

## Opérateur LIRE\_RESU

---

### 1 But

---

Lire des champs et les stocker dans une structure de données de type `resultat`

Le type du format du fichier lu est : soit le format MED, soit le format universel IDEAS, soit le format ENSIGHT.

Produit une structure de données de type `resultat` (`evol_noli`, `evol_ther`, ...).

## 2 Syntaxe générale

```
resu = LIRE_RESU (
```

### # Choix du format du fichier à lire :

```

♦ / FORMAT = 'IDEAS',
  ♦ NOM_CHAM = l_nomch, [l_Kn]
  ◇ UNITE = / iunit, [I]
           / 19, [DEFAULT]

  ◇ FORMAT_IDEAS = _F ( ... # voir [ § 3.2.1.2 ] )

/ FORMAT = 'IDEAS_DS58',
  ♦ UNITE = /iunit, [I]
           / 19, [DEFAULT]

  ♦ NOM_CHAM = l_nomch, [l_Kn]
  ◇ REDEFI_ORIENT = _F(
  ◇ CODE_DIR = /1,
               /2,
               /3,
               ◇ DIRECTION = (dx,dy,dz), [l_R]
               ◇ NOEUD = l_no, [l_noeud]
               ),

/ FORMAT = 'ENSIGHT',
  ♦ NOM_CHAM = l_nomch, [l_Kn]
  ◇ NOM_FICHER = fichier, [Kn]

/ FORMAT = 'MED'
  ◇ UNITE = /iunit, [I]
           /81

  ♦ FORMAT_MED =
  ( _F(
    ♦ NOM_CHAM = nomch, [Kn]
    ♦ / NOM_CHAM_MED = nommed, [Kn]
    / NOM_RESU = nomres, [Kn]
    ◇ NOM_CMP = lcmp, [l_Kn]
    ◇ NOM_CMP_MED = lcmpmed, [l_Kn]
  ), ),

```

### # Quelle structure de données faut-il créer ? :

```

♦ TYPE_RESU = / 'EVOL_ELAS',
              / 'EVOL_CHAR',
              / 'EVOL_THER',
              / 'EVOL_NOLI',
              / 'DYNA_TRANS',
              / 'DYNA_HARMO',
              / 'MODE_MECA',
              / 'MODE_MECA_C',

```

### # Si TYPE\_RESU= 'EVOL\_NOLI' :

```

◇ NB_VARI = nbvar,

```

### # Si TYPE\_RESU= 'MODE\_MECA' OU 'MODE\_MECA\_C' :

```

◇ MATR_RIGI = matr_rigi, [matr_asse_DEPL_R]

```

n                   ◇   MATR\_MASS                   =   matr\_mass,           [matr\_asse\_DEPL\_R]

**# autres informations :**

◆   /   MAILLAGE                   =   ma,                   [maillage]  
   /   MODELE                    =   mo,                   [modele]  
◇   COMPORTEMENT               =   \_F (voir le document [U4.51.11]),  
◇   CHAM\_MATER                  =   chmat,                [cham\_mater]  
◇   AFFE\_CARA\_ELEM              =   carac,                [cara\_elem]

**# choix des numéros d'ordre à lire :**

◇   /   TOUT\_ORDRE               =   'OUI',                [DEFAULT]  
   /   NUME\_ORDRE               =   lordr,                [l\_I]  
   /   LIST\_ORDRE               =   lordr,                [listis]  
   /   INST                      =   linst,                [l\_R]  
   /   LIST\_INST                =   linst,                [listr8]  
   /   FREQ                      =   lfreq,                [l\_R]  
   /   LIST\_FREQ                =   lfreq,                [listr8]  
◇   |   PRECISION               =   /prec,                [R]  
   |   CRITERE                 =   /'RELATIF',            [DEFAULT]  
   |                             /'ABSOLU'                [DEFAULT]

**# Si TYPE\_RESU= 'EVOL\_THER' :**

◇   EXCIT =\_F(  
   ◆   CHARGE                   =   char,                [char\_ther]  
   ◇   FONC\_MULT                =   fonc,                [fonction]  
   ),

