

Manuel de Descriptif Informatique
Fascicule D6.06 :
Document : D6.06.01

Utilisation des structures de données *table*

Résumé :

Ce document décrit les routines permettant d'utiliser les structures de données *table* décrites en [D4.02.05].

Table des matières

1 Généralités	3
1.1 Qu'est-ce qu'une table ? :	3
1.2 Quelques propriétés des tables	3
2 Liste des utilitaires pour les tables.....	4
3 Routines de gestion d'une table	4
3.1 Routine TBCRSO : Créer une nouvelle table.....	4
3.2 Routine TBAJPA : Ajouter des paramètres à la table.....	4
3.3 Routine TBEXIP : Tester l'existence et le type d'un paramètre	5
3.4 Routine TBFUTB : Fusionner plusieurs tables en une seule table	5
3.5 Détruire une table.....	6
3.6 Dupliquer une table	6
4 Routines d'écriture de valeurs dans une table	6
4.1 Routine TBAJLI : Ajouter une ligne à la table	6
4.2 Routine TBNULI : Permet de récupérer un numéro de ligne dans une table	7
4.3 Exemples	7
5 Routines de lecture de valeurs dans une table	8
5.1 Routine TBEXVE : Lecture des valeurs d'une colonne d'une table	8
5.2 Routine TBEXFO : Crée une fonction à partir d'une table.....	8
5.3 Routine TBLIVA : Lecture de la valeur d'une cellule.....	8
6 Routines de filtrage et de tri d'une table	9
6.1 Mécanisme de filtrage de lignes dans une table.....	9
6.1.1 Cas particulier des critères d'égalité (ou de non égalité) pour des nombres "flottants"	10
6.2 Routine TBEXTB : Filtrage et extraction d'une nouvelle table	11
6.3 Routine TBTRTB : Tri de la table.....	12
7 Routine TBIMPR : Impression d'une table.....	13
7.1 FORMAT : 'EXCEL'	13
7.2 FORMAT : 'TABLEAU'	13
7.3 FORMAT : 'MOT_CLE'	13
7.4 FORMAT 'EXCEL' avec pagination (définie ici par le paramètre 'NOEUD')	14
8 Routine TBEXLR : Transformation d'une table en une SD LISTR8.....	14
8.1 But de la routine	14
8.2 Interface de TBEXLR	14
8.3 Restrictions d'utilisation de TBEXLR.....	14
8.3.1 Exemples de tables diagonales par blocs	15
8.3.2 Exemples de tables non diagonales par blocs.....	15
8.4 Comment une table est-elle transformée en LISTR8 ?	15
8.4.1 étape 1 : on ne retient que les colonnes non vides de type 'I' ou 'R'	16
8.4.2 étape 2 : reconnaissance des « blocs » de la table « diagonale ».....	16
8.4.3 étape 3 : mise en vecteur des valeurs retenues, conversion des entiers en réels :	16
9 Exemple simple de création et d'exploitation d'une table.....	17

1 Généralités

1.1 Qu'est-ce qu'une table ? :

Une table est une structure de données de caractère informatique permettant de stocker un ensemble de valeurs entières, réelles, complexes ou chaînes de caractères.

Une table est comparable à une base de données EXCEL (version 5), c'est-à-dire qu'on peut la voir comme une liste de colonnes (le terme de "colonne" est ici pour faire le rapprochement avec une liste EXCEL ; cela ne veut pas dire que l'impression d'une table se fasse toujours en colonne (voir [§7])) en vis-à-vis. Chaque colonne a un nom de champ, que nous appelons **paramètre**, et contient des données similaires de type : I, R, C, K8, K16, K24 ou K32.

Exemple 1 : T1

NUME_ORDRE	INST	NOEUD	G_LOCAL
1	10.	N1	5.
1	10.	N2	6.
1	10.	N3	7.
1	10.	N4	8.
2	20.	N1	9.
2	20.	N2	9.
2	20.	N3	8.
2	20.	N4	8.
3	30.	N1	7.
3	30.	N2	6.
3	30.	N3	5.
3	30.	N4	4.

Exemple 2 : T2

ACTION	NUME_ORDRE	INST	NOEUD	DX	DY	MAILLE	SIXX
INTITULE 1	1	10.	N1	3.	5.		
INTITULE 1	1	10.	N2	6.	7.		
INTITULE 1	1	10.	N3	8.	9.		
INTITULE 1	2	20.	N1	11.	12.		
INTITULE 1	2	20.	N2	15.	13.		
INTITULE 1	2	20.	N3	19.	18.		
INTITULE 2	2	20.				MA1	-12.
INTITULE 2	2	20.				MA2	-14.

1.2 Quelques propriétés des tables

- Une table a un nombre limité de colonnes (ou de paramètres). Ces paramètres sont choisis par les développeurs des commandes créant des tables. Le nom d'un paramètre est une chaîne d'au plus 16 caractères.
- En revanche, le nombre de lignes d'une table est souvent « dynamique » : il dépend en général des choix de l'utilisateur : noeuds de dépouillement, instants de calculs, ...
- Les valeurs contenues dans une colonne d'une table sont toutes de même type FORTRAN : réels, complexes, entiers ou textes. On peut stocker sous forme « texte » dans une table des noms de SD Aster ou des noms d'objets JEVEUX ; par exemple des noms de fonctions.
- Une table est dite « pleine » lorsque toutes ses lignes contiennent des valeurs pour tous les paramètres de la table. La table t1 ci-dessus est pleine. Une table qui n'est pas pleine est dite « creuse » (table t2 ci-dessus).

- Les lignes d'une table sont naturellement ordonnées par leur ordre d'insertion dans la table (routine TBAJLI).
- Les colonnes d'une table sont naturellement ordonnées par l'ordre de déclaration de leurs paramètres (routine TBAJPA).
- Une table a au moins une ligne. Sur une ligne, il peut y avoir des cellules vides : le paramètre associé n'est pas affecté (table t2 ci-dessus).

2 Liste des utilitaires pour les tables

Fonction	nom	paragraphe
Créer une nouvelle table	TBCRSD	[§3.1]
Déclarer les paramètres	TBAJPA	[§3.2]
Récupérer l'existence et le type d'un paramètre	TBEXIP	[§3.3]
Fusionner plusieurs tables en une seule	TBFUTB	[§3.4]
Détruire une table	DETRSD	[§3.5]
Dupliquer une table	COPISD	[§3.6]
Construire une table en y ajoutant des lignes une à une	TBAJLI	[§4.1]
Récupérer le numéro d'une ligne dans une table	TBNULI	[§4.2]
Imprimer une table sur listing	TBIMPR	[§7]
Filtrer les lignes d'une table en imposant des critères sur un ou plusieurs paramètres pour créer une nouvelle table de taille plus petite	TBEXTB	[§6.2]
Trier les lignes d'une table en fonction de certains paramètres sélectionnés. Le résultat du tri est une table dont les lignes ont été réordonnées.	TBTRTB	[§6.3]
Recueillir dans un vecteur les valeurs correspondant à un paramètre donné	TBEXVE	[§5.1]
Créer une fonction à partir de 2 colonnes d'une table	TBEXFO	[§5.2]
Lire la valeur associée à un paramètre donné pour une ligne d'une table	TBLIVA	[§5.3]
Récupérer toutes les valeurs numériques d'une table dans une SD LISTR8	TBEXLR	[§8]

3 Routines de gestion d'une table

3.1 Routine TBCRSD : Créer une nouvelle table

TBCRSD	(nomtab, base)		
nomtab	in	K19	Nom de la nouvelle table à créer Si elle existe déjà, on la détruit + émission <A>
base	in	K1	Base de création de la table table ('G', 'V', ...)

3.2 Routine TBAJPA : Ajouter des paramètres à la table

TBAJPA	(nomtab, nbpar, nompar, typpar)		
nomtab	in	K19	Nom de la table où l'on veut ajouter des paramètres
nbpar	in	I	Nombre de paramètres à ajouter
nompar	in	V(K16)	Liste des noms des paramètres à ajouter
typpar	in	V(K8)	Liste des types des paramètres : 'R', 'I', 'C', 'K8', 'K16', 'K24', 'K32'

3.3 Routine TBEXIP : Tester l'existence et le type d'un paramètre

TBEXIP (nomtab, para, exist, typpar)			
nomtab	in	K19	Nom de la table à examiner
para	in	K16	paramètre à tester
exist	out	L	.TRUE. : le paramètre existe déjà dans la table nomtab
typpar	out	K8	type des paramètre s'il existe déjà dans la table : 'R', 'I', 'C', 'K8', 'K16', 'K24', 'K32'

3.4 Routine TBFUTB : Fusionner plusieurs tables en une seule table

TBFUTB (tabout, basout, ntab, ltabin, para, typpar, vi, vr, vc, vk)			
tabout	in	K19	Nom de la table que l'on veut créer
basout	in	K1	Base de création de la table tabout ('G', 'V', ...)
ntab	in	I	Nombre de tables que l'on veut fusionner
ltabin	in	K19	Noms des tables que l'on veut fusionner
para	in	K16	Nouveau paramètre (facultatif) qui permettra de distinguer l'origine de chacune des lignes de la nouvelle table si para=' ' les arguments suivants ne sont pas utilisés.
typpar	in	K8	Type du nouveau paramètre (facultatif)
vi	in	V(I)	Liste des valeurs pour le nouveau paramètre 'I' (facultatif)
vr	in	V(R)	Liste des valeurs pour le nouveau paramètre 'R' (facultatif)
vc	in	V(C)	Liste des valeurs pour le nouveau paramètre 'C' (facultatif)
vk	in	V(K*)	Liste des valeurs pour le nouveau paramètre 'K' (facultatif)

Cette routine peut être pratique pour créer une table creuse, le développeur peut réaliser chaque sous-table séparément et utiliser la routine TBFUTB pour les fusionner en une seule table.

Exemple :

On veut fusionner les 2 tables : T1 et T2

A	B	C	D
x1	x2		x3
	x4		x5
x6	x7	x8	

A	B	E
y1	y2	y3
y4		y5

Si l'on écrit :

```
ltabin(1)=T1
ltabin(2)=T2
CALL TBFUTB (T3, 'V', 2, ltabin, ' ', kbid, ibid, rbid, cbid, kbid)
```

On obtient la table : T3

A	B	C	D	E
x1	x2		x3	
	x4		x5	
x6	x7	x8		
y1	y2			y3
y4				y5

Si l'on écrit :

```
ltabin(1)=T1
ltabin(2)=T2
VK(1)='ACTION1'
VK(2)='ACTION2'
CALL TBFUTB (T3,'V',2,ltabin,'N','K8',ibid,rbid,cbid,VK)
```

On obtient la table : T3

N	A	B	C	D	E
ACTION1	x1	x2		x3	
ACTION1		x4		x5	
ACTION1	x6	x7	x8		
ACTION2	y1	y2			y3
ACTION2	y4				y5

Remarques sur l'ordre des lignes et des colonnes de la nouvelle table :

Les lignes de la nouvelle table sont ordonnées en mettant bout à bout les lignes des tables que l'on fusionne.

Pour ordonner les paramètres on adopte les règles suivantes :

- le nouveau paramètre (facultatif) est numéroté en premier,
- les paramètres de la 1^{ère} table de *ltabin* sont ensuite ajoutés dans l'ordre qu'ils ont dans *ltabin(1)*
- les paramètres de la 2^{ème} table de *ltabin* sont ensuite ajoutés (sauf ceux déjà présents dans *ltabin(1)*) dans l'ordre qu'ils ont dans *ltabin(2)*
- ...

3.5 Détruire une table

```
CALL DETRSD( 'TABLE', nomtab)
```

3.6 Dupliquer une table

```
CALL COPISD( 'TABLE','V',tabin,tabout)
```

4 Routines d'écriture de valeurs dans une table

4.1 Routine TBAJLI : Ajouter une ligne à la table

TBAJLI	(nomtab,nbpar,nompar,vi,vr,vc,vk,nume)			
nomtab	in jxvar	K19	Nom de la table où l'on veut ajouter une ligne	
nbpar	in	I	Nombre de paramètres pour la ligne	
nompar	in	V(K16)	Liste des noms des paramètres de la ligne	
vi	in	V(I)	Liste des valeurs pour les paramètres 'I'	
vr	in	V(R)	Liste des valeurs pour les paramètres 'R'	
vc	in	V(C)	Liste des valeurs pour les paramètres 'C'	
vk	in	V(K*)	Liste des valeurs pour les paramètres 'K'	
nume	in	I	/ 0 : on ajoute la ligne au bout de la table / i : on remplace la i ^{ème} ligne de la table	

Exemple :

Soit une table contenant les paramètres de type "entiers" 'I1', 'I2', 'I3', les paramètres "réels" 'R1' et 'R2' et les paramètres "chaînes de caractères" 'K1' et 'K2'

Supposons que l'on veuille y ajouter une ligne contenant : I2=i2, R1=r1, K2=k2, K1=k1.

On pourra appeler la routine TBAJLI avec les arguments :

```
nbpar = 4
nompri = ('I2', 'R1', 'K2', 'K1')
vi = (i2)
vr = (r1)
vk = (k2, k1)
nume = 0
```

4.2 Routine TBNULI : Permet de récupérer un numéro de ligne dans une table

TBNULI	(tabin, npacri, lipacr, vi, vr, vc, vk, lprec, lcrit, nume)		
tabin	in	K19	Nom de la table dont on veut récupérer une ligne
npacri	in	I	Nombre de paramètres impliqués dans les critères de sélection de la ligne (dimension de lipacr)
lipacr	in	V(K16)	Liste des noms des paramètres critères
vi	in	V(I)	Valeurs des critères pour les paramètres 'I'
vr	in	V(R)	Valeurs des critères pour les paramètres 'R'
vc	in	V(C)	Valeurs des critères pour les paramètres 'C'
vk	in	V(K)	Valeurs des critères pour les paramètres 'K'
lcrit	in	V(K8)	Liste des critères secondaires d'égalité pour les paramètres flottants : 'EGAL', 'RELA', 'ABSO' [§6.1.1].
lprec	in	V(R)	Liste des précisions (pour les critères d'égalité des paramètres flottants [§6.1.1].)
nume	out	I	= 0 : Il n'y a pas de ligne correspondant aux critères. = i : La ligne i est la seule qui correspond aux critères de sélection < 0 : Il y a plusieurs lignes correspondant aux critères

On recherche une ligne dans la table tabin en imposant des conditions sur ses paramètres.

Le mécanisme de sélection d'une ligne dans une table (arguments lipacr, lcrit, lprec, vi, vr, vc, vk) est expliqué au [§6.1].

Lorsque cette ligne est trouvée (et unique), on rend son numéro ce qui permet de modifier cette ligne à l'aide de la routine TBAJLI.

4.3 Exemples

Soit la table : T3

A	B	C
x1	x2	
x6	x7	x8

On ajoute une ligne à T3 :

```
CALL TBAJLI (T3, 1, 'B', ibid, z1, cbid, kbid, 0)
```

On obtient alors la table :

A	B	C
x1	x2	
x6	x7	x8

	z1	
--	----	--

On récupère le numéro de la ligne telle que A=x1 :

```
CALL TBNULI (T3,1,'A',ibid,x1,cbid,kbid,1.d-6,'RELA',ilig)
```

On modifie la ligne ilig :

```
CALL TBAJLI (T3,1,'C',ibid,z2,cbid,kbid,ilig)
```

On obtient alors la table :

A	B	C
		z2
x6	x7	x8
	z1	

5 Routines de lecture de valeurs dans une table

5.1 Routine TBEXVE : Lecture des valeurs d'une colonne d'une table

TBEXVE	(nomtab,para,nomobj,basobj,nbval,typval)		
nomtab	in jxin	K19	Nom de la table dans laquelle on veut extraire une colonne
para	in	K16	Paramètre désignant la colonne à extraire
nomobj	in jxout	K24	Nom de l'objet JEVEUX contenant les valeurs lues dans la table
basobj	in	K1	Base 'G', 'V' sur laquelle on crée le vecteur nomobj
nbval	out	I	Nombres de valeurs extraites
typval	out	K4	Type JEVEUX des valeurs extraites : I/R/C/K8,K16,...

5.2 Routine TBEXFO : Crée une fonction à partir d'une table

TBEXFO	(nomtab,parax,paray,nomfo,basfon)		
nomtab	in jxin	K19	Nom de la table dans laquelle on veut extraire une fonction.
parax	in	K16	Paramètre abscisse désignant la colonne à extraire
paray	in	K16	Paramètre ordonnée désignant la colonne à extraire
nomfo	in jxout	K24	Nom de la fonction à créer
basfon	in	K1	Base 'G', 'V' sur laquelle on crée la fonction

5.3 Routine TBLIVA : Lecture de la valeur d'une cellule

TBLIVA	(nomtab,npacri,lipacr,vi,vr,vc,vk,lcrit,lprec,para,ctype, vali, valr, valc, valk, ier)		
nomtab	in jxin	K19	Nom de la table dans laquelle on veut extraire la valeur d'une cellule
npacri	in	I	Nombre de paramètres impliqués dans les critères de choix de la ligne (dimension de lipacr)
lipacr	in	V(K16)	Liste des paramètres critères
vi	in	V(I)	Valeurs des critères pour les paramètres 'I'
vr	in	V(R)	Valeurs des critères pour les paramètres 'R'
vc	in	V(C)	Valeurs des critères pour les paramètres 'C'

vk	in	V(K*)	Valeurs des critères pour les paramètres 'K'
lcrit	in	V(K8)	Liste des critères secondaires d'égalité pour les paramètres flottants : 'EGAL', 'RELA', 'ABSO' [§6.1.1]
lprec	in	V(R)	Liste des précisions (pour les critères d'égalité des paramètres flottants [§6.1.1])
para	in	K16	Paramètre associé à la colonne de la valeur cherchée
ctype	out	K8	Type de la valeur trouvée
vali	out	I	Valeur trouvée si paramètre 'I'
valr	out	R	Valeur trouvée si paramètre 'R'
valc	out	C	Valeur trouvée si paramètre 'C'
valk	out	K*	Valeur trouvée si paramètre 'K'
ier	out	I	Code retour : 0 : OK 1 : para n'existe pas dans la table pour la ligne trouvée 2 : pas de ligne trouvée correspondant aux critères 3 : plusieurs lignes trouvées correspondant aux critères

Cette routine permet de lire la valeur associée à un paramètre donné pour une ligne sélectionnée dans une table. On sélectionne la ligne en imposant des valeurs à certains paramètres. Le mécanisme de sélection d'une ligne dans une table (arguments lipacr, lcrit, lprec, vi, vr, vc, vk) est expliqué au [§6.1.1].

Exemple :

Soit la table: T3

A	B	C
7.	4	'Z'
12.	0	'A1'
	4	'A2'

```
lipacr(1)='A', vr(1)=6.999, lcrit (1)='RELA', lprec(1)=0.01
```

```
lipacr(2)='B', vi(1)= 4
```

```
CALL TBLIVA (T3,2,lipacr,vi,vr,vc,vk,lcrit,lprec,'C',ctype,
             vali,valr,valc,valk,ier)
```

en sortie on a :

```
valk='Z'
ier=0
ctype='K8'
```

6 Routines de filtrage et de tri d'une table

6.1 Mécanisme de filtrage de lignes dans une table

Dans les routines TBLIVA, TBNULI et TBEXTB, il est nécessaire de "filtrer" une table existante pour n'en retenir qu'une (ou plusieurs) lignes. C'est ce mécanisme que nous expliquons ici.

Pour filtrer une table, l'utilisateur impose des critères à certains paramètres. Il dira par exemple de ne retenir que les lignes de la table pour lesquelles NUME_ORDRE=1. Il peut utiliser plusieurs critères de sélection des lignes et un même paramètre peut apparaître plusieurs fois dans la liste des critères.

Le "types" de critère de choix possibles sont :

EQ	"égalité" pour les entiers, les textes, les réels ou les complexes. Pour les nombres flottants (réels ou complexes), ce critère est complété par un critère "secondaire" expliqué ci-dessous (Cf. 6.1.1).
NE	"non-égalité" (Cf. EQ)
LT	"plus petit que" Relations d'ordre : - naturelle pour les entiers et les réels - alphabétique pour les textes - invalide pour les complexes
GT	"plus grand que" (Cf. LT)
LE	"plus petit ou égal à" (Cf. LT)
GE	"plus grand ou égal à" (Cf. LT)
VIDE	cellule vide
NON_VIDE	Cellule non vide

Remarque :

Dans les routines *TBLIVA* et *TBNULI*, le type de critère de sélection est (pour l'instant) **toujours** l'égalité ('EQ'). La liste des critères donnée dans ces 2 routines (*lcrit*) est donc la liste des critères "secondaires" (Cf. 6.1.1).

Les arguments : *vi*, *vr*, *vc*, *vk* de ces routines contiennent les valeurs associées à ces critères (selon le type associé à chacun des paramètres sur lesquels portent les critères).

Nous allons expliquer l'utilisation de tous ces arguments sur un exemple (sans nombres flottants) :

Soit une table contenant des paramètres *I1*, *I2*, *I3* de type "entier" et *K1*, *K2* de type "caractère".

On veut en extraire les lignes qui satisfassent aux critères suivants :

- 1) la valeur de *K1* est différente de 'ACTION1'
- 2) la valeur de *I2* est comprise entre 12 et 21
- 3) la valeur de *I3* vaut 999
- 4) la valeur de *I1* est "non vide"
- 5) la valeur de *K2* est "supérieure" (au sens de l'ordre alphabétique) à 'III'

On donnera comme arguments de *TBEXTB* :

```
npacri=6 Il y a 6 critères de choix car le 2ème critère est double : (x>12) et (x<21)
lipacr = ('K1', 'I2', 'I2', 'I3', 'I1', 'K2')
lcrpa = ('NE', 'GT', 'LT', 'EQ', 'NON_VIDE', 'GT')
vk(1)='ACTION1', vi(1)=12, vi(2)=21, vi(3)=999, vk(2)='III'
```

Remarque :

Les types de critère 'VIDE' et 'NON_VIDE' ne nécessitent pas d'arguments dans les tableaux *vi*, *vr*, ...

6.1.1 Cas particulier des critères d'égalité (ou de non égalité) pour des nombres "flottants"

L'égalité des nombres flottants est une notion dangereuse en informatique car elle peut dépendre de certaines troncatures : erreurs d'arrondi par exemple. Pour ces critères de sélection, on utilise donc une liste de critères secondaires qui précisent quelle égalité est voulue. Il y a trois type possibles pour les critères secondaires :

'EGAL'	égalité "exacte" des 2 nombres flottants
'ABSO'	égalité des 2 nombres flottants à un epsilon près (<i>eps</i>) en comparaison absolue : vrai si $ x1-x2 < eps$
'RELA'	égalité des 2 nombres flottants à un epsilon près (<i>eps</i>) en comparaison relative : vrai si $ x1-x2 < eps * x1 $

Exemple :

Soit une table contenant les paramètres entiers : I1 et I2, les paramètres réels R1 et R2 et la paramètres complexes C1 et C2. On veut en extraire les lignes correspondant aux critères suivants :

- 1) I1 > I2
- 2) C1 = c1 à 0.01 près en "absolu"
- 3) R1 = r1 à 0.1 près en "relatif"
- 4) C2 /= c2 à 0.02 près en "relatif" (les nombres qui **ne sont pas** dans le disque de centre c2 et de rayon 0.002*c2)
- 5) R2 /= r2 à 0.2 près en "absolu" (les nombres qui **ne sont pas** dans l'intervalle de centre r2 et de rayon 0.2)

On donnera comme arguments de TBEXTB :

```
npacri=5
lipacr = ('I1', 'C1', 'R1', 'C2', 'R2')
lcrpa = ('GT', 'EQ', 'EQ', 'NE', 'NE')
vi(1) = 12, vr(1)=r1, vr(2)=r2, vc(1)=c1, vc(2)=c2
lcrit = ('ABSO', 'RELA', 'RELA', 'ABSO')
lprec = (0.01, 0.1, 0.02, 0.2)
```

6.2 Routine TBEXTB : Filtrage et extraction d'une nouvelle table

TBEXTB	(tabin,basout,tabout,npacri,lipacr,lcrpa,vi,vr,vc,vk,lprec,lcrit)		
tabin	in jxin	K19	Nom de la table dont on veut extraire des lignes
basout	in	K1	'G', 'V' : base de création de tabout
tabout	in jxout	K19	Nom de la table qui contiendra les lignes extraites de tabin
npacri	in	I	Nombre de paramètres impliqués dans les critères d'extraction (dimension de lipacr et de lcrpa)
lipacr	in	V(K16)	Liste des paramètres critères [§6.1]
lcrpa	in	V(K10)	Liste des critères de sélection : EQ , LT, GT , NE, LE, GE, VIDE, NON_VIDE La signification de ces critères est donnée au [§6.1].
lcrit	in	V(K8)	Liste des critères secondaires d'égalité pour les paramètres flottants : 'EGAL', 'RELA', 'ABSO' [§6.1.1]..
lprec	in	V(R)	Liste des précisions (pour les critères d'égalité des paramètres flottants [§6.1.1].)
vi	in	V(I)	Valeurs des critères pour les paramètres 'I'
vr	in	V(R)	Valeurs des critères pour les paramètres 'R'
vc	in	V(C)	Valeurs des critères pour les paramètres 'C'
vk	in	V(K)	Valeurs des critères pour les paramètres 'K'

Remarques :

Lorsque l'on a récupéré dans tabin les lignes satisfaisant à tous les critères donnés, on ne conserve dans la table tabout que les colonnes pour lesquelles il existe au moins une valeur.
Un même paramètre peut apparaître plusieurs fois dans la liste des critères (lcrpa).

Exemple :

Soit la table: T3

A	B	C
	4	'Z'
11.	0	'A1'
24.	9	'A2'
240.	9	'A2'

12.	12.	
-----	-----	--

```
lipacr(1)='A'
lcrpa (1)='NON_VIDE'
```

```
lipacr(2)='B'
lcrpa (2)='GT'
vi      (1)= 1
```

```
lipacr(3)='A'
lcrpa (3)='LE'
vr      (1)= 1.d2
```

```
CALL TBEXTB (T3, 'G', T3B, 3, lipacr, lcrpa, vi, vr, vc, vk, lprec, lcrit)
```

en sortie T3B contient :

A	B	C
24.	9	'A2'
12.	12.	

6.3 Routine TBTRTB : Tri de la table

TBTRTB	(tabin,basout,tabout,npara,lipara,lcrit)
--------	--

tabin	in jxin	K19	Nom de la table dont on veut classer les lignes
basout	in	K1	Base de création de tabout : G/V/L
tabout	in jxout	K19	Nom de la table contenant toutes les lignes de tabin classées selon les critères suivants
npara	in	I	Nombre de paramètres impliqués dans les critères de tri (dimension de lipara et de lcrit)
lipara	in	V(K16)	Liste des paramètres critères
lcrit	in	V(K2)	Liste des types de critères : 'CR' ou 'DR' 'CR' : ordre croissant 'DR' : ordre décroissant

Cette routine sert à créer une nouvelle table (tabout) en permutant l'ordre des lignes d'une table existante (tabin) selon certains critères de tri.

Tri

Valeurs R, I : Par valeurs croissantes ou décroissantes

Valeurs K8, K16, K24, K32 : Par ordre alphabétique croissant ou décroissant

Exemple :

```
npara = 2
lipara = ('NOEUD', 'INST')
lcrit  = ('CR', 'DR')
```

La nouvelle table (tabout) sera ordonnée :

- premièrement par ordre alphabétique croissant des noms de noeuds,
- deuxièmement par ordre décroissant des instants de calcul.

Remarques :

Le 2ème critère ne s'applique que s'il y a égalité au sens du premier critère.
Les cellules vides sont classées en "tête" (ce sont les plus petites pour la relation d'ordre).

7 Routine TBIMPR : Impression d'une table

TBIMPR	(table, format, fichier, nparim, lipaim, nparpg, lipapg, formar, formac)		
table	in	K19	Nom de la table que l'on veut imprimer
format	in	K8	Format d'impression de la table : 'EXCEL', 'TABLEAU', 'MOT_CLE'
fichier	in	K*	Nom du fichier listing : 'RESULTAT', ...
nparim	in	I	Nombre de paramètres à imprimer si nparim=0, on imprime TOUS les paramètres.
lipaim	in	V(K16)	Liste des paramètres à imprimer. L'ordre d'impression des paramètres est celui de la liste lipaim
nparpg	in	I	Nombre de paramètres de "pagination"
lipapg	in	V(K16)	Liste des paramètres de pagination
formar	in	K8	/ Format d'écriture des valeurs réelles / ' ' => valeur par défaut : 1PE12.5
formac	in	K2	Convention d'écriture des nombres complexes : / ' ' => valeur par défaut : 'RI' / 'RI' (partie réelle, partie imaginaire) / 'MP' (module, phase)

L'utilisateur a la possibilité d'imprimer ses résultats sous les formats suivants :

7.1 FORMAT : 'EXCEL'

NUME_ORDRE	INST	NOEUD	DX	DY
1	4.	N7	3.4	3.8
1	4.	N4	2.4	2.8
1	4.	N2	1.4	1.8
4	8.	N7	3.4	3.8
4	8.	N4	2.4	2.8
4	8.	N2	1.4	1.8
7	20.	N7	3.4	3.8
7	20.	N4	2.4	2.8
7	20.	N2	1.4	1.8

7.2 FORMAT : 'TABLEAU'

DX		INST		
		4.	8.	20.
NOEUD	N7	3.4	3.4	3.4
	N4	2.4	2.4	2.4
	N2	1.4	1.4	1.4

7.3 FORMAT : 'MOT_CLE'

```

NUME_ORDRE: 1 INST: 4. NOEUD: N7 DX: 3.4 DY: 3.8
NUME_ORDRE: 1 INST: 4. NOEUD: N4 DX: 2.4 DY: 2.8
NUME_ORDRE: 1 INST: 4. NOEUD: N2 DX: 1.4 DY: 1.8
NUME_ORDRE: 4 INST: 8. NOEUD: N7 DX: 3.4 DY: 3.8
...
```

7.4 FORMAT 'EXCEL' avec pagination (définie ici par le paramètre 'NOEUD')

NOEUD: N7

NUME_ORDRE	INST	DX	DY
1	4.	3.4	3.8
4	8.	3.4	3.8
7	20.	3.4	3.8

NOEUD: N4

NUME_ORDRE	INST	DX	DY
1	4.	2.4	2.8
4	8.	2.4	2.8
7	20.	2.4	2.8

NOEUD: N2

NUME_ORDRE	INST	DX	DY
1	4.	1.4	1.8
4	8.	1.4	1.8
7	20.	1.4	1.8

Par défaut le format d'impression est le format 'EXCEL', c'est à dire présentation en colonnes des différents paramètres sélectionnés.

8 Routine TBEXLR : Transformation d'une table en une SD LISTR8

8.1 But de la routine

Pour accéder rapidement (au niveau des routines *te00ij* par exemple) à des informations contenues dans des tables, on peut transformer ces tables en des SD de type LISTR8. C'est l'objet de la routine TBEXLR.

Remarque :

La lecture de ce paragraphe 8 peut être sautée par tous ceux qui ne s'intéressent pas à la routine très particulière qu'est TBEXLR.

8.2 Interface de TBEXLR

TBEXLR	(table,lr8,base)		
table	in	K19	Nom de la table que l'on veut transformer en SD LISTR8
lr8	in jxout	K8	Nom de la SD LISTR8 résultat
base	in	K1	Base de création de lr8: 'G', 'V', ...

8.3 Restrictions d'utilisation de TBEXLR

La routine TBEXLR ne peut transformer une table en LISTR8 que si cette table est « diagonale par blocs ». Si ce n'est pas le cas, la routine s'arrêtera en erreur fatale.

8.3.1 Exemples de tables diagonales par blocs

A	B	C	D	E	F
1.	2.				
2.1	3.				
			8.	9.	
			10.	11.	
					19.

A	B	C	D	E
1.				
	3.			
			10.	11.

8.3.2 Exemples de tables non diagonales par blocs

A	B	C	D	E	F
1.	2.				
2.1	3.		5.	7.	
			8.	9.	
			10.	11.	
					19.

A	B	C	D	E	F
1.	2.				
2.1	3.				
			8.	9.	
			10.	11.	
	4.				19.

8.4 Comment une *table* est-elle transformée en *LISTR8* ?

Soit la table

paramètre	B	C	D	E	F	G	H	I	K
type valeurs	K1	R	I	K1	R	R	I	I	R
	w	1.	2	s	3.				
		2.	2	s	3.				
	w	3.	2	s	3.				
				s			1	2	
				s			1	3	
				s			1	4	
				s			1	5	
				s					12.
									13.
									14.
				s					15.

				S					16.
--	--	--	--	---	--	--	--	--	-----

8.4.1 étape 1 : on ne retient que les colonnes non vides de type 'I' ou 'R'

paramètre	C	D	F	H	I	K
type valeurs	R	I	R	I	I	R
	1.	2	3.			
	2.	2	3.			
	3.	2	3.			
				1	2	
				1	3	
				1	4	
				1	5	
						12.
						13.
						14.
						15.
						16.

8.4.2 étape 2 : reconnaissance des « blocs » de la table « diagonale »

paramètre	C	D	F	H	I	K
type valeurs	R	I	R	I	I	R
	1.	2	3.			
	2.	2	3.			
	3.	2	3.			
				1	2	
				1	3	
				1	4	
				1	5	
						12.
						13.
						14.
						15.
						16.

8.4.3 étape 3 : mise en vecteur des valeurs retenues, conversion des entiers en réels :

On met bout à bout dans le vecteur de réels les valeurs numériques trouvées dans les blocs de la table :

```
nb_blocs , nb_col(bloc1), nb_lig(bloc1), valeurs (bloc1), nb_col(bloc2),
nb_lig(bloc2), valeurs (bloc2), ...
```

Les valeurs d'un bloc sont écrites ligne par ligne. La table ci-dessus devient alors le vecteur de réels ci-dessous.

```
3.
3.    3.    1.    2.    3.    2.    2.    3.    3.    2.    3.
2.    4.    1.    2.    1.    3.    1.    4.    1.    5.
1.    5.    12.   13.   14.   15.   16.
```

Remarque :

Nous avons présenté la liste de réels sur 4 lignes pour faciliter la lecture du résultat de la transformation, mais la *LISTR8* produite contient $(3*3+2*4+1*5) + 3*2 + 1$ réels simplement mis bout à bout.

9 Exemple simple de création et d'exploitation d'une table

Soit l'exemple de la table T2 :

ACTION	NUME_ORDRE	INST	NOEUD	DX	DY	MAILLE	SIXX
INTITULE 1	1	10.	N1	3.	5.		
INTITULE 1	1	10.	N2	6.	7.		
INTITULE 1	1	10.	N3	8.	9.		
INTITULE 1	2	20.	N1	11.	12.		
INTITULE 1	2	20.	N2	15.	13.		
INTITULE 1	2	20.	N3	19.	18.		
INTITULE 2	5	20.				MA1	-12.
INTITULE 2	5	20.				MA2	-14.

Remarque préliminaire :

Les interfaces détaillées des routines utilitaires sont données au paragraphe [§2]. Mais les quelques lignes de fortran ci-dessous peuvent se comprendre sans leur lecture.

% Déclaration de la table sur la base GLOBALE :

```
CALL TBCRS('T2','G')
```

% Déclarations des paramètres de la table et des types de leurs données :

```
CALL TBAJPA('T2',1,'ACTION','K8')
CALL TBAJPA('T2',1,'NUME_ORDRE','I')
CALL TBAJPA('T2',1,'INST','R')
CALL TBAJPA('T2',1,'NOEUD','K8')
CALL TBAJPA('T2',1,'DX','R')
CALL TBAJPA('T2',1,'DY','R')
CALL TBAJPA('T2',1,'MAILLE','K8')
CALL TBAJPA('T2',1,'SIXX','R')
```

% Ajout des lignes dans la table :

```
REAL*8          VR(3)
CHARACTER*8      VK(2)
INTEGER          VI(1)
CHARACTER*16     LPARA1(6), LPARA2(5)
DATA LPARA1 / 'ACTION', 'NUME_ORDRE', 'INST', 'NOEUD', 'DX', 'DY' /
DATA LPARA2 / 'ACTION', 'NUME_ORDRE', 'INST', 'MAILLE', 'SIXX' /
%
VK(1) = action
IF action = intitule1
  DO nume_ordre = 1,2
    VI(1) = nume_ordre
    VR(1) = t = instant(nume_ordre)
    DO noeud = N1, N2, N3
      VK(2) = noeud
      VR(2) = DX(noeud,t)
      VR(3) = DY(noeud,t)
      CALL TBAJLI('T2',6,LPARA1,VI,VR,CBID,VK,0)
    CONTINUE
  CONTINUE
ELSE IF action = intitule2
  VI(1) = 5 = nume_ordre
  VR(1) = t = instant(5)
  DO maille = MA1, MA2
    VK(2) = maille
    VR(2) = SIXX(maille,t)
    CALL TBAJLI('T2',5,LPARA2,VI,VR,CBID,VK,0)
  CONTINUE
ENDIF
```

Page laissée intentionnellement blanche.