

Procédure *POURSUITE*

1 But

Poursuivre une étude à partir de la sauvegarde au format JEVEUX ou au format HDF de sa base 'GLOBALE'.

La syntaxe apparemment complexe de cette procédure ne doit pas inquiéter l'utilisateur, l'appel avec les opérandes par défaut, est suffisant dans la plupart des cas : *POURSUITE* ()

L'usage de cette commande est tout à fait semblable à celui de *DEBUT*. Les mots-clés dont le comportement est identique dans *DEBUT* et *POURSUITE* sont uniquement décrits dans [U4.11.01].

2 Syntaxe

```
POURSUITE
(
  ◇ PAR_LOT = / 'OUI' , [DEFAULT]
                / 'NON' ,
  ◇ IMPR_MACRO = / 'NON', [DEFAULT]
                / 'OUI',

  ◇ LANG = lang, [TXM]

  ◇ BASE = _F (
      ◆ FICHIER = 'VOLATILE',
                ◇ / | LONG_ENRE= lenr, [I]
                  | NMAX_ENRE= nenr, [I]
                  | LONG_REPE= lrep, [I]
                ),
  ◇ CODE = _F (
      ◆ NOM = nom code, [K8]
      ◇ UNITE = | 15 , [DEFAULT]
                | unitd, [I]
                ),
  ◇ ERREUR = _F (
      ERREUR_F = / 'ABORT', [DEFAULT]
                  / 'EXCEPTION',
                ),
  IGNORE_ALARM = l_vale, [l_Kn]

  ◇ DEBUG = _F(
      ◇ JXVERI = / 'OUI',
                  / 'NON',
      ◇ ENVIMA = 'TEST', [l_Kn]
      ◇ JEVEUX = / 'OUI',
                  / 'NON',
      ◇ SDVERI = / 'OUI',
                  / 'NON',
      ◇ HIST_ETAPE = / 'NON',
                     / 'OUI',
  ),

  ◇ MESURE_TEMPS = _F (... voir commande DEBUT),
  ◇ MEMOIRE = _F( ... voir commande DEBUT),

  ◇ RESERVE_CPU = _F( / VALE = vale [R]
                      / POURCENTAGE = pcent [R]
                      ◇ BORNE = / bv, [R]
                          / 180. [DEFAULT]
                      ),
  ◇ FORMAT_HDF = / 'NON', [DEFAULT]
                  / 'OUI',
)
)
```

3 Principe de fonctionnement

Cette procédure affecte, en outre, les ressources mémoire nécessaires à la poursuite du calcul.

Les opérandes de la commande sont homologues de ceux de la procédure **DEBUT** [U4.11.01]. Ils permettent de préciser certaines ressources affectées à la nouvelle exécution.

L'étude engagée précédemment se poursuit par un ensemble de commandes commençant par **POURSUITE** et se terminant par **FIN** [U4.11.02].

Les commandes placées avant **POURSUITE** (sauf évidemment **DEBUT**) ou après **FIN**, si elles sont syntaxiquement correctes, sont ignorées.

La procédure **POURSUITE** qui est exécutée, dès sa lecture par le superviseur, effectue les tâches suivantes :

- définition des unités logiques des fichiers utilisés en impression,
- allocation des fichiers associés aux bases de données gérées par **JEVEUX**,
- lecture des catalogues de commandes mais pas des catalogues des éléments qui ont été recopiés sur la base de données lors de la première exécution.

Les opérandes sont à utiliser pour dérouter les différents fichiers sur des numéros d'unité logique différents des numéros affectés par défaut ou pour ajuster certains paramètres de fichiers.

Les concepts de python simples (de type variable) créés lors d'une exécution précédente sont conservés dans un fichier associé à la base **JEVEUX** (*pick.1*). Lors de l'exécution de la procédure **POURSUITE** ces concepts sont régénérés et peuvent donc être utilisés sous le nom sous lequel ils ont été créés.

Remarque

| Ne sont pas sauvegardés dans le *pick.1*, les objets python de type classe, fonction et type.

4 Opérandes

L'opérande **PAR_LOT** et les mots clés **LANG** et **DEBUG** sont identiques à ceux de la procédure **DEBUT** [U4.11.01].

Le mot clé **BASE** est différent pour la procédure **POURSUITE**.

Le mot clé **HDF** permet la relecture d'une base stockée au format « Hierarchical Data Format ».