**# Si TYPE\_RESU= 'EVOL\_NOLI' :**

◇   EXCIT =\_F(  
   ◆   CHARGE                   =   char,                [char\_meca]  
   ◇   TYPE\_CHARGE              =   / 'FIXE\_CSTE'  
   /   'FIXE\_PILO'  
   /   'FIXE\_PILO'  
   /   'SUIV'  
   /   'DIDI'  
   ◆   /   FONC\_MULT            =   fonc,                [fonction]  
   /   ◆   DEPL                 =   depl,                [fonction]  
   /   ◆   VITE                 =   vite,                [fonction]  
   /   ◆   ACCE                 =   acce,                [fonction]  
   ◇   MULT\_APPUI               =   / 'OUI',                [DEFAULT]  
   /   'NON'  
   ◇   DIRECTION               =   (d1, d2, d3),           [l\_R]  
   ◇   /   NOEUD                =   lno,                [l\_noeud]  
   /   GROUP\_NO                =   lgrno,               [l\_gr\_noeud]  
   ),

**# Si TYPE\_RESU= 'EVOL\_ELAS' :**

◇   EXCIT =\_F(  
   ◆   CHARGE                   =   char,                [char\_meca]  
   ◇   FONC\_MULT                =   fonc,                [fonction]  
   ◇   TYPE\_CHARGE              =   / type                [l\_Kn]

```
                                / 'FIXE'  
                                ),  
  
    ◇ TITRE                      = l_titre,          [l_Kn]  
    ◇ INFO                       = /1,  
                                /2  
    )
```

## 3 Opérandes

### 3.1 Opérandes FORMAT / UNITE / NOM\_FICHER

/ FORMAT = 'IDEAS' ou 'IDEAS\_DS58'

Lecture du fichier au format IDEAS.

◇ UNITE

Numéro d'unité logique du fichier au format universel IDEAS, par défaut 19.

/ FORMAT = 'MED'

Lecture du fichier au format MED.

◇ UNITE

Numéro d'unité logique du fichier au format MED, par défaut 81.

/ FORMAT = 'ENSIGHT'

Lecture de fichiers au format Ensignt, concerne la lecture de champs de pression aux nœuds et est associé à TYPE\_RESU = 'EVOL\_CHAR' et à NOM\_CHAM = 'PRES'.

◇ NOM\_FICHER

Nom de fichier (en minuscules et entre quotes) Ensignt de type 'Measured Results' (voir Ensignt User Manuel pages 3-49, 3-50) contenant notamment :

- le nom du fichier 'Measured Geometry' contenant des numéros de nœuds suivis de leurs coordonnées,
- le nom du (ou des) fichier(s) 'Scalar Variable' contenant les valeurs de la pression aux nœuds spécifiés dans le fichier 'Measured Geometry'.

#### Remarque :

Le numéro d'unité logique peut avoir été associé à un fichier à l'aide de la commande `DEFI_FICHER [U4.12.03]`.

### 3.2 Opérandes si FORMAT = 'IDEAS'

On ne lit pas les *datasets* 58 (mais seulement les *datasets* 55,57 et 2414)

#### 3.2.1 Mot clé FORMAT\_IDEAS

##### 3.2.1.1 Objectif

Les champs à lire dans le fichier universel sont écrits sous forme de *datasets*. Chaque *dataset* est composé d'un en-tête "*carte d'identité*" et d'un ensemble de valeurs (résultats aux nœuds ou par élément aux nœuds). Cette carte d'identité est composée de plusieurs enregistrements "record", composés de champs "*field*". L'objectif de ce mot clé est de permettre à l'utilisateur de définir sa propre "carte d'identité" en spécifiant ses propres critères de recherche.

#### Remarques :

Un certain nombre de "cartes d'identité" sont définies par défaut [§6]. On peut les "surcharger" en utilisant le mot clé `FORMAT_IDEAS`.

Dans certains cas particuliers (tests de non-régression) le maillage associé au modèle n'a pas été créé par `PRE_IDEAS`. Si on ne retrouve pas dans le maillage le nœud associé au numéro qui figure dans le *dataset* 55, alors, on récupère le nom du nœud en commentaire dans le fichier universel, tout en vérifiant que le numéro associé correspond bien au nom récupéré.

