM  
DYNAMIC

RELATIVE KEY ref1  
FILE STATUS ref2.

---

NOTES

## ↵ INSTRUCTIONS

### □ OUVERTURE DES FICHIERS

OPEN	[	INPUT	Nf
		OUPUT	
		I-O	
		EXTEND	

### □ ACCES SELECTIF

```

READ Nf [ INTO wref ] [ INVALID KEY ... ]
          [ NOT INVALID KEY ... ]
[ END-READ ]

```

Exemple :

```

MOVE 5 TO Ref1
READ Nf INVALID KEY ...

```

Lecture des données stockées dans l'enregistrement numéro 5 si celui-ci est servi sinon, exécution de la séquence INVALID KEY.



**❑ ACCES SEQUENTIEL**

```

READ Nf NEXT [ INTO wref ]      [ AT END ... ]
                                [ NOT AT END ... ]
[END-READ ]

```

Exemple :

```

READ Nf NEXT AT END ...

```

Lecture des données stockées dans l'enregistrement sur lequel le pointeur d'article est positionné.

**❑ POSITIONNEMENT**

```

START Nf KEY [ =
              >
              NOT < ou Ref1 [ INVALID KEY... ]
              >=
              [ NOT INVALID KEY... ]
[ END-START ]

```

Exemple :

```

MOVE 5 TO Ref1

```

```

START Nf KEY > = Ref1 INVALID KEY ...

```

Positionne le pointeur sur l'enregistrement numéro 5 si celui-ci est servi sinon le positionnement a lieu sur l'enregistrement immédiatement servi.

**❑ CREATION D'ENREGISTREMENT**

```
WRITE  Ne  [ FROM  wref ] [ INVALID KEY ]
                               [ NOT INVALID KEY ]
[ END-WRITE ]
```

Exemple :

```
MOVE 5 TO Ref1
WRITE Ne INVALID KEY ...
```

Stocke les données dans l'enregistrement 5 si celui-ci est libre sinon, exécution de la séquence INVALID KEY.

**❑ MODIFICATION D'ENREGISTREMENT**

```
REWRITE Ne [ FROM  wref ] [ INVALID KEY ]
                               [ NOT INVALID KEY ]
[ END-REWRITE ]
```

Exemple :

```
MOVE 5 TO Ref1
READ Nf INVALID KEY ...
Modifications
REWRITE Ne INVALID KEY ..
```

Modifie les données stockées dans l'enregistrement 5 si celui-ci est servi sinon, exécution de la séquence INVALID KEY.

**❑ SUPPRESSION D'ENREGISTREMENT**

```
DELETE Nf [ INVALID KEY ]  
          [ NOT INVALID KEY ]  
[ END-DELETE ]
```

Exemple :

```
MOVE 5 TO Ref1  
DELETE Nf INVALID KEY ...
```

Annule l'enregistrement 5 si celui-ci est servi sinon, exécution de la séquence INVALID KEY.

**❑ FERMETURE DES FICHIERS**

```
CLOSE Nf [ WITH LOCK ]
```

# Exercice CI CA

## 1. EXERCICE SUR LA MISE À JOUR D'UN FICHIER SÉQUENTIEL INDEXÉ

A partir du fichier ci-dessous, issu du chargement initial, veuillez effectuer les différentes mises à jour indiquées :

INDEX	040	1	175	2	200	3		C.I.
C.A. INITIAL	010		015		020		040	C.I.
	099		100		150		175	C.I.
	190		200					C.I.
								C.I.

- CREATION 

041
-----

INDEX								C.I.
C.A.								C.I.
								C.I.
								C.I.
								C.I.

• **CREATION**      012

INDEX        C.I.

C.A.								C.I.
								C.I.
								C.I.
								C.I.

• **CREATION**      140

INDEX        C.I.

C.A.								C.I.
								C.I.
								C.I.
								C.I.

• CREATION 175

	INDEX		C.I.
C.A.	[		C.I.
			C.I.
			C.I.
			C.I.

• ANNULATION 100

	INDEX		C.I.
C.A.	[		C.I.
			C.I.
			C.I.
			C.I.

• CREATION      160

	INDEX		C.I.
C.A.	[		C.I.
			C.I.
			C.I.
			C.I.