4.1 Mot clé **BASE**

BASE =

La fonctionnalité de ce mot clé est de redéfinir les valeurs des paramètres des fichiers d'accès direct associés aux «base de données» dans le cas où l'on ne désire pas utiliser ceux fixés par défaut. La taille maximum du ou des fichiers associés, et par conséquent le nombre maximum d'enregistrements, peut être redéfini à l'aide du paramètre passé sur la ligne de commande derrière le mot clé

En mode **POURSUITE**, certaines caractéristiques de la base **GLOBALE** ne peuvent plus être modifiées. Valeurs par défaut des paramètres associés aux bases de données

VOLATILE

NMAX_ENRE	62914	
LONG_ENRE	100	K mots
LONG_REPE	2000	

Le mot vaut 8 octets sur plate-forme 64 bits sous LINUX 64, TRU64 et IRIX 64, 4 octets sur plate-forme 32 bits sous SOLARIS, HP-UX et WINDOWS-NT, LINUX.

Sous Linux 64, la procédure **POURSUITE**, avec les valeurs par défaut, allouera un fichier d'accès direct d'au plus 62914 enregistrements de 100 *Kmot* (le *K* vaut 1024) pour la base 'VOLATILE'.

Remarques :

La taille réelle du fichier est dynamique ; elle dépend du volume d'informations à stocker effectivement. Mais cette taille est limitée par les conditions d'exploitation et un paramètre défini parmi les valeurs caractérisant la plate-forme. Sur la plate-forme de référence Linux 64 la taille maximum est fixée à 48 *Go*. Cette valeur peut-être modifiée en passant un argument sur la ligne de commande de l'exécutable derrière le mot clé `-max_base` taille où *taille* est une valeur réelle mesurée en *Mo*, ou dans les arguments de ASTK [U1.04.00].

Sur les plates-formes 32 bits, la taille maximum est fixée à 2.047 Go (2 147 483 647), mais le code gère plusieurs fichiers pour aller au delà de cette limite lorsque le paramètre `-max_base` est passé en argument.

Pour la base Globale, qui peut être sauvegardée et ré-utilisée en donnée d'un calcul, la taille maximum en « **POURSUITE** » est conservée telle quelle si le paramètre `-max_base` n'est pas utilisé, mais peut-être redéfini au besoin de cette manière.

4.1.1 Opérande **FICHIER**

♦ **FICHIER** =

Nom symbolique de la base considérée.

Seul le paramètre de la base de données 'VOLATILE' peut être redéfini.

4.1.2 Opérandes **LONG_ENRE** / **NMAX_ENRE** / **LONG_REPE**

Définition des paramètres de la base de données (fichiers d'accès direct).

/ | **LONG_ENRE** = *lenr*

lenr est la longueur des enregistrements en *Kmots* des fichiers d'accès directs utilisés.

Remarque :

Le gestionnaire de mémoire JEVEUX utilise ce paramètre pour déterminer deux types d'objets : les gros objets qui seront découpés en autant d'enregistrements que nécessaire, et les petits objets qui seront accumulés dans un tampon de la taille d'un enregistrement avant d'être déchargé.

| **NMAX_ENRE** = *nenr*

nenr est le nombre d'enregistrements par défaut, cette valeur est déterminée à partir de **LONG_ENRE** et d'un paramètre d'exploitation sous LINUX_64 fixé à 12 *Go* (51 539 607 552 octets) pour la taille maximale du fichier associé à une base de données.

Remarque :

*Les deux opérandes **LONG_ENRE** et **NMAX_ENRE** doivent être utilisés avec précaution, un mauvais usage pouvant conduire à l'arrêt brutal du programme par saturation des fichiers d'accès direct. La cohérence entre la taille maximale du fichier et la valeur résultant du produit des deux paramètres **LONG_ENRE** et **NMAX_ENRE** est vérifiée en début d'exécution.*

| **LONG_REPE** = *lrep*

lrep est la longueur initiale du répertoire (nombre maximal d'objets adressables par JEVEUX), elle est gérée dynamiquement par le gestionnaire de mémoire qui étend la taille du répertoire et de tous les objets système associés au fur et à mesure des besoins.

4.2 Mot clé **FORMAT_HDF**

`FORMAT_HDF = 'OUI'`

Permet de relire une base `GLOBALE` sauvegardée dans un fichier au format HDF (cf commande `FIN` [U4.11.02]). La base est alors reconstruite à partir des objets `JEVEUX` stockés dans le fichier, ce fichier peut avoir été construit sur une plate-forme différente (système d'exploitation, plate-forme 32 ou 64 bits). Les caractéristiques de la base originale sont relues dans le fichier et la base est reconstruite à l'identique (on conserve par exemple la longueur des enregistrements).

Le fichier associé à la base `GLOBALE` au format HDF est nommé `bhdf.1` dans le répertoire d'exécution du code.

5 Exemple d'utilisation

L'utilisation standard de cette procédure est :

`POURSUITE ( )`

Les tests `yyy100a` et `yyy100b` illustrent l'utilisation de `RESERVE_CPU`

Les tests `forma04b`, `ssnv156a`, `ssnv166b`, `yyy108`, illustrent l'utilisation de `MEMOIRE`.