##### 3.2.1.2 Syntaxe

```
FORMAT_IDEAS = _F(  
♦ NOM_CHAM = nomch, [Kn]  
◇ NUME_DATASET = /55,  
/57,  
/2414,  
◇ | RECORD_3 = r3, [1_I]  
◇ | RECORD_6 = r6, [1_I]  
◇ | RECORD_9 = r9, [1_I]  
♦ POSI_ORDRE = po, [1_I]  
◇ POSI_NUME_MODE = pnm, [1_I]  
◇ POSI_MASS_GENE = pmg, [1_I]  
◇ POSI_AMOR_GENE = pag, [1_I]  
◇ / POSI_INST = pi, [1_I]  
/ POSI_FREQ = pf, [1_I]  
♦ NOM_CMP = lcmp, [1_Kn]  
◇ NB_VARI = nbvari, [I]  
) ,
```

### 3.2.1.3 Opérandes

♦ `NOM_CHAM = nomch`

Nom symbolique du champ pour lequel l'utilisateur définit les critères de recherche. Voir le mot clé `NOM_CHAM` hors des mots clés facteurs [§3.5.1].

♦ `NUME_DATASET =`

Numéro du *dataset* à partir duquel sera extrait les résultats :

- : valeurs aux nœuds,
- : valeurs aux nœuds par élément
- : valeurs

```
◇ | RECORD_3  
◇ | RECORD_6  
◇ | RECORD_9
```

Chacun de ces mots clés est composé du mot `RECORD` et d'un nombre. Le nombre indique le numéro de l'enregistrement pour lequel on va définir les critères de recherche. Chaque opérande permet de définir au maximum 10 valeurs entières.

Ex : `RECORD_6 = (1, 4, 9999, 8, 2, 6),`

Dans cet exemple, si le *dataset* lu contient au niveau de l'enregistrement n°6 les valeurs (1 4 9999 8 2 6), il sera retenu pour la suite des recherches. La valeur 9999 est un joker permettant d'ignorer la valeur lue dans le *dataset*.

♦ `/ POSI_ORDRE`

Vecteur de deux entiers permettant de localiser le numéro d'ordre

$V(1)$  : Numéro de l'enregistrement

$V(2)$  : Position du numéro d'ordre

◇ `/ POSI_NUME_MODE`

Vecteur de deux entiers permettant de localiser le numéro de mode

$V(1)$  : Numéro de l'enregistrement

$V(2)$  : Position du numéro de mode

◇ `/ POSI_MASS_GENE`

Vecteur de deux entiers permettant de localiser la masse généralisée

$V(1)$  : Numéro de l'enregistrement

$V(2)$  : Position de la masse généralisée

◇ / POSI\_AMOR\_GENE

Vecteur de deux entiers permettant de localiser l'amortissement généralisé

$V(1)$  : Numéro de l'enregistrement

$V(2)$  : Position de l'amortissement généralisé

◇ / POSI\_INST

Vecteur de deux entiers permettant de localiser l'instant

$V(1)$  : Numéro de l'enregistrement

$V(2)$  : Position de l'instant

◇ / POSI\_FREQ

Vecteur de deux entiers permettant de localiser la fréquence

$V(1)$  : Numéro de l'enregistrement

$V(2)$  : Position de la fréquence

◆ NOM\_CMP

Nom des composantes à lire.

Ex : `NOM_CMP = ('DX' , 'DY' , 'DZ' , 'XXX' , 'DRX' , 'XXX' , 'DRZ' , )`,

La chaîne de caractères 'XXX' est un joker permettant d'ignorer la composante lors de la lecture des valeurs.

Si le nombre de composantes à lire est supérieur au nombre de composantes présentes dans le fichier .unv, celles-ci sont ignorées.

#### Remarque importante :

Lorsqu'on lit un *cham\_elem*, celui-ci est dimensionné conformément aux éléments finis du modèle (voir mot clé *MODELE* ci-dessous). Par exemple, si on lit un champ de contraintes sur un modèle 2D, les composantes portées par les éléments seront *SIXX*, *SIYY*, *SIXY* et *SIZZ*. Si dans le fichier *IDEAS*, on trouve les composantes : *SIXX*, *SIZZ*, *SIXZ*, les composantes *SIXZ* seront ignorées. En revanche, toutes les composantes non trouvées dans le fichier (*SIYY* et *SIXY* dans notre exemple) seront mises à zéro.

#### Remarque sur l'affectation du champ au nœud dans les datasets 55 :

Dans un fichier au format universel (*Ideas*), un nœud du maillage est connu par son numéro. Dans *Aster*, un nœud est connu par son nom. La retranscription se fait en affectant au nœud, un nom qui commence par *N* suivi de son numéro *Ideas*. Lors de la lecture du champ au nœud dans un dataset 55, on vérifie que le nœud existe dans le maillage *Aster*. Si tel n'est pas le cas, alors on considère le nom donné, habituellement, en commentaire à côté du numéro du nœud dans le dataset 55.

◇ NB\_VARI

Nombre de variables internes pour les champs de type ' VARI '

◇ PROL\_ZERO =

Si `PROL_ZERO = 'OUI'` : on met les valeurs des composantes aux nœuds à 0 où le champ n'est pas défini, et on émet une information du type: " Les valeurs non existantes du champ *TEMP* lues sur le maillage donné sont considérées nulles."

Si `PROL_ZERO = 'NON'` : on garde le champ tel quel.

Ce mot-clé est dédié uniquement aux champs aux nœuds, car pour les champs aux éléments, le champ est prolongé par 0 par défaut.

### 3.3 Opérandes si `FORMAT = 'IDEAS_DS58'`

On ne lit que les datasets 58.

### 3.3.1 Opérande NOM\_CHAM

♦ NOM\_CHAM = nomch,

Nom symbolique du ou des champs à lire. On peut lire les champs aux nœuds correspondant à :  
NOM\_CHAM = 'DEPL', 'VITE', 'ACCE', 'SIEF\_NOEU', 'EPSI\_NOEU'.

### 3.3.2 Mot clé REDEFI\_ORIENT

Ce mot clé facteur facultatif permet de redéfinir éventuellement la direction sensible du capteur en certains points de mesure. Cette redéfinition n'est traitée actuellement que pour les NOM\_CHAM = 'DEPL', 'VITE' et 'ACCE'. On redéfinit autant d'orientations que nécessaire. Si ce mot clé est absent, on adopte la convention habituelle rappelée dans le tableau suivant :

Code direction	'DEPL', 'VITE', 'ACCE'	'SIEF_NOEU'	'EPSI_NOEU'
1	DX	SIXX	EPXX
2	DY	SIYY	EPYY
3	DZ	SIZZ	EPZZ
4		SIXY	EPXY
5		SIXZ	EPXZ
6		SIYZ	EPYZ

#### 3.3.2.1 Opérandes

♦ CODE\_DIR = /1,  
/2,  
/3,

Code décrivant la direction de mesure, donné dans l'en-tête du dataset 58.

♦ DIRECTION = (dx, dy, dz)

Vecteur directeur, exprimé dans le repère global, indiquant la direction sensible à associer à CODE\_DIR

♦ NOEUD = l\_no,

Liste des nœuds où CODE\_DIR doit être associé au vecteur directeur DIRECTION.

### 3.3.3 Remarque : structure du champ créé

Les fichiers universels au format dataset 58 contiennent des champs « à trou » : chaque dataset contient les données associée à un nœud et une direction. Le champ ne peut pas être prolongé par des valeurs nulles, comme c'est le cas pour les fichiers de type dataset 55, 57 ou 2414. On crée un champ dont les composantes sont, pour chaque nœud :

- $D_i$ , le numéro  $i$  étant associé au code direction dans le repère local,
- $D_iX$ ,  $D_iY$  et  $D_iZ$ , donnant la direction locale dans le repère global.

Remarque : il n'est pas possible d'imprimer et de visualiser directement le champ lu dans les formats standards (type MED). Par contre, le champ lu peut être utilisé dans des opérateurs de corrélation calcul-essai. Le champ résultat contient les composantes standard et peut être imprimé.

## 3.4 Opérandes si FORMAT = 'MED'

Au format MED il est pas possible de lire de structures de données de type suivant : DYNA\_TRAN, DYNA\_HARMO, MODE\_MECA, MODE\_MECA\_C. Toutefois, il faut noter que cette relecture ne permet pas de remplir les paramètres de la structure de données. Le calcul peut s'arrêter si ces paramètres sont nécessaires à l'exécution d'une commande.



## 3.4.1 Opérande **NOM\_CHAM**

♦ `NOM_CHAM = nomch,`

Nom symbolique du champ à lire. On peut lire a priori les champs aux nœuds, les champs par éléments aux nœuds (`ELNO`), les champs constants (`ELEM`) et des champs aux points d'intégration (`ELGA`). Voir le mot clé `NOM_CHAM` hors des mots clés facteurs [§3.5.1]

## 3.4.2 Opérande **NOM\_CHAM\_MED/NOM\_RESU**

♦ `/ NOM_CHAM_MED = nommed,`  
`/ NOM_RESU = nommed,`

Permet de définir le nom du champ MED, soit de façon explicite soit de façon automatique :

- `NOM_CHAM_MED` : Nom selon la convention MED du champ à lire dans le fichier. C'est une chaîne de 32 caractères,
- `NOM_RESU` : Préfixe du nom du champ MED à lire dans le fichier. Il s'agit du nom du résultat associé au champ MED. C'est une chaîne d'au plus 8 caractères. Il permet de définir le nom du champ med de façon automatique avec la donnée fournie à `NOM_CHAM`.

## 3.4.3 Opérandes **NOM\_CMP/NOM\_CMP\_MED**

♦ `NOM_CMP = lcmp,`  
♦ `NOM_CMP_MED = lcmpmed,`

Ces deux listes doivent être de même longueur. On lit dans le fichier MED les composantes listées dans `lcmpmed`, puis on les affecte dans les composantes au sens de *Code\_Aster*, de même rang dans la liste `lcmp`.

Les noms des composantes MED sont limités à 16 caractères.

Si ces 2 listes sont absentes, on suppose que les noms des composantes sont les mêmes pour MED et pour Aster.

## 3.4.4 Opérande **PROL\_ZERO**

Lors de la lecture d'un champ aux points de Gauss au format MED, on vérifie la cohérence entre les points de Gauss tels qu'ils sont définis dans le fichier MED et dans Aster.

Si le nombre de points de Gauss "MED" est différent du nombre de points de Gauss "Aster" on émet simplement une alarme et on ne remplit pas le champ sur les éléments en cause. On aura donc des valeurs non définies (NaN) là où le champ n'est pas défini. Le mot-clé `PROL_ZERO = 'OUI'` permet à l'utilisateur mettre des valeurs nulles s'il le désire.

## 3.5 Autres opérandes

### 3.5.1 Opérandes **TYPE\_RESU/NOM\_CHAM**

♦ `TYPE_RESU`  
Type de la structure de données `resultat` créée.

Pour le type `EVOL_VARC` et pour le format MED, si le nom du champ est 'IRRA' alors `NOMGD='IRRA_R'`.

♦ `NOM_CHAM = l_nomch`

Nom symbolique du ou des champs à lire. C'est sous ce (ou ces noms) que les champs seront stockés dans la structure de données `resultat`.

A priori, pour le format 'IDEAS', on peut lire les champs aux nœuds (`NOEU`) ou constant par élément (`ELEM`) ou par nœuds aux éléments (`ELNO`).

Pour le format 'ENSIGHT', on ne sait lire que les champs de pression ('PRES') des SD de type evol\_char.

Pour le format 'IDEAS\_DS58', on ne traite pour l'instant les champs aux nœuds suivants : déplacement, vitesse, accélération, contrainte et déformation.

Pour le format 'MED', le choix de la famille de points de Gauss se fait à partir du nom fourni par l'utilisateur, par exemple :

```
U3 =LIRE_RESU( TYPE_RESU ='EVOL_NOLI',  
              FORMAT   ='MED',MODELE=MOMECA,  
              FORMAT_MED=(  
                _F(NOM_CHAM_MED ='U_____VARI_ELGA_____',  
                  NOM_CMP_MED  =('V1','V2',),  
                  NOM_CHAM     ='VARI_ELGA'), )  
            )
```

### 3.5.2 Opérande NB\_VARI

◇ NB\_VARI = nbvar,

Nombre de variables internes à lire pour les champs de variables internes (VARI\_R).

### 3.5.3 Opérandes MATR\_RIGI/MATR\_MASS

◇ MATR\_RIGI = matr\_rigi,

◇ MATR\_MASS = matr\_mass,

Lorsque l'on relit un concept de type mode\_meca et que l'on veut se servir de ce mode\_meca dans certains opérateurs (par exemple PROJ\_MATR\_BASE), il faut que :

- 1) le mode\_meca produit référence les 2 matr\_asse (rigidité et masse) qui ont servi à le calculer,
- 2) les champs du mode\_meca soient numérotés de la même manière que les inconnues de ces 2 matr\_asse.

Pour que cette numérotation cohérente soit établie dans LIRE\_RESU, il faut utiliser ces 2 mots-clé facultatifs MATR\_RIGI et MATR\_MASS (voir par exemple le test sdnv102a).

#### Attention :

| Cette possibilité n'est disponible qu'au format 'IDEAS' .

### 3.5.4 Opérandes MAILLAGE / MODELE

◆ / MAILLAGE = ma,

Maillage sur lequel on affecte le ou les champs lus.

/ MODELE = mo,

Nom du modèle où sont définis les types d'éléments finis affectés sur le maillage. Si on veut lire un cham\_elem, il faut donner le nom du modèle.

### 3.5.5 Opérande COMPORTEMENT

La syntaxe de ce mot-clé commun à plusieurs commandes est décrite dans le document [U4.51.11]. Ce mot-clef doit être renseigné dans le cas de la mécanique non-linéaire car il sert en reprise de calcul dans STAT\_NON\_LINE et DYNA\_NON\_LINE pour vérifier la compatibilité des comportements (nombre de variables internes en particulier). Si on ne le renseigne pas, la structure sera considérée avoir comportement élastique (COMPORTEMENT='ELAS') en petites déformations (RELATION='PETIT').

Le concept `EVOL_THER` sert à la thermique linéaire et à la thermique non-linéaire. Dans ce dernier cas, le comportement thermique (mot-clef `COMP_THER_NL` dans `THER_NON_LINE` ) n'est pas disponible dans `LIRE_RESU` . Ce qui veut dire que le concept `EVOL_THER` créé par `LIRE_RESU` ne contient pas la carte de comportement thermique ( `COMPOR_THER` ) et n'est donc pas tout à fait conforme. Toutefois, ce problème n'est pas gênant dans la mesure où la carte de comportement est créée dans `THER_NON_LINE` , y compris en reprise, et que les comportements thermiques non-linéaires n'ont pas de variables internes, il n'est donc pas nécessaire de vérifier la compatibilité des champs non-linéaires en reprise, il n'y a pas de risque de résultats faux

### 3.5.6 Opérandes `CHAM_MATER/CARA_ELEM/EXCIT`

Pour qu'un résultat issu de la commande `LIRE_RESU` soit exploitable avec Stanley, il est nécessaire de disposer des informations relatives au champ de matériau et aux caractéristiques élémentaires. Par souci de complétude, les informations concernant le chargement peuvent aussi être stockées dans la structure de données résultat.

### 3.5.7 Opérandes `TOUT_ORDRE/ NUME_ORDRE / LIST_ORDRE / INST / LIST_INST / FREQ / LIST_FREQ / PRECISION / CRITERE`

Sélection dans une structure de données `resultat` [U4.71.00].

### 3.5.8 Lecture des `MODE_MECA`

On peut lire des modes propres stockés au format `IDEAS`. Mais pour pouvoir les réutiliser dans les opérateurs de dynamique (en particulier `DYNA_TRAN_MODAL`), on a besoin des matrices assemblées (rigidité et masse) associées à ces modes. Les mots-clés `MATR_RIGI` et `MATR_MASS` (rappelant ceux de l'opérateur de calcul modal `CALC_MODES`) permettent de renseigner ces deux matrices.

### 3.5.9 A propos d'une alarme

Il peut arriver que les valeurs lues sur le fichier ne puissent être recopiées dans les champs de la `SD_RESULTAT` produite. Par exemple, un champ de pression existant sur des mailles surfaciques, ne peut être recopié sur un modèle ne contenant que des éléments 3D. Dans ce cas, le code émet un message d'alarme ressemblant à :

```
<A> <LIRE_RESU> <LRCEME>
      VALEURS NON AFFECTEES DANS LE CHAMP : 3699
      VALEURS LUES DANS LE FICHER          : 3699
```

### 3.5.10 Opérande `TITRE`

◇ `TITRE`

Titre que l'on veut donner au résultat [U4.03.01].

### 3.5.11 Opérande `INFO`

◇ `INFO = inf`

Permet d'imprimer dans le fichier `MESSAGE` des informations liées aux déroulement de l'opérateur.

## 4 Exemples

### 4.1 Exemple 1 : lecture d'un résultat de type '`dyna_trans`'

On lit sur le fichier universel `IDEAS`, les champs de déplacement, de vitesse et d'accélération aux instants 1., 2., 3., 4. et 5.,

```
resu = LIRE_RESU ( FORMAT = 'IDEAS' ,
```

```

MODELE = mo,
TYPE_RESU = 'DYNA_TRANS' ,
NOM_CHAM = ( 'DEPL' , 'VITE' , 'ACCE' , ),
INST = ( 1. , 2. , 3. , 4. , 5. , ),
    )
    
```

## 4.2 Exemple 2 : lecture d'un resultat de type 'evol\_noli' en définissant les critères de recherche

On lit sur le fichier universel IDEAS, les champs de variables internes et de déformations à l'instant 15. en tenant compte de critères de recherche utilisateur.

```

INIT =LIRE_RESU( MODELE = mo,
                  FORMAT = 'IDEAS' ,
                  TYPE_RESU = 'EVOL_NOLI' ,
                  NOM_CHAM = ('VARI_ELNO', 'EPSA_ELNO'),
                  NB_VARI = 2,
                  INST = 15.,
                  FORMAT_IDEAS = ( _F(NOM_CHAM = 'VARI_ELNO' ,
                                     NUME_DATASET = 57,
                                     RECORD_6 = (1, 4, 3, 9999, 2, 6),
                                     POSI_ORDRE = (7, 4, ),
                                     POSI_INST = (8 1)
                                     NOM_CMP = ('V1', 'V2', 'V3', 'V4', )),
                                _F(NOM_CHAM = 'EPSA_ELNO' ,
                                     RECORD_6 = (1, 4, 4, 3, 2, 6),
                                     NOM_CMP = ('EPXX', 'XXX', 'EPZZ',
                                                'EPXY', 'EPXZ', 'EPYZ'))),
    )
    
```

### Fichiers IDEAS à lire

```

-1
57 %VALEURS AUX NOEUDS DES ELEMENTS
ASTER 3.05.30 CONCEPT U CALCULE LE - CHAMP PAR ELEMENT AUX NOEUDS DE NOM
CHAMP PAR ELEMENT AUX NOEUDS DE NOM SYMBOLIQUE VARI_ELNO_ELGA - VARI_1 (ELNO)
ASTER 3.05.30 CONCEPT U CALCULE LE 29/12/95 A 09:56:55 DE TYPE EVOL_NOLI
CHAMP PAR ELEMENT AUX NOEUDS DE NOM SYMBOLIQUE VARI_ELNO_ELGA

Record 6 →
Record 7 →
Record 8 → 0.15000E+02 ← POSI_INST (8,1) 1 ← POSI_ORDRE (7,4)

0 2 6
6 % MAILLE MA2
2.07919E-05 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00
2.07919E-05 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00
2.07919E-05 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00
2.07919E-05 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00
2.07919E-05 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00
2.07919E-05 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00
2.07919E-05 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00
-1
-1
57 %VALEURS AUX NOEUDS DES ELEMENTS
ASTER 3.05.30 CONCEPT U CALCULE LE - CHAMP PAR ELEMENT AUX NOEUDS DE NOM
CHAMP PAR ELEMENT AUX NOEUDS DE NOM SYMB - EPXX EPXY EPYY EPXZ EPYZ EPZZ (ELNO)
ASTER 3.05.30 CONCEPT U CALCULE LE 29/12/95 A 09:56:55 DE TYPE EVOL_NOLI
CHAMP PAR ELEMENT AUX NOEUDS DE NOM SYMBOLIQUE EPSA_ELNO

1 4 4 3 2 6
2 1 1 1
0.15000E+02
1 8 6 % MAILLE MA2
0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00
0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00
0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00
0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00
0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00
0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00
0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00
0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00
-1
    
```

## 4.3 Exemple 3 : lecture d'un résultat de type 'evol\_ther' en définissant les critères de recherche

On lit sur le fichier universel IDEAS, le champ de température pour l'instant 0.8 en tenant compte de critères de recherche utilisateur.

```
TEMP = LIRE_RESU (  MAILLAGE      = mail,
                   UNITE          = 19,
                   FORMAT         = 'IDEAS' ,
                   TYPE_RESU      = 'EVOL_THER' ,
                   NOM_CHAM       = 'TEMP' ,
                   INST           = 0.8,
                   FORMAT_IDEAS   =_F(  NOM_CHAM      = 'TEMP' ,
                                         NUME_DATASET = 2414,
                                         RECORD_3    = (1,) ,
                                         RECORD_9    = (2,4,1,5,2,1) ,
                                         POSI_ORDRE   = (10,7) ,
                                         POSI_INST    = (12,1) ,
                                         ),
                   )
```

## 4.4 Exemple 4 : lecture d'un résultat de type 'evol\_char' au format 'ENSIGHT'

On lit sur le fichier ENSIGHT tous les champs de pression

```
pres = LIRE_RESU (  FORMAT        = 'ENSIGHT' ,
                   NOM_FICHIER    = 'face6.result' ,
                   MODELE         = mo,
                   TYPE_RESU      = 'EVOL_CHAR' ,
                   NOM_CHAM       = 'PRES' ,
                   TOUT_ORDRE     = 'OUI' ,
                   )
```

### Fichiers ENSIGHT à lire

```
FICHIER face6.result
*****
```

```
      1      0      1
      3
      .50000E-03 .10000E-02 .15000E-02
      1      1
face6.geom
presface6.***  pression
```

```
FICHIER face6.geom
*****
```

```
Fichier Enight measured geometry file
particles coordinates
      10
      13  1.48000e+03-7.77010e+027.42010e+02
      14  1.48000e+03-7.77010e+027.42010e+02
```

15 1.48000e+03-7.77010e+027.42010e+02  
16 1.48000e+03-7.77010e+027.42010e+02  
17 1.48000e+03-7.77010e+027.42010e+02  
18 1.48000e+03-7.77010e+027.42010e+02  
19 1.48000e+03-7.77010e+027.42010e+02  
20 1.48000e+03-7.77010e+027.42010e+02  
21 1.48000e+03-7.77010e+027.42010e+02  
22 1.48000e+03-7.77010e+027.42010e+02

FICHIER presface6.\*\*\*  
\*\*\*\*\*

presface6.001  
.10000E+00 .10000E+00 .10000E+00 .10000E+00 .10000E+00  
.10000E+00  
.10000E+00 .10000E+00 .10000E+00

presface6.002  
.10000E+01 .10000E+01 .10000E+01 .10000E+01 .10000E+01  
.10000E+01  
.10000E+01 .10000E+01 .10000E+01

presface6.003  
.10000E+02 .10000E+02 .10000E+02 .10000E+02 .10000E+02  
.10000E+02  
.10000E+02 .10000E+02 .10000E+02

## 4.5 Exemple 5 : lecture d'un evol\_ther au format MED

```
LIRE_RESU ( FORMAT = 'MED', MAILLAGE = MA,
            UNITE = 21, TOUT_ORDRE = 'OUI', TYPE_RESU = 'EVOL_THER',
            FORMAT_MED = _F ( NOM_CHAM = TEMP,
                              NOM_CHAM_MED = 'THERDEP_TEMP', )
          )
```

## Annexe 1 : FORMAT\_IDEAS : valeurs par défaut

Dans ce paragraphe, nous présentons pour chaque champ (NOM\_CHAM) les critères de recherche par défaut utilisés pour localiser dans le fichier universel les résultats à lire.

CHAM_NO				
NOM_CHAM	'DEPL'	'VITE'	'ACCE'	'TEMP'
NUME_DATASET	55	55	55	55
RECORD_3				
RECORD_6	1 4 3 8 2 6	1 4 3 11 2 6	1 4 3 12 2 6	2 4 1 5 2 1
RECORD_9				
POSI_ORDRE	7 4	7 4	7 4	7 4
POSI_INST	8 1	8 1	8 1	8 1
POSI_FREQ				
NOM_CMP	'DX' 'DY' 'DZ' 'DRX' 'DRY' 'DRZ'	'DX' 'DY' 'DZ' 'DRX' 'DRY' 'DRZ'	'DX' 'DY' 'DZ' 'DRX' 'DRY' 'DRZ'	'TEMP' 'TEMP_MIL' 'TEMP_INF' TEMP_SUP'

CHAM_ELEM				
NOM_CHAM	'VARI_ELNO'	'EPSA_ELNO'	'SIEF_ELNO'	'PRES'
NUME_DATASET	57	57	57	57
RECORD_3				
RECORD_6	1 4 3 0 2 6	1 4 4 3 2 6	1 4 4 2 2 6	1 4 1 15 2 1
RECORD_9				
POSI_ORDRE	7 4	7 4	7 4	7 4
POSI_INST	8 1	8 1	8 1	8 1
POSI_FREQ				
NOM_CMP	'V1' 'V2' 'V3' 'V4' 'V5' 'V6' 'V7' 'V8' ... .. 'V9' 'V30'	'EPXX' 'EPXY' 'EPYY' 'EPXZ' 'EPYZ' 'EPZZ'	'SIXX' 'SIXY' 'SIYY' 'SIXZ' 'SIYZ' 'SIZZ'	'PRES'