• ANNULATION      020

	INDEX		C.I.
[		C.I.	
		C.I.	
		C.I.	
		C.I.	

• CREATION      **013**

INDEX      \_\_\_\_\_ C.I.

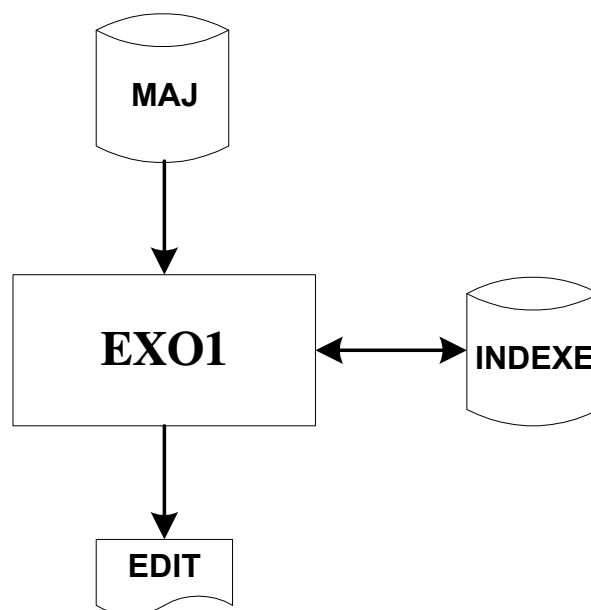
[	_____	C.I.
	_____	C.I.
	_____	C.I.
	_____	C.I.



---

**1. SUJET**

---

**1.1. ORGANIGRAMME****ATTENTION POUR LES TP.**

Utilisez les squelettes de programmes déjà créés :

EXO1 : c:\STAGE\GESTFIC\EXO1\EXO1.CBL

EXO2 : c:\STAGE\GESTFIC\EXO2\EXO2.CBL

EXO3 : c:\STAGE\GESTFIC\EXO3\EXO3.CBL

## 1.2. LE TRAITEMENT.

↪ Listage du fichier séquentiel indexé.

↪ Mise à jour du fichier séquentiel indexé à partir des enregistrements du fichier MAJ.

Le code mise à jour situé en première position du fichier MAJ peut avoir une valeur comprise entre 1 et 4 qui correspond aux modifications suivantes :

**CODE 1** : Création d'un enregistrement

description de l'enregistrement MAJ :

pos 2 à 13 : Prénom

pos 14 à 17 : Date (MM / JJ)

**CODE 2** : Annulation d'un enregistrement

description de l'enregistrement MAJ :

pos 2 à 13 : Prénom

pos 14 à 17 : Espaces

**CODE 3** : Modification d'un enregistrement (mise à jour de la date)

description de l'enregistrement MAJ :

pos 2 à 13 : Prénom

pos 14 à 17 : Date (MM / JJ)

**CODE 4** : Suppression globale

Supprimer sur le fichier séquentiel indexé tous les enregistrements dont la clé commence par le caractère indiqué en 2ème position du fichier mouvement.

description de l'enregistrement MAJ :

pos 2 : Caractère alphabétique

pos 3 à 17 : Espaces

↪ Listage du fichier séquentiel indexé après mise à jour.

**REMARQUES**

Il n'y a aucun contrôle à effectuer sur les articles du fichier mouvement.

Les listes sont à réaliser sur le fichier EDIT à raison de 4 prénoms par ligne.

**1.2.1. TRAITEMENT DES ANOMALIES**

Lorsqu'une anomalie est détectée au cours d'un accès au fichier séquentiel indexé (Clé en double, inexistante etc.), un enregistrement est créé sur le fichier EDIT selon le format suivant :

pos 1 à 10 : Libellé de l'E/S demandé (READ/WRITE/DELETE/REWRITE/START)  
pos 11 à 20 : Le texte "\* CLE---->"  
pos 21 à 32 : Valeur de la clé en anomalie  
pos 33 à 51 : Le texte " \* FILE STATUS---->"  
pos 52 à 53 : Valeur du FILE STATUS  
pos 54 à 74 : Libellé du type d'erreur (ABSENCE D'ARTICLE/CLE EN DOUBLE/AUTRE)  
pos 75 à 80 : Espaces

**1.3. LES FICHIERS****MAJ**

Organisation	Séquentielle
Longueur article	17 c.
Nom externe	MAJ

*Description article :*

Pos 1	- Code mise à jour
Pos 2 à 13	- Prénom
Pos 14 à 17	- Date

**INDEXE**

Organisation	Indexé
Longueur article	16 c.
Position de la clé	1 à 12
Nom externe	INDEXE

*Description article :*

Pos 1 à 12	- Prénom
Pos 13 à 16	- Date

**EDIT**

Organisation	Séquentielle
Longueur article	80 c.
Nom externe	EDIT

---

**2. LES FICHIERS AVANT LE TRAITEMENT**

---

**FICHER INDEXE EN ENTREE**

ADELE	0908
ADELINE	0908
ALAIN	0908
ALICE	1230
ALINE	0908
CASIMIR	0204
CHRISTELLE	0724
CHRISTIAN	1112
CHRISTIANE	1112
CHRISTINE	0724
DAVID	1230
DIDIER	0523
FLORENT	0204
GERARD	1023
JACQUES	0725
JEROME	0930
LINE	0000
MICHEL	0929
MICHELE	0929
MICHELLE	0929
NICOLE	1206
OLIVE	0712
OLIVIER	0712
PASCAL	0517
PATRICE	0317
REGINE	0907
ROGER	1230
ROMAINE	0223
SERGE	0930
SOLANGE	0510
VERONIQUE	0204
YVES	0519

**FICHER MISE A JOUR**

4Y	
4M	
4Q	
2REGINE	
3LINE	9999
3NICOLE	4444
1NOEL	1225
1SYLVIE	1106
1GERARD	1023
2YVES	
3MONIQUE	6666

**3. LES EDITIONS****LISTAGE DU FICHIER INDEXE AVANT MISE A JOUR**

ADELE	0908	ADELINE	0908	ALAIN	0908	ALICE	1230
ALINE	0908	CASIMIR	0204	CHRISTELLE	0724	CHRISTIAN	1112
CHRISTIANE	1112	CHRISTINE	0724	DAVID	1230	DIDIER	0523
FLORENT	0204	GERARD	1023	JACQUES	0725	JEROME	0930
LINE	0000	MICHEL	0929	MICHELE	0929	MICHELLE	0929
NICOLE	1206	OLIVE	0712	OLIVIER	0712	PASCAL	0517
PATRICE	0317	REGINE	0907	ROGER	1230	ROMAINE	0223
SERGE	0930	SOLANGE	0510	VERONIQUE	0204	YVES	0519

**LISTAGE DES ANOMALIES DETECTEES LORS DE LA MISE A JOUR**

START	* CLE---> Q	* FILE STATUS----> 23	ABSENCE D'ARTICLE
WRITE	* CLE---> GERARD	* FILE STATUS----> 22	CLE EN DOUBLE
DELETE	* CLE---> YVES	* FILE STATUS----> 23	ABSENCE D'ARTICLE
READ	* CLE---> MONIQUE	* FILE STATUS----> 23	ABSENCE D'ARTICLE

**LISTAGE DU FICHER INDEXE APRES MISE MISE A JOUR**

ADELE	0908	ADELINE	0908	ALAIN	0908	ALICE	1230
ALINE	0908	CASIMIR	0204	CHRISTELLE	0724	CHRISTIAN	1112
CHRISTIANE	1112	CHRISTINE	0724	DAVID	1230	DIDIER	0523
FLORENT	0204	GERARD	1023	JACQUES	0725	JEROME	0930
LINE	9999	NICOLE	4444	NOEL	1225	OLIVE	0712
OLIVIER	0712	PASCAL	0517	PATRICE	0317	ROGER	1230
ROMAINE	0223	SERGE	0930	SOLANGE	0510	SYLVIE	1106
VERONIQUE	0204						



## Exercice clés secondaires

### 1. EXERCICE D'APPLICATION SUR LES CLES SECONDAIRES

---

A partir du fichier des trains décrit dans le support de cours (pages 34 et 35), complétez la partie en pointillés par le contenu de la zone de communication.

En cas d'anomalie précisez la valeur du File status et si cette anomalie est une clé invalide ou une fin de fichier.

**IDENTIFICATION DIVISION.  
PROGRAM-ID. EXERCICE.**

**ENVIRONMENT DIVISION.  
INPUT-OUTPUT SECTION.  
FILE-CONTROL.**

**SELECT FICH ASSIGN TO FICH  
ORGANIZATION IS INDEXED  
ACCES MODE IS DYNAMIC  
RECORD KEY IS NUM  
ALTERNATE RECORD KEY DEST WITH DUPLICATES  
FILE STATUS IS FS.**

**DATA DIVISION.  
FILE SECTION.**

**FD FICH.  
01 ENR.**

**05 NUM.**  
    **10 N1**                   **PIC 9(02).**  
    **10 N2**                   **PIC X(02).**  
**05 DEST.**  
    **10 D1**                   **PIC X.**  
    **10 D2**                   **PIC X.**  
**05 FREQ**                   **PIC X(03).**  
**05 DONNEES**               **PIC X(91).**

**WORKING-STORAGE SECTION.**

**01 FS**                       **PIC X(02).**

**PROCEDURE DIVISION.**

**OPEN I-O FICH.**

**MOVE "LC" TO DEST.**

**START FICH KEY = DEST INVALID KEY PERFORM ERREUR.**

**MOVE "1625" TO NUM.**

**READ FICH NEXT AT END PERFORM ERREUR.**

\*\*\*\*\* ENR = ..... IK AE FS =

**MOVE "1700" TO NUM.**

**MOVE "DK" TO DEST.**

**MOVE "TLJ" TO FREQ.**

**MOVE "DONNEES" TO DONNEES.**

**WRITE ENR INVALID KEY DISPLAY "WRITE" PERFORM ERREUR.**

**READ FICH NEXT AT END PERFORM ERREUR.**

\*\*\*\*\* ENR = ..... IK AE FS =

**MOVE "DK" TO DEST.**

**READ FICH NEXT AT END PERFORM ERREUR.**

\*\*\*\*\* ENR = ..... IK AE FS =

**MOVE "1625" TO NUM.**

**READ FICH INVALID KEY PERFORM ERREUR.**

\*\*\*\*\* ENR = ..... IK AE FS =

**READ FICH NEXT AT END PERFORM ERREUR.**

\*\*\*\*\* ENR = ..... IK AE FS =

**MOVE "B " TO DEST.**

**START FICH INVALID KEY PERFORM ERREUR.**

**READ FICH NEXT AT END PERFORM ERREUR.**

\*\*\*\*\* ENR = ..... IK AE FS =

**MOVE "1100" TO NUM**

**READ FICH INVALID KEY PERFORM ERREUR.**

\*\*\*\*\* ENR = ..... IK AE FS =

**READ FICH NEXT AT END PERFORM ERREUR.**

\*\*\*\*\* ENR = ..... IK AE FS =

**PROCEDURE DIVISION.**

**MOVE SPACE TO ENR.  
 OPEN INPUT FICH.  
 START FICH KEY = DEST INVALID KEY PERFORM ERREUR.  
 MOVE "1600" TO NUM.  
 READ FICH INVALID KEY PERFORM ERREUR.**

\*\*\*\*\* ENR = ..... IK AE FS =

**READ FICH NEXT AT END PERFORM ERREUR.**

\*\*\*\*\* ENR = ..... IK AE FS =

**MOVE "1550" TO NUM.  
 READ FICH NEXT AT END PERFORM ERREUR.**

\*\*\*\*\* ENR = ..... IK AE FS =

**START FICH KEY > DEST INVALID KEY PERFORM ERREUR.  
 READ FICH NEXT AT END PERFORM ERREUR.**

\*\*\*\*\* ENR = ..... IK AE FS =

**MOVE "1247" TO NUM.  
 READ FICH NEXT AT END PERFORM ERREUR.**

\*\*\*\*\* ENR = ..... IK AE FS =

**MOVE "1247" TO NUM.  
 READ FICH INVALID KEY PERFORM ERREUR.**

\*\*\*\*\* ENR = ..... IK AE FS =

**MOVE "54" TO N2.  
 READ FICH INVALID KEY PERFORM ERREUR.**

\*\*\*\*\* ENR = ..... IK AE FS =

**MOVE "L" TO DEST  
 ADD 2 TO N1.  
 READ FICH NEXT AT END PERFORM ERREUR.**

\*\*\*\*\* ENR = ..... IK AE FS =

**START FICH KEY > D1 INVALID KEY PERFORM ERREUR.  
 READ FICH NEXT AT END PERFORM ERREUR.**

\*\*\*\*\* ENR = ..... IK AE FS =

**MOVE LOW-VALUE TO DEST.  
 READ FICH NEXT AT END PERFORM ERREUR.**

\*\*\*\*\* ENR = ..... IK AE FS =

**MOVE SPACE TO ENR.  
 START FICH KEY > DEST INVALID KEY PERFORM ERREUR.  
 READ FICH NEXT AT END PERFORM ERREUR.**

\*\*\*\*\* ENR = ..... IK AE FS =

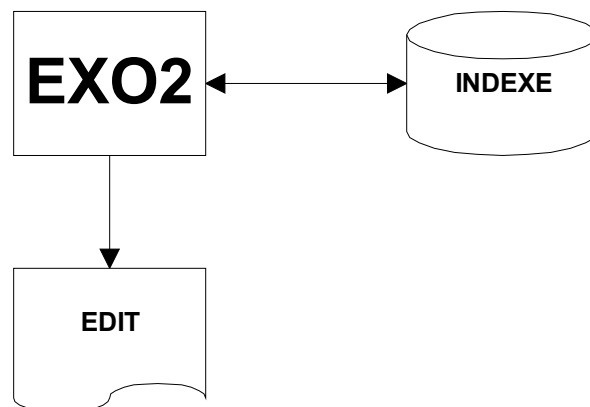
**MOVE "L" TO ENR.  
 START FICH KEY > DEST INVALID KEY PERFORM ERREUR.  
 READ FICH NEXT AT END PERFORM ERREUR.**

\*\*\*\*\* ENR = ..... IK AE FS =

---

**1. SUJET**

---

**1.1. ORGANIGRAMME.****1.2. LE TRAITEMENT.**

A partir du même fichier INDEXE que celui du programme EXO1, établir selon la forme présentée ci-dessous, les listes suivantes :

- 1** Les prénoms dont la fête est à souhaiter au mois de septembre
- 2** Les prénoms des jours pairs de chaque mois pour lesquels il y a plusieurs fêtes à souhaiter (seul les 5 premiers jours correspondants aux conditions seront édités).
- 3** Les autres fêtes à souhaiter le jour de la St Dominique et de la St Olivier

**↳ DESCRIPTION DE L'ETAT :****1 LISTE DES FETES A SOUHAITER EN SEPTEMBRE**

5 prénom1  
 9 prénom2 prénom3

**2 PLUSIEURS FETES A SOUHAITER**

JANVIER	8	prénom1	prénom2	
	14	prénom3	prénom4	prénom5
MARS	12	prénom6	prénom7	

**3 AUTRES FETES A SOUHAITER**

St Dominique : .....

St Olivier : .....

**REMARQUES**

- Il peut n'y avoir aucune fête à souhaiter certains jours ou pour la totalité de certains mois.
- Limiter les listes à 4 prénoms par jour.

### 1.3. LES FICHIERS

#### INDEXE

Organisation	Indexé
Longueur article	16 c.
Position de la clé	1 à 12
Nom externe	INDEXE

*Description article :*

Pos 1 à 12	- Prénom
Pos 13 à 16	- Date

#### EDIT

Organisation	Séquentielle
Longueur article	80 c.
Nom externe	EDIT

---

**2. LES EDITIONS**

---

**I - LISTE DES FETES A SOUHAITER EN SEPTEMBRE**

7	REGINE			
8	ADELE	ADELINE	ALAIN	ALINE
29	MICHEL	MICHELE	MICHELLE	
30	JEROME	SERGE		

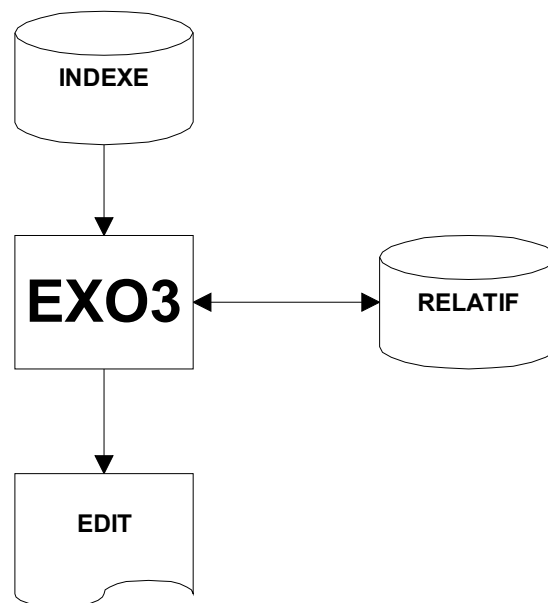
**II - PLUSIEURS FETES A SOUHAITER**

FEVRIER	4	CASIMIR	FLORENT	VERONIQUE	
JUILLET	12	OLIVE	OLIVIER		
	24	CHRISTELLE	CHRISTINE		
SEPTEMBRE	8	ADELE	ADELINE	ALAIN	ALINE
	30	JEROME	SERGE		
NOVEMBRE	12	CHRISTIAN	CHRISTIANE		
DECEMBRE	30	ALICE	DAVID	ROGER	

**III - AUTRES FETES A SOUHAITER**

ST DOMINIQUE	:	NEANT
ST OLIVIER	:	OLIVE



**1. SUJET****1.1. ORGANIGRAMME.****1.2. TRAITEMENT.**

A partir du même fichier INDEXE que celui du programme EXO1, exécutez les phases suivantes :

- 1** Chargement du fichier RELATIF à partir du fichier INDEXE en déterminant le numéro d'enregistrement d'après le rang dans l'alphabet de la première lettre du prénom.  
La gestion des doubles sera réalisée avec un enregistrement de gestion situé en première position.
- 2** Liste des articles du fichier relatif dans l'ordre de stockage.
- 3** Liste des articles du fichier relatif par ordre alphabétique.

**↳ PRESENTATION DES LISTES SUR LE FICHIER EDIT**

Pos 1 à 2	Numéro d'enregistrement
Pos 5 à 16	Prénom
Pos 20 à 21	Lien de chaînage

**1.3. LES FICHIERS****INDEXE**

Organisation	Séquentielle
Longueur article	16c
Nom externe	INDEXE

Description article :

Pos 1 à 12 - Libellé prénom  
 Pos 13 à 16 - Date

**RELATIF**

Organisation	Relative
Longueur article	50c
Nom externe	RELATIF

Description article :

Pos 1 à 12 - Libellé prénom  
 Pos 13 à 16 - Date  
 Pos 17 à 19 - Filler  
 Pos 20 à 24 - Chaînage  
 Pos 25 à 50 - Filler

**EDIT**

Organisation	Séquentielle
Longueur article	80c
Nom externe	EDIT

---

**2. LES EDITIONS**

---

**Liste par numero de poste**

NUM ENR.	1	PRENOM 00045	CHAINAGE	0
NUM ENR.	2	PRENOM ADELE	CHAINAGE	28
NUM ENR.	4	PRENOM CASIMIR	CHAINAGE	32
NUM ENR.	5	PRENOM DAVID	CHAINAGE	36
NUM ENR.	7	PRENOM FLORENT	CHAINAGE	0
NUM ENR.	8	PRENOM GERARD	CHAINAGE	0
NUM ENR.	11	PRENOM JACQUES	CHAINAGE	37
NUM ENR.	13	PRENOM LINE	CHAINAGE	0
NUM ENR.	14	PRENOM MICHEL	CHAINAGE	38
NUM ENR.	15	PRENOM NICOLE	CHAINAGE	0
NUM ENR.	16	PRENOM OLIVE	CHAINAGE	40
NUM ENR.	17	PRENOM PASCAL	CHAINAGE	41
NUM ENR.	19	PRENOM REGINE	CHAINAGE	42
NUM ENR.	20	PRENOM SERGE	CHAINAGE	44
NUM ENR.	23	PRENOM VERONIQUE	CHAINAGE	0
NUM ENR.	26	PRENOM YVES	CHAINAGE	0
NUM ENR.	28	PRENOM ADELINE	CHAINAGE	29
NUM ENR.	29	PRENOM ALAIN	CHAINAGE	30
NUM ENR.	30	PRENOM ALICE	CHAINAGE	31
NUM ENR.	31	PRENOM ALINE	CHAINAGE	0
NUM ENR.	32	PRENOM CHRISTELLE	CHAINAGE	33
NUM ENR.	33	PRENOM CHRISTIAN	CHAINAGE	34
NUM ENR.	34	PRENOM CHRISTIANE	CHAINAGE	35
NUM ENR.	35	PRENOM CHRISTINE	CHAINAGE	0
NUM ENR.	36	PRENOM DIDIER	CHAINAGE	0
NUM ENR.	37	PRENOM JEROME	CHAINAGE	0
NUM ENR.	38	PRENOM MICHELE	CHAINAGE	39
NUM ENR.	39	PRENOM MICHELLE	CHAINAGE	0
NUM ENR.	40	PRENOM OLIVIER	CHAINAGE	0
NUM ENR.	41	PRENOM PATRICE	CHAINAGE	0
NUM ENR.	42	PRENOM ROGER	CHAINAGE	43
NUM ENR.	43	PRENOM ROMAINE	CHAINAGE	0
NUM ENR.	44	PRENOM SOLANGE	CHAINAGE	0

## LISTE PAR ORDRE ALPHABETIQUE

NUM ENR.	2	PRENOM ADELE	CHAINAGE	28
NUM ENR.	28	PRENOM ADELINE	CHAINAGE	29
NUM ENR.	29	PRENOM ALAIN	CHAINAGE	30
NUM ENR.	30	PRENOM ALICE	CHAINAGE	31
NUM ENR.	31	PRENOM ALINE	CHAINAGE	0
NUM ENR.	4	PRENOM CASIMIR	CHAINAGE	32
NUM ENR.	32	PRENOM CHRISTELLE	CHAINAGE	33
NUM ENR.	33	PRENOM CHRISTIAN	CHAINAGE	34
NUM ENR.	34	PRENOM CHRISTIANE	CHAINAGE	35
NUM ENR.	35	PRENOM CHRISTINE	CHAINAGE	0
NUM ENR.	5	PRENOM DAVID	CHAINAGE	36
NUM ENR.	36	PRENOM DIDIER	CHAINAGE	0
NUM ENR.	7	PRENOM FLORENT	CHAINAGE	0
NUM ENR.	8	PRENOM GERARD	CHAINAGE	0
NUM ENR.	11	PRENOM JACQUES	CHAINAGE	37
NUM ENR.	37	PRENOM JEROME	CHAINAGE	0
NUM ENR.	13	PRENOM LINE	CHAINAGE	0
NUM ENR.	14	PRENOM MICHEL	CHAINAGE	38
NUM ENR.	38	PRENOM MICHELE	CHAINAGE	39
NUM ENR.	39	PRENOM MICHELLE	CHAINAGE	0
NUM ENR.	15	PRENOM NICOLE	CHAINAGE	0
NUM ENR.	16	PRENOM OLIVE	CHAINAGE	40
NUM ENR.	40	PRENOM OLIVIER	CHAINAGE	0
NUM ENR.	17	PRENOM PASCAL	CHAINAGE	41
NUM ENR.	41	PRENOM PATRICE	CHAINAGE	0
NUM ENR.	19	PRENOM REGINE	CHAINAGE	42
NUM ENR.	42	PRENOM ROGER	CHAINAGE	43
NUM ENR.	43	PRENOM ROMAINE	CHAINAGE	0
NUM ENR.	20	PRENOM SERGE	CHAINAGE	44
NUM ENR.	44	PRENOM SOLANGE	CHAINAGE	0
NUM ENR.	23	PRENOM VERONIQUE	CHAINAGE	0
NUM ENR.	26	PRENOM YVES	CHAINAGE